M/s

Managing distributed workloads

Benjamin Hanser (bwh2124)
Miranda Li (mjl2206)
Mengdi Lin (ml3567)

Table of contents

1. Introduction **5**
Motivation 5
Key Features 5

2. Language Tutorial **5**
Environment Setup 5
Installation under Ubuntu 15.10 5
To run and test: 6
To compile a M/s program file called remote_test.ms: 6
To run a M/s program on multiple machines: 7
Syntax examples 7
Job syntax 7
Primitives 7
Scoping 7
Operators 8
Control flow 8

3. Language Reference Manual **8**
1. Introduction 8
2. Lexical elements 8
2.1 Comments 8
2.2 Delimiters 8
2.3 Identifiers 8
2.4 Data types 9
Primitive Types 9
Vectors 9
Strings 9
Struct (C-like Struct) 10
Jobs 10
3. Expressions and operators 11
3.1 Expressions 11
3.2 Arithmetic operators 11
3.3 Logical operators 11
3.4 Assignment, = 12
3.5 Vector operators, [ ] and :: 13
3.6 Struct operator, -> 13
3.7 Job status operator, . 13
4. Keywords 13

4.1 Control flow 13
4.2 Printing 14
4.3 Function (job) definition 14
job 14
return 14
void 14
4.4 Jobs 14
Job states 14
failed 14
finished 15
running 15
Job operators 15
get 15
cancel 15
4.5 master 15
4.6 Remote keyword 15
5. Operator and Keyword precedence 16
6. Error handling 16
7. Job Scheduling 17
8. Scope 17
9. Program Structure 17
10. Frequently Asked Questions 17
11. Grammar 19
12. Sample program (gcd) 19
13. References 20

4. Project Plan 20
4.1 Planning process 20
4.2 Programming style guide 20
4.3 Timeline 21
4.4 Member roles and responsibilities 23
4.5 Software development environment 24
4.6 Project log 24

5. Architectural Design 25
5.1 Block Diagrams 25
5.2 Compiler components 26
The Scanner 26
The Parser and AST 26
The Semantic checker 26

The Code generator 26
5.3 Runtime Implementation 26
5.4 Network Protocol 28

6. Test Plan 28
6.1 Sample programs, source and target 28
6.2 Test suites and automation 28
Runtime tests 28
Compiler tests 29
6.3 List of tests 29

7. Lessons Learned 32
7.1 Member reflections 32
Ben 32
Mengdi 32
Miranda 33
7.2 Advice to future groups 33

8. Appendix: That Unevolved Vestige That Gives You Appendicitis 34

1. Introduction

Motivation

M/s stands for Master/slave and is a language for implementing a distributed system (master-slaves relationship). It allows a master server distribute work across slave nodes. The user defines the master function (akin to a main function), as well as job functions that can be run either locally or on slaves without worrying about socket handling, threading, and network packet serialization for their job inputs.

Key Features

M/s provides a default runtime that does socket handling, threading, and job scheduling. Code generation provides serialization and deserialization procedures for the inputs and outputs of user-defined job functions. We have removed the fuzziness that comes from manipulating data structures that contain pointers and references because all assignments of variables are copy assignments by default. All function arguments are passed by value.

M/s supports C++-like data structure called vectors in place of C arrays, and also supports C-like structs. M/s has automatic garbage collection; once variables go out of scope, they will be cleaned up. What the users need to provide to M/s are simply the definitions of a master function (similar to C's main function), and the definitions of job functions.

The syntax of remote jobs mimic those of Java's Futures: the `remote` keyword is a non-blocking operation used to submit a job to be run remotely. `get` keyword is a blocking operation used to retrieve the output of a remote job. If the job has not finished running when `get` is called, `get` blocks until the job finishes running and provides an output.

2. Language Tutorial

Environment Setup

M/s needs the OCaml llvm library, which is most easily installed through opam. Install LLVM and its development libraries, the m4 macro preprocessor, and opam, then use opam to install llvm. The version of the OCaml llvm library should match the version of the LLVM system installed on your system.

Installation under Ubuntu 15.10

LLVM 3.6 is the default under 15.10, so we ask for a matching version of the OCaml library.

```
sudo apt-get install -y ocaml m4 llvm opam
opam init
```

```
opam install llvm.3.6 ocamlfind
eval `opam config env`
```

To run and test:

In the base directory, make builds the compiler, builds the runtime, and does the necessary linking between the two:

```
$ make
ocamlbuild -use-ocamlfind -pkgs llvm,llvm.analysis -cflags -w,+a-4 \
              mscompile.native
Finished, 22 targets (0 cached) in 00:00:02.
/Applications/Xcode.app/Contents/Developer/usr/bin/make -C runtime
gcc -g -Wall   -c -o master.o master.c
gcc -g -Wall   -c -o job_list.o job_list.c
gcc -g -Wall   -c -o slave_queue.o slave_queue.c
gcc -g -Wall   -c -o util.o util.c
ar rc libmsmaster.a master.o job_list.o slave_queue.o util.o
ranlib libmsmaster.a
gcc -g -Wall   -c -o slave.o slave.c
ar rc libmsslave.a slave.o util.o
ranlib libmsslave.a
cp runtime/libms* compiler

Testing the compiler:
$ cd compiler
$ ./testall.sh
test-arith1...OK
test-arith2...OK
test-arith3...OK
test-fib...OK
...
fail-while1...OK
fail-while2...OK

You can also run tests individually by running:
$ ./testall.sh tests/test-arith1.ms
```

To compile a M/s program file called remote_test.ms:

```
Inside the compiler directory:
$ ./mscompile.native < remote_test.ms > remote_test.ll
$ llc remote_test.ll
$ gcc -L. remote_test.s -lmsmaster -pthread -lm -o master
$ gcc -L. remote_test.s -lmsslave -pthread -lm -o slave
$ ./master & ./slave
```

To run a M/s program on multiple machines:

Start the master binary first, and then start the slave binaries. The launch syntax looks as follows:

Master session:
```
$ ./master [PORT]
```
Slave sessions:
```
$ ./slave [HOST] [PORT]
```

Master will start a server, which all slaves will connect to. Once the slaves are connected, they will begin receiving jobs from master, and will execute those jobs.

Syntax examples

Please see the Language Reference Manual section for a more detailed description. This provides some examples of how to use the features of our language.

Job syntax

Defining a jobs (as functions that can be sent to slaves):
```
job int f(int a) {
  return 2*a;
}
```

Defining a job object (to refer to a running job):
```
job<int> a = remote f(6);
//job object a contains an int output
```

Primitives

Our primitives are int; bool; double; string.

Scoping

Master and jobs have their own hierarchy of scope. There are no globally accessible variables accessible to any combination of the two.

Vector:
```
Declaration: vector<int> a;
Nested vectors: vector<vector<string>> a;
Access: a[0] = 2;
```

Struct:
Declaration is done at the same level as master and jobs:
```
struct e {
       int a;
```

```
        vector<int> b;
        struct inner_e x;
};
struct inner_e {
        int c;
};
Instantiation is done anywhere within master or jobs: struct e s;
Field access: s->x->c;
```

Operators

Binary operators: == != < > <= >= && ||
Unary operators: !

Control flow

If, else, while

3. Language Reference Manual

1. Introduction

M/s is a language based on C. M/s can be used as a general programming language, but is intended for managing distributed systems, in which a master machine or terminal can send functions to be done remotely on slave machines. M/s uses the LLVM IR.

2. Lexical elements

2.1 Comments

Characters // start a single-line comment. Characters /* start a multi-line comment, and */ end a multi-line comment.

2.2 Delimiters

Semicolon delimits a statement; space, tab, and newline delimit a token.

2.3 Identifiers

An identifier is a case-sensitive ASCII string of letters, underscores, and digits where the first character must be a letter. M/s interprets an identifier based on its type. All variables of M/s are statically typed.

2.4 Data types

Primitive Types

M/s has four primitive types: int, double, boolean, and string. An integer is a two's complement signed 32-bit integer number. A double is a two's complement signed 64-bit floating point number. A boolean is a one-byte value that can either be "true" or "false". There is no implicit casting of primitive types, i.e. a one-character string cannot be casted into an integer and vice versa. We also consider string to be a primitive type; see the section on string, *infra*. All primitive types have initial values upon declaration. Int and double have an initial value of 0. A boolean is always initialized to false. Vectors have an initial value of size 0, with no memory allocated.

Vectors

A vector is a contiguous sequence of elements of the same type. Internally, a vector is implemented as a LLVM struct with three fields: size, length, and pointer. Size field indicates the number of elements in the vector. Length indicates the amount of memory allocated to the vector. A vector of *n* elements shall allocate storage for 2^(ceil(log_2(*n*))) elements, providing amortized constant time append. A vector has an initial value of an empty vector by default. To declare a vector, you must declare it in the form "vector<type>" where type is the type of the elements in the vector. Vectors support copy pushback via "::" operator and copy assignments. M/s provides automatic garbage collection for data stored inside vectors.

Use:

```
vector<int> a;
a::2; a::3; a::4;
vector<int> b = a; //copied
print(a[0]); // prints "2"
```

Strings

A string is a sequence of characters. In the backend, it is implemented as a vector of characters, even though the character type is not exposed to the user. We consider string to be a primitive type. Initial value of a string is an empty string. String has a concatenation operator "<<" which can be used to concatenate two strings and create a new string.

Use:

```
string a="hello";
string b = "hey";
string c = a << b;

prints(c); // prints "hellohey"
```

Struct (C-like Struct)

A struct contains a list of variables under one name in a block of memory. The name is used to access different variables within the name via the “->” operator. M/s supports nested structs.

Struct types should be declared at the same level as the master block and jobs (the order of declaration does not matter) as follows:

```
struct example {
    int e;
    int c;
    struct example_inner inner;
};
struct example_inner {
    int x;
    struct example_inner_inner inner;
};
```

Structs can then be initialized as follows:

```
struct example a;
a->inner->x = 1;
```

Jobs

A job (data type) is an entity that represents a job (thread of execution). Internally, it is implemented as an integer. This integer represents a unique job ID corresponding to a specific job that has been submitted to a slave. This integer is used by the runtime to correctly populate the return results of a job to the right output pointer. It also contains the boolean states “running,” “finished,” and “failed.” See more about these states under Jobs section. Further discussions on failures are under the topic “Error Handling”, *infra* .

Type	**Description**
int	a 32-bit integer
double	two’s complement signed 64-bit floating point number
string	standard string
boolean	true or false
vector	As in C++, push back via “::” operator
struct	As in C struct. Variables within the struct can be accessed via “->” operator.

job	● Represents a job, and also is used as the keyword to define a new job ● To assign a job: job<type> c = remote f() ● To retrieve output: get job ● To cancel a job: cancel job ● Is pronounced /dʒoʊb/

3. Expressions and operators

3.1 Expressions

An expression contains at least one operand, and may contain unary or binary operators specifying operations on the operand.
All unary operators have format <*operator*> <*expression*>
All binary operators have format <*expression*> <*operator*> <*expression*>

3.2 Arithmetic operators

Binary operators: + - * / %
These operators can only be applied to ints and doubles.

3.3 Logical operators

Unary operators

- !
    - *! expr1 is true if expr1 is false, false otherwise*
    - *Only works for booleans*

Binary operators

- ==
    - *expr1 == expr2 is true if the values of the two expressions are equal, false otherwise*
    - *Works for ints, doubles, strings, booleans, strings, vectors, and jobs*
- !=
    - *expr1 != expr2 is true if the values of the two expressions are not equal, false otherwise*
    - *Works for ints, doubles, strings, booleans, strings, vectors, and jobs*
- <
    - *expr1 < expr2 is true if the value of expr1 is less than the value of expr2, false otherwise*
    - *Only works for ints and doubles*
- >
    - *expr1 > expr2 is true if the value of expr1 is greater than the value of expr2, false otherwise*
    - *Only works for ints and doubles*

- <=
  - *expr1 < expr2 is true if the value of expr1 is less than or equal to the value of expr2, false otherwise*
  - *Only works for ints and doubles*
- >=
  - *expr1 < expr2 is true if the value of expr1 is greater than or equal to the value of expr2, false otherwise*
  - *Only works for ints and doubles*
- &&
  - *expr1 && expr2 is true if expr1 is true and expr2 is true, false otherwise*
  - *Only works for booleans*
- ||
  - *expr1 || expr2 is true if expr1 is true or expr2 is true, false otherwise*
  - *Only works for booleans*

3.4 Assignment, =

Assigns a value to a variable.
There is different behavior for different *left hand side types*:

- Primitives, string, struct, vectors *when RHS is not* `get`*-ting from a job*
  - Performs a deep copy
  - *Use*
    - `int a = 3;`
- Primitives, string, struct, vectors *when RHS is* `get`*-ting from a job*
  - Blocks until job is finished
  - Performs a shallow copy (because the result from the job will not be used until we *get* it)
  - *Use*
    - `int a = 3;`
    - `int b = 6;`
    - `int x = get remote gcd(a,b); // blocks until gcd finishes`
- Job
  - *Does not block;* it is a lazy assignment
  - Will only block when get <job_name> is called
  - *Use*
    - `int a = 3;`
    - `int b = 6;`
    - `job<int> c = remote gcd(a,b); // does not block`
    - `int x = get c; //blocks`
    - `print("x is ready when this statement is run");`

3.5 Vector operators, [ ] and ::

- [x]
    - Random access operator for an element of the vector.
- vector<int> a; a::1;
    - Push back operator that copies an element into the end of the vector

3.6 Struct operator, ->

-> is used to access variables within a struct.

- e.g. a->var

3.7 Job status operator, .

To access boolean statuses in a job, use the operator '.' See section on Job states under Jobs, *infra.*

- e.g. j.running, j.failed, j.finished

4. Keywords

4.1 Control flow

while

Continues looping on a given set of statements while the given expression evaluates to true

```
while (<expression>) {
	//statements;
}
```

if

Runs a given set of statements if the given expression evaluates to true. Otherwise skips the statements

```
if (<expression>) {
	//statements;
}
```

else

After an if statement, jump to statements in else block if provided

```
if (<expression>) {
	//statements;
}
else {
	//other statements;
}
```

If, else, and while will include the following line automatically without need for brackets. Because of this, we get `else if` for free.

4.2 Printing

There are different *print* functions for different types. Each prints to the command line

- *print* for ints
- *printb* for bools
- *printd* for doubles
- *prints* for strings

Use:

```
int a = 2;
string s = “hello”;
double d = 3.4;
print(a);
printd(d)
prints(s);
// Prints 23.4hello to terminal
```

4.3 Function (job) definition

job

A job is a function that can be sent to slave nodes to run with remote, run on a separate thread in master with local, or directly run by master.

Use:

```
job <return_type> <job_name> ( <input type 1> <input 1>, <input type 2> <input 2> ...) {
        <job_definition>;
}
```

return

The return keyword indicates the value to return from a job or a function.

void

The return type for jobs and functions that do not return anything.

4.4 Jobs

Declaring a job variable:

```
job<int> a = remote gcd(4,6);
```

Job states

failed

Initial state: false

failed state is set to true if and only if the job encountered network errors, runtime exceptions, or general errors. Note that cancelling a job does not set this state to true.

finished

Initial state: false
done state is set to true if and only if the job has successfully finished executing and is ready to return its result.

running

Initial state: true.
running state is set to true if and only if the job is still running.

Job operators

get

Get the return value from a job. Will block if job is not done.
Use:

```
job a = remote f();
int i = get a; //blocks until a is done
```

cancel

Cancels a running job. Any lazy assignments that were made by that job are reverted. Canceling a failed, done, or non-existent job will generate an error.

Use:

```
int a = 3;
int b = 6;
job c = remote gcd(a,b); // c refers to the job running gcd
cancel c;
int y = get c; // generates an error
```

4.5 master

Contains “server side code” written by the user that issues jobs to slave nodes. The master section of the code, indicated by the “master” keyword, (see program structure and samples section) is like the main function. The internal implementation of master functionalities (such as socket handling and asynchronous data assignment) will be done in C++ and linked against the compiled user codes.

4.6 Remote keyword

Denotes a job should be executed on a slave node. This keyword can only be used within master block.

Use:

```
// gcd with inputs a and b are run on a slave node, referenced by job c.
int a = 3;
int b = 6;
job<int> c = remote gcd(a,b);
```

5. Operator and Keyword precedence

In order of precedence:

1. Function calls ()
2. Parentheses '()'
3. -> [ ]
4. !
5. concat
6. * /
7. + -
8. < <= > >=
9. == !=
10. &&
11. ||
12. size
13. remote
14. = (assignment)
15. get cancel
16. type declaration
17. control flow
18. return

6. Error handling

An exception occurs if a remote job fails and the program tries to access the variable associated with the output of the job.

```
job<int> a = remote gcd(3,5); //remote job runs and then fails because of network error
int x = get a; //program crashes because user is trying to access a and it has failed
```

To fix the above code such that the program doesn't crash, the user should check whether a is ready and if not, whether job of variable a has failed as follows:

```
job<int> a = remote gcd(3,5); //remote job fails because of network error
int x;
while(!a.finished && !a.failed) {}  //wait for a to finish or to fail

if (a.failed) {
    //error handling
```

```
    x = -1;
}
```

Basic information about the job's status is maintained by the runtime. There are three possible statuses that the user can access, which are not mutually exclusive: running, finished, and failed.

7. Job Scheduling

Every time a slave joins the network, it is added to the list of slaves, and it will receive jobs from the head of the pending queue in competition with any other slaves. When a slave disconnects from the network, it is removed from the list of slaves, and any jobs running on that slave are restarted up to two times on other slaves. Once a job reaches its attempt cap, its status is set to 'failed'.

8. Scope

Master and jobs have their own hierarchy of scope. There are no globally accessible variables accessible to any combination of master and jobs.

9. Program Structure

Codes reside either in the master block ("master {...}") or jobs. The master block is similar to a main function in that user codes in this block will be executed.

For the program to work, a machine must have a master binary and another machine must have the corresponding slave binary. To obtain the two binaries, simply compile the user codes with flags and link the compiled *.s files with libmsmaster.a and libmsslave.a. The linkings will produce two binaries: master and slave. Both binaries must be of the same version for the program to work properly since both binaries should have the same job definitions.

10. Frequently Asked Questions[1]

1. *When can the result of a "remote" job be obtained? (For example, your "remote" keyboard* [sic] *seems to be some sort of "fork" operation that immediately returns yet starts a process in the background. When can the results of that process be obtained?)*

   There is only one way to obtain the result of a job, whether local or remote: through the get operator.

   ```
   job<int> a = gcd(10,12);
   int result = get a; //blocking call
   ```

[1] Frequency of 1, by Professor Edwards

The result of the argument is obtained at the execution of the second line. This line is blocking. The program will block on this line until the result of gcd returns and is assigned to the variable “result”.

2. *When does the system wait for processes to terminate and produce their results? What happens when a process attempts to access a variable whose value has yet to be assigned by a process running in parallel?*
   From the user’s perspective, the system waits for a job to terminate when the get keyword is invoked on that job. The implementation may actually reap the job earlier, asynchronously.

3. *How, if at all, can processes communicate?*
   Jobs cannot communicate with each other in any way. Master may only communicate with jobs through arguments passed at job creation and a return value passed at job termination.

4. *How will you implement this?!?*
   Here’s the rundown:

The compiler will be composed of two parts:

1) Master-Slave networking libraries written in C
2) The compiler that compiles everything that is user-defined (jobs, code within master, etc.) with ability to link to the master-slave networking libraries to produce the master and the slaves binaries

The compiler will create two files, master and slave, to be run on the respective machines.

Master will communicate with slaves through sockets. Master will send each slave a packet according to the following schema:

job to run (ordinal)	int
unique job id (to identify return value)	int
length	int
arguments (parsed according to signature)	`length` many bytes

Slave will send a packet back when it is done according to the following schema:

job that was run (ordinal)	int
unique job id (to identify return value)	int
length	int

return value	`length` many bytes

Master will run user code, and fire off jobs to the slaves. Slave will run one thread that listens to the socket for incoming jobs, parse each job request, and create a new thread for each job. Slave's binary will include all the code for all the jobs.

11. Grammar

Please see the scanner, parser, and AST files in the appendix for the full grammar of the language.

12. Sample program (gcd)

```
master {
	int a = 100;
	int b = 40;
	job<int> j1 = remote m3(a); // select a slave, tell it to create a thread of function
m3 on
				// input a;
	job<int> j2 = remote m4(b); // this remote job runs concurrently with the job above
	job<int> j3 = remote gcd(get j1, get j2); // waits for both j1 and j2 to be ready
	job<int> j4 = remote gcd(a,b); // doesn't run until j1 and j2 are ready because the
line
				// above waits
	job<int> c = remote m3(a);
	print(get j3, " ", get j4); // prints "20 20"
	while (!c.finished && !c.failed) {}
	if (!c.failed) { // c has finished successfully
		int output = get c; // blocking assignment from job c
		print(output); // only prints "300" if job c finished before the ~ operator
				//checked if job c was ready


}

job int gcd(int i, int j) { // function gcd takes int i and int j, returns int
	if (i - j == 0)
		return i;
	if (i > j)
		return gcd(i-j, j); // function call, not thread creation
	return gcd(i, j-i);
}

job int m4(int i) {
	return i * 4;
}
```

```
job int m3(int i) {
        return i * 3;
}
```

13. References

Ritchie, Dennis. *C Language Reference Manual*.
Stroustrup, Bjarne. *A Tour of C++*.

4. Project Plan

4.1 Planning process

The M/s team first assigned project roles and set up a weekly meeting time between ourselves (Sunday nights) and a meeting time with our advisor, Professor Edwards (Monday afternoons)[2]. Every week, we would assign a task to complete by next Sunday, and often we would work together on Sunday afternoons as well as separately during the week.

We used an open GitHub issue to keep track of milestones to reach (see section on timeline). Every time we had a meeting, we kept track of meeting minutes and decisions for reference. In our initial 3 meetings while we were writing the language reference manual, we aimed to be as specific as possible with the specification of the language. In general, we implemented features without much ambiguity about the specification.

For testing, Miranda setup the test framework and constantly urged everyone to write tests for the features they implemented, keeping her own testing work at a minimum; often, this required pestering people's pull requests :). Near the end of the project, she revisited all the code and made sure there were no gaps in testing, writing fail tests for every semant error and passing tests for new features.

4.2 Programming style guide

We used the following conventions while programming our M/s compiler, in order to ensure consistency, readability, and transparency:

- OCaml editing and formatting style for compiler architecture
- C language editing and formatting style for M/s programs
- Ben created the runtime library, with clear header files describing how to call functions from his library, which we all followed when calling runtime functions

[2] It must be said that these monday afternoon meetings were the absolute highlights of our weeks, of course!

- File names end in .ms
- Variable identifiers begin with lowercase letter and use underscores
- Function identifiers begin with lowercase letter and use underscores
- Always include a master block in M/s programs

We also followed collaboration guidelines[3]:

- Always write tests when implementing a feature; should be in the same pull request as the feature implemented
- Never push code to master that will break the compiler
- Create a separate branch when submitting a pull request

4.3 Timeline

Below is our overall timeline for the project

Date	**Milestone**
Feb 1	Initial discussion of ideas and roles assigned
Feb 8	Proposal
Feb 22	Language reference manual
March 5	Scanner, Parser, Semant, and AST for initial end-to-end
March 10	Runtime library and interface set-up by Ben
March 24	Codegen for initial end-to-end
March 27	Hello world
April 2	Job table, dynamic vector, int and bool serialization
April 9	Double end-to-end and serialization, structs
April 16	Vector end-to-end, struct end-to-end, job states interface, cancel
April 30	Struct-in-vector and vector-in-struct support, vector and struct serialization; string implementation with vector, struct copy
May 10	Error cleanup, update and create new tests Final report, demo, and presentation

Below is the specific features milestone we created for ourselves after we had end-to-end initially working:

[3] Though some of us were better at others than following it! Sorry-Miranda :)

4/2:

- 1d dynamic vector completes. multi-d vector segfaults on push_back and size. static vector works just file. (mengdi)
- job table accepts more than "f" (ben)
- int, bool serialization (ben)
- testing framework setup (miranda)

4/9:

- multidimensional dynamic vector should work for push_back and size (mengdi)
- struct end-to-end (miranda)
- struct end-to-end if miranda fails (mengdi)
- struct end-to-end if mengdi fails (ben)
- debug runtime to find out why runtime segfaults on OSX (ben)
- help debug multidimensional dynamic vector (ben)
- floats codegen end-to-end (miranda)
- float serialization (miranda)
- check miranda's float serialization (ben)
- get `Get` return value working after meeting with sedwards (ben)

4/16

- re-write vector codegen (ben)
- struct codegen (mengdi)
- semant for vector type checking (mengdi)
- job states interface (miranda) -- in job_list.h: assigned (running), unassigned, finished, failed, lost. Lost means you asked for it and I looked up in the job table and it was not there. Anything >0 for job states means it is running. 0 means job is waiting to be assigned.
- interface:
- .finished, .running, .failed
- get `Get` return value working after meeting with sedwards (ben)

4/23

- vector with struct support part 1 (ben)
- struct serialization (ben)
- vector serialization (mengdi)
- vector semant and job data type semant check (mengdi)
- string support with vector: string literal --> pushback sequentially into a vector (miranda)
- cancel (miranda)

4/30

- vector with struct support part 2 (mengdi/ben)
- semant loose ends

- parser shift reduce errors (miranda)
- Why are the tests failing?? :-((miranda)
- making string WORK (miranda)

5/7

- presentation / cool .ms demo
- vector of struct types support part 3 (mengdi/ben) - vector to vector copy assignment refer to old pull request
- serialization with vector of struct types (mengdi/ben)
- struct serialization (mengdi)
- vector serialization
- final report (everyone)

5/10

- presentation / cool .ms demo
- Readme and general cleanup (mimi)
- make more tests of things we've added, especially fail tests (mimi)
- semant loose ends (mimi)
- print vector and string (mengdi)
- local function arguments (ben)
- struct copy
- struct's vector (inner level) garbage collection
- final report (everyone)
- project deadline

4.4 Member roles and responsibilities

We collaborated on many features, doing different parts and helping each other out when stuck. This was the general division-of-labor:

Benjamin Hanser: System architect[4]

- Create runtime library
- Serialization
- Vector end-to-end
- Struct in vector and vector in struct
- Help everyone debug

Mengdi Lin: Language guru[5]

- Initial end-to-end parser, scanner, and ast

[4] Ben is our team's very own assembly x86-man
[5] Mengdi is an actual life guru, as well

- Vector end-to-end
- Struct codegen
- Serialization
- Struct in vector and vector in struct

Miranda Li: Manager and tester[6]

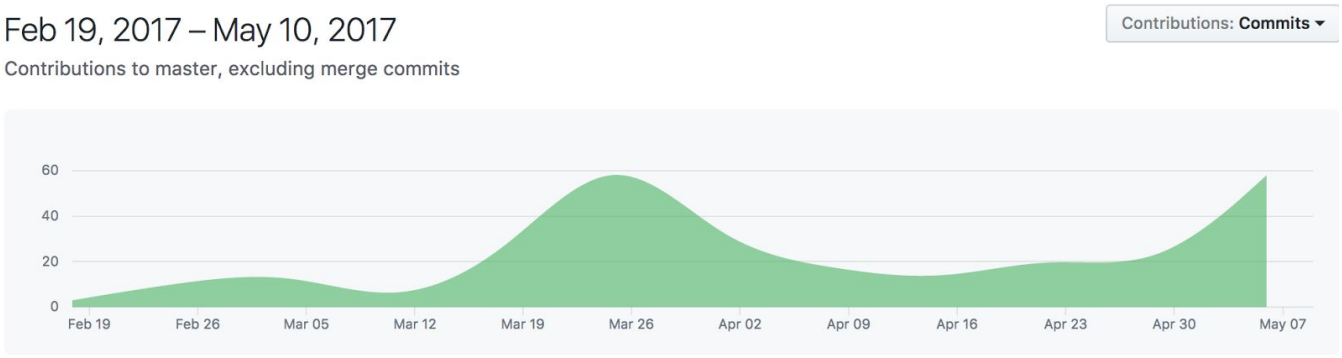
- Initial end-to-end parser, scanner, and ast
- Set up testing framework and make sure tests were comprehensive
- Struct scanner, parser, AST
- Job states and operators
- Strings with vector
- Keep meeting notes and try to be on schedule, though arguably was the least timely person of the group

4.5 Software development environment

Operating Systems: Mac OS Systems, Ubuntu 15.10 on Virtual Box
Languages: OCaml (used OPAM to install), C (runtime libraries)
Text Editor: Sublime, Vim
Version Control: Git, GitHub
Documentation: Google Docs

4.6 Project log

Commit overview:



Commit log:
Please see appendix, section 8. Our commit log is 34 pages long.

[6] Excudes overall awesomeness. Can't be denied!

5. Architectural Design

5.1 Block Diagrams

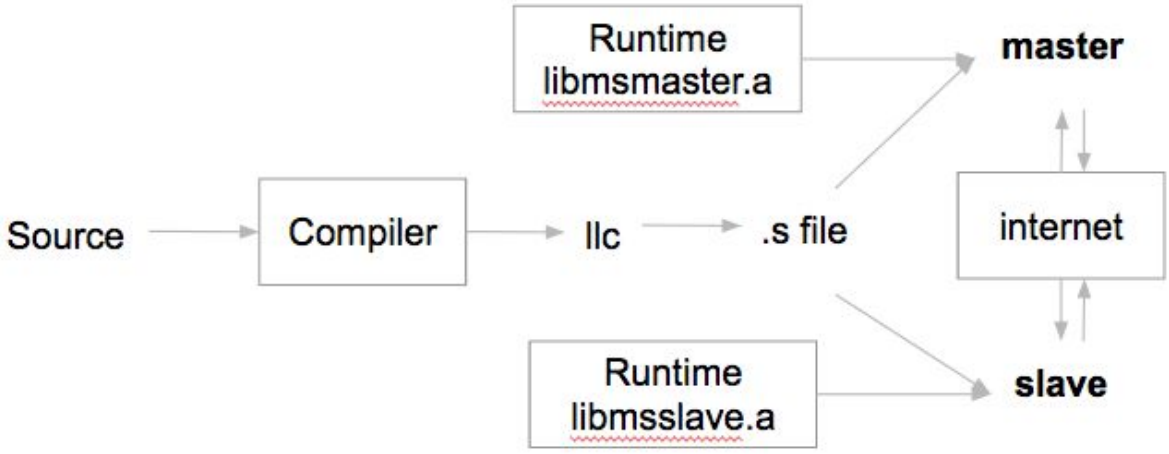


Diagram 1: Compiler-runtime interface

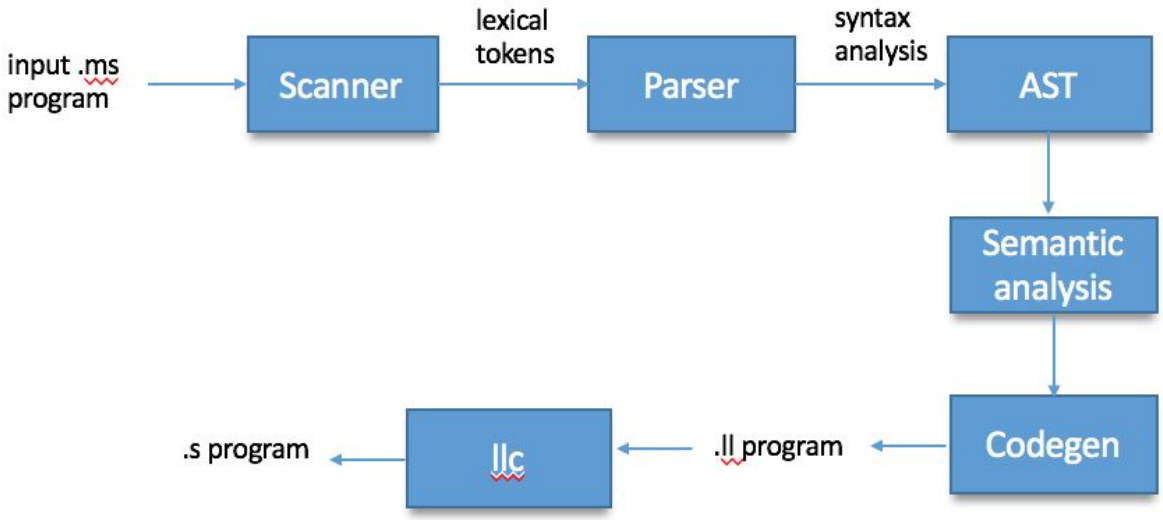


Diagram 2: Compiler components

5.2 Compiler components

The Scanner

Worked on by Mengdi and Miranda
The scanner takes as input a M/s source program and generates tokens for identifiers, keywords, operators, and values, as specified by the lexical conventions. Aside from that, its most complex task is processing escape sequences into single characters, and dealing with comments.

The Parser and AST

Worked on by Mengdi and Miranda
The parser takes in the generated tokens and generates an abstract syntax tree (AST) based on the M/s grammar. For most of the time, single tokens refer to single AST nodes, or none at all (in the case of punctuation). There are separate cases for function declaration, struct declaration, variable declaration. If the code can be successfully parsed, that means it is syntactically (although not necessarily semantically) correct.

The Semantic checker

Worked on by Mengdi and Miranda (mostly), and Ben (sometimes)
In semantic checking, the compiler verifies the validity of the AST. For every binary operator, the compiler checks that the two elements of the operation have the same type since M/s does not support type casting. A symbol table is used to verify that any identifier which gets called has previously been declared, is in scope and is of the correct type.

The Code generator

Worked on by Mengdi and Ben (mostly), and Miranda (sometimes)
The code generator traverses the AST tree to generate code, with each node returning the code necessary to evaluate the value for the handle. In addition, this stage also generates the code necessary to deal with memory management of vectors; keeping track of struct names; remote function calls; serializing and deserializing packets; linking with the runtime library.

5.3 Runtime Implementation

The runtime is written in C and compiles to two static libraries, libmsmaster.a and libmsslave.a. The compiled M/s code is linked against libmsmaster.a to produce a master binary, and against libmsslave.a to produce a slave binary.

Libmsmaster.a provides the following functionality:

- Provides a main function that calls the compiled M/s code's "master" function
- Maintains a table of currently active slaves. The table is a sorted vector of file

descriptors. Every time a slave connects, it is added to the table with its socket file descriptor. When a slave's connection is broken, it is removed from the list of slaves, and the job table (see below) is notified.

- Starts one global thread to accept new slaves, and two threads per slave, one to grab jobs from the job table and send them to the corresponding slave, and one to read return values from the socket and update the job table to reflect the completion of the job.
- Provides a global job table. The job table contains a vector (ordered by jid) of all the currently pending, running, finished, and failed jobs in the system. There is also a linked list over the elements in the vector of jobs that are awaiting assignment to a slave. When a job is started, it is enqueued at the end of the list. The writer threads for each slave pull jobs off the head of the list. The job arguments are maintained in the table until a return value is received, at which time they are freed and replaced by the return value.
- Exposes various function handles, which are called by compiled M/s code:
    - `int start_job(uint32_t ord, uint8_t *data, uint32_t len)`
        - This adds a job to the job table, and enqueues it to be sent to a slave.
    - `void reap_job(int jid, uint8_t **data, uint32_t *len)`
        - This waits for a job to be in a non-running state (finished, failed, or no longer in the job table) and then removes it from the job table, returning its return value to the user.
    - `int get_job_status(int jid)`
    - `void cancel_job(int jid)`
        - This removes the job from the job table, and adds a new job to the job table that represents a cancellation request. A cancellation request is sent to a specific slave on which the original job was running, and is enqueued at the head of the queue, in front of all other pending jobs. A cancellation request works by setting ordinal to the negation of the jid that should be cancelled. Thus, positive ordinals represent normal jobs, and zero or negative ordinals represent cancellation requests.
- Manages the restarting of jobs. A job is by default restartable up to two times. If a slave is disconnected, the job table is updated to set the job back into pending state. It is enqueued at the head of the queue, and is dispatched to a new slave. After two restarting attempts, a job is set to failed. The motivation is that if a job contains a bug that causes a slave to crash, it should not be able to bring down the entire network.

Libmsslave.a is much simpler. It contains a simple main that connects to a master server and receives job requests from the socket. For each job request, it creates a new thread to run that job. When a job is done, it writes its return value to the socket.

5.4 Network Protocol

Master and slaves communicate by sending job requests, and job completion records, over the network. Each of these communications has a 12-byte header, set up as follows:

job to run (ordinal)	int
unique job id (to identify return value)	int
length of data	int

The ordinal is a positive integer representing the job function to be run. The compiler creates a static table of function pointers, one for each job function. The ordinal corresponds to an offset within this table.

The arguments (from master to slave) or return value (slave to master) are serialized as follows, using the following inductive process:

- All arguments are serialized sequentially
- Structs are serialized by serializing each field sequentially
- Vectors are serialized by writing a 4-byte length (in elements, not bytes), and then by serializing each element sequentially.
- Primitives are serialized byte-for-byte

Note that all integer values are little-endian.

6. Test Plan

6.1 Sample programs, source and target

Please see the appendix, section 8. Each LLIR output file is over 1000 lines of code.

6.2 Test suites and automation

Runtime tests

Ben wrote tests for developing the runtime library. In `runtime/test`, we provide several tests (written in C) that verify the expected runtime functionality. Each test comes with separate master and slave C files, which will make use of the function handles exposed by the master and slave runtime libraries. To compile and link the tests, go to `runtime/test` and run `make TEST=N`, where `N` is the integer number of the test you want to run. These numbers are clearly given in the file names.

Compiler tests

On the compilers side, we adapted testall.sh to automate our integration tests. They key changes made were to have a separate Check() for tests that include remote calls, and tests that do not. For the tests that include remote calls, we start a master and a slave process as follows:

```
generatedfiles="$generatedfiles ${basename}.ll ${basename}.out" &&
Run "$MSCOMPILE" "<" $1 ">" "${basename}.ll" &&
Run "llc ${basename}.ll" &&
Run "gcc -L. ${basename}.s -lmsmaster -pthread -lm -o ${basename}-master" &&
Run "gcc -L. ${basename}.s -lmsslave -pthread -lm -o ${basename}-slave" &&
# Change port number if tests are freezing
Run './${basename}-master $PORT > ${basename}.out & PID=$! ; while [ -z "`netstat -an | grep $PORT`" ] ; do :
    ; done ; ./${basename}-slave $PORT ; wait $PID' &&
Compare ${basename}.out ${reffile}.out ${basename}.diff
```

For the tests that do not require remote, we simply just run the master process:

```
generatedfiles="$generatedfiles ${basename}.ll ${basename}.out" &&
Run "$MSCOMPILE" "<" $1 ">" "${basename}.ll" &&
Run "llc ${basename}.ll" &&
Run "gcc -L. ${basename}.s -lmsmaster -pthread -lm -o ${basename}-master" &&
Run "gcc -L. ${basename}.s -lmsslave -pthread -lm -o ${basename}-slave" &&
# Change port number if tests are freezing
Run "./${basename}-master $PORT > ${basename}.out" &&
Compare ${basename}.out ${reffile}.out ${basename}.diff
```

In terms of writing the tests, we each wrote tests for the features we implemented. Near end of the project, Miranda went back through semant and checked that all the semantic fail cases were accounted for in the fail tests, as well as each feature implemented had passing tests.

We adapted the initial suite of tests provided by Professor Edwards into M/s compatible code These are the key additional features we tested, with both passing and failing tests:

- Jobs: assignment, get, cancel, job states
- Vector: creation, pushback, access, assignment
- Structs: declaration, instantiation, field access, assignment
- Vectors in structs and structs in vectors
- Remote calls, memory freeing
- Primitives, doubles, strings

6.3 List of tests

When running ./testall.sh from the compilers directory (on a mac, hence the -n):

```
Miranda@dyn-160-39-11-3:~/Desktop/plt-ms/compiler(mimi-temp)$ ./testall.sh
test-add1...OK
test-arith1...OK
test-arith2...OK
test-arith3...OK
test-cancel...OK
test-decl-assign...OK
test-double1...OK
test-double-add...OK
test-fib...OK
test-for1...OK
test-for2...OK
test-func1...OK
test-func2...OK
test-func3...OK
test-func4...OK
test-func5...OK
test-func6...OK
test-func8...OK
test-gcd2...OK
test-gcd...OK
test-hello...OK
test-if1...OK
test-if2...OK
test-if3...OK
test-if4...OK
test-if5...OK
test-local1...OK
test-local2...OK
test-ops1...OK
test-ops2...OK
test-remote-doubles...OK
test-remote-int...OK
test-remote-job-get...OK
test-remote-job-states...OK
test-remote-many-ints...OK
test-remote-primes...OK
test-remote-struct-serialize...OK
test-remote-vector-serialize...OK
test-string1...OK
test-string2...OK
test-string-concat...OK
test-struct-field-copy...OK
test-struct-in-vector...OK
test-struct-nocopy...OK
test-struct-return...OK
test-var1...OK
test-vector-args...OK
test-vector-assign...OK
test-vector...OK
```

```
test-vector-struct-copy-assign...OK
test-vector-struct-copy-free...OK
test-while1...OK
test-while2...OK
fail-assign1...OK
fail-assign2...OK
fail-assign3...OK
fail-assign-double...OK
fail-assign-string1...OK
fail-assign-string...OK
fail-dead1...OK
fail-dead2...OK
fail-expr1...OK
fail-expr2...OK
fail-for1...OK
fail-for2...OK
fail-for3...OK
fail-for4...OK
fail-for5...OK
fail-func10...OK
fail-func11...OK
fail-func1...OK
fail-func2...OK
fail-func3...OK
fail-func4...OK
fail-func5...OK
fail-func6...OK
fail-func7...OK
fail-func8...OK
fail-func9...OK
fail-if1...OK
fail-if2...OK
fail-if3...OK
fail-job-cancel...OK
fail-job-get2...OK
fail-job-get...OK
fail-job-state1...OK
fail-job-state2...OK
fail-nomaster...OK
fail-remote1...OK
fail-remote...OK
fail-return1...OK
fail-return2...OK
fail-string-concat...OK
fail-struct1...OK
fail-struct2...OK
fail-struct3...OK
fail-struct4...OK
fail-struct5...OK
fail-struct6...OK
```

```
fail-vector2...OK
fail-vector3...OK
fail-vector4...OK
fail-vector...OK
fail-while1...OK
fail-while2...OK
```

7. Lessons Learned

7.1 Member reflections

Ben

Through the many days and nights - or whatever quantification of time exists in the world of M/s - I spent staring at mysterious pointer-arithmetic bugs, I learned not to overcomplicate things. No need to build a linked list inside a vector, where every deletion and reallocation requires adjusting the pointers, when using binary lookup with unique ids will do in place of raw pointers. No need to implement in LLIR what could reasonably be outsourced to the runtime library.

And, by the end of the project, I could use `List.fold_left` without looking at the documentation every time.

Mengdi

M/s' true meaning, through the perspective of one of its creators (aka me), should really be "mastering self-mutilation" because embarking on this project is really embarking on a journey to become the master of masochistic self-mutilation. But as Kelly Clarkson sings (***ahem* Professor Edwards, if you are reading this, this song is for you**),

What doesn't kill you make you stronger
Doesn't mean I'm lonely when I'm alone
What doesn't kill you makes a fighter
Footsteps even lighter
Doesn't mean I'm over cause you're gone
Thanks to you I got a new thing started
Thanks to you I'm not the broken hearted
Thanks to you I'm finally thinking about me
You know in the end the day you left was just my beginning
In the end
What doesn't kill you makes you stronger
Stand a little taller
Doesn't mean I'm lonely when I'm alone
What doesn't kill you makes a fighter
Footsteps even lighter

Doesn't mean I'm over cause you're gone
What doesn't kill you makes you stronger, stronger
Just me, myself and I
What doesn't kill you makes you stronger
Stand a little taller
Doesn't mean I'm lonely when I'm alone
What doesn't kill you makes you stronger, stronger
Just me, myself and I
What doesn't kill you makes you stronger
Stand a little taller
Doesn't mean I'm lonely when I'm alone

Miranda

This project was akin to an ascent up the ladder to the realm of forms. At first, I was a bit sad because ascending the staircase meant that I had to leave behind an old world of bodily attachments - Java and object oriented programming - in order to ascend to the heavenly perfection of OCaml and functional programming. But slowly, I grew wings, the wings of understanding the “perfect” method of programming - that is, programming without side effects - and the mental acuity of knowing how to navigate OCaml lists, maps, and let-in statements. These wings let me ascend even faster towards what I thought must be the ultimate good. I was excited not only to complete this project and finally graduate, but also to understand computer science as a mathematical discipline, learn what it means to be truly just and moral, and how to fix the depravity prevalent all around the world.

By the end of the project, I was soaring, because I felt like I understood, a bit more than before, what a compiler’s purpose is, how it works, and how it’s implemented; distributed systems and master-slave relationships; how to program functionally and how it can actually be useful; how to work with teammates and make sure they’re not mad at you; the importance of git pulling before rebasing; my own work strengths and weaknesses; the sentiments, motivations, and emotions that drive me, and drive all of us toward being the best people we can be.

And then, I slipped on a shift-reduce error in the ladder, and fell back to bodily earth. Oh well.

7.2 Advice to future groups

If you are here, then welcome…! Welcome..! Tributes, I welcome you. I salute your courage and your sacrifice… and may the odds be ever in your favor!

Let’s be honest. Who’s actually reading this? Probably not even Professor Edwards, or any of the TA’s. Nevertheless, we will add our hoarse shouts to the chasm of never-ending un-listened

to pieces of advice that are already out there, because we were told to, and as students, we are supposed to do what we're told (right? wait until the real world, kids). Keep in mind, we are giving you advice as people who never listened to any advice that was given to them. Nor will we actually follow the advice we're about to give you in our future endeavors. Basically, we're completely unqualified. So let's begin!

- Start "early". Pfffft. As if!
- Get to know OCaml early. Ha. Ha. Ha. Why would you learn something before you really needed to use it, last minute?
    - "Actually, I learned Haskell last year. And I quite liked it!" -Miranda, jaded functional "programmer" (?)
    - "OCaml is not bad. One of the better parts of the project" -Mengdi
- Use GitHub, or something.
- LLVM documentation isn't great. Just get over it, it gets better after a while. It really does!
    - "You are never alone in your suffering, and in your togetherness you will find the true meaning of the human spirit, and how it can prevail [over moe moe moe anime girls]..." -Miranda, wannabe philosopher and true instropect
- *Important* to have competent and helpful teammates and TA advisor
- Don't be too serious on yourself. This is a college class, and your performance doesn't mean you suck at programming, or are great at programming[7].
- In a world where freedom is history, brutality is law…. Oh wait. That's Planet of the Apes. A fitting analogy.
- If you are really struggling, like REALLY struggling to the point where you have to *read past groups' advice sections* of their final reports, you must be in a very dark place. I am really, really sorry for that. Sending you lots of sarcasm and a bit of love.

8. Appendix: That Unevolved Vestige That Gives You Appendicitis

Appended to the final report are:

1) Git commit log for the project
2) 2 sample .ms programs and their corresponding .ll files
    a) test-remote-vector-serialize.ms and corresponding .ll
    b) test-vector-struct-copy-free and corresponding .ll
3) Our source code
    a) Module authorship: Ben wrote the runtime library, and we all collaborated on the compiler side, writing tests for the features we created
4) Compiler and runtime library tests and output/error files

[7] Actually, maybe it does, just a little bit.

```
commit e99ca71c9fa2d3950e6a1b2fc0633add39b29dc1
Merge: e03c123 0b2274f
Author: mengdilin <mengdilin95@gmail.com>
Date:   Wed May 10 21:47:54 2017 -0400

    Merge pull request #70 from bwhanser/mimi-temp

    Updated readme + makefile

commit 0b2274fac04b04b419250b29d2d986a24b5ca8b4
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 21:42:52 2017 -0400

    hopefully final readme changes

commit e5a679ed417748b7a45254919bbb80d76d6f949c
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 21:30:42 2017 -0400

    small readme changes

commit edf587faf46029dbcc976e6e22dcc19069c8b526
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 20:58:44 2017 -0400

    updated makefile tarball to incorporate new test

commit 5578dd4dca04f03cdc8aabd04f08f9706a747106
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 19:51:38 2017 -0400

    updated tarfile portion of makefile

commit e03c1239e16b3ea72138dcfabfe0260fb603e0d9
Merge: 80eb33d 159c744
Author: Miranda Li <mirandali1995@gmail.com>
Date:   Wed May 10 20:47:39 2017 -0400

    Merge pull request #69 from bwhanser/demo-merge

    Prime demo

commit 159c7449a3c9faef7dc0a8997a048331ae2bd6cb
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed May 10 20:42:01 2017 -0400

    finalized prime tests

commit f47e9d5013667816ecada61acbd1806c9c17dd89
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed May 10 05:38:09 2017 -0400

    demo fix

commit 6d43d3df37dd12e62744e3722d83e836942daefb
Merge: d55188a 80eb33d
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed May 10 20:43:37 2017 -0400

    Merge branch 'master' of https://github.com/bwhanser/plt-ms

commit 80eb33d8a82bf5af0aabb5bdcfa157ea7a2759ca
Author: mengdilin <mengdilin95@gmail.com>
Date:   Wed May 10 19:37:49 2017 -0400

    Update README.md

commit d39dda8ab3fcd767ed7e05693145ec1eca9de2a6
Author: mengdilin <mengdilin95@gmail.com>
Date:   Wed May 10 19:37:11 2017 -0400
```

```
    Rename README.txt to README.md

commit aa38f5c076f0e41afbe0379458b068c084d23ff4
Author: mengdilin <mengdilin95@gmail.com>
Date:   Wed May 10 19:36:47 2017 -0400

    README -> markup format

commit fcae183152da2a7581d48af88cf23d3f51b3f6d5
Merge: 8635db5 23258b8
Author: mengdilin <mengdilin95@gmail.com>
Date:   Wed May 10 19:31:18 2017 -0400

    Merge pull request #68 from bwhanser/mimi-temp

    General cleanup + readme update

commit 23258b881ca3636fee5eebf28dc30e738d8d2265
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 19:21:19 2017 -0400

    deleted unneeded tests in compiler folder

commit 477c543f6924c87090ac504708b8b2ed7c0a19c5
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 18:08:10 2017 -0400

    readme update

commit cd59b0ae3f56e60136e6a21ca9e351e452a3bb6e
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 10:27:14 2017 -0400

    updated readme

commit 8431656400ab76b6afbd65e15403b3370370aa53
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 10:15:44 2017 -0400

    removed the vector struct and semant folders in test with garbage tests

commit 8635db56793495f4bc721e90f6394f4c49f762bd
Merge: a83d6fb 803f628
Author: mengdilin <mengdilin95@gmail.com>
Date:   Wed May 10 05:37:27 2017 -0400

    Merge pull request #67 from bwhanser/mimi-temp

    Fail-tests with semant changes

commit d55188a871c3852c580bbdfe5b54448c514fffe9
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed May 10 05:35:52 2017 -0400

    fixed demo

commit 803f628f8f272f8485b205b902212dbd5d95c9a8
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 05:16:54 2017 -0400

    merge in demo with changes

commit 1f4cda73e97b95d4e1a6ac58c8504081ae50f2a2
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 05:02:09 2017 -0400

    vector and remote fail tests

commit df8ccd8c46f89f6330386e24a5a65b4ef69f1931
```

```
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 03:45:17 2017 -0400

    remove local keyword

commit 8f67ba2d95be4e10038f56f7bbe63942da811570
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 03:28:39 2017 -0400

    undeclared struct caught in semant

commit f80ae2c35140f5c2d4834e5ef3a67f0c39fe3570
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 03:06:29 2017 -0400

    job states semant update and more fail tests

commit f270c63b267722cc047b34a89f494332deb6f193
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 02:14:35 2017 -0400

    struct fail tests

commit 7689ec0506092277e5cd2a2b50e19e9ee726ee86
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 01:35:50 2017 -0400

    string and double tests

commit a83d6fb8776290040db71002882c4e4cc7f91273
Merge: a895f00 17d7474
Author: Miranda Li <mirandali1995@gmail.com>
Date:   Wed May 10 05:00:56 2017 -0400

    Merge pull request #63 from bwhanser/semant

    Semant Rejects Vector decl inside for loop

commit 17d74746cfd85c5251cec83dadc5acf5e1720e42
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Wed May 10 08:48:43 2017 +0000

    remove submodule PLT

commit 0715b0d16a260c27532c0d1844aa98204e08a0fa
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Wed May 10 07:43:50 2017 +0000

    add jprime

commit a895f00eee2d1aeb08e74afa506e8471df64c374
Merge: c22a136 58f6085
Author: mengdilin <mengdilin95@gmail.com>
Date:   Wed May 10 03:34:24 2017 -0400

    Merge pull request #65 from bwhanser/ben-struct-func-arg-merged2

    Clean up edge cases involving function calls

commit fc49d5a4b2920b17d6b7e0346cd2d367dd599363
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Wed May 10 07:10:54 2017 +0000

    remove

commit 58f6085d6745fb0a3ee9c82e2dde5ef697c60154
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed May 10 03:09:49 2017 -0400

    Allow nested function calls
```

```
commit 4abbe4d14e3cbca1de771d270d3359b3dba9981e
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Wed May 10 07:09:26 2017 +0000

    remove binaries

commit e87e3d1bc698d2c9084ecae11b0e2c4a784b6ee3
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Wed May 10 07:08:39 2017 +0000

    update gitignore

commit ec8be7feb2320ba11a16d4d3c9a8a7a41e499700
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Wed May 10 07:06:25 2017 +0000

    remove test.ms

commit 56fba1ff469d373999661063c87adb55df7637e7
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Wed May 10 07:03:27 2017 +0000

    add semant rules to reject vector decls in for or while loop

commit ca3afc97dac9e50bd3f4120e3fa352fa3d128bba
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Wed May 10 02:25:26 2017 -0400

    test.ms nonworking string decl inside for loop

commit 5e8dcb622bca31dfa08cafe3110b00d6552f945a
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed May 10 02:17:33 2017 -0400

    avoid various memory leaks

commit c22a1368bdf207eed71ac5afefba5897c5afc388
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Wed May 10 06:13:52 2017 +0000

    segfault on vector get inside for loop

commit 3063b438f97e7cd451a2dce6e3f6a52838436f56
Merge: b95fcfb 810bbd0
Author: mengdilin <mengdilin95@gmail.com>
Date:   Wed May 10 01:22:05 2017 -0400

    Merge pull request #60 from bwhanser/ben-struct-func-arg-merged2

    Struct function arguments and serialization

commit 810bbd053074cfb73976257b615165969bd6fdbd
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed May 10 00:51:54 2017 -0400

    Struct as function argument

commit 18b167bc5dc794607b6e3d4b733cb3062f07b3b1
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed May 10 00:17:21 2017 -0400

    Finalized struct serialization fix

commit cdf6527fc235a534eef3bf8b3ee5bf90ecb63fe6
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Tue May 9 22:23:03 2017 +0000

    serialize compute_size fix
```

```
commit 4c011d94e506ac66caadaa709aacd91a5fde4fb7
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Tue May 9 17:54:41 2017 -0400

    still broken - something with struct op that copies vector

commit 2d15e41e8b735f2b9ccce1f44be804185d89c23f
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Tue May 9 15:51:26 2017 -0400

    partially working serialization of structs

commit b95fcfb15a5ef13abadf75713825ec398aae7950
Merge: 763e8df fc5bb5d
Author: bwhanser <bwhanser@users.noreply.github.com>
Date:   Wed May 10 00:59:51 2017 -0400

    Merge pull request #58 from bwhanser/mimi-temp

    Some small changes

commit fc5bb5d12dd0ce5c638d18a0ff83e745e4d02f58
Merge: 5181e63 65f4e73
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 00:39:10 2017 -0400

    unnecessary merge?

commit 5181e6383a20bf2d39da06f44ba9b66dcc9843b6
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 00:36:43 2017 -0400

    removed mod and leaving it up to user :)

commit bca8bda7d78be3ba8a677cc1855fc747b9d56e06
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 00:06:32 2017 -0400

    delete test_bak.c

commit 763e8df195e92bd5c5f9d33e4062f4dd9098b076
Merge: 9722fb3 430374b
Author: Miranda Li <mirandali1995@gmail.com>
Date:   Wed May 10 00:37:42 2017 -0400

    Merge pull request #59 from bwhanser/struct

    update test

commit 65f4e739e69c39a0cfdd42dd8dc4cd9084628717
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 00:36:43 2017 -0400

    removed mod and leaving it up to user :)

commit 430374bfa4203322cea42a21c8e20747d1a15fd6
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Wed May 10 04:31:02 2017 +0000

    update test

commit eadb5e99fa0f26db7ab101535a5dd4cd519b6e2e
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 10 00:06:32 2017 -0400

    delete test_bak.c

commit 9722fb3401f0f8e827b32350c8f5d60bb897a675
Merge: 2cf9a22 d844c89
Author: Miranda Li <mirandali1995@gmail.com>
```

```
Date:   Wed May 10 00:02:07 2017 -0400

    Merge pull request #57 from bwhanser/struct

    string concat and printf up

commit d844c898528fceceb175d3c4ea8892912501b88c
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Tue May 9 22:44:25 2017 +0000

    update test files

commit f98905bd9b02213c443b8b9d183f71ba834c9679
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Tue May 9 22:00:22 2017 +0000

    working concat

commit 997f33a6314f7dc8516627322da686bc8a983e38
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Tue May 9 21:05:38 2017 +0000

    working concat. memleak

commit 1d5ea5aed9456193ba658cb933f2c5367bc0dcdb
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Tue May 9 20:49:49 2017 +0000

    concat works w/o assignment

commit 2cf9a227eda1a064be7ff290cdf9e1a09f872651
Merge: dfed046 b83f5ef
Author: mengdilin <mengdilin95@gmail.com>
Date:   Tue May 9 15:49:30 2017 -0400

    Merge pull request #56 from bwhanser/ben-misc-fixes

    Bugfixes

commit b83f5efb62d9611be902c421fb9341591f824eae
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Tue May 9 12:57:47 2017 -0400

    fix vector assign of struct bug

commit fb14a6eafb111d659900a4fe6afc3e37f9089438
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Tue May 9 12:10:56 2017 -0400

    housekeeping

commit dfed0465de36d92ac917f17e483ae64dce3e5457
Merge: f4bbd86 c79a48b
Author: bwhanser <bwhanser@users.noreply.github.com>
Date:   Tue May 9 11:23:02 2017 -0400

    Merge pull request #55 from bwhanser/struct

    Struct Copy and Deletion

commit c79a48b9fc705dfd9019b96d2303229a9cc5ea82
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Tue May 9 07:00:28 2017 +0000

    add test file

commit aecc0db79a1efb99835d2458c08697f0fd0d501e
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Tue May 9 06:31:28 2017 +0000
```

```
    finish struct copy and delete

commit 496d7104c0d863cf5013d05eef606fde86dde946
Merge: 349b817 2ef6b5a
Author: mengdilin <mengdilin95@gmail.com>
Date:   Tue May 9 02:23:49 2017 -0400

    Merge pull request #54 from bwhanser/ben-struct-fix

    Fix struct issue

commit 2ef6b5a0ee51e6544030b31ffc98045b12d9ef01
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Tue May 9 00:18:23 2017 -0400

    Fix pushback problem

commit 423f7db4280ecd7f1001eb522ff5a839495d7755
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Mon May 8 23:40:54 2017 -0400

    moved struct_copy

commit 349b817aa5e8fab290d05e5f79e549c307e51ea3
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon May 8 08:35:23 2017 +0000

    working vector<struct> copy. memleak

commit 0bf93e4427c729f79761e9d19eb84009aa287c54
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon May 8 06:15:27 2017 +0000

    nonworking vector<struct> copy assignment

commit 78d7c3f26619fc0a8019fd177e945958a01fa0c7
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon May 8 04:10:57 2017 +0000

    fix vector<struct> access bug

commit c23d7e2ed482c0f143f474787048b7d7f666622d
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon May 8 03:22:28 2017 +0000

    working basic struct's vector delete

commit 15a19921403e13898d47ebbe3a759c688025fc4e
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon May 8 01:41:52 2017 +0000

    update test-vector-struct-copy-assign

commit 0e8458c5c5daa386aa156bd018774411d65f3142
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon May 8 01:36:43 2017 +0000

    add test

commit f6e64b5e798e20d0e91c0e973afc56760989c5d9
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon May 8 01:35:58 2017 +0000

    working basic vector<struct> copy assignment

commit 1e754ded8520670c6f43d39a91019e659c8c2a12
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon May 8 00:29:09 2017 +0000

    copy assignment of struct
```

```
commit 4dd2f328802c676871e991423d8577129f4ac72c
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun May 7 23:20:09 2017 +0000

    broken struct's vector assignment

commit 0488010be04dac0f1c09effd59b590918491f727
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun May 7 22:12:17 2017 +0000

    nonworking struct copy assign

commit a583f8d2b39cdd51e42a1bcef390d4cd5e9bef23
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun May 7 21:30:01 2017 +0000

    working field assign with vector

commit 6f2d6911f56c3a19b4afdaf6fd752a5b8fae36c0
Merge: 8763e81 f4bbd86
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon May 8 00:22:26 2017 +0000

    merged struct vector fix

commit f4bbd861db7b506fb2969eefebe6cf323550a771
Merge: e7ea75f 16e098d
Author: mengdilin <mengdilin95@gmail.com>
Date:   Sun May 7 20:20:20 2017 -0400

    Merge pull request #52 from bwhanser/ben-fix-vector-in-struct-bug

    Fix vector-in-struct bug

commit 16e098dbe34503e4e99e8cb6d9361a72f21f2cbb
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun May 7 20:19:11 2017 -0400

    Fix vector-in-struct bug

commit 8763e8133f51846e09f1da47fcd3896fd9cea9de
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun May 7 21:16:55 2017 +0000

    working struct field copy assign

commit e7ea75f1b15928f4e0e6eb0271478cd6234f5e09
Merge: c7bff07 2416c50
Author: mengdilin <mengdilin95@gmail.com>
Date:   Sun May 7 14:17:37 2017 -0400

    Merge pull request #50 from bwhanser/ben-vec-serialization

    Vector serialization

commit c7bff07f096c938ced7270feff21858d2889bb51
Merge: 48ce6dd f1264f4
Author: mengdilin <mengdilin95@gmail.com>
Date:   Sun May 7 14:16:40 2017 -0400

    Merge pull request #49 from bwhanser/ben-vector-args

    Allow vectors as arguments to non-remote function calls

commit 2416c5049e27dd50086454ff8e4c50f598ffb5bb
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Thu May 4 18:46:09 2017 -0400

    Improved vec serialization test; renamed file to work with testall.sh
```

```
commit fef1e968ef5e5f1669d5707fc63a153599e99f5f
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Thu May 4 18:15:29 2017 -0400

    Vector serialization/deserialization now works in both directionS

commit fe89955d7e81aaa17463c0b09a14846959ecfd9e
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Thu May 4 04:37:45 2017 -0400

    Serialization now works!  Also fixed bug with job table

commit d350e4c0e0f302202e27b880c71d258fefd6f395
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Thu May 4 03:13:33 2017 -0400

    Serialization now working in simple cases but broken in others

commit 828b4781788ee1963600e02cef67ba39dad49cc1
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Thu May 4 02:16:06 2017 -0400

    Vector serialization milestone: unbroke existing code, but serialization still broken

commit f1264f49b4c263384b1adf7c5e3c775d825444cd
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed May 3 20:43:53 2017 -0400

    combined declaration and assignment finally works!

commit 884d705c7fadfb02c43c4b6ad8a186a9678a7a8e
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed May 3 20:20:13 2017 -0400

    Vector as argument to local function

commit 48ce6dd2416f0dd073f5be5e3d33d1632467a954
Merge: 1125560 427f27e
Author: bwhanser <bwhanser@users.noreply.github.com>
Date:   Wed May 3 13:28:10 2017 -0400

    Merge pull request #48 from bwhanser/mimi-string

    moved string tests into tests folder

commit 427f27e90b5059d511fff01c9f5429698461a200
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 3 11:29:33 2017 -0400

    moved string tests into tests folder

commit 11255608eab67236dc123ff5d2af6604a6bdd51d
Merge: 1adefe7 c2f100c
Author: Miranda Li <mirandali1995@gmail.com>
Date:   Wed May 3 11:21:59 2017 -0400

    Merge pull request #47 from bwhanser/ben-cleanup-merged

    Cleanup

commit c2f100cea4ad0bfbb203d527415927ea779ceae0
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed May 3 03:24:32 2017 -0400

    fix minor bug in string codegen

commit 305e3b402130e4d6b9d79152ae72b9205481d31f
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed May 3 03:13:20 2017 -0400
```

```
    Fixed compile warnings and de-hacked tester

commit 02fb0f5e4cfbd80e6d585f81a5bade0bfd357af9
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed May 3 00:10:07 2017 -0400

    Fixed parser and cleaned up errors

commit 1adefe7fb715ae4b4797f0411b5f787f8b92c8ea
Merge: d162dd9 107e538
Author: bwhanser <bwhanser@users.noreply.github.com>
Date:   Wed May 3 03:18:39 2017 -0400

    Merge pull request #46 from bwhanser/mimi-string

    Strings using vector

commit 107e538e8b730a64e1ba8423d735d8019a8bd380
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 3 01:32:57 2017 -0400

    small changes to string codegen and test

commit 66f0ab764582678d1eea1a9cda26d26eab71f7ec
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Wed May 3 01:23:45 2017 -0400

    String using vector

commit 098754bfe646909e32e8df7b96231667a900f0ee
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Tue May 2 21:06:39 2017 -0400

    small change in slave runtime

commit ef3c62e31d5112379a24e7190f4a2567dcda3bc2
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 23 22:38:26 2017 -0400

    parsing string literals and string types

commit 41b19ed4c8e2ba67556424a3b294987f612e309b
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Mon Apr 24 01:12:52 2017 -0400

    string tests, test 2 is broken

commit 42502767af30c626faf22dd9b7636e5a6af6532c
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Mon Apr 24 01:11:59 2017 -0400

    string codegen as pointer type

commit a8e13faa326741b948b91963f45430b4d83d47a1
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 23 22:38:26 2017 -0400

    parsing string literals and string types

commit 60457b80100786a6d79f5645d93f92eb25ab9d6d
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 23 22:24:23 2017 -0400

    delete those DYSM folders whatever they are

commit d162dd90f05f864a3bf7f109dc8d518af30c5910
Merge: e36b913 ac91152
Author: Miranda Li <mirandali1995@gmail.com>
Date:   Tue May 2 21:22:36 2017 -0400
```

```
    Merge pull request #45 from bwhanser/fix-shift-reduce

    get rid of shift/reduce conflicts FINALLY

commit ac911521a449d918253430d1f6a4a43f4e2bc36c
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Tue May 2 21:18:18 2017 -0400

    get rid of shift/reduce conflicts FINALLY

commit e36b913be83f63edafaddf3d8f53e66ccc1d0968
Merge: 5da1699 9522f40
Author: mengdilin <mengdilin95@gmail.com>
Date:   Mon Apr 24 01:50:13 2017 -0400

    Merge pull request #44 from bwhanser/ben-struct-in-vector-merged

    Misc struct/vector bugfixes

commit 9522f4086a1fbf43bce18675b28a2f424069fe78
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Mon Apr 24 01:04:22 2017 -0400

    garbage collection of first-level vects in a struct

commit 1de680667040a243d8c09b242fb18bba8b218099
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Mon Apr 24 00:27:19 2017 -0400

    copy into the vector of a struct

commit 6c6e31055d2092f0e6ffe0874bdf7faf9411a516
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Apr 23 23:56:37 2017 -0400

    added failing test for assignment to a struct's vector

commit 1647a7c22c1e12d0ab0d72de1267f7545b25843e
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Apr 23 23:46:55 2017 -0400

    added test case for deleting vector when overwriting with return val

commit 5da16995dd3ba42e83fa52406fb4701074d6b9b4
Merge: 21e38c5 cd4b687
Author: mengdilin <mengdilin95@gmail.com>
Date:   Sun Apr 23 23:41:59 2017 -0400

    Merge pull request #43 from bwhanser/ben-struct-in-vector-merged

    Misc. struct/vector improvements

commit 5bcd9edba630aea069bfecf7ba52b67a30afe059
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Apr 23 23:40:51 2017 -0400

    vector assignment bugfixes

commit cd4b6875b1b9f425e53c911bf793526bbabd7dec
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Apr 23 23:22:10 2017 -0400

    fix miranda's job cancel to work with new expr

commit 819e3517162ee4ba8aaed677fc3d2935619d5c77
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Apr 23 23:16:19 2017 -0400

    vector assignment to and get size of partially dereferenced vector
```

```
commit 02458cee1ec6eb960164b1964a1e21b78a37410d
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Apr 23 21:06:30 2017 -0400

    vector assignment now works from struct

commit 1a8f4810e3e1515611d8fd6f58863767eb7c5229
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Apr 23 20:55:35 2017 -0400

    vector assignment now does deep copy

commit fcd1b59b2245dcbd8f252446e3dda76785e26f2b
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Apr 23 19:04:47 2017 -0400

    expr now returns a builder

commit 1ef5710f26407fc5f2d84316536347bb568585b1
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Apr 23 17:58:19 2017 -0400

    vector in struct partially working

commit 63dc893d846376fe612aecdb9b02aed27908b2d7
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Apr 23 17:39:37 2017 -0400

    basic struct inside vector now works

commit 6a205f7f17500c6cc857e7ee0c3a0edffd53b5f7
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Apr 23 16:59:03 2017 -0400

    broken struct in vector codes

commit 21e38c59e3a81587770e74020494a1134dafe52b
Merge: 2e40ef7 cf4fae4
Author: bwhanser <bwhanser@users.noreply.github.com>
Date:   Sun Apr 23 23:18:09 2017 -0400

    Merge pull request #42 from bwhanser/mimi-job-states

    Canceling jobs

commit cf4fae4d42a037a13268465e0a48988d665f13fa
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 23 22:32:25 2017 -0400

    modified test-remote-cancel

commit bc070fceb97485552de89fe0c8f9fefbe8d4f230
Merge: 14f8ca4 84b6811
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 23 21:27:58 2017 -0400

    merge

commit 14f8ca40860a21a29390812bf0c01156382d51f1
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 23 21:26:40 2017 -0400

    framework for test for cancel

commit 594c7c1c76f51a8f63317b92524fe2262509220b
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 23 21:26:10 2017 -0400

    canceling jobs on compiler side
```

```
commit 84b681107c440ca5993f3c82e3ea8e59d60e5122
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 23 21:26:40 2017 -0400

    framework for test for cancel

commit f3c950842f5f8d2285d88f15e5fe66ca5143c79d
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 23 21:26:10 2017 -0400

    canceling jobs on compiler side

commit 2e40ef72d43732f8359a8a895342daf3fe0c3b84
Merge: 62daf9f 3c85b85
Author: bwhanser <bwhanser@users.noreply.github.com>
Date:   Sun Apr 23 16:27:55 2017 -0400

    Merge pull request #41 from bwhanser/struct

    Struct works independent of vector

commit 3c85b85f5077e4e813014663f68bf820a483aea3
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon Apr 17 00:14:56 2017 -0400

    merge conflict

commit 8e3180629029c69675f7e5461018ac46251b3281
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Apr 16 23:55:36 2017 -0400

    working struct with nested structs

commit eb584ad74dcab91962313fdfbaffc33f4d75b8da
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Apr 16 20:47:51 2017 -0400

    working struct assignment

commit e100990eb370e8ea1bf4a7b7ff5af864dfa9c4ce
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Apr 16 19:03:13 2017 -0400

    non-working recursive struct 1

commit 8d37fa39df2a45c20d02bcf422361eca0f3a342a
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Apr 16 17:43:56 2017 -0400

    non-working recursive struct

commit 5c5b230dedcc6c2a349c9fc90d3dab7e53014395
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Apr 16 16:53:07 2017 -0400

    add struct type in semant

commit adfa8907c26f763c9adf5b0548bef228b1549371
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Apr 16 15:55:35 2017 -0400

    struct field access and assign works for int

commit 1c52a25dc75281cd1c525b4e4e7466ff42d2022e
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Apr 16 15:20:13 2017 -0400

    working struct decl for primitives
```

```
commit 62daf9f61152beaa150edf7ae168f66565ef9825
Merge: 3cc2712 96bab19
Author: bwhanser <bwhanser@users.noreply.github.com>
Date:   Sun Apr 16 23:45:14 2017 -0400

    Merge pull request #40 from bwhanser/mimi-job-states

    Job states

commit 96bab19161b0cf4e4833277e77412db880b464a5
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 16 23:38:59 2017 -0400

    changed existing tests to work with printb

commit 07d47c45271d94b1ebd9dd795dd0a930e54f87da
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 16 23:38:37 2017 -0400

    job states codegen and tests

commit 047cfbf33a42203245edda2c85739503a5cff00c
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 16 21:18:02 2017 -0400

    parsing job states

commit 3cc2712cc89cb475da663f27346839c8914ba26c
Merge: 84f2367 7a7f7f2
Author: mengdilin <mengdilin95@gmail.com>
Date:   Sun Apr 16 13:53:43 2017 -0400

    Merge pull request #39 from bwhanser/ben-vector-copy

    Dynamic vector; garbage collection; copy on pushback

commit 7a7f7f2480539928b68e5028ef104d04ab1a9509
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Thu Apr 13 23:28:01 2017 -0400

    fixed bug in type computation for partially dereferenced vector return

commit 28f08b4f93eb9c5d99c77bd1c488cd3590a3c785
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Thu Apr 13 17:42:55 2017 -0400

    functions may now return a vector

commit 9206fc3521522d233696678d0764ef33dbc94d12
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Thu Apr 13 03:02:38 2017 -0400

    cleaned up tests

commit 35cafcf21f9a978dfa90b2c36f8e12d2804d56cf
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Thu Apr 13 02:40:11 2017 -0400

    vectors are now automatically garbage collected at the end of each function

commit fe20b781dec6d760faf8d5053db5f76cd2e8f020
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed Apr 12 21:26:01 2017 -0400

    cleaned up files

commit 47af6f3ed3c6db514236914f05dcf22f8fbd27e3
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed Apr 12 18:57:43 2017 -0400
```

```
    fixed vector copy bug where malloced for size rather than len

commit b232992eec840c45bfd1a5927f7dcd9ff4991eca
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed Apr 12 18:40:32 2017 -0400

    vector copy on pushback now seems to work

commit 5adb0c55e0e7872ebcb46bcc5e5d973ad1adc8ba
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed Apr 12 01:47:50 2017 -0400

    vector realloc; vector type checking; codegen type computations; push_back to partial
ly dereferenced vector, eg v[0]::val

commit c2c6f997a738cfec26681df6af619eec1869d41b
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Tue Apr 11 10:06:48 2017 -0400

    rewritten vector now working except for dynamic resize

commit 3d85b39e8a94a6195059b738b1898111caff49f4
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon Apr 10 05:32:36 2017 +0000

    struct sample ll files

commit 27a6a6b72d2cc2ac5c515091e33723afe4be2baf
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon Apr 10 03:42:44 2017 +0000

    non-working dynamic size vector

commit 963f0a9522e4e37e57a62f519d77f8915b6c945f
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon Apr 10 03:00:04 2017 +0000

    fix: extra allocation bug

commit 02ae1622f079b41cae10efaeca8ba276ddcd32fe
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon Apr 10 02:12:25 2017 +0000

    broken vector resizing in for loop

commit dcddccbcfc47c9128ae7160b77d94e4f6465b6c4
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon Apr 10 01:48:48 2017 +0000

    dynamic vector resizing works for 2d vector

commit 5c44e6ebbfb0efdb14e59bd098ec2b4b2c23dd89
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Apr 9 22:50:35 2017 +0000

    working 2d vector pushback and size

commit 84f2367eb2c0f3dbd4e655bb7cb1fed071a750e5
Merge: 883906f 58bfc51
Author: Miranda Li <mirandali1995@gmail.com>
Date:   Wed Apr 12 23:32:40 2017 -0400

    Merge pull request #38 from bwhanser/ben-fix-ret

    fix auto-generated return bug for version 3.7.1

commit 58bfc5191f162f9cdf91971b94ea05b792c14072
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed Apr 12 23:30:37 2017 -0400
```

```
    fix auto-generated return bug for version 3.7.1

commit 883906f5f52a984343b06e10118ce2b5c61d4c74
Merge: 9b56e13 79b9cc5
Author: Miranda Li <mirandali1995@gmail.com>
Date:   Tue Apr 11 22:47:51 2017 -0400

    Merge pull request #36 from bwhanser/ben-get

    Job<typ> syntax; get now works with primitive types

commit 79b9cc56d3289255de6546d28c5b13c01b427849
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Mon Apr 10 12:57:39 2017 -0400

    Job<typ> syntax; get now works with primitive types

commit 9b56e138971ef42607369eb717e025d1b57549f9
Merge: cd0c9ac cec6aa6
Author: Miranda Li <mirandali1995@gmail.com>
Date:   Wed Apr 5 16:05:41 2017 -0400

    Merge pull request #35 from bwhanser/ben

    fixed double serialization and job table bug; added double test case

commit cec6aa6ce44dc2efdc3be8735c22c2b5cd591ceb
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed Apr 5 15:34:25 2017 -0400

    fixed double serialization and job table bug; added double test case

commit cd0c9ac57edecd530986278d732ee54f8d2caee3
Merge: 9d99cdf 3d57178
Author: mengdilin <mengdilin95@gmail.com>
Date:   Wed Apr 5 12:13:42 2017 -0400

    Merge pull request #34 from bwhanser/mimi

    Double end to end, global make, and small changes to tests

commit 3d57178a1df74305ce2b315fc2c6e38a95c5a319
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Tue Apr 4 00:28:35 2017 -0400

    global makefile

commit 89c0fae082138ec04ca7edb84557ded6294a2762
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Tue Apr 4 00:16:09 2017 -0400

    test double add

commit 41ebef142f0cd3bcd6937c9e20c1b1872fefbd98
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Mon Apr 3 21:28:38 2017 -0400

    double arithmetic in codegen

commit 0c266dc5a0409f50dd1c36ebd7a0c84efff07249
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Mon Apr 3 21:07:10 2017 -0400

    added test remote many ints to tests folder

commit 020ce1060d481f70cc2ac1ff6fab396ee8d6e1e9
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Mon Apr 3 21:02:08 2017 -0400

    removed testfloat tests:
```

```
commit 9847e1c0ef26ae74d1d4f02ae0449748799f0edd
Merge: 88d3461 204b257
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Mon Apr 3 21:00:26 2017 -0400

    merge

commit 88d34610662b539373c90105b997597cb91a2054
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Mon Apr 3 20:54:57 2017 -0400

    changed float into double

commit a970d0d0ed63ac3e7d619c63d96205048ed18c58
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Mon Apr 3 20:36:24 2017 -0400

    stuff

commit 4e8bd69145d0f9132bea54d6e31713c82ff9c6a8
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Mon Apr 3 20:31:51 2017 -0400

    print float works with printf

commit 6673ec3bb740fa74208da274b9491e62a209277b
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 2 23:30:34 2017 -0400

    skeleton global makefile

commit 0dc6449e1c9b88c6ad878c4a0764ae02443733c6
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 2 22:12:02 2017 -0400

    trying to print float

commit 9cb56e598fea6dce648a84f198d1671ddfb0a311
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 2 21:54:51 2017 -0400

    float checking

commit 006edd394850e523dcd094a2867ed6ee067b148c
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 2 21:11:06 2017 -0400

    print float using printf, test floats

commit e785a6924b7fee00ee1970b0fe94f38b0717fb1f
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 2 19:57:58 2017 -0400

    added float to semant

commit ea6dfe1840a7283dbd77ae9caac8fe2ca093aad4
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 2 19:37:10 2017 -0400

    making semant a bit more readable

commit 258e1748af69b2a258123805349eb4713e06950d
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 2 19:36:59 2017 -0400

    floats in parser and ast

commit 9d99cdf191c56eaa960ad84fd165491e4fc79abb
Merge: e3bf0f9 8c6de5f
```

```
Author: bwhanser <bwhanser@users.noreply.github.com>
Date:   Sun Apr 2 23:29:02 2017 -0400

    Merge pull request #33 from bwhanser/ben-runtime-debug

    Ben Fixes Runtime and Codegen Bug

commit 8c6de5fe50f26f63bbd410a67c070f291fa25988
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Apr 2 23:25:30 2017 -0400

    fixed job ordinal calculation regression

commit 43e0ccdf1f44ee2d9642bec028d247c6daa7097a
Merge: e3bf0f9 e9afd13
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Apr 2 22:55:36 2017 -0400

    resolve codegen conflict

commit 204b25710438f25d3ade6d39e8fda6c8b6f33f70
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 2 22:12:02 2017 -0400

    trying to print float

commit 8ef89b883baeee157271b3f5ab9b646b5ec387ac
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 2 21:54:51 2017 -0400

    float checking

commit e9afd13c384ecdbf5185b653109ae49ae3ee51bd
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Apr 2 21:53:21 2017 -0400

    added job tester

commit f3944f2b2d376b7f5ec66defdc7f35d788db1ee7
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 2 21:11:06 2017 -0400

    print float using printf, test floats

commit 4315d8af07539ad225868c478a4000c1fd91f2bb
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 2 19:57:58 2017 -0400

    added float to semant

commit 9d73f2538aa692997da4d6034cbe01b1e82d7b02
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 2 19:37:10 2017 -0400

    making semant a bit more readable

commit 2bdf439bc8b267cf014627a3d15718584537f905
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Apr 2 19:36:59 2017 -0400

    floats in parser and ast

commit e3bf0f9ab2aff91be8df5b6143fd87e5b1e8b5f5
Merge: 92b7cb8 f757949
Author: mengdilin <mengdilin95@gmail.com>
Date:   Sun Apr 2 21:01:20 2017 -0400

    Merge pull request #30 from bwhanser/mimi

    Integration testing framework setup
```

```
commit 4024721ea97731107ab7844d01b64a1c8a49a6cb
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Apr 2 20:52:34 2017 -0400

    some garbage

commit f757949074e4a883872ad1b19b215c4976e5b97a
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Apr 1 21:52:12 2017 -0400

    moving mengdi's file into the mscompile folder from microc

commit 6501c2d424d263d4ef96351e171389a291e74316
Merge: b88a88a fd1cc09
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Apr 1 21:48:15 2017 -0400

    merge

commit b88a88a01d0aeff8dda89a25c7739674d9a93378
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Apr 1 16:42:09 2017 -0400

    different case for remote tests

commit 666eafa89992a2729f78e8c6d9fc1e8e31e0c54d
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Apr 1 16:07:51 2017 -0400

    Floats in parser and scanner working

commit 3faf9579fcb4f56f6836366c639212ec2c5d33b5
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Apr 1 13:02:30 2017 -0400

    temporarily removing test many ints from tests folder

commit 092ebf6c358371e50b951a1ff1288f86109f9095
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Apr 1 12:59:12 2017 -0400

    deleted unused test for struct

commit 971d1609d7dd5d730ade4f103f2f623a8968488c
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Apr 1 12:58:48 2017 -0400

    tests for remote int and remote many ints

commit 462f66fa076ef795ecc6cceee5f87c87046bae59
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Thu Mar 30 19:23:45 2017 -0400

    copied remoteint test and output into tests folder

commit 3f04c5e6a785a61fcd71f09fa467a1749a3c8a5a
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Thu Mar 30 19:23:18 2017 -0400

    updated Makefile for new testing framework

commit e4ee6d8b508c7bb88bac92d98c0eebfb77fe9818
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Thu Mar 30 19:22:31 2017 -0400

    changed existing tests to ms tests, deleted irrelevant tests

commit c0c0c7183fa0d1f26afa16e482e6f90f44c394f4
Author: Miranda Li <mjl2206@columbia.edu>
```

```
Date:   Thu Mar 30 19:01:33 2017 -0400

    changed testall.sh to run master and slave binaries

commit 63c77916805d19ad85062f23f50475174cec03a2
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Thu Mar 30 14:56:14 2017 -0400

    renamed project from microc to mscompile

commit 38725d8adc08cb95ef5cdb796475a288ba2222a1
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Thu Mar 30 11:33:05 2017 -0400

    makefile update

commit 46f08e6912e3f726c5290d0b0aaa544f00f06739
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Thu Mar 30 11:31:43 2017 -0400

    merge makefile changes

commit 92b7cb8e96664c295214e082ac33a480a0260b76
Merge: 0e4cba2 0b2d1f3
Author: Miranda Li <mirandali1995@gmail.com>
Date:   Sat Apr 1 21:46:44 2017 -0400

    Merge pull request #31 from bwhanser/mengdi

    Simple Vector Access and Assign Works

commit 0b2d1f34bc95e3abfa304c4109b03936a28cacf0
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Apr 2 00:40:08 2017 +0000

    working SIZE and PUSH_BACK op

commit 057d9c1666cd71529e1f038a308377082e23dd4e
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sat Apr 1 22:45:38 2017 +0000

    working 2d vector for vector assign and vector access. see testfile for example

commit 9bab59a3cf77201b5f165d92d6d95ca3bc7e4c0c
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sat Apr 1 22:23:45 2017 +0000

    working 2d vector that allows 2d access but does not support 2d assign b[0][0]=1

commit 24827b5a642cba64d13ac6259e1f254c0b8d5395
Merge: f8a5766 a3919fc
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sat Apr 1 21:30:39 2017 +0000

    fix merge conflict

commit f8a5766df917cb900c3b570255f31db7e7def63c
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sat Apr 1 21:29:37 2017 +0000

    working 1d vector for single element access and assign

commit dceca727728f5de0769778e75c0cdab8435adb73
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sat Apr 1 19:34:00 2017 +0000

    nonworking vector

commit fd1cc093838edcbbbb8d77150a3134ceb6274b9a
Author: Miranda Li <mjl2206@columbia.edu>
```

```
Date:   Sat Apr 1 16:42:09 2017 -0400

    different case for remote tests

commit fad43704bcec76136452a45a2a6baffc2dc1d8a2
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Apr 1 16:07:51 2017 -0400

    Floats in parser and scanner working

commit c8d13fd2c3229773da2e76d78aa023b3121b55ff
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Thu Mar 30 04:49:11 2017 +0000

    added more comments to codegen

commit 295f6fdc7cb58f22b3c51ce7dd3d5affb9978cc3
Author: vagrant <vagrant@vagrant.vm>
Date:   Mon Mar 27 19:23:54 2017 +0000

    add comments to codegen

commit c88eecfc7bca75bea9792acad23f7d6c0d11412e
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Apr 1 13:02:30 2017 -0400

    temporarily removing test many ints from tests folder

commit 55bba1cec0fd0ea25181ee5b575fda13f1cb9ce6
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Apr 1 12:59:12 2017 -0400

    deleted unused test for struct

commit ff3aa3751b182233c617e8cd79e92fe82b7b0120
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Apr 1 12:58:48 2017 -0400

    tests for remote int and remote many ints

commit 65f0286291d0c1cc60b1dc6f6c4c6e241b642873
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Thu Mar 30 19:23:45 2017 -0400

    copied remoteint test and output into tests folder

commit 1222a7ca65a1d7bbf4afd107ac1b0e1f9bfd8a03
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Thu Mar 30 19:23:18 2017 -0400

    updated Makefile for new testing framework

commit 4dd74645cdfbe2f2ff8c1c46d0607909da2c4c6c
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Thu Mar 30 19:22:31 2017 -0400

    changed existing tests to ms tests, deleted irrelevant tests

commit fa87b1d5a573d655a129191f2de11bb6f14a01ab
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Thu Mar 30 19:01:33 2017 -0400

    changed testall.sh to run master and slave binaries

commit a2447fd8d8ca9c1c3d5ef0d9094d61dbede412dc
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Thu Mar 30 14:56:14 2017 -0400

    renamed project from microc to mscompile
```

```
commit 6b23fae905b8f3fe6c486b5ce7a0d07a54045cba
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Thu Mar 30 11:33:05 2017 -0400

    makefile update

commit 4d42fa6e0398e773d4df2b5574975ae0d8b20f66
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Thu Mar 30 11:31:43 2017 -0400

    merge makefile changes

commit 0e4cba23addd83eb0ec808a72ff3cc8152e5afa2
Merge: 7d323e4 74d69e0
Author: mengdilin <mengdilin95@gmail.com>
Date:   Thu Mar 30 00:50:41 2017 -0400

    Merge pull request #29 from bwhanser/ben

    Works with arbitrarily many jobs taking arbitrarily many ints and bools

commit 74d69e0f81bc0bdbd85da1fcb59656af4c061e18
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Thu Mar 30 00:13:32 2017 -0400

    added comments for mengdi

commit 59c7fee754515931ec5479f509d3d248dd791884
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed Mar 29 23:20:50 2017 -0400

    ints and bools now work as job arguments

commit 15344c461b34a306c4970be8873b684a708cba2c
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed Mar 29 18:43:38 2017 -0400

    now allows remote calls to all job functions

commit c84d99cf00dfab8708ecf6de1be5f2e1a0939625
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Wed Mar 29 02:23:02 2017 -0400

    jobs may now take many ints

commit a3919fcbb09f4f95f383c886cf6e0023f80d7d27
Merge: 7789c4a fad55a1
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon Mar 27 19:32:33 2017 +0000

    Merge branch 'mengdi' of github.com:bwhanser/plt-ms into mengdi

commit 7789c4a79d2dfa2fcbfd53e38bc69cff3ec54eb2
Author: vagrant <vagrant@vagrant.vm>
Date:   Mon Mar 27 19:23:54 2017 +0000

    add comments to codegen

commit fad55a106731af1b6861beb7e21778d282aa08fa
Author: vagrant <vagrant@vagrant.vm>
Date:   Mon Mar 27 19:23:54 2017 +0000

    add comments to codegen

commit 7d323e4653d824c3e4aac9bcba9de735323fd8ef
Merge: 7d36caa 22cf750
Author: mengdilin <mengdilin95@gmail.com>
Date:   Mon Mar 27 15:22:10 2017 -0400

    Merge pull request #27 from bwhanser/ben
```

```
    fix memory leaks of job arguments and return value

commit 22cf75086e32d5d453c2ac5575b1a12057edc42b
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Mon Mar 27 15:13:19 2017 -0400

    fix memory leaks of job arguments and return value

commit 7d36caa160c20d23091b02635eda26d6ff58cf37
Merge: 0d0bb1f 7c82059
Author: bwhanser <bwhanser@users.noreply.github.com>
Date:   Sun Mar 26 22:35:24 2017 -0400

    Merge pull request #26 from bwhanser/mengdi

    job var declaration works

commit 7c820596d0fe120f07d980bddda319704644fde7
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 26 22:31:47 2017 -0400

    semantic checking complete for job decl

commit cb28affbbf8fd97e71918af7fe7857a0b310434f
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 26 22:11:47 2017 -0400

    job a; works, but job a = 1; is valid as well

commit 496a0e7b5d8a6a7552917315470ffd0ebcbda8c2
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 26 21:40:38 2017 -0400

    update readme to include running ./master and ./slave

commit 0d0bb1fb4653711f8ef1d82355e00a99b04e06ca
Merge: f930fbe a6e89f7
Author: bwhanser <bwhanser@users.noreply.github.com>
Date:   Sun Mar 26 21:35:25 2017 -0400

    Merge pull request #23 from bwhanser/mimi

    Parser supports struct declaration, field access and assignment

commit f930fbe4145f6ec390fd6cb3dd653e9cbc1b26f5
Merge: bd5db7d 3517b68
Author: mengdilin <mengdilin95@gmail.com>
Date:   Sun Mar 26 21:32:26 2017 -0400

    Merge pull request #24 from bwhanser/runtime-experimental

    Update runtime to use queues, fix linkability

commit bd5db7d2b837c50faf24d035b087ed9f8c2e0a95
Merge: b696c62 3d33266
Author: mengdilin <mengdilin95@gmail.com>
Date:   Sun Mar 26 21:28:14 2017 -0400

    Merge pull request #25 from bwhanser/mengdi

    Compiler now works end-to-end!

commit 3d3326654b42b01832b59d3d94dcc6996763af2a
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 26 21:14:15 2017 -0400

    compiler now working end-to-end!

commit 3517b682426af5e54c3eb695b96562335f37ac92
```

```
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 26 21:12:47 2017 -0400

    add dummy function handles to slave to facilitate linking

commit a5c3f03b209fc32843585b697a4729b570040659
Merge: 96c9fa7 1f1bfcb
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 26 21:03:49 2017 -0400

    Merge branch 'mengdi' of https://github.com/bwhanser/plt-ms into mengdi

commit 96c9fa7747ff10d802a5e02e82a797132f703303
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 26 21:02:38 2017 -0400

    added basic job table to codegen

commit 1f1bfcb0efeb82884cee2bfa8cc680fb3e50b261
Author: vagrant <vagrant@vagrant.vm>
Date:   Mon Mar 27 01:01:37 2017 +0000

    codegen generates a simple f_job properly

commit edbd9fa1eda388b6fe566ef0110bd06714e1d385
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 26 20:04:17 2017 -0400

    codegen does not generate master_job

commit a6e89f7f2cb0ddddb79d4c5fe80630afefe7b057
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Mar 26 20:00:17 2017 -0400

    because I'm bad at merging

commit 39044c876b8c7968b72cfb4e721f82880b3aff5a
Merge: ac316f0 b8b438a
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Mar 26 19:57:54 2017 -0400

    merge

commit ac316f0885aad8b9674374e7a1d8ebfb5b2c3925
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Mar 26 19:55:05 2017 -0400

    parser supports field assignment, separate from regular assignment

commit ae73d6f9e7944204bbd44b971770e9c55daec46c
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Mar 26 19:26:38 2017 -0400

    parser accepts the -> field access symbol

commit c2da2b37fee48199c271f3c58c2d25c7ab873de8
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Mar 26 19:20:23 2017 -0400

    structdecl is working, but it is not a type

commit 93d9861e029067e1a49bf4ab8f3447f04d6c4069
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Mar 26 18:55:30 2017 -0400

    removed float stuff, added some semi-working struct stuff

commit 735a4b270f12f52b2a01f193437f40ff0cbe6eec
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Mar 26 17:40:24 2017 -0400
```

```
    making it compile

commit 712939effe8d0c954613dc5a63c67a9a66f93270
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Mar 26 15:18:28 2017 -0400

    floats, single line comments, added tokens for structs

commit 57281107a005c30a9126fcf6b355992e4eeafb79
Merge: 2450548 e199a9c
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 26 19:45:30 2017 -0400

    Merge branch 'mengdi' of https://github.com/bwhanser/plt-ms into mengdi

commit 2450548a12ed21a349d888a5ac440151ef54572c
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 26 19:44:06 2017 -0400

    strange %entry error

commit e199a9c8c793f8c2bed4e2e1e6e675bde7bdd995
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 26 19:37:46 2017 -0400

    new error in codegen

commit 3089b0a195e4af05b09575f98efa4b56e16cc7b4
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 26 19:31:43 2017 -0400

    failing codegen

commit b9d3af13c697c3deb75bafe1dc7a9582ad066899
Merge: cbbc4a9 874cf0d
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 26 19:28:10 2017 -0400

    Merge branch 'mengdi' of github.com:bwhanser/plt-ms into mengdi

commit b8b438ac43f701f441a066fd69b3478a910dfde4
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Mar 26 19:26:38 2017 -0400

    parser accepts the -> field access symbol

commit cbbc4a99c4974295497ad2f00580918086e580ca
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 26 19:26:19 2017 -0400

    resolve merge conflict

commit b31455e356c15226ac9c9aaed7981a2299cf12bb
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 26 19:12:09 2017 -0400

    working on deserialization

commit bc58915a73ca53b77d152a1a98e0a628cc311ca3
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 26 17:36:37 2017 -0400

    non-working deserialization

commit b696c62d3acc8a0fbec8142a6562112f2f9018b1
Merge: cfb4ef4 ec61e09
Author: mengdilin <mengdilin95@gmail.com>
Date:   Sun Mar 26 19:21:29 2017 -0400
```

```
    Merge pull request #22 from bwhanser/ben-improve-serialization

    Get now in codegen

commit ec61e0969fb50fd4f5e6ea860646b0bf39944382
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 26 19:20:41 2017 -0400

    modified test driver

commit d66d84161f3a622d0ad84f34ed74db6e5f3c8143
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Mar 26 19:20:23 2017 -0400

    structdecl is working, but it is not a type

commit 874cf0daea43890233ad43b34442c16b0c4272fc
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 26 19:12:09 2017 -0400

    working on deserialization

commit 0bdbb88dbbbe7e5694e8d43c26abc7a77048ad1e
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Mar 26 18:55:30 2017 -0400

    removed float stuff, added some semi-working struct stuff

commit e544607b171dc933094a5ee353668916ca7f84bc
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Mar 26 17:40:24 2017 -0400

    making it compile

commit e181240ac6788d113a50a0c8e0429d22354402c7
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Mar 26 15:18:28 2017 -0400

    floats, single line comments, added tokens for structs

commit cfb4ef4af3d0a29329b3f6e0544ca992903655f0
Merge: 28bd4c2 43825f5
Author: Miranda Li <mirandali1995@gmail.com>
Date:   Sun Mar 26 18:54:53 2017 -0400

    Merge pull request #21 from bwhanser/mengdi-parser-job

    job declaration is supported

commit 907f38c49dceda7c33f321f8c83b42586bca102c
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 26 18:50:13 2017 -0400

    Get now working in codegen

commit ad79c90674299ce2619be1a26735e7be1a897555
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 26 17:36:37 2017 -0400

    non-working deserialization

commit 43825f5c067310d88ee6a7b0aceafbb905a52de3
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 26 18:30:40 2017 -0400

    job declaration is supported

commit 28bd4c27d2bb136c555c5d92d54f32cf50bf9342
Merge: c916beb a6e512c
Author: mengdilin <mengdilin95@gmail.com>
```

```
Date:   Sun Mar 26 17:59:17 2017 -0400

    Merge pull request #20 from bwhanser/ben-improve-serialization

    improve serialization function

commit a6e512cdb0ccbb5e0f098dd4c696748c07dea272
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 26 17:33:00 2017 -0400

    improve serialization function

commit c916bebc6bad2f2b8f72f88a4440c70b894c5987
Merge: 53788ad 3a4a611
Author: bwhanser <bwhanser@users.noreply.github.com>
Date:   Sun Mar 26 15:43:58 2017 -0400

    Merge pull request #19 from bwhanser/mimi

    Parser accepts get and cancel job

commit 53788ad525841af726c1b7f160206c7fccf7509b
Merge: 41fbc59 6c96bcf
Author: bwhanser <bwhanser@users.noreply.github.com>
Date:   Sun Mar 26 15:43:22 2017 -0400

    Merge pull request #18 from bwhanser/mengdi

    "main" -> "master" & formals and local var decls have the same namespace

commit 3a4a6117e5ae24552e4fb38bca359e3bb52e8e2b
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Mar 26 14:38:05 2017 -0400

    cancel in ast

commit d013cf8d4b8aecb8764d00f901a79bda1f9ca8cb
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sun Mar 26 14:35:29 2017 -0400

    parser accepts get and cancel keywords

commit 6c96bcfd9248f5bbdba6825ca60e16b281e74066
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 26 11:41:41 2017 -0400

    semant disallows same formal and local var decl

commit b31673d7d0a39d7e4e09266b958484187b9261d1
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 26 11:29:31 2017 -0400

    codegen produces master instead of main

commit 41fbc590565b7a796b32b0b0a6e43f451381764d
Merge: d904c8e e93446d
Author: mengdilin <mengdilin95@gmail.com>
Date:   Sun Mar 26 11:05:46 2017 -0400

    Merge pull request #17 from bwhanser/ben-remote

    remote sorta works

commit d904c8ec0cf3015fe3288c5c11db69555f50a08b
Merge: 8f0ee21 1d80645
Author: mengdilin <mengdilin95@gmail.com>
Date:   Sun Mar 26 11:05:11 2017 -0400

    Merge pull request #16 from bwhanser/ben-mimi-mod
```

```
    fix local and remote syntax

commit 8f0ee21422042863333579 26b1da900ce73df46b
Merge: 4d99be8 245b01b
Author: mengdilin <mengdilin95@gmail.com>
Date:   Sun Mar 26 11:05:03 2017 -0400

    Merge pull request #15 from bwhanser/ben-simple-runtime

    Get rid of semaphores in runtime, basic job_cancel

commit 4d99be884ec31a6ad2a148ffe8a7d66116880110
Merge: 4d6a543 ad47afc
Author: mengdilin <mengdilin95@gmail.com>
Date:   Sun Mar 26 11:04:52 2017 -0400

    Merge pull request #14 from bwhanser/mimi

    Parser accepts local and remote syntax

commit e93446d1504210a4d68bde1ef99e87167b375265
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 26 04:45:45 2017 -0400

    remote -> start_job function call works

commit 3fc72e8ce82f747edffe06dfc6a04896efe7835f
Merge: 4d6a543 1d80645
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 26 02:47:30 2017 -0400

    Merge remote-tracking branch 'origin/ben-mimi-mod'

commit 4d6a54395104a05a88efc97d0e543c5953055a0e
Merge: cc94f8b 2e073f5
Author: bwhanser <bwhanser@users.noreply.github.com>
Date:   Sun Mar 26 02:43:53 2017 -0400

    Merge pull request #13 from bwhanser/mengdi

    local var decls, assignment, & function call work in codegen

commit 1d806455caf79d592852baa862563d4cfc07f211
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 26 02:37:46 2017 -0400

    fix bugs in parser

commit cef8f210ad86c78444d3dc4afb1469d001f6405c
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 26 01:08:19 2017 -0400

    restarted jobs get priority

commit 738ee416f6db4028da97f5cddbec624001365780
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 26 00:52:12 2017 -0400

    removed old slave_queue code

commit ec5d8b458007e5ddbe5dc6706a4f7b7b009d0572
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 26 00:45:59 2017 -0400

    master now uses wait queue and per-socket threads - avoids deadlock

commit cb2c7b878d9cf4140d81db14e2931e98ab64afc0
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sat Mar 25 18:15:08 2017 -0400
```

```
    fixed job allocation issues

commit 9ab56ebb6414fcec95f946e92cfeb0970b5de117
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sat Mar 25 15:48:51 2017 -0400

    work on creating job pending queue

commit 245b01beeacf144b2fc52603f75692a6708cbe25
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sat Mar 25 16:04:11 2017 -0400

    fixed cancellation bug

commit 39fda192ca19b4d836592cdc2e0ce08052b2a085
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sat Mar 25 15:51:34 2017 -0400

    removed semaphores

commit ad47afcf38fa4f2e31bbfd0235cebba527436af6
Merge: 7f94633 fd320fb
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Mar 25 10:29:17 2017 -0400

    merging

commit 7f946338141575309cb1a99eb5e86816e7ab8d4c
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Mar 25 10:25:05 2017 -0400

    random stuff

commit 2e073f5faf3c711649cbae8d6becfbe979416cb0
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sat Mar 25 05:17:25 2017 -0400

    working local function call

commit 8ec15068897163292d77fb96d4885b2e172bd34a
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sat Mar 25 04:55:33 2017 -0400

    codegen handles simple local variable decl and assignment

commit ab839fc463797d0bc46e5f0a14e05278dcd135be
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sat Mar 25 04:36:28 2017 -0400

    update semant such that local var checking works

commit c18cdff7bc5c2fbfd2a7ec239e39e1cfec7dbecc
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Mar 25 03:06:16 2017 -0400

    commit before pulling from master

commit e7109ccccbbafe2ec3da12ec8dcd45e547d4ba17
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Mar 25 03:09:26 2017 -0400

    resolving merge conflict

commit cc94f8b65111111c1f1917c0526ec409f79c1a77
Merge: 95c1a61 a34acb5
Author: bwhanser <bwhanser@users.noreply.github.com>
Date:   Sat Mar 25 03:05:04 2017 -0400

    Merge pull request #12 from bwhanser/mengdi
```

```
    Codegen compiles

commit a34acb5705fe3ffc1619c52688048403a4064be6
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sat Mar 25 01:57:59 2017 -0400

    ast provides RemoteCall

commit a42c9f423ad81d6f60f6345b3a141f365ccb4355
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sat Mar 25 01:50:00 2017 -0400

    codegen works with microc-llvm/testfile where you can print

commit cbb9ebeb04d3c05ea070fd3a41efdcf006abd6af
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sat Mar 25 00:20:40 2017 -0400

    add linking instruction to readme

commit 4342e500fc6302e665a0d52eb0e79a54f5060eb4
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Fri Mar 24 23:54:02 2017 -0400

    added api functions, fixed synchronization bugs

commit fd320fbc8611574a5aed7e34bf59580c690cc6be
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Fri Mar 24 14:44:57 2017 -0400

    added remote and local to scanner parser ast

commit 95c1a61d8e3a5fc8728a91fd58aa8693127053c2
Merge: 23f6475 1fbfa65
Author: mengdilin <mengdilin95@gmail.com>
Date:   Fri Mar 24 13:47:51 2017 -0400

    Merge pull request #11 from bwhanser/mimi

    Changes to makefile

commit 1fbfa651afd0dfa1ae2e803c8065c443b2088e1d
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Fri Mar 24 13:44:23 2017 -0400

    more updates to makefile

commit 23f647503cfd348a66cc0da9d9b3c321c720fb9d
Merge: e37576b b672660
Author: mengdilin <mengdilin95@gmail.com>
Date:   Fri Mar 24 13:42:15 2017 -0400

    Merge pull request #9 from bwhanser/ben-simple-runtime

    Simple runtime libraries

commit 71e09dcf2c3a4bc71cfa376d5245747fc947ad90
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Fri Mar 24 13:40:22 2017 -0400

    updated make clean

commit e37576b7a9ef545717e9153a77a2e129ac4cfe6b
Merge: 1dec559 420bb7b
Author: Miranda Li <mirandali1995@gmail.com>
Date:   Fri Mar 24 13:24:02 2017 -0400

    Merge pull request #6 from bwhanser/mengdi

    Parser accepts syntax for vector
```

```
commit 420bb7b65eb0117dc502f68626bacb443210a05b
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Tue Mar 21 12:29:26 2017 -0400

    add master pretty_printing

commit 0b347b1e69dbac38c501f0a6d3f6b94e71ea2c72
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Tue Mar 21 12:23:34 2017 -0400

    Remove fdecl's locals field from ast

commit ce1bb003e5f192651ed61f157007b011f86cd0e1
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Tue Mar 21 12:11:29 2017 -0400

    update ast

commit 27ae93b0070e4a9ed7324528836e48a8dfe15b4a
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Tue Mar 21 09:40:24 2017 -0400

    all blocks must be enclosed in curly braces

commit 7c45db1a3e30957299c5ca572aad3274c6ec34fc
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon Mar 20 20:53:13 2017 -0400

    working parser for declaration + assignment

commit 299bc40c7eb42b1be032c98a12146505dc0dfa0e
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Mon Mar 20 19:49:05 2017 -0400

    declaration out of order works in function body

commit b672660a9cff28ac1a8811b11df59e795dc965ad
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Tue Mar 7 19:32:07 2017 -0500

    fix behavior on abnormal client termination

commit c9ab8e16022dcf620ddb7252e5bd1e40de617413
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Tue Mar 7 17:16:15 2017 -0500

    fixed concurrency bugs; dividing workload among many slaves now works

commit b9cc8bc7737779e91a066601ea7a414c7cc6acd9
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Tue Mar 7 15:40:13 2017 -0500

    added new test case

commit 4265f55acf47b3b9a13d749f66f42d23b2cedf9b
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Tue Mar 7 01:55:04 2017 -0500

    fixed locking bugs, added new test case

commit 0ca45bfe5294aadc4673c684492979c3ec735306
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Mon Mar 6 23:07:06 2017 -0500

    misc bugfixes and rearrangement

commit 1dec559df0825d5ca3c4fc781300857341fd6c72
Merge: 2bbbaa0 30d4a77
Author: Miranda Li <mirandali1995@gmail.com>
```

```
Date:   Sun Mar 5 07:50:03 2017 -0500

    Merge pull request #2 from bwhanser/mengdi

    Mengdi

commit d20d9403a03dfd2c17826f9aab27e99fc68a8ca2
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 5 03:32:47 2017 -0500

    added README to runtime library

commit ecb72d6fb5314444698222e07c522770ca6f0947
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 5 03:13:12 2017 -0500

    added testing script

commit 79458b4d6a5c769b0940d43cc559371826666b9f
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 5 02:43:42 2017 -0500

    tweaks and improve handling of client failure

commit 07d22a884fcbf5293026f72d314055a8a25a8225
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 5 02:37:08 2017 -0500

    improved job list implementation

commit 30d4a778fab8c239446d7da581efd131b6c288fe
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 5 02:09:41 2017 -0500

    add new test case for vector

commit 642fdb8adf4e14ec9459058b7c9e1153211a0312
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 5 02:05:44 2017 -0500

    remove extraneous test files

commit e1e968a534de9fe630d6f34523fc6ddae18d3bbc
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 5 02:03:43 2017 -0500

    added test cases for vector

commit 51553f936da29f35d8f5c9b6df402d762269371a
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 5 02:03:02 2017 -0500

    semi-working out of order dec and assign. Parser doesn't throw an error but semant.ml
 does

commit 1cfc5cfc14fb1e57574dcc458971ac41822b5797
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 5 00:45:29 2017 -0500

    moved tests to subdirectory, now links via static library

commit f2b677f261a0e006c8f86a1bf84690741a9e0ce4
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sun Mar 5 00:37:57 2017 -0500

    working list literals

commit c9ba4853cc90b1c3e3b619f8e2c0a700ff7a094a
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sun Mar 5 00:23:19 2017 -0500
```

```
    improves slave queue implementation

commit 2bbbaa0cff8e4e6c34a55cfbdc8e7333d2d99bdb
Merge: 052bfcf f87ee17
Author: Miranda Li <mirandali1995@gmail.com>
Date:   Sat Mar 4 23:30:42 2017 -0500

    Merge pull request #1 from bwhanser/mengdi

    Delete Executable Files

commit f87ee177c6c6d4b9ff7870ea9a8c95ff4561234b
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sat Mar 4 23:24:58 2017 -0500

    working on list literal

commit 5ed2a88ca04e86abf1a4a3412162ab5a593beaeb
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sat Mar 4 23:22:42 2017 -0500

    rebase test commit

commit 052bfcf8568efcac4db045d193e7b14e3c31e639
Author: Miranda Li <mjl2206@columbia.edu>
Date:   Sat Mar 4 23:16:10 2017 -0500

    removed skeleton files. changes in microc

commit 972c872cd44df33bf0f46edf430cf35a7d00bd2f
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sat Mar 4 22:30:01 2017 -0500

    add debug.sh

commit 35119be4bb3ec9bfe795c61d856ea81daac5c983
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sat Mar 4 22:07:01 2017 -0500

    misc simplifications

commit 19e855df68ad724b24e5e026f76446a5fe32310e
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sat Mar 4 20:26:22 2017 -0500

    simplification of code

commit fcfedef3a51c080bc23832effd62cbcf2c2c8103
Author: Mengdi Lin <mengdilin95@gmail.com>
Date:   Sat Mar 4 18:38:11 2017 -0500

    able to parse vector<data type> a; declaration

commit 0529f970b1726e2073e86ac41363597522adf338
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sat Mar 4 04:28:49 2017 -0500

    basic master slave functionality verified; uploaded test driver

commit 133a5f57a254a66395a97077f1360a0f00b953b5
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sat Mar 4 02:34:06 2017 -0500

    changed tabs to spaces

commit a299b376bb40f8531c3f2f65685ef9830f46db03
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sat Mar 4 02:25:23 2017 -0500
```

```
    Basic implementation of master; misc bugfixes

commit 9130ef41ffa123d38a7e165a536becb9f6814c8d
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Fri Mar 3 23:27:08 2017 -0500

    Basic slave implementation

commit 5ba50de251aacf857a8a9332d6d51e08ec8a3043
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sat Feb 25 16:58:39 2017 -0500

    tester skeleton

commit 6e69a7f9d69b9df301aad165dd3ab8cdf4168c4a
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sat Feb 25 16:46:01 2017 -0500

    directory structure

commit c1d7b27db4c033bb4d74bb2e543c71d38230b8a0
Author: Benjamin Hanser <bwh2124@columbia.edu>
Date:   Sat Feb 25 16:15:04 2017 -0500

    beg
```

```
job int foo (vector<int> vec)
{
    return vec[4];
}

job int bar (vector<double> vec)
{
    return size vec;
}

job vector<int> baz (vector<vector<int>> vec)
{
    return vec[1];
}

job int vecsum (vector<int> vec)
{
    int i;
    int sum = 0;
    for (i = 0; i < size vec; i = i + 1)
        sum = sum + vec[i];
    return sum;
}

master
{
    vector<int> a;
    a::4;
    a::6;
    a::8;
    a::10;
    a::12;
    vector<double> d;
    d::2.2;
    d::3.3;
    d::4.4;
    vector<vector<int>> aa;
    aa::a;
    vector<int> b;
    b::3;
    b::5;
    aa::b;
    job<int> j = remote foo(a);
    job<int> k = remote bar(d);
    job<vector<int>> l = remote baz(aa);
    print(get j);
    print(get k);
    vector<int> zz = get l;
    print(zz[1]);
    job<int> sj = remote vecsum(a);
    print(get sj);
}
```

```
; ModuleID = 'M/s'
source_filename = "M/s"

%vec_t = type { %vec_t*, i32, i32 }

@fmt = private unnamed_addr constant [4 x i8] c"%d\0A\00"
@fmt.1 = private unnamed_addr constant [4 x i8] c"%f\0A\00"
@true_str = private unnamed_addr constant [6 x i8] c"true\0A\00"
@false_str = private unnamed_addr constant [7 x i8] c"false\0A\00"
@false_str.2 = private unnamed_addr constant [4 x i8] c"%s\0A\00"
@fmt.3 = private unnamed_addr constant [4 x i8] c"%d\0A\00"
@fmt.4 = private unnamed_addr constant [4 x i8] c"%f\0A\00"
@true_str.5 = private unnamed_addr constant [6 x i8] c"true\0A\00"
@false_str.6 = private unnamed_addr constant [7 x i8] c"false\0A\00"
@false_str.7 = private unnamed_addr constant [4 x i8] c"%s\0A\00"
@fmt.8 = private unnamed_addr constant [4 x i8] c"%d\0A\00"
@fmt.9 = private unnamed_addr constant [4 x i8] c"%f\0A\00"
@true_str.10 = private unnamed_addr constant [6 x i8] c"true\0A\00"
@false_str.11 = private unnamed_addr constant [7 x i8] c"false\0A\00"
@false_str.12 = private unnamed_addr constant [4 x i8] c"%s\0A\00"
@fmt.13 = private unnamed_addr constant [4 x i8] c"%d\0A\00"
@fmt.14 = private unnamed_addr constant [4 x i8] c"%f\0A\00"
@true_str.15 = private unnamed_addr constant [6 x i8] c"true\0A\00"
@false_str.16 = private unnamed_addr constant [7 x i8] c"false\0A\00"
@false_str.17 = private unnamed_addr constant [4 x i8] c"%s\0A\00"
@fmt.18 = private unnamed_addr constant [4 x i8] c"%d\0A\00"
@fmt.19 = private unnamed_addr constant [4 x i8] c"%f\0A\00"
@true_str.20 = private unnamed_addr constant [6 x i8] c"true\0A\00"
@false_str.21 = private unnamed_addr constant [7 x i8] c"false\0A\00"
@false_str.22 = private unnamed_addr constant [4 x i8] c"%s\0A\00"
@job_funcs = global [4 x void (i8*, i32, i8**, i32*)*] [void (i8*, i32, i8**, i32*)* @foo
_job, void (i8*, i32, i8**, i32*)* @bar_job, void (i8*, i32, i8**, i32*)* @baz_job, void
(i8*, i32, i8**, i32*)* @vecsum_job]
@num_jobs = global i32 4

declare i32 @printf(i8*, ...)

declare i32 @start_job(i32, i8*, i32)

declare void @reap_job(i32, i8**, i32*)

declare i32 @get_job_status(i32)

declare i32 @cancel_job(i32)

; Function Attrs: argmemonly nounwind
declare void @llvm.memcpy.p0i8.p0i8.i32(i8* nocapture writeonly, i8* nocapture readonly,
i32, i32, i1) #0

define i32 @foo(%vec_t %vec) {
entry:
  %vec1 = alloca %vec_t
  store %vec_t %vec, %vec_t* %vec1
  %retval = alloca i32
  store i32 0, i32* %retval
  %elem_arr_p = getelementptr %vec_t, %vec_t* %vec1, i32 0, i32 0
  %elem_arr_load = load %vec_t*, %vec_t** %elem_arr_p
  %elem_arr_cast = bitcast %vec_t* %elem_arr_load to i32*
  %elem = getelementptr i32, i32* %elem_arr_cast, i32 4
  %element = load i32, i32* %elem
  store i32 %element, i32* %retval
  br label %return_loc

return_loc:                                       ; preds = %entry
  %ptr_p = getelementptr %vec_t, %vec_t* %vec1, i32 0, i32 0
  %size_p = getelementptr %vec_t, %vec_t* %vec1, i32 0, i32 1
  %ptr = load %vec_t*, %vec_t** %ptr_p
  %size = load i32, i32* %size_p
  %iszerosize = icmp slt i32 %size, 1
  br i1 %iszerosize, label %merge, label %del_nonzerosize
```

```
del_nonzerosize:                                  ; preds = %return_loc
  %0 = bitcast %vec_t* %ptr to i8*
  tail call void @free(i8* %0)
  br label %merge

merge:                                            ; preds = %return_loc, %del_nonzerosize
  %final_retval = load i32, i32* %retval
  ret i32 %final_retval
}

define i32 @bar(%vec_t %vec) {
entry:
  %vec1 = alloca %vec_t
  store %vec_t %vec, %vec_t* %vec1
  %retval = alloca i32
  store i32 0, i32* %retval
  %size_p = getelementptr %vec_t, %vec_t* %vec1, i32 0, i32 1
  %size = load i32, i32* %size_p
  store i32 %size, i32* %retval
  br label %return_loc

return_loc:                                       ; preds = %entry
  %ptr_p = getelementptr %vec_t, %vec_t* %vec1, i32 0, i32 0
  %size_p2 = getelementptr %vec_t, %vec_t* %vec1, i32 0, i32 1
  %ptr = load %vec_t*, %vec_t** %ptr_p
  %size3 = load i32, i32* %size_p2
  %iszerosize = icmp slt i32 %size3, 1
  br i1 %iszerosize, label %merge, label %del_nonzerosize

del_nonzerosize:                                  ; preds = %return_loc
  %0 = bitcast %vec_t* %ptr to i8*
  tail call void @free(i8* %0)
  br label %merge

merge:                                            ; preds = %return_loc, %del_nonzerosize
  %final_retval = load i32, i32* %retval
  ret i32 %final_retval
}

define %vec_t @baz(%vec_t %vec) {
entry:
  %vec1 = alloca %vec_t
  store %vec_t %vec, %vec_t* %vec1
  %retval = alloca %vec_t
  store %vec_t zeroinitializer, %vec_t* %retval
  %vect_gep = getelementptr %vec_t, %vec_t* %vec1, i32 0, i32 0
  %vec_load = load %vec_t*, %vec_t** %vect_gep
  %elem = getelementptr %vec_t, %vec_t* %vec_load, i32 1
  %ptr_p = getelementptr %vec_t, %vec_t* %elem, i32 0, i32 0
  %size_p = getelementptr %vec_t, %vec_t* %elem, i32 0, i32 1
  %len_p = getelementptr %vec_t, %vec_t* %elem, i32 0, i32 2
  %ptr = load %vec_t*, %vec_t** %ptr_p
  %size = load i32, i32* %size_p
  %len = load i32, i32* %len_p
  %out_vec = alloca %vec_t
  %out_ptr = getelementptr %vec_t, %vec_t* %out_vec, i32 0, i32 0
  %out_size = getelementptr %vec_t, %vec_t* %out_vec, i32 0, i32 1
  %out_len = getelementptr %vec_t, %vec_t* %out_vec, i32 0, i32 2
  store i32 %size, i32* %out_size
  store i32 %len, i32* %out_len
  %iszerosize = icmp slt i32 %size, 1
  br i1 %iszerosize, label %zerosize, label %nonzerosize

return_loc:                                       ; preds = %merge
  %ptr_p2 = getelementptr %vec_t, %vec_t* %vec1, i32 0, i32 0
  %size_p3 = getelementptr %vec_t, %vec_t* %vec1, i32 0, i32 1
  %ptr4 = load %vec_t*, %vec_t** %ptr_p2
  %size5 = load i32, i32* %size_p3
  %iter = alloca i32
```

```
  store i32 0, i32* %iter
  %iszerosize6 = icmp slt i32 %size5, 1
  br i1 %iszerosize6, label %mergeagain, label %check_size

zerosize:                                         ; preds = %entry
  store %vec_t* null, %vec_t** %out_ptr
  br label %merge

nonzerosize:                                      ; preds = %entry
  %mallocsize = mul i32 %len, ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to i32
)
  %malloccall = tail call i8* @malloc(i32 %mallocsize)
  %newvec = bitcast i8* %malloccall to i32*
  %newvec_cast = bitcast i32* %newvec to %vec_t*
  store %vec_t* %newvec_cast, %vec_t** %out_ptr
  %src_cast = bitcast %vec_t* %ptr to i8*
  %dst_cast = bitcast i32* %newvec to i8*
  %sz_to_copy = mul i32 %size, ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to i3
2)
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast, i8* %src_cast, i32 %sz_to_copy, i32
 1, i1 true)
  br label %merge

merge:                                            ; preds = %nonzerosize, %zerosize
  %out_vec_val = load %vec_t, %vec_t* %out_vec
  store %vec_t %out_vec_val, %vec_t* %retval
  br label %return_loc

check_size:                                       ; preds = %merge13, %return_loc
  %iter_val = load i32, i32* %iter
  %keeplooping = icmp slt i32 %iter_val, %size5
  br i1 %keeplooping, label %del_vec_elem, label %merge14

del_vec_elem:                                     ; preds = %check_size
  %iter_val7 = load i32, i32* %iter
  %dest_ptr = getelementptr %vec_t, %vec_t* %ptr4, i32 %iter_val7
  %ptr_p8 = getelementptr %vec_t, %vec_t* %dest_ptr, i32 0, i32 0
  %size_p9 = getelementptr %vec_t, %vec_t* %dest_ptr, i32 0, i32 1
  %ptr10 = load %vec_t*, %vec_t** %ptr_p8
  %size11 = load i32, i32* %size_p9
  %iszerosize12 = icmp slt i32 %size11, 1
  br i1 %iszerosize12, label %merge13, label %del_nonzerosize

del_nonzerosize:                                  ; preds = %del_vec_elem
  %0 = bitcast %vec_t* %ptr10 to i8*
  tail call void @free(i8* %0)
  br label %merge13

merge13:                                          ; preds = %del_vec_elem, %del_nonzerosi
ze
  %new_iter_val = add i32 %iter_val7, 1
  store i32 %new_iter_val, i32* %iter
  br label %check_size

merge14:                                          ; preds = %check_size
  %1 = bitcast %vec_t* %ptr4 to i8*
  tail call void @free(i8* %1)
  br label %mergeagain

mergeagain:                                       ; preds = %merge14, %return_loc
  %final_retval = load %vec_t, %vec_t* %retval
  ret %vec_t %final_retval
}

define i32 @vecsum(%vec_t %vec) {
entry:
  %vec1 = alloca %vec_t
  store %vec_t %vec, %vec_t* %vec1
  %retval = alloca i32
  store i32 0, i32* %retval
```

```
  %i = alloca i32
  %sum = alloca i32
  store i32 0, i32* %sum
  store i32 0, i32* %i
  br label %while

return_loc:                                       ; preds = %merge
  %ptr_p = getelementptr %vec_t, %vec_t* %vec1, i32 0, i32 0
  %size_p9 = getelementptr %vec_t, %vec_t* %vec1, i32 0, i32 1
  %ptr = load %vec_t*, %vec_t** %ptr_p
  %size10 = load i32, i32* %size_p9
  %iszerosize = icmp slt i32 %size10, 1
  br i1 %iszerosize, label %merge11, label %del_nonzerosize

while:                                            ; preds = %while_body, %entry
  %i6 = load i32, i32* %i
  %size_p = getelementptr %vec_t, %vec_t* %vec1, i32 0, i32 1
  %size = load i32, i32* %size_p
  %tmp7 = icmp slt i32 %i6, %size
  br i1 %tmp7, label %while_body, label %merge

while_body:                                       ; preds = %while
  %sum2 = load i32, i32* %sum
  %i3 = load i32, i32* %i
  %elem_arr_p = getelementptr %vec_t, %vec_t* %vec1, i32 0, i32 0
  %elem_arr_load = load %vec_t*, %vec_t** %elem_arr_p
  %elem_arr_cast = bitcast %vec_t* %elem_arr_load to i32*
  %elem = getelementptr i32, i32* %elem_arr_cast, i32 %i3
  %element = load i32, i32* %elem
  %tmp = add i32 %sum2, %element
  store i32 %tmp, i32* %sum
  %i4 = load i32, i32* %i
  %tmp5 = add i32 %i4, 1
  store i32 %tmp5, i32* %i
  br label %while

merge:                                            ; preds = %while
  %sum8 = load i32, i32* %sum
  store i32 %sum8, i32* %retval
  br label %return_loc

del_nonzerosize:                                  ; preds = %return_loc
  %0 = bitcast %vec_t* %ptr to i8*
  tail call void @free(i8* %0)
  br label %merge11

merge11:                                          ; preds = %return_loc, %del_nonzerosize
  %final_retval = load i32, i32* %retval
  ret i32 %final_retval
}

define i32 @master() {
entry:
  %retval = alloca i32
  store i32 0, i32* %retval
  %vec_alloc = alloca %vec_t
  %ptr = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  store %vec_t* null, %vec_t** %ptr
  %size = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 1
  store i32 0, i32* %size
  %len = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 2
  store i32 0, i32* %len
  %size_p = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 1
  %size_var = load i32, i32* %size_p
  %len_p = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 2
  %leng_var = load i32, i32* %len_p
  %size_var_cmp_leng = icmp slt i32 %size_var, %leng_var
  br i1 %size_var_cmp_leng, label %pushback, label %resize

return_loc:                                       ; preds = %merge587
```

```
  %ptr_p598 = getelementptr %vec_t, %vec_t* %vec_alloc502, i32 0, i32 0
  %size_p599 = getelementptr %vec_t, %vec_t* %vec_alloc502, i32 0, i32 1
  %ptr600 = load %vec_t*, %vec_t** %ptr_p598
  %size601 = load i32, i32* %size_p599
  %iszerosize602 = icmp slt i32 %size601, 1
  br i1 %iszerosize602, label %merge604, label %del_nonzerosize603

resize:                                           ; preds = %entry
  %size_var_nonzero = icmp slt i32 %leng_var, 1
  br i1 %size_var_nonzero, label %resize_zero, label %resize_nonzero

resize_zero:                                      ; preds = %resize
  store i32 1, i32* %len_p
  br label %resize_merge

resize_nonzero:                                   ; preds = %resize
  %shl = shl i32 %leng_var, 1
  store i32 %shl, i32* %len_p
  br label %resize_merge

resize_merge:                                     ; preds = %resize_nonzero, %resize_zero
  %new_len_val = load i32, i32* %len_p
  %ptr_p = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %ptr_val = load %vec_t*, %vec_t** %ptr_p
  %mallocsize = mul i32 %new_len_val, ptrtoint (i32* getelementptr (i32, i32* null, i32 1
) to i32)
  %malloccall = tail call i8* @malloc(i32 %mallocsize)
  %vec_mem = bitcast i8* %malloccall to i32*
  %vec_mem_byte_cast = bitcast i32* %vec_mem to i8*
  %ptr_val_byte_cast = bitcast %vec_t* %ptr_val to i8*
  %sz_to_copy = mul i32 ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to i32), %si
ze_var
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast, i8* %ptr_val_byte_cast, i3
2 %sz_to_copy, i32 1, i1 true)
  %0 = bitcast %vec_t* %ptr_val to i8*
  tail call void @free(i8* %0)
  %vec_mem_cast = bitcast i32* %vec_mem to %vec_t*
  store %vec_t* %vec_mem_cast, %vec_t** %ptr_p
  br label %pushback

pushback:                                         ; preds = %resize_merge, %entry
  %elem_arr_p = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %elem_arr_load = load %vec_t*, %vec_t** %elem_arr_p
  %elem_arr_cast = bitcast %vec_t* %elem_arr_load to i32*
  %elem = getelementptr i32, i32* %elem_arr_cast, i32 %size_var
  store i32 4, i32* %elem
  %new_size = add i32 %size_var, 1
  store i32 %new_size, i32* %size_p
  %size_p1 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 1
  %size_var2 = load i32, i32* %size_p1
  %len_p3 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 2
  %leng_var4 = load i32, i32* %len_p3
  %size_var_cmp_leng5 = icmp slt i32 %size_var2, %leng_var4
  br i1 %size_var_cmp_leng5, label %pushback22, label %resize6

resize6:                                          ; preds = %pushback
  %size_var_nonzero7 = icmp slt i32 %leng_var4, 1
  br i1 %size_var_nonzero7, label %resize_zero8, label %resize_nonzero9

resize_zero8:                                     ; preds = %resize6
  store i32 1, i32* %len_p3
  br label %resize_merge11

resize_nonzero9:                                  ; preds = %resize6
  %shl10 = shl i32 %leng_var4, 1
  store i32 %shl10, i32* %len_p3
  br label %resize_merge11

resize_merge11:                                   ; preds = %resize_nonzero9, %resize_zer
o8
```

```
  %new_len_val12 = load i32, i32* %len_p3
  %ptr_p13 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %ptr_val14 = load %vec_t*, %vec_t** %ptr_p13
  %mallocsize15 = mul i32 %new_len_val12, ptrtoint (i32* getelementptr (i32, i32* null, i
32 1) to i32)
  %malloccall16 = tail call i8* @malloc(i32 %mallocsize15)
  %vec_mem17 = bitcast i8* %malloccall16 to i32*
  %vec_mem_byte_cast18 = bitcast i32* %vec_mem17 to i8*
  %ptr_val_byte_cast19 = bitcast %vec_t* %ptr_val14 to i8*
  %sz_to_copy20 = mul i32 ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to i32), %
size_var2
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast18, i8* %ptr_val_byte_cast19
, i32 %sz_to_copy20, i32 1, i1 true)
  %1 = bitcast %vec_t* %ptr_val14 to i8*
  tail call void @free(i8* %1)
  %vec_mem_cast21 = bitcast i32* %vec_mem17 to %vec_t*
  store %vec_t* %vec_mem_cast21, %vec_t** %ptr_p13
  br label %pushback22

pushback22:                                       ; preds = %resize_merge11, %pushback
  %elem_arr_p23 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %elem_arr_load24 = load %vec_t*, %vec_t** %elem_arr_p23
  %elem_arr_cast25 = bitcast %vec_t* %elem_arr_load24 to i32*
  %elem26 = getelementptr i32, i32* %elem_arr_cast25, i32 %size_var2
  store i32 6, i32* %elem26
  %new_size27 = add i32 %size_var2, 1
  store i32 %new_size27, i32* %size_p1
  %size_p28 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 1
  %size_var29 = load i32, i32* %size_p28
  %len_p30 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 2
  %leng_var31 = load i32, i32* %len_p30
  %size_var_cmp_leng32 = icmp slt i32 %size_var29, %leng_var31
  br i1 %size_var_cmp_leng32, label %pushback49, label %resize33

resize33:                                         ; preds = %pushback22
  %size_var_nonzero34 = icmp slt i32 %leng_var31, 1
  br i1 %size_var_nonzero34, label %resize_zero35, label %resize_nonzero36

resize_zero35:                                    ; preds = %resize33
  store i32 1, i32* %len_p30
  br label %resize_merge38

resize_nonzero36:                                 ; preds = %resize33
  %shl37 = shl i32 %leng_var31, 1
  store i32 %shl37, i32* %len_p30
  br label %resize_merge38

resize_merge38:                                   ; preds = %resize_nonzero36, %resize_ze
ro35
  %new_len_val39 = load i32, i32* %len_p30
  %ptr_p40 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %ptr_val41 = load %vec_t*, %vec_t** %ptr_p40
  %mallocsize42 = mul i32 %new_len_val39, ptrtoint (i32* getelementptr (i32, i32* null, i
32 1) to i32)
  %malloccall43 = tail call i8* @malloc(i32 %mallocsize42)
  %vec_mem44 = bitcast i8* %malloccall43 to i32*
  %vec_mem_byte_cast45 = bitcast i32* %vec_mem44 to i8*
  %ptr_val_byte_cast46 = bitcast %vec_t* %ptr_val41 to i8*
  %sz_to_copy47 = mul i32 ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to i32), %
size_var29
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast45, i8* %ptr_val_byte_cast46
, i32 %sz_to_copy47, i32 1, i1 true)
  %2 = bitcast %vec_t* %ptr_val41 to i8*
  tail call void @free(i8* %2)
  %vec_mem_cast48 = bitcast i32* %vec_mem44 to %vec_t*
  store %vec_t* %vec_mem_cast48, %vec_t** %ptr_p40
  br label %pushback49

pushback49:                                       ; preds = %resize_merge38, %pushback22
  %elem_arr_p50 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
```

```
  %elem_arr_load51 = load %vec_t*, %vec_t** %elem_arr_p50
  %elem_arr_cast52 = bitcast %vec_t* %elem_arr_load51 to i32*
  %elem53 = getelementptr i32, i32* %elem_arr_cast52, i32 %size_var29
  store i32 8, i32* %elem53
  %new_size54 = add i32 %size_var29, 1
  store i32 %new_size54, i32* %size_p28
  %size_p55 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 1
  %size_var56 = load i32, i32* %size_p55
  %len_p57 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 2
  %leng_var58 = load i32, i32* %len_p57
  %size_var_cmp_leng59 = icmp slt i32 %size_var56, %leng_var58
  br i1 %size_var_cmp_leng59, label %pushback76, label %resize60

resize60:                                         ; preds = %pushback49
  %size_var_nonzero61 = icmp slt i32 %leng_var58, 1
  br i1 %size_var_nonzero61, label %resize_zero62, label %resize_nonzero63

resize_zero62:                                    ; preds = %resize60
  store i32 1, i32* %len_p57
  br label %resize_merge65

resize_nonzero63:                                 ; preds = %resize60
  %shl64 = shl i32 %leng_var58, 1
  store i32 %shl64, i32* %len_p57
  br label %resize_merge65

resize_merge65:                                   ; preds = %resize_nonzero63, %resize_ze
ro62
  %new_len_val66 = load i32, i32* %len_p57
  %ptr_p67 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %ptr_val68 = load %vec_t*, %vec_t** %ptr_p67
  %mallocsize69 = mul i32 %new_len_val66, ptrtoint (i32* getelementptr (i32, i32* null, i
32 1) to i32)
  %malloccall70 = tail call i8* @malloc(i32 %mallocsize69)
  %vec_mem71 = bitcast i8* %malloccall70 to i32*
  %vec_mem_byte_cast72 = bitcast i32* %vec_mem71 to i8*
  %ptr_val_byte_cast73 = bitcast %vec_t* %ptr_val68 to i8*
  %sz_to_copy74 = mul i32 ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to i32), %
size_var56
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast72, i8* %ptr_val_byte_cast73
, i32 %sz_to_copy74, i32 1, i1 true)
  %3 = bitcast %vec_t* %ptr_val68 to i8*
  tail call void @free(i8* %3)
  %vec_mem_cast75 = bitcast i32* %vec_mem71 to %vec_t*
  store %vec_t* %vec_mem_cast75, %vec_t** %ptr_p67
  br label %pushback76

pushback76:                                       ; preds = %resize_merge65, %pushback49
  %elem_arr_p77 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %elem_arr_load78 = load %vec_t*, %vec_t** %elem_arr_p77
  %elem_arr_cast79 = bitcast %vec_t* %elem_arr_load78 to i32*
  %elem80 = getelementptr i32, i32* %elem_arr_cast79, i32 %size_var56
  store i32 10, i32* %elem80
  %new_size81 = add i32 %size_var56, 1
  store i32 %new_size81, i32* %size_p55
  %size_p82 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 1
  %size_var83 = load i32, i32* %size_p82
  %len_p84 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 2
  %leng_var85 = load i32, i32* %len_p84
  %size_var_cmp_leng86 = icmp slt i32 %size_var83, %leng_var85
  br i1 %size_var_cmp_leng86, label %pushback103, label %resize87

resize87:                                         ; preds = %pushback76
  %size_var_nonzero88 = icmp slt i32 %leng_var85, 1
  br i1 %size_var_nonzero88, label %resize_zero89, label %resize_nonzero90

resize_zero89:                                    ; preds = %resize87
  store i32 1, i32* %len_p84
  br label %resize_merge92
```

```
resize_nonzero90:                                 ; preds = %resize87
  %shl91 = shl i32 %leng_var85, 1
  store i32 %shl91, i32* %len_p84
  br label %resize_merge92

resize_merge92:                                   ; preds = %resize_nonzero90, %resize_ze
ro89
  %new_len_val93 = load i32, i32* %len_p84
  %ptr_p94 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %ptr_val95 = load %vec_t*, %vec_t** %ptr_p94
  %mallocsize96 = mul i32 %new_len_val93, ptrtoint (i32* getelementptr (i32, i32* null, i
32 1) to i32)
  %malloccall97 = tail call i8* @malloc(i32 %mallocsize96)
  %vec_mem98 = bitcast i8* %malloccall97 to i32*
  %vec_mem_byte_cast99 = bitcast i32* %vec_mem98 to i8*
  %ptr_val_byte_cast100 = bitcast %vec_t* %ptr_val95 to i8*
  %sz_to_copy101 = mul i32 ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to i32),
%size_var83
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast99, i8* %ptr_val_byte_cast10
0, i32 %sz_to_copy101, i32 1, i1 true)
  %4 = bitcast %vec_t* %ptr_val95 to i8*
  tail call void @free(i8* %4)
  %vec_mem_cast102 = bitcast i32* %vec_mem98 to %vec_t*
  store %vec_t* %vec_mem_cast102, %vec_t** %ptr_p94
  br label %pushback103

pushback103:                                      ; preds = %resize_merge92, %pushback76
  %elem_arr_p104 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %elem_arr_load105 = load %vec_t*, %vec_t** %elem_arr_p104
  %elem_arr_cast106 = bitcast %vec_t* %elem_arr_load105 to i32*
  %elem107 = getelementptr i32, i32* %elem_arr_cast106, i32 %size_var83
  store i32 12, i32* %elem107
  %new_size108 = add i32 %size_var83, 1
  store i32 %new_size108, i32* %size_p82
  %vec_alloc109 = alloca %vec_t
  %ptr110 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 0
  store %vec_t* null, %vec_t** %ptr110
  %size111 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 1
  store i32 0, i32* %size111
  %len112 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 2
  store i32 0, i32* %len112
  %size_p113 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 1
  %size_var114 = load i32, i32* %size_p113
  %len_p115 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 2
  %leng_var116 = load i32, i32* %len_p115
  %size_var_cmp_leng117 = icmp slt i32 %size_var114, %leng_var116
  br i1 %size_var_cmp_leng117, label %pushback134, label %resize118

resize118:                                        ; preds = %pushback103
  %size_var_nonzero119 = icmp slt i32 %leng_var116, 1
  br i1 %size_var_nonzero119, label %resize_zero120, label %resize_nonzero121

resize_zero120:                                   ; preds = %resize118
  store i32 1, i32* %len_p115
  br label %resize_merge123

resize_nonzero121:                                ; preds = %resize118
  %shl122 = shl i32 %leng_var116, 1
  store i32 %shl122, i32* %len_p115
  br label %resize_merge123

resize_merge123:                                  ; preds = %resize_nonzero121, %resize_z
ero120
  %new_len_val124 = load i32, i32* %len_p115
  %ptr_p125 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 0
  %ptr_val126 = load %vec_t*, %vec_t** %ptr_p125
  %mallocsize127 = mul i32 %new_len_val124, ptrtoint (double* getelementptr (double, doub
le* null, i32 1) to i32)
  %malloccall128 = tail call i8* @malloc(i32 %mallocsize127)
  %vec_mem129 = bitcast i8* %malloccall128 to double*
```

```
  %vec_mem_byte_cast130 = bitcast double* %vec_mem129 to i8*
  %ptr_val_byte_cast131 = bitcast %vec_t* %ptr_val126 to i8*
  %sz_to_copy132 = mul i32 ptrtoint (double* getelementptr (double, double* null, i32 1)
to i32), %size_var114
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast130, i8* %ptr_val_byte_cast1
31, i32 %sz_to_copy132, i32 1, i1 true)
  %5 = bitcast %vec_t* %ptr_val126 to i8*
  tail call void @free(i8* %5)
  %vec_mem_cast133 = bitcast double* %vec_mem129 to %vec_t*
  store %vec_t* %vec_mem_cast133, %vec_t** %ptr_p125
  br label %pushback134

pushback134:                                      ; preds = %resize_merge123, %pushback10
3
  %elem_arr_p135 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 0
  %elem_arr_load136 = load %vec_t*, %vec_t** %elem_arr_p135
  %elem_arr_cast137 = bitcast %vec_t* %elem_arr_load136 to double*
  %elem138 = getelementptr double, double* %elem_arr_cast137, i32 %size_var114
  store double 2.200000e+00, double* %elem138
  %new_size139 = add i32 %size_var114, 1
  store i32 %new_size139, i32* %size_p113
  %size_p140 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 1
  %size_var141 = load i32, i32* %size_p140
  %len_p142 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 2
  %leng_var143 = load i32, i32* %len_p142
  %size_var_cmp_leng144 = icmp slt i32 %size_var141, %leng_var143
  br i1 %size_var_cmp_leng144, label %pushback161, label %resize145

resize145:                                        ; preds = %pushback134
  %size_var_nonzero146 = icmp slt i32 %leng_var143, 1
  br i1 %size_var_nonzero146, label %resize_zero147, label %resize_nonzero148

resize_zero147:                                   ; preds = %resize145
  store i32 1, i32* %len_p142
  br label %resize_merge150

resize_nonzero148:                                ; preds = %resize145
  %shl149 = shl i32 %leng_var143, 1
  store i32 %shl149, i32* %len_p142
  br label %resize_merge150

resize_merge150:                                  ; preds = %resize_nonzero148, %resize_z
ero147
  %new_len_val151 = load i32, i32* %len_p142
  %ptr_p152 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 0
  %ptr_val153 = load %vec_t*, %vec_t** %ptr_p152
  %mallocsize154 = mul i32 %new_len_val151, ptrtoint (double* getelementptr (double, doub
le* null, i32 1) to i32)
  %malloccall155 = tail call i8* @malloc(i32 %mallocsize154)
  %vec_mem156 = bitcast i8* %malloccall155 to double*
  %vec_mem_byte_cast157 = bitcast double* %vec_mem156 to i8*
  %ptr_val_byte_cast158 = bitcast %vec_t* %ptr_val153 to i8*
  %sz_to_copy159 = mul i32 ptrtoint (double* getelementptr (double, double* null, i32 1)
to i32), %size_var141
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast157, i8* %ptr_val_byte_cast1
58, i32 %sz_to_copy159, i32 1, i1 true)
  %6 = bitcast %vec_t* %ptr_val153 to i8*
  tail call void @free(i8* %6)
  %vec_mem_cast160 = bitcast double* %vec_mem156 to %vec_t*
  store %vec_t* %vec_mem_cast160, %vec_t** %ptr_p152
  br label %pushback161

pushback161:                                      ; preds = %resize_merge150, %pushback13
4
  %elem_arr_p162 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 0
  %elem_arr_load163 = load %vec_t*, %vec_t** %elem_arr_p162
  %elem_arr_cast164 = bitcast %vec_t* %elem_arr_load163 to double*
  %elem165 = getelementptr double, double* %elem_arr_cast164, i32 %size_var141
  store double 3.300000e+00, double* %elem165
  %new_size166 = add i32 %size_var141, 1
```

```
  store i32 %new_size166, i32* %size_p140
  %size_p167 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 1
  %size_var168 = load i32, i32* %size_p167
  %len_p169 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 2
  %leng_var170 = load i32, i32* %len_p169
  %size_var_cmp_leng171 = icmp slt i32 %size_var168, %leng_var170
  br i1 %size_var_cmp_leng171, label %pushback188, label %resize172

resize172:                                        ; preds = %pushback161
  %size_var_nonzero173 = icmp slt i32 %leng_var170, 1
  br i1 %size_var_nonzero173, label %resize_zero174, label %resize_nonzero175

resize_zero174:                                   ; preds = %resize172
  store i32 1, i32* %len_p169
  br label %resize_merge177

resize_nonzero175:                                ; preds = %resize172
  %shl176 = shl i32 %leng_var170, 1
  store i32 %shl176, i32* %len_p169
  br label %resize_merge177

resize_merge177:                                  ; preds = %resize_nonzero175, %resize_z
ero174
  %new_len_val178 = load i32, i32* %len_p169
  %ptr_p179 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 0
  %ptr_val180 = load %vec_t*, %vec_t** %ptr_p179
  %mallocsize181 = mul i32 %new_len_val178, ptrtoint (double* getelementptr (double, doub
le* null, i32 1) to i32)
  %malloccall182 = tail call i8* @malloc(i32 %mallocsize181)
  %vec_mem183 = bitcast i8* %malloccall182 to double*
  %vec_mem_byte_cast184 = bitcast double* %vec_mem183 to i8*
  %ptr_val_byte_cast185 = bitcast %vec_t* %ptr_val180 to i8*
  %sz_to_copy186 = mul i32 ptrtoint (double* getelementptr (double, double* null, i32 1)
to i32), %size_var168
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast184, i8* %ptr_val_byte_cast1
85, i32 %sz_to_copy186, i32 1, i1 true)
  %7 = bitcast %vec_t* %ptr_val180 to i8*
  tail call void @free(i8* %7)
  %vec_mem_cast187 = bitcast double* %vec_mem183 to %vec_t*
  store %vec_t* %vec_mem_cast187, %vec_t** %ptr_p179
  br label %pushback188

pushback188:                                      ; preds = %resize_merge177, %pushback16
1
  %elem_arr_p189 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 0
  %elem_arr_load190 = load %vec_t*, %vec_t** %elem_arr_p189
  %elem_arr_cast191 = bitcast %vec_t* %elem_arr_load190 to double*
  %elem192 = getelementptr double, double* %elem_arr_cast191, i32 %size_var168
  store double 4.400000e+00, double* %elem192
  %new_size193 = add i32 %size_var168, 1
  store i32 %new_size193, i32* %size_p167
  %vec_alloc194 = alloca %vec_t
  %ptr195 = getelementptr %vec_t, %vec_t* %vec_alloc194, i32 0, i32 0
  store %vec_t* null, %vec_t** %ptr195
  %size196 = getelementptr %vec_t, %vec_t* %vec_alloc194, i32 0, i32 1
  store i32 0, i32* %size196
  %len197 = getelementptr %vec_t, %vec_t* %vec_alloc194, i32 0, i32 2
  store i32 0, i32* %len197
  %size_p198 = getelementptr %vec_t, %vec_t* %vec_alloc194, i32 0, i32 1
  %size_var199 = load i32, i32* %size_p198
  %len_p200 = getelementptr %vec_t, %vec_t* %vec_alloc194, i32 0, i32 2
  %leng_var201 = load i32, i32* %len_p200
  %size_var_cmp_leng202 = icmp slt i32 %size_var199, %leng_var201
  br i1 %size_var_cmp_leng202, label %pushback218, label %resize203

resize203:                                        ; preds = %pushback188
  %size_var_nonzero204 = icmp slt i32 %leng_var201, 1
  br i1 %size_var_nonzero204, label %resize_zero205, label %resize_nonzero206

resize_zero205:                                   ; preds = %resize203
```

```
  store i32 1, i32* %len_p200
  br label %resize_merge208

resize_nonzero206:                                ; preds = %resize203
  %shl207 = shl i32 %leng_var201, 1
  store i32 %shl207, i32* %len_p200
  br label %resize_merge208

resize_merge208:                                  ; preds = %resize_nonzero206, %resize_z
ero205
  %new_len_val209 = load i32, i32* %len_p200
  %ptr_p210 = getelementptr %vec_t, %vec_t* %vec_alloc194, i32 0, i32 0
  %ptr_val211 = load %vec_t*, %vec_t** %ptr_p210
  %mallocsize212 = mul i32 %new_len_val209, ptrtoint (%vec_t* getelementptr (%vec_t, %vec
_t* null, i32 1) to i32)
  %malloccall213 = tail call i8* @malloc(i32 %mallocsize212)
  %vec_mem214 = bitcast i8* %malloccall213 to %vec_t*
  %vec_mem_byte_cast215 = bitcast %vec_t* %vec_mem214 to i8*
  %ptr_val_byte_cast216 = bitcast %vec_t* %ptr_val211 to i8*
  %sz_to_copy217 = mul i32 ptrtoint (%vec_t* getelementptr (%vec_t, %vec_t* null, i32 1)
to i32), %size_var199
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast215, i8* %ptr_val_byte_cast2
16, i32 %sz_to_copy217, i32 1, i1 true)
  %8 = bitcast %vec_t* %ptr_val211 to i8*
  tail call void @free(i8* %8)
  store %vec_t* %vec_mem214, %vec_t** %ptr_p210
  br label %pushback218

pushback218:                                      ; preds = %resize_merge208, %pushback18
8
  %vect_gep = getelementptr %vec_t, %vec_t* %vec_alloc194, i32 0, i32 0
  %vec_load = load %vec_t*, %vec_t** %vect_gep
  %elem219 = getelementptr %vec_t, %vec_t* %vec_load, i32 %size_var199
  %ptr_p220 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %size_p221 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 1
  %len_p222 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 2
  %ptr223 = load %vec_t*, %vec_t** %ptr_p220
  %size224 = load i32, i32* %size_p221
  %len225 = load i32, i32* %len_p222
  %out_vec = alloca %vec_t
  %out_ptr = getelementptr %vec_t, %vec_t* %out_vec, i32 0, i32 0
  %out_size = getelementptr %vec_t, %vec_t* %out_vec, i32 0, i32 1
  %out_len = getelementptr %vec_t, %vec_t* %out_vec, i32 0, i32 2
  store i32 %size224, i32* %out_size
  store i32 %len225, i32* %out_len
  %iszerosize = icmp slt i32 %size224, 1
  br i1 %iszerosize, label %zerosize, label %nonzerosize

zerosize:                                         ; preds = %pushback218
  store %vec_t* null, %vec_t** %out_ptr
  br label %merge

nonzerosize:                                      ; preds = %pushback218
  %mallocsize226 = mul i32 %len225, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
to i32)
  %malloccall227 = tail call i8* @malloc(i32 %mallocsize226)
  %newvec = bitcast i8* %malloccall227 to i32*
  %newvec_cast = bitcast i32* %newvec to %vec_t*
  store %vec_t* %newvec_cast, %vec_t** %out_ptr
  %src_cast = bitcast %vec_t* %ptr223 to i8*
  %dst_cast = bitcast i32* %newvec to i8*
  %sz_to_copy228 = mul i32 %size224, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
 to i32)
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast, i8* %src_cast, i32 %sz_to_copy228,
i32 1, i1 true)
  br label %merge

merge:                                            ; preds = %nonzerosize, %zerosize
  %out_vec_val = load %vec_t, %vec_t* %out_vec
  store %vec_t %out_vec_val, %vec_t* %elem219
```

```
  %new_size229 = add i32 %size_var199, 1
  store i32 %new_size229, i32* %size_p198
  %vec_alloc230 = alloca %vec_t
  %ptr231 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 0
  store %vec_t* null, %vec_t** %ptr231
  %size232 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 1
  store i32 0, i32* %size232
  %len233 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 2
  store i32 0, i32* %len233
  %size_p234 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 1
  %size_var235 = load i32, i32* %size_p234
  %len_p236 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 2
  %leng_var237 = load i32, i32* %len_p236
  %size_var_cmp_leng238 = icmp slt i32 %size_var235, %leng_var237
  br i1 %size_var_cmp_leng238, label %pushback255, label %resize239

resize239:                                        ; preds = %merge
  %size_var_nonzero240 = icmp slt i32 %leng_var237, 1
  br i1 %size_var_nonzero240, label %resize_zero241, label %resize_nonzero242

resize_zero241:                                   ; preds = %resize239
  store i32 1, i32* %len_p236
  br label %resize_merge244

resize_nonzero242:                                ; preds = %resize239
  %shl243 = shl i32 %leng_var237, 1
  store i32 %shl243, i32* %len_p236
  br label %resize_merge244

resize_merge244:                                  ; preds = %resize_nonzero242, %resize_z
ero241
  %new_len_val245 = load i32, i32* %len_p236
  %ptr_p246 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 0
  %ptr_val247 = load %vec_t*, %vec_t** %ptr_p246
  %mallocsize248 = mul i32 %new_len_val245, ptrtoint (i32* getelementptr (i32, i32* null,
 i32 1) to i32)
  %malloccall249 = tail call i8* @malloc(i32 %mallocsize248)
  %vec_mem250 = bitcast i8* %malloccall249 to i32*
  %vec_mem_byte_cast251 = bitcast i32* %vec_mem250 to i8*
  %ptr_val_byte_cast252 = bitcast %vec_t* %ptr_val247 to i8*
  %sz_to_copy253 = mul i32 ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to i32),
%size_var235
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast251, i8* %ptr_val_byte_cast2
52, i32 %sz_to_copy253, i32 1, i1 true)
  %9 = bitcast %vec_t* %ptr_val247 to i8*
  tail call void @free(i8* %9)
  %vec_mem_cast254 = bitcast i32* %vec_mem250 to %vec_t*
  store %vec_t* %vec_mem_cast254, %vec_t** %ptr_p246
  br label %pushback255

pushback255:                                      ; preds = %resize_merge244, %merge
  %elem_arr_p256 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 0
  %elem_arr_load257 = load %vec_t*, %vec_t** %elem_arr_p256
  %elem_arr_cast258 = bitcast %vec_t* %elem_arr_load257 to i32*
  %elem259 = getelementptr i32, i32* %elem_arr_cast258, i32 %size_var235
  store i32 3, i32* %elem259
  %new_size260 = add i32 %size_var235, 1
  store i32 %new_size260, i32* %size_p234
  %size_p261 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 1
  %size_var262 = load i32, i32* %size_p261
  %len_p263 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 2
  %leng_var264 = load i32, i32* %len_p263
  %size_var_cmp_leng265 = icmp slt i32 %size_var262, %leng_var264
  br i1 %size_var_cmp_leng265, label %pushback282, label %resize266

resize266:                                        ; preds = %pushback255
  %size_var_nonzero267 = icmp slt i32 %leng_var264, 1
  br i1 %size_var_nonzero267, label %resize_zero268, label %resize_nonzero269

resize_zero268:                                   ; preds = %resize266
```

```
  store i32 1, i32* %len_p263
  br label %resize_merge271

resize_nonzero269:                                ; preds = %resize266
  %shl270 = shl i32 %leng_var264, 1
  store i32 %shl270, i32* %len_p263
  br label %resize_merge271

resize_merge271:                                  ; preds = %resize_nonzero269, %resize_z
ero268
  %new_len_val272 = load i32, i32* %len_p263
  %ptr_p273 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 0
  %ptr_val274 = load %vec_t*, %vec_t** %ptr_p273
  %mallocsize275 = mul i32 %new_len_val272, ptrtoint (i32* getelementptr (i32, i32* null,
 i32 1) to i32)
  %malloccall276 = tail call i8* @malloc(i32 %mallocsize275)
  %vec_mem277 = bitcast i8* %malloccall276 to i32*
  %vec_mem_byte_cast278 = bitcast i32* %vec_mem277 to i8*
  %ptr_val_byte_cast279 = bitcast %vec_t* %ptr_val274 to i8*
  %sz_to_copy280 = mul i32 ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to i32),
%size_var262
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast278, i8* %ptr_val_byte_cast2
79, i32 %sz_to_copy280, i32 1, i1 true)
  %10 = bitcast %vec_t* %ptr_val274 to i8*
  tail call void @free(i8* %10)
  %vec_mem_cast281 = bitcast i32* %vec_mem277 to %vec_t*
  store %vec_t* %vec_mem_cast281, %vec_t** %ptr_p273
  br label %pushback282

pushback282:                                      ; preds = %resize_merge271, %pushback25
5
  %elem_arr_p283 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 0
  %elem_arr_load284 = load %vec_t*, %vec_t** %elem_arr_p283
  %elem_arr_cast285 = bitcast %vec_t* %elem_arr_load284 to i32*
  %elem286 = getelementptr i32, i32* %elem_arr_cast285, i32 %size_var262
  store i32 5, i32* %elem286
  %new_size287 = add i32 %size_var262, 1
  store i32 %new_size287, i32* %size_p261
  %size_p288 = getelementptr %vec_t, %vec_t* %vec_alloc194, i32 0, i32 1
  %size_var289 = load i32, i32* %size_p288
  %len_p290 = getelementptr %vec_t, %vec_t* %vec_alloc194, i32 0, i32 2
  %leng_var291 = load i32, i32* %len_p290
  %size_var_cmp_leng292 = icmp slt i32 %size_var289, %leng_var291
  br i1 %size_var_cmp_leng292, label %pushback308, label %resize293

resize293:                                        ; preds = %pushback282
  %size_var_nonzero294 = icmp slt i32 %leng_var291, 1
  br i1 %size_var_nonzero294, label %resize_zero295, label %resize_nonzero296

resize_zero295:                                   ; preds = %resize293
  store i32 1, i32* %len_p290
  br label %resize_merge298

resize_nonzero296:                                ; preds = %resize293
  %shl297 = shl i32 %leng_var291, 1
  store i32 %shl297, i32* %len_p290
  br label %resize_merge298

resize_merge298:                                  ; preds = %resize_nonzero296, %resize_z
ero295
  %new_len_val299 = load i32, i32* %len_p290
  %ptr_p300 = getelementptr %vec_t, %vec_t* %vec_alloc194, i32 0, i32 0
  %ptr_val301 = load %vec_t*, %vec_t** %ptr_p300
  %mallocsize302 = mul i32 %new_len_val299, ptrtoint (%vec_t* getelementptr (%vec_t, %vec
_t* null, i32 1) to i32)
  %malloccall303 = tail call i8* @malloc(i32 %mallocsize302)
  %vec_mem304 = bitcast i8* %malloccall303 to %vec_t*
  %vec_mem_byte_cast305 = bitcast %vec_t* %vec_mem304 to i8*
  %ptr_val_byte_cast306 = bitcast %vec_t* %ptr_val301 to i8*
  %sz_to_copy307 = mul i32 ptrtoint (%vec_t* getelementptr (%vec_t, %vec_t* null, i32 1)
```

```
to i32), %size_var289
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast305, i8* %ptr_val_byte_cast3
06, i32 %sz_to_copy307, i32 1, i1 true)
  %11 = bitcast %vec_t* %ptr_val301 to i8*
  tail call void @free(i8* %11)
  store %vec_t* %vec_mem304, %vec_t** %ptr_p300
  br label %pushback308

pushback308:                                      ; preds = %resize_merge298, %pushback28
2
  %vect_gep309 = getelementptr %vec_t, %vec_t* %vec_alloc194, i32 0, i32 0
  %vec_load310 = load %vec_t*, %vec_t** %vect_gep309
  %elem311 = getelementptr %vec_t, %vec_t* %vec_load310, i32 %size_var289
  %ptr_p312 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 0
  %size_p313 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 1
  %len_p314 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 2
  %ptr315 = load %vec_t*, %vec_t** %ptr_p312
  %size316 = load i32, i32* %size_p313
  %len317 = load i32, i32* %len_p314
  %out_vec318 = alloca %vec_t
  %out_ptr319 = getelementptr %vec_t, %vec_t* %out_vec318, i32 0, i32 0
  %out_size320 = getelementptr %vec_t, %vec_t* %out_vec318, i32 0, i32 1
  %out_len321 = getelementptr %vec_t, %vec_t* %out_vec318, i32 0, i32 2
  store i32 %size316, i32* %out_size320
  store i32 %len317, i32* %out_len321
  %iszerosize322 = icmp slt i32 %size316, 1
  br i1 %iszerosize322, label %zerosize323, label %nonzerosize324

zerosize323:                                      ; preds = %pushback308
  store %vec_t* null, %vec_t** %out_ptr319
  br label %merge332

nonzerosize324:                                   ; preds = %pushback308
  %mallocsize325 = mul i32 %len317, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
to i32)
  %malloccall326 = tail call i8* @malloc(i32 %mallocsize325)
  %newvec327 = bitcast i8* %malloccall326 to i32*
  %newvec_cast328 = bitcast i32* %newvec327 to %vec_t*
  store %vec_t* %newvec_cast328, %vec_t** %out_ptr319
  %src_cast329 = bitcast %vec_t* %ptr315 to i8*
  %dst_cast330 = bitcast i32* %newvec327 to i8*
  %sz_to_copy331 = mul i32 %size316, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
 to i32)
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast330, i8* %src_cast329, i32 %sz_to_cop
y331, i32 1, i1 true)
  br label %merge332

merge332:                                         ; preds = %nonzerosize324, %zerosize323
  %out_vec_val333 = load %vec_t, %vec_t* %out_vec318
  store %vec_t %out_vec_val333, %vec_t* %elem311
  %new_size334 = add i32 %size_var289, 1
  store i32 %new_size334, i32* %size_p288
  %j = alloca i32
  %a = load %vec_t, %vec_t* %vec_alloc
  %size335 = alloca i32
  store i32 0, i32* %size335
  %e_store = alloca %vec_t
  store %vec_t %a, %vec_t* %e_store
  %vec_sz_var = alloca i32
  store i32 0, i32* %vec_sz_var
  %vec_ptr = getelementptr %vec_t, %vec_t* %e_store, i32 0, i32 0
  %ptr336 = load %vec_t*, %vec_t** %vec_ptr
  %vec_sz_p = getelementptr %vec_t, %vec_t* %e_store, i32 0, i32 1
  %vec_sz = load i32, i32* %vec_sz_p
  %size_val = load i32, i32* %size335
  %size_plus4 = add i32 %size_val, 4
  store i32 %size_plus4, i32* %size335
  br label %check_size

check_size:                                       ; preds = %merge332, %del_vec_elem
```

```
  %iter_val = load i32, i32* %vec_sz_var
  %keeplooping = icmp slt i32 %iter_val, %vec_sz
  br i1 %keeplooping, label %del_vec_elem, label %merge339

del_vec_elem:                                     ; preds = %check_size
  %iter_val337 = load i32, i32* %vec_sz_var
  %dest_ptr = getelementptr %vec_t, %vec_t* %ptr336, i32 %iter_val337
  %origin_ptr_cast = bitcast %vec_t* %dest_ptr to i32*
  %inner_vec = load i32, i32* %origin_ptr_cast
  %size_val338 = load i32, i32* %size335
  %size_add = add i32 %size_val338, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
to i32)
  store i32 %size_add, i32* %size335
  %new_iter_val = add i32 %iter_val337, 1
  store i32 %new_iter_val, i32* %vec_sz_var
  br label %check_size

merge339:                                         ; preds = %check_size
  %serializelen = load i32, i32* %size335
  %serializebuf = alloca i8*
  %mallocsize340 = mul i32 %serializelen, ptrtoint (i8* getelementptr (i8, i8* null, i32
1) to i32)
  %serializearg = tail call i8* @malloc(i32 %mallocsize340)
  store i8* %serializearg, i8** %serializebuf
  %target = load i8*, i8** %serializebuf
  %size_32 = bitcast i8* %target to i32*
  %e_store342 = alloca %vec_t
  store %vec_t %a, %vec_t* %e_store342
  %vec_ptr343 = getelementptr %vec_t, %vec_t* %e_store342, i32 0, i32 0
  %ptr344 = load %vec_t*, %vec_t** %vec_ptr343
  %vec_sz_p345 = getelementptr %vec_t, %vec_t* %e_store342, i32 0, i32 1
  %vec_sz346 = load i32, i32* %vec_sz_p345
  store i32 %vec_sz346, i32* %size_32
  %next_target = getelementptr i8, i8* %target, i32 4
  store i8* %next_target, i8** %serializebuf
  %vec_sz_var347 = alloca i32
  store i32 0, i32* %vec_sz_var347
  br label %check_size348

check_size348:                                    ; preds = %merge339, %del_vec_elem351
  %iter_val349 = load i32, i32* %vec_sz_var347
  %keeplooping350 = icmp slt i32 %iter_val349, %vec_sz346
  br i1 %keeplooping350, label %del_vec_elem351, label %merge356

del_vec_elem351:                                  ; preds = %check_size348
  %iter_val352 = load i32, i32* %vec_sz_var347
  %ptr_v_cast = bitcast %vec_t* %ptr344 to i32*
  %dest_ptr353 = getelementptr i32, i32* %ptr_v_cast, i32 %iter_val352
  %inner_elem = load i32, i32* %dest_ptr353
  %target354 = load i8*, i8** %serializebuf
  %elem_nn = bitcast i8* %target354 to i32*
  store i32 %inner_elem, i32* %elem_nn
  %newtarget = getelementptr i8, i8* %target354, i32 ptrtoint (i32* getelementptr (i32, i
32* null, i32 1) to i32)
  store i8* %newtarget, i8** %serializebuf
  %new_iter_val355 = add i32 %iter_val352, 1
  store i32 %new_iter_val355, i32* %vec_sz_var347
  br label %check_size348

merge356:                                         ; preds = %check_size348
  %start_job = call i32 @start_job(i32 1, i8* %serializearg, i32 %serializelen)
  store i32 %start_job, i32* %j
  %k = alloca i32
  %d = load %vec_t, %vec_t* %vec_alloc109
  %size357 = alloca i32
  store i32 0, i32* %size357
  %e_store358 = alloca %vec_t
  store %vec_t %d, %vec_t* %e_store358
  %vec_sz_var359 = alloca i32
  store i32 0, i32* %vec_sz_var359
```

```
  %vec_ptr360 = getelementptr %vec_t, %vec_t* %e_store358, i32 0, i32 0
  %ptr361 = load %vec_t*, %vec_t** %vec_ptr360
  %vec_sz_p362 = getelementptr %vec_t, %vec_t* %e_store358, i32 0, i32 1
  %vec_sz363 = load i32, i32* %vec_sz_p362
  %size_val364 = load i32, i32* %size357
  %size_plus4365 = add i32 %size_val364, 4
  store i32 %size_plus4365, i32* %size357
  br label %check_size366

check_size366:                                    ; preds = %merge356, %del_vec_elem369
  %iter_val367 = load i32, i32* %vec_sz_var359
  %keeplooping368 = icmp slt i32 %iter_val367, %vec_sz363
  br i1 %keeplooping368, label %del_vec_elem369, label %merge377

del_vec_elem369:                                  ; preds = %check_size366
  %iter_val370 = load i32, i32* %vec_sz_var359
  %dest_ptr371 = getelementptr %vec_t, %vec_t* %ptr361, i32 %iter_val370
  %origin_ptr_cast372 = bitcast %vec_t* %dest_ptr371 to double*
  %inner_vec373 = load double, double* %origin_ptr_cast372
  %size_val374 = load i32, i32* %size357
  %size_add375 = add i32 %size_val374, ptrtoint (double* getelementptr (double, double* n
ull, i32 1) to i32)
  store i32 %size_add375, i32* %size357
  %new_iter_val376 = add i32 %iter_val370, 1
  store i32 %new_iter_val376, i32* %vec_sz_var359
  br label %check_size366

merge377:                                         ; preds = %check_size366
  %serializelen378 = load i32, i32* %size357
  %serializebuf379 = alloca i8*
  %mallocsize380 = mul i32 %serializelen378, ptrtoint (i8* getelementptr (i8, i8* null, i
32 1) to i32)
  %serializearg382 = tail call i8* @malloc(i32 %mallocsize380)
  store i8* %serializearg382, i8** %serializebuf379
  %target383 = load i8*, i8** %serializebuf379
  %size_32384 = bitcast i8* %target383 to i32*
  %e_store385 = alloca %vec_t
  store %vec_t %d, %vec_t* %e_store385
  %vec_ptr386 = getelementptr %vec_t, %vec_t* %e_store385, i32 0, i32 0
  %ptr387 = load %vec_t*, %vec_t** %vec_ptr386
  %vec_sz_p388 = getelementptr %vec_t, %vec_t* %e_store385, i32 0, i32 1
  %vec_sz389 = load i32, i32* %vec_sz_p388
  store i32 %vec_sz389, i32* %size_32384
  %next_target390 = getelementptr i8, i8* %target383, i32 4
  store i8* %next_target390, i8** %serializebuf379
  %vec_sz_var391 = alloca i32
  store i32 0, i32* %vec_sz_var391
  br label %check_size392

check_size392:                                    ; preds = %merge377, %del_vec_elem395
  %iter_val393 = load i32, i32* %vec_sz_var391
  %keeplooping394 = icmp slt i32 %iter_val393, %vec_sz389
  br i1 %keeplooping394, label %del_vec_elem395, label %merge404

del_vec_elem395:                                  ; preds = %check_size392
  %iter_val396 = load i32, i32* %vec_sz_var391
  %ptr_v_cast397 = bitcast %vec_t* %ptr387 to double*
  %dest_ptr398 = getelementptr double, double* %ptr_v_cast397, i32 %iter_val396
  %inner_elem399 = load double, double* %dest_ptr398
  %target400 = load i8*, i8** %serializebuf379
  %elem_nn401 = bitcast i8* %target400 to double*
  store double %inner_elem399, double* %elem_nn401
  %newtarget402 = getelementptr i8, i8* %target400, i32 ptrtoint (double* getelementptr (
double, double* null, i32 1) to i32)
  store i8* %newtarget402, i8** %serializebuf379
  %new_iter_val403 = add i32 %iter_val396, 1
  store i32 %new_iter_val403, i32* %vec_sz_var391
  br label %check_size392

merge404:                                         ; preds = %check_size392
```

```
  %start_job405 = call i32 @start_job(i32 2, i8* %serializearg382, i32 %serializelen378)
  store i32 %start_job405, i32* %k
  %l = alloca i32
  %aa = load %vec_t, %vec_t* %vec_alloc194
  %size406 = alloca i32
  store i32 0, i32* %size406
  %e_store407 = alloca %vec_t
  store %vec_t %aa, %vec_t* %e_store407
  %vec_sz_var408 = alloca i32
  store i32 0, i32* %vec_sz_var408
  %vec_ptr409 = getelementptr %vec_t, %vec_t* %e_store407, i32 0, i32 0
  %ptr410 = load %vec_t*, %vec_t** %vec_ptr409
  %vec_sz_p411 = getelementptr %vec_t, %vec_t* %e_store407, i32 0, i32 1
  %vec_sz412 = load i32, i32* %vec_sz_p411
  %size_val413 = load i32, i32* %size406
  %size_plus4414 = add i32 %size_val413, 4
  store i32 %size_plus4414, i32* %size406
  br label %check_size415

check_size415:                                    ; preds = %merge404, %merge441
  %iter_val416 = load i32, i32* %vec_sz_var408
  %keeplooping417 = icmp slt i32 %iter_val416, %vec_sz412
  br i1 %keeplooping417, label %del_vec_elem418, label %merge443

del_vec_elem418:                                  ; preds = %check_size415
  %iter_val419 = load i32, i32* %vec_sz_var408
  %dest_ptr420 = getelementptr %vec_t, %vec_t* %ptr410, i32 %iter_val419
  %inner_vec421 = load %vec_t, %vec_t* %dest_ptr420
  %e_store422 = alloca %vec_t
  store %vec_t %inner_vec421, %vec_t* %e_store422
  %vec_sz_var423 = alloca i32
  store i32 0, i32* %vec_sz_var423
  %vec_ptr424 = getelementptr %vec_t, %vec_t* %e_store422, i32 0, i32 0
  %ptr425 = load %vec_t*, %vec_t** %vec_ptr424
  %vec_sz_p426 = getelementptr %vec_t, %vec_t* %e_store422, i32 0, i32 1
  %vec_sz427 = load i32, i32* %vec_sz_p426
  %size_val428 = load i32, i32* %size406
  %size_plus4429 = add i32 %size_val428, 4
  store i32 %size_plus4429, i32* %size406
  br label %check_size430

check_size430:                                    ; preds = %del_vec_elem418, %del_vec_el
em433
  %iter_val431 = load i32, i32* %vec_sz_var423
  %keeplooping432 = icmp slt i32 %iter_val431, %vec_sz427
  br i1 %keeplooping432, label %del_vec_elem433, label %merge441

del_vec_elem433:                                  ; preds = %check_size430
  %iter_val434 = load i32, i32* %vec_sz_var423
  %dest_ptr435 = getelementptr %vec_t, %vec_t* %ptr425, i32 %iter_val434
  %origin_ptr_cast436 = bitcast %vec_t* %dest_ptr435 to i32*
  %inner_vec437 = load i32, i32* %origin_ptr_cast436
  %size_val438 = load i32, i32* %size406
  %size_add439 = add i32 %size_val438, ptrtoint (i32* getelementptr (i32, i32* null, i32
1) to i32)
  store i32 %size_add439, i32* %size406
  %new_iter_val440 = add i32 %iter_val434, 1
  store i32 %new_iter_val440, i32* %vec_sz_var423
  br label %check_size430

merge441:                                         ; preds = %check_size430
  %new_iter_val442 = add i32 %iter_val419, 1
  store i32 %new_iter_val442, i32* %vec_sz_var408
  br label %check_size415

merge443:                                         ; preds = %check_size415
  %serializelen444 = load i32, i32* %size406
  %serializebuf445 = alloca i8*
  %mallocsize446 = mul i32 %serializelen444, ptrtoint (i8* getelementptr (i8, i8* null, i
32 1) to i32)
```

```
  %serializearg448 = tail call i8* @malloc(i32 %mallocsize446)
  store i8* %serializearg448, i8** %serializebuf445
  %target449 = load i8*, i8** %serializebuf445
  %size_32450 = bitcast i8* %target449 to i32*
  %e_store451 = alloca %vec_t
  store %vec_t %aa, %vec_t* %e_store451
  %vec_ptr452 = getelementptr %vec_t, %vec_t* %e_store451, i32 0, i32 0
  %ptr453 = load %vec_t*, %vec_t** %vec_ptr452
  %vec_sz_p454 = getelementptr %vec_t, %vec_t* %e_store451, i32 0, i32 1
  %vec_sz455 = load i32, i32* %vec_sz_p454
  store i32 %vec_sz455, i32* %size_32450
  %next_target456 = getelementptr i8, i8* %target449, i32 4
  store i8* %next_target456, i8** %serializebuf445
  %vec_sz_var457 = alloca i32
  store i32 0, i32* %vec_sz_var457
  br label %check_size458

check_size458:                                    ; preds = %merge443, %merge486
  %iter_val459 = load i32, i32* %vec_sz_var457
  %keeplooping460 = icmp slt i32 %iter_val459, %vec_sz455
  br i1 %keeplooping460, label %del_vec_elem461, label %merge488

del_vec_elem461:                                  ; preds = %check_size458
  %iter_val462 = load i32, i32* %vec_sz_var457
  %dest_ptr463 = getelementptr %vec_t, %vec_t* %ptr453, i32 %iter_val462
  %inner_elem464 = load %vec_t, %vec_t* %dest_ptr463
  %target465 = load i8*, i8** %serializebuf445
  %size_32466 = bitcast i8* %target465 to i32*
  %e_store467 = alloca %vec_t
  store %vec_t %inner_elem464, %vec_t* %e_store467
  %vec_ptr468 = getelementptr %vec_t, %vec_t* %e_store467, i32 0, i32 0
  %ptr469 = load %vec_t*, %vec_t** %vec_ptr468
  %vec_sz_p470 = getelementptr %vec_t, %vec_t* %e_store467, i32 0, i32 1
  %vec_sz471 = load i32, i32* %vec_sz_p470
  store i32 %vec_sz471, i32* %size_32466
  %next_target472 = getelementptr i8, i8* %target465, i32 4
  store i8* %next_target472, i8** %serializebuf445
  %vec_sz_var473 = alloca i32
  store i32 0, i32* %vec_sz_var473
  br label %check_size474

check_size474:                                    ; preds = %del_vec_elem461, %del_vec_el
em477
  %iter_val475 = load i32, i32* %vec_sz_var473
  %keeplooping476 = icmp slt i32 %iter_val475, %vec_sz471
  br i1 %keeplooping476, label %del_vec_elem477, label %merge486

del_vec_elem477:                                  ; preds = %check_size474
  %iter_val478 = load i32, i32* %vec_sz_var473
  %ptr_v_cast479 = bitcast %vec_t* %ptr469 to i32*
  %dest_ptr480 = getelementptr i32, i32* %ptr_v_cast479, i32 %iter_val478
  %inner_elem481 = load i32, i32* %dest_ptr480
  %target482 = load i8*, i8** %serializebuf445
  %elem_nn483 = bitcast i8* %target482 to i32*
  store i32 %inner_elem481, i32* %elem_nn483
  %newtarget484 = getelementptr i8, i8* %target482, i32 ptrtoint (i32* getelementptr (i32
, i32* null, i32 1) to i32)
  store i8* %newtarget484, i8** %serializebuf445
  %new_iter_val485 = add i32 %iter_val478, 1
  store i32 %new_iter_val485, i32* %vec_sz_var473
  br label %check_size474

merge486:                                         ; preds = %check_size474
  %new_iter_val487 = add i32 %iter_val462, 1
  store i32 %new_iter_val487, i32* %vec_sz_var457
  br label %check_size458

merge488:                                         ; preds = %check_size458
  %start_job489 = call i32 @start_job(i32 3, i8* %serializearg448, i32 %serializelen444)
  store i32 %start_job489, i32* %l
```

```
  %j490 = load i32, i32* %j
  %out_data = alloca i8*
  %out_len491 = alloca i32
  call void @reap_job(i32 %j490, i8** %out_data, i32* %out_len491)
  %out_data_val = load i8*, i8** %out_data
  %ptr_val492 = load i8*, i8** %out_data
  %src = bitcast i8* %ptr_val492 to i32*
  %new_ptr_val = getelementptr i8, i8* %ptr_val492, i32 ptrtoint (i32* getelementptr (i32
, i32* null, i32 1) to i32)
  store i8* %new_ptr_val, i8** %out_data
  %deserialized_typ = load i32, i32* %src
  tail call void @free(i8* %out_data_val)
  %printf = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4 x i8]* @
fmt.18, i32 0, i32 0), i32 %deserialized_typ)
  %k493 = load i32, i32* %k
  %out_data494 = alloca i8*
  %out_len495 = alloca i32
  call void @reap_job(i32 %k493, i8** %out_data494, i32* %out_len495)
  %out_data_val496 = load i8*, i8** %out_data494
  %ptr_val497 = load i8*, i8** %out_data494
  %src498 = bitcast i8* %ptr_val497 to i32*
  %new_ptr_val499 = getelementptr i8, i8* %ptr_val497, i32 ptrtoint (i32* getelementptr (
i32, i32* null, i32 1) to i32)
  store i8* %new_ptr_val499, i8** %out_data494
  %deserialized_typ500 = load i32, i32* %src498
  tail call void @free(i8* %out_data_val496)
  %printf501 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4 x i8]
* @fmt.18, i32 0, i32 0), i32 %deserialized_typ500)
  %vec_alloc502 = alloca %vec_t
  %ptr503 = getelementptr %vec_t, %vec_t* %vec_alloc502, i32 0, i32 0
  store %vec_t* null, %vec_t** %ptr503
  %size504 = getelementptr %vec_t, %vec_t* %vec_alloc502, i32 0, i32 1
  store i32 0, i32* %size504
  %len505 = getelementptr %vec_t, %vec_t* %vec_alloc502, i32 0, i32 2
  store i32 0, i32* %len505
  %l506 = load i32, i32* %l
  %out_data507 = alloca i8*
  %out_len508 = alloca i32
  call void @reap_job(i32 %l506, i8** %out_data507, i32* %out_len508)
  %out_data_val509 = load i8*, i8** %out_data507
  %ptr_val510 = load i8*, i8** %out_data507
  %size511 = bitcast i8* %ptr_val510 to i32*
  %size_val512 = load i32, i32* %size511
  %iter = alloca i32
  store i32 0, i32* %iter
  %new_ptr = getelementptr i8, i8* %ptr_val510, i32 4
  store i8* %new_ptr, i8** %out_data507
  %vec_ptr513 = alloca %vec_t
  %vec_ptr_p = getelementptr %vec_t, %vec_t* %vec_ptr513, i32 0, i32 0
  %vec_ptr_sz = getelementptr %vec_t, %vec_t* %vec_ptr513, i32 0, i32 1
  %vec_ptr_ln = getelementptr %vec_t, %vec_t* %vec_ptr513, i32 0, i32 2
  %mallocsize514 = mul i32 %size_val512, ptrtoint (i32* getelementptr (i32, i32* null, i3
2 1) to i32)
  %malloccall515 = tail call i8* @malloc(i32 %mallocsize514)
  %buf = bitcast i8* %malloccall515 to i32*
  %buf_cast = bitcast i32* %buf to %vec_t*
  store %vec_t* %buf_cast, %vec_t** %vec_ptr_p
  store i32 %size_val512, i32* %vec_ptr_sz
  store i32 %size_val512, i32* %vec_ptr_ln
  br label %check_size516

check_size516:                                    ; preds = %merge488, %del_vec_elem519
  %iter_val517 = load i32, i32* %iter
  %keeplooping518 = icmp slt i32 %iter_val517, %size_val512
  br i1 %keeplooping518, label %del_vec_elem519, label %merge527

del_vec_elem519:                                  ; preds = %check_size516
  %iter_val520 = load i32, i32* %iter
  %dest_ptr521 = getelementptr i32, i32* %buf, i32 %iter_val520
  %ptr_val522 = load i8*, i8** %out_data507
```

```
  %src523 = bitcast i8* %ptr_val522 to i32*
  %new_ptr_val524 = getelementptr i8, i8* %ptr_val522, i32 ptrtoint (i32* getelementptr (
i32, i32* null, i32 1) to i32)
  store i8* %new_ptr_val524, i8** %out_data507
  %deserialized_typ525 = load i32, i32* %src523
  store i32 %deserialized_typ525, i32* %dest_ptr521
  %new_iter_val526 = add i32 %iter_val520, 1
  store i32 %new_iter_val526, i32* %iter
  br label %check_size516

merge527:                                         ; preds = %check_size516
  %vec = load %vec_t, %vec_t* %vec_ptr513
  tail call void @free(i8* %out_data_val509)
  %ptr_p528 = getelementptr %vec_t, %vec_t* %vec_alloc502, i32 0, i32 0
  %size_p529 = getelementptr %vec_t, %vec_t* %vec_alloc502, i32 0, i32 1
  %ptr530 = load %vec_t*, %vec_t** %ptr_p528
  %size531 = load i32, i32* %size_p529
  %iszerosize532 = icmp slt i32 %size531, 1
  br i1 %iszerosize532, label %merge533, label %del_nonzerosize

del_nonzerosize:                                  ; preds = %merge527
  %12 = bitcast %vec_t* %ptr530 to i8*
  tail call void @free(i8* %12)
  br label %merge533

merge533:                                         ; preds = %merge527, %del_nonzerosize
  store %vec_t %vec, %vec_t* %vec_alloc502
  %elem_arr_p534 = getelementptr %vec_t, %vec_t* %vec_alloc502, i32 0, i32 0
  %elem_arr_load535 = load %vec_t*, %vec_t** %elem_arr_p534
  %elem_arr_cast536 = bitcast %vec_t* %elem_arr_load535 to i32*
  %elem537 = getelementptr i32, i32* %elem_arr_cast536, i32 1
  %element = load i32, i32* %elem537
  %printf538 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4 x i8]
* @fmt.18, i32 0, i32 0), i32 %element)
  %sj = alloca i32
  %a539 = load %vec_t, %vec_t* %vec_alloc
  %size540 = alloca i32
  store i32 0, i32* %size540
  %e_store541 = alloca %vec_t
  store %vec_t %a539, %vec_t* %e_store541
  %vec_sz_var542 = alloca i32
  store i32 0, i32* %vec_sz_var542
  %vec_ptr543 = getelementptr %vec_t, %vec_t* %e_store541, i32 0, i32 0
  %ptr544 = load %vec_t*, %vec_t** %vec_ptr543
  %vec_sz_p545 = getelementptr %vec_t, %vec_t* %e_store541, i32 0, i32 1
  %vec_sz546 = load i32, i32* %vec_sz_p545
  %size_val547 = load i32, i32* %size540
  %size_plus4548 = add i32 %size_val547, 4
  store i32 %size_plus4548, i32* %size540
  br label %check_size549

check_size549:                                    ; preds = %merge533, %del_vec_elem552
  %iter_val550 = load i32, i32* %vec_sz_var542
  %keeplooping551 = icmp slt i32 %iter_val550, %vec_sz546
  br i1 %keeplooping551, label %del_vec_elem552, label %merge560

del_vec_elem552:                                  ; preds = %check_size549
  %iter_val553 = load i32, i32* %vec_sz_var542
  %dest_ptr554 = getelementptr %vec_t, %vec_t* %ptr544, i32 %iter_val553
  %origin_ptr_cast555 = bitcast %vec_t* %dest_ptr554 to i32*
  %inner_vec556 = load i32, i32* %origin_ptr_cast555
  %size_val557 = load i32, i32* %size540
  %size_add558 = add i32 %size_val557, ptrtoint (i32* getelementptr (i32, i32* null, i32
1) to i32)
  store i32 %size_add558, i32* %size540
  %new_iter_val559 = add i32 %iter_val553, 1
  store i32 %new_iter_val559, i32* %vec_sz_var542
  br label %check_size549

merge560:                                         ; preds = %check_size549
```

```
  %serializelen561 = load i32, i32* %size540
  %serializebuf562 = alloca i8*
  %mallocsize563 = mul i32 %serializelen561, ptrtoint (i8* getelementptr (i8, i8* null, i
32 1) to i32)
  %serializearg565 = tail call i8* @malloc(i32 %mallocsize563)
  store i8* %serializearg565, i8** %serializebuf562
  %target566 = load i8*, i8** %serializebuf562
  %size_32567 = bitcast i8* %target566 to i32*
  %e_store568 = alloca %vec_t
  store %vec_t %a539, %vec_t* %e_store568
  %vec_ptr569 = getelementptr %vec_t, %vec_t* %e_store568, i32 0, i32 0
  %ptr570 = load %vec_t*, %vec_t** %vec_ptr569
  %vec_sz_p571 = getelementptr %vec_t, %vec_t* %e_store568, i32 0, i32 1
  %vec_sz572 = load i32, i32* %vec_sz_p571
  store i32 %vec_sz572, i32* %size_32567
  %next_target573 = getelementptr i8, i8* %target566, i32 4
  store i8* %next_target573, i8** %serializebuf562
  %vec_sz_var574 = alloca i32
  store i32 0, i32* %vec_sz_var574
  br label %check_size575

check_size575:                                    ; preds = %merge560, %del_vec_elem578
  %iter_val576 = load i32, i32* %vec_sz_var574
  %keeplooping577 = icmp slt i32 %iter_val576, %vec_sz572
  br i1 %keeplooping577, label %del_vec_elem578, label %merge587

del_vec_elem578:                                  ; preds = %check_size575
  %iter_val579 = load i32, i32* %vec_sz_var574
  %ptr_v_cast580 = bitcast %vec_t* %ptr570 to i32*
  %dest_ptr581 = getelementptr i32, i32* %ptr_v_cast580, i32 %iter_val579
  %inner_elem582 = load i32, i32* %dest_ptr581
  %target583 = load i8*, i8** %serializebuf562
  %elem_nn584 = bitcast i8* %target583 to i32*
  store i32 %inner_elem582, i32* %elem_nn584
  %newtarget585 = getelementptr i8, i8* %target583, i32 ptrtoint (i32* getelementptr (i32
, i32* null, i32 1) to i32)
  store i8* %newtarget585, i8** %serializebuf562
  %new_iter_val586 = add i32 %iter_val579, 1
  store i32 %new_iter_val586, i32* %vec_sz_var574
  br label %check_size575

merge587:                                         ; preds = %check_size575
  %start_job588 = call i32 @start_job(i32 4, i8* %serializearg565, i32 %serializelen561)
  store i32 %start_job588, i32* %sj
  %sj589 = load i32, i32* %sj
  %out_data590 = alloca i8*
  %out_len591 = alloca i32
  call void @reap_job(i32 %sj589, i8** %out_data590, i32* %out_len591)
  %out_data_val592 = load i8*, i8** %out_data590
  %ptr_val593 = load i8*, i8** %out_data590
  %src594 = bitcast i8* %ptr_val593 to i32*
  %new_ptr_val595 = getelementptr i8, i8* %ptr_val593, i32 ptrtoint (i32* getelementptr (
i32, i32* null, i32 1) to i32)
  store i8* %new_ptr_val595, i8** %out_data590
  %deserialized_typ596 = load i32, i32* %src594
  tail call void @free(i8* %out_data_val592)
  %printf597 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4 x i8]
* @fmt.18, i32 0, i32 0), i32 %deserialized_typ596)
  br label %return_loc

del_nonzerosize603:                               ; preds = %return_loc
  %13 = bitcast %vec_t* %ptr600 to i8*
  tail call void @free(i8* %13)
  br label %merge604

merge604:                                         ; preds = %return_loc, %del_nonzerosize
603
  %ptr_p605 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 0
  %size_p606 = getelementptr %vec_t, %vec_t* %vec_alloc230, i32 0, i32 1
  %ptr607 = load %vec_t*, %vec_t** %ptr_p605
```

```
  %size608 = load i32, i32* %size_p606
  %iszerosize609 = icmp slt i32 %size608, 1
  br i1 %iszerosize609, label %merge611, label %del_nonzerosize610

del_nonzerosize610:                               ; preds = %merge604
  %14 = bitcast %vec_t* %ptr607 to i8*
  tail call void @free(i8* %14)
  br label %merge611

merge611:                                         ; preds = %merge604, %del_nonzerosize61
0
  %ptr_p612 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 0
  %size_p613 = getelementptr %vec_t, %vec_t* %vec_alloc109, i32 0, i32 1
  %ptr614 = load %vec_t*, %vec_t** %ptr_p612
  %size615 = load i32, i32* %size_p613
  %iszerosize616 = icmp slt i32 %size615, 1
  br i1 %iszerosize616, label %merge618, label %del_nonzerosize617

del_nonzerosize617:                               ; preds = %merge611
  %15 = bitcast %vec_t* %ptr614 to i8*
  tail call void @free(i8* %15)
  br label %merge618

merge618:                                         ; preds = %merge611, %del_nonzerosize61
7
  %ptr_p619 = getelementptr %vec_t, %vec_t* %vec_alloc194, i32 0, i32 0
  %size_p620 = getelementptr %vec_t, %vec_t* %vec_alloc194, i32 0, i32 1
  %ptr621 = load %vec_t*, %vec_t** %ptr_p619
  %size622 = load i32, i32* %size_p620
  %iter623 = alloca i32
  store i32 0, i32* %iter623
  %iszerosize624 = icmp slt i32 %size622, 1
  br i1 %iszerosize624, label %mergeagain, label %check_size625

check_size625:                                    ; preds = %merge637, %merge618
  %iter_val626 = load i32, i32* %iter623
  %keeplooping627 = icmp slt i32 %iter_val626, %size622
  br i1 %keeplooping627, label %del_vec_elem628, label %merge639

del_vec_elem628:                                  ; preds = %check_size625
  %iter_val629 = load i32, i32* %iter623
  %dest_ptr630 = getelementptr %vec_t, %vec_t* %ptr621, i32 %iter_val629
  %ptr_p631 = getelementptr %vec_t, %vec_t* %dest_ptr630, i32 0, i32 0
  %size_p632 = getelementptr %vec_t, %vec_t* %dest_ptr630, i32 0, i32 1
  %ptr633 = load %vec_t*, %vec_t** %ptr_p631
  %size634 = load i32, i32* %size_p632
  %iszerosize635 = icmp slt i32 %size634, 1
  br i1 %iszerosize635, label %merge637, label %del_nonzerosize636

del_nonzerosize636:                               ; preds = %del_vec_elem628
  %16 = bitcast %vec_t* %ptr633 to i8*
  tail call void @free(i8* %16)
  br label %merge637

merge637:                                         ; preds = %del_vec_elem628, %del_nonzer
osize636
  %new_iter_val638 = add i32 %iter_val629, 1
  store i32 %new_iter_val638, i32* %iter623
  br label %check_size625

merge639:                                         ; preds = %check_size625
  %17 = bitcast %vec_t* %ptr621 to i8*
  tail call void @free(i8* %17)
  br label %mergeagain

mergeagain:                                       ; preds = %merge639, %merge618
  %ptr_p640 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %size_p641 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 1
  %ptr642 = load %vec_t*, %vec_t** %ptr_p640
  %size643 = load i32, i32* %size_p641
```

```
  %iszerosize644 = icmp slt i32 %size643, 1
  br i1 %iszerosize644, label %merge646, label %del_nonzerosize645

del_nonzerosize645:                               ; preds = %mergeagain
  %18 = bitcast %vec_t* %ptr642 to i8*
  tail call void @free(i8* %18)
  br label %merge646

merge646:                                         ; preds = %mergeagain, %del_nonzerosize
645
  %final_retval = load i32, i32* %retval
  ret i32 %final_retval
}

declare void @free(i8*)

declare noalias i8* @malloc(i32)

define void @foo_job(i8* %input_data, i32 %input_leng, i8** %output_data, i32* %output_le
ng) {
entry:
  %input_store = alloca i8*
  store i8* %input_data, i8** %input_store
  %ptr_val = load i8*, i8** %input_store
  %size = bitcast i8* %ptr_val to i32*
  %size_val = load i32, i32* %size
  %iter = alloca i32
  store i32 0, i32* %iter
  %new_ptr = getelementptr i8, i8* %ptr_val, i32 4
  store i8* %new_ptr, i8** %input_store
  %vec_ptr = alloca %vec_t
  %vec_ptr_p = getelementptr %vec_t, %vec_t* %vec_ptr, i32 0, i32 0
  %vec_ptr_sz = getelementptr %vec_t, %vec_t* %vec_ptr, i32 0, i32 1
  %vec_ptr_ln = getelementptr %vec_t, %vec_t* %vec_ptr, i32 0, i32 2
  %mallocsize = mul i32 %size_val, ptrtoint (i32* getelementptr (i32, i32* null, i32 1) t
o i32)
  %malloccall = tail call i8* @malloc(i32 %mallocsize)
  %buf = bitcast i8* %malloccall to i32*
  %buf_cast = bitcast i32* %buf to %vec_t*
  store %vec_t* %buf_cast, %vec_t** %vec_ptr_p
  store i32 %size_val, i32* %vec_ptr_sz
  store i32 %size_val, i32* %vec_ptr_ln
  br label %check_size

check_size:                                       ; preds = %entry, %del_vec_elem
  %iter_val = load i32, i32* %iter
  %keeplooping = icmp slt i32 %iter_val, %size_val
  br i1 %keeplooping, label %del_vec_elem, label %merge

del_vec_elem:                                     ; preds = %check_size
  %iter_val1 = load i32, i32* %iter
  %dest_ptr = getelementptr i32, i32* %buf, i32 %iter_val1
  %ptr_val2 = load i8*, i8** %input_store
  %src = bitcast i8* %ptr_val2 to i32*
  %new_ptr_val = getelementptr i8, i8* %ptr_val2, i32 ptrtoint (i32* getelementptr (i32,
i32* null, i32 1) to i32)
  store i8* %new_ptr_val, i8** %input_store
  %deserialized_typ = load i32, i32* %src
  store i32 %deserialized_typ, i32* %dest_ptr
  %new_iter_val = add i32 %iter_val1, 1
  store i32 %new_iter_val, i32* %iter
  br label %check_size

merge:                                            ; preds = %check_size
  %vec = load %vec_t, %vec_t* %vec_ptr
  %foo_result = call i32 @foo(%vec_t %vec)
  %size3 = alloca i32
  store i32 0, i32* %size3
  %size_val4 = load i32, i32* %size3
  %size_add = add i32 %size_val4, ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to
```

```
 i32)
  store i32 %size_add, i32* %size3
  %serializelen = load i32, i32* %size3
  %serializebuf = alloca i8*
  %mallocsize5 = mul i32 %serializelen, ptrtoint (i8* getelementptr (i8, i8* null, i32 1)
 to i32)
  %serializearg = tail call i8* @malloc(i32 %mallocsize5)
  store i8* %serializearg, i8** %serializebuf
  %target = load i8*, i8** %serializebuf
  %elem_nn = bitcast i8* %target to i32*
  store i32 %foo_result, i32* %elem_nn
  %newtarget = getelementptr i8, i8* %target, i32 ptrtoint (i32* getelementptr (i32, i32*
 null, i32 1) to i32)
  store i8* %newtarget, i8** %serializebuf
  store i8* %serializearg, i8** %output_data
  store i32 %serializelen, i32* %output_leng
  ret void
}

define void @bar_job(i8* %input_data, i32 %input_leng, i8** %output_data, i32* %output_le
ng) {
entry:
  %input_store = alloca i8*
  store i8* %input_data, i8** %input_store
  %ptr_val = load i8*, i8** %input_store
  %size = bitcast i8* %ptr_val to i32*
  %size_val = load i32, i32* %size
  %iter = alloca i32
  store i32 0, i32* %iter
  %new_ptr = getelementptr i8, i8* %ptr_val, i32 4
  store i8* %new_ptr, i8** %input_store
  %vec_ptr = alloca %vec_t
  %vec_ptr_p = getelementptr %vec_t, %vec_t* %vec_ptr, i32 0, i32 0
  %vec_ptr_sz = getelementptr %vec_t, %vec_t* %vec_ptr, i32 0, i32 1
  %vec_ptr_ln = getelementptr %vec_t, %vec_t* %vec_ptr, i32 0, i32 2
  %mallocsize = mul i32 %size_val, ptrtoint (double* getelementptr (double, double* null,
 i32 1) to i32)
  %malloccall = tail call i8* @malloc(i32 %mallocsize)
  %buf = bitcast i8* %malloccall to double*
  %buf_cast = bitcast double* %buf to %vec_t*
  store %vec_t* %buf_cast, %vec_t** %vec_ptr_p
  store i32 %size_val, i32* %vec_ptr_sz
  store i32 %size_val, i32* %vec_ptr_ln
  br label %check_size

check_size:                                       ; preds = %entry, %del_vec_elem
  %iter_val = load i32, i32* %iter
  %keeplooping = icmp slt i32 %iter_val, %size_val
  br i1 %keeplooping, label %del_vec_elem, label %merge

del_vec_elem:                                     ; preds = %check_size
  %iter_val1 = load i32, i32* %iter
  %dest_ptr = getelementptr double, double* %buf, i32 %iter_val1
  %ptr_val2 = load i8*, i8** %input_store
  %src = bitcast i8* %ptr_val2 to double*
  %new_ptr_val = getelementptr i8, i8* %ptr_val2, i32 ptrtoint (double* getelementptr (do
uble, double* null, i32 1) to i32)
  store i8* %new_ptr_val, i8** %input_store
  %deserialized_typ = load double, double* %src
  store double %deserialized_typ, double* %dest_ptr
  %new_iter_val = add i32 %iter_val1, 1
  store i32 %new_iter_val, i32* %iter
  br label %check_size

merge:                                            ; preds = %check_size
  %vec = load %vec_t, %vec_t* %vec_ptr
  %bar_result = call i32 @bar(%vec_t %vec)
  %size3 = alloca i32
  store i32 0, i32* %size3
  %size_val4 = load i32, i32* %size3
```

```
  %size_add = add i32 %size_val4, ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to
 i32)
  store i32 %size_add, i32* %size3
  %serializelen = load i32, i32* %size3
  %serializebuf = alloca i8*
  %mallocsize5 = mul i32 %serializelen, ptrtoint (i8* getelementptr (i8, i8* null, i32 1)
 to i32)
  %serializearg = tail call i8* @malloc(i32 %mallocsize5)
  store i8* %serializearg, i8** %serializebuf
  %target = load i8*, i8** %serializebuf
  %elem_nn = bitcast i8* %target to i32*
  store i32 %bar_result, i32* %elem_nn
  %newtarget = getelementptr i8, i8* %target, i32 ptrtoint (i32* getelementptr (i32, i32*
 null, i32 1) to i32)
  store i8* %newtarget, i8** %serializebuf
  store i8* %serializearg, i8** %output_data
  store i32 %serializelen, i32* %output_leng
  ret void
}

define void @baz_job(i8* %input_data, i32 %input_leng, i8** %output_data, i32* %output_le
ng) {
entry:
  %input_store = alloca i8*
  store i8* %input_data, i8** %input_store
  %ptr_val = load i8*, i8** %input_store
  %size = bitcast i8* %ptr_val to i32*
  %size_val = load i32, i32* %size
  %iter = alloca i32
  store i32 0, i32* %iter
  %new_ptr = getelementptr i8, i8* %ptr_val, i32 4
  store i8* %new_ptr, i8** %input_store
  %vec_ptr = alloca %vec_t
  %vec_ptr_p = getelementptr %vec_t, %vec_t* %vec_ptr, i32 0, i32 0
  %vec_ptr_sz = getelementptr %vec_t, %vec_t* %vec_ptr, i32 0, i32 1
  %vec_ptr_ln = getelementptr %vec_t, %vec_t* %vec_ptr, i32 0, i32 2
  %mallocsize = mul i32 %size_val, ptrtoint (%vec_t* getelementptr (%vec_t, %vec_t* null,
 i32 1) to i32)
  %malloccall = tail call i8* @malloc(i32 %mallocsize)
  %buf = bitcast i8* %malloccall to %vec_t*
  store %vec_t* %buf, %vec_t** %vec_ptr_p
  store i32 %size_val, i32* %vec_ptr_sz
  store i32 %size_val, i32* %vec_ptr_ln
  br label %check_size

check_size:                                       ; preds = %entry, %merge
  %iter_val = load i32, i32* %iter
  %keeplooping = icmp slt i32 %iter_val, %size_val
  br i1 %keeplooping, label %del_vec_elem, label %merge22

del_vec_elem:                                     ; preds = %check_size
  %iter_val1 = load i32, i32* %iter
  %dest_ptr = getelementptr %vec_t, %vec_t* %buf, i32 %iter_val1
  %ptr_val2 = load i8*, i8** %input_store
  %size3 = bitcast i8* %ptr_val2 to i32*
  %size_val4 = load i32, i32* %size3
  %iter5 = alloca i32
  store i32 0, i32* %iter5
  %new_ptr6 = getelementptr i8, i8* %ptr_val2, i32 4
  store i8* %new_ptr6, i8** %input_store
  %vec_ptr7 = alloca %vec_t
  %vec_ptr_p8 = getelementptr %vec_t, %vec_t* %vec_ptr7, i32 0, i32 0
  %vec_ptr_sz9 = getelementptr %vec_t, %vec_t* %vec_ptr7, i32 0, i32 1
  %vec_ptr_ln10 = getelementptr %vec_t, %vec_t* %vec_ptr7, i32 0, i32 2
  %mallocsize11 = mul i32 %size_val4, ptrtoint (i32* getelementptr (i32, i32* null, i32 1
) to i32)
  %malloccall12 = tail call i8* @malloc(i32 %mallocsize11)
  %buf13 = bitcast i8* %malloccall12 to i32*
  %buf_cast = bitcast i32* %buf13 to %vec_t*
  store %vec_t* %buf_cast, %vec_t** %vec_ptr_p8
```

```
  store i32 %size_val4, i32* %vec_ptr_sz9
  store i32 %size_val4, i32* %vec_ptr_ln10
  br label %check_size14

check_size14:                                     ; preds = %del_vec_elem, %del_vec_elem1
7
  %iter_val15 = load i32, i32* %iter5
  %keeplooping16 = icmp slt i32 %iter_val15, %size_val4
  br i1 %keeplooping16, label %del_vec_elem17, label %merge

del_vec_elem17:                                   ; preds = %check_size14
  %iter_val18 = load i32, i32* %iter5
  %dest_ptr19 = getelementptr i32, i32* %buf13, i32 %iter_val18
  %ptr_val20 = load i8*, i8** %input_store
  %src = bitcast i8* %ptr_val20 to i32*
  %new_ptr_val = getelementptr i8, i8* %ptr_val20, i32 ptrtoint (i32* getelementptr (i32,
 i32* null, i32 1) to i32)
  store i8* %new_ptr_val, i8** %input_store
  %deserialized_typ = load i32, i32* %src
  store i32 %deserialized_typ, i32* %dest_ptr19
  %new_iter_val = add i32 %iter_val18, 1
  store i32 %new_iter_val, i32* %iter5
  br label %check_size14

merge:                                            ; preds = %check_size14
  %vec = load %vec_t, %vec_t* %vec_ptr7
  store %vec_t %vec, %vec_t* %dest_ptr
  %new_iter_val21 = add i32 %iter_val1, 1
  store i32 %new_iter_val21, i32* %iter
  br label %check_size

merge22:                                          ; preds = %check_size
  %vec23 = load %vec_t, %vec_t* %vec_ptr
  %baz_result = call %vec_t @baz(%vec_t %vec23)
  %size24 = alloca i32
  store i32 0, i32* %size24
  %e_store = alloca %vec_t
  store %vec_t %baz_result, %vec_t* %e_store
  %vec_sz_var = alloca i32
  store i32 0, i32* %vec_sz_var
  %vec_ptr25 = getelementptr %vec_t, %vec_t* %e_store, i32 0, i32 0
  %ptr = load %vec_t*, %vec_t** %vec_ptr25
  %vec_sz_p = getelementptr %vec_t, %vec_t* %e_store, i32 0, i32 1
  %vec_sz = load i32, i32* %vec_sz_p
  %size_val26 = load i32, i32* %size24
  %size_plus4 = add i32 %size_val26, 4
  store i32 %size_plus4, i32* %size24
  br label %check_size27

check_size27:                                     ; preds = %merge22, %del_vec_elem30
  %iter_val28 = load i32, i32* %vec_sz_var
  %keeplooping29 = icmp slt i32 %iter_val28, %vec_sz
  br i1 %keeplooping29, label %del_vec_elem30, label %merge35

del_vec_elem30:                                   ; preds = %check_size27
  %iter_val31 = load i32, i32* %vec_sz_var
  %dest_ptr32 = getelementptr %vec_t, %vec_t* %ptr, i32 %iter_val31
  %origin_ptr_cast = bitcast %vec_t* %dest_ptr32 to i32*
  %inner_vec = load i32, i32* %origin_ptr_cast
  %size_val33 = load i32, i32* %size24
  %size_add = add i32 %size_val33, ptrtoint (i32* getelementptr (i32, i32* null, i32 1) t
o i32)
  store i32 %size_add, i32* %size24
  %new_iter_val34 = add i32 %iter_val31, 1
  store i32 %new_iter_val34, i32* %vec_sz_var
  br label %check_size27

merge35:                                          ; preds = %check_size27
  %serializelen = load i32, i32* %size24
  %serializebuf = alloca i8*
```

```
  %mallocsize36 = mul i32 %serializelen, ptrtoint (i8* getelementptr (i8, i8* null, i32 1
) to i32)
  %serializearg = tail call i8* @malloc(i32 %mallocsize36)
  store i8* %serializearg, i8** %serializebuf
  %target = load i8*, i8** %serializebuf
  %size_32 = bitcast i8* %target to i32*
  %e_store38 = alloca %vec_t
  store %vec_t %baz_result, %vec_t* %e_store38
  %vec_ptr39 = getelementptr %vec_t, %vec_t* %e_store38, i32 0, i32 0
  %ptr40 = load %vec_t*, %vec_t** %vec_ptr39
  %vec_sz_p41 = getelementptr %vec_t, %vec_t* %e_store38, i32 0, i32 1
  %vec_sz42 = load i32, i32* %vec_sz_p41
  store i32 %vec_sz42, i32* %size_32
  %next_target = getelementptr i8, i8* %target, i32 4
  store i8* %next_target, i8** %serializebuf
  %vec_sz_var43 = alloca i32
  store i32 0, i32* %vec_sz_var43
  br label %check_size44

check_size44:                                     ; preds = %merge35, %del_vec_elem47
  %iter_val45 = load i32, i32* %vec_sz_var43
  %keeplooping46 = icmp slt i32 %iter_val45, %vec_sz42
  br i1 %keeplooping46, label %del_vec_elem47, label %merge52

del_vec_elem47:                                   ; preds = %check_size44
  %iter_val48 = load i32, i32* %vec_sz_var43
  %ptr_v_cast = bitcast %vec_t* %ptr40 to i32*
  %dest_ptr49 = getelementptr i32, i32* %ptr_v_cast, i32 %iter_val48
  %inner_elem = load i32, i32* %dest_ptr49
  %target50 = load i8*, i8** %serializebuf
  %elem_nn = bitcast i8* %target50 to i32*
  store i32 %inner_elem, i32* %elem_nn
  %newtarget = getelementptr i8, i8* %target50, i32 ptrtoint (i32* getelementptr (i32, i3
2* null, i32 1) to i32)
  store i8* %newtarget, i8** %serializebuf
  %new_iter_val51 = add i32 %iter_val48, 1
  store i32 %new_iter_val51, i32* %vec_sz_var43
  br label %check_size44

merge52:                                          ; preds = %check_size44
  store i8* %serializearg, i8** %output_data
  store i32 %serializelen, i32* %output_leng
  %vec_store = alloca %vec_t
  store %vec_t %baz_result, %vec_t* %vec_store
  %ptr_p = getelementptr %vec_t, %vec_t* %vec_store, i32 0, i32 0
  %size_p = getelementptr %vec_t, %vec_t* %vec_store, i32 0, i32 1
  %ptr53 = load %vec_t*, %vec_t** %ptr_p
  %size54 = load i32, i32* %size_p
  %iszerosize = icmp slt i32 %size54, 1
  br i1 %iszerosize, label %merge55, label %del_nonzerosize

del_nonzerosize:                                  ; preds = %merge52
  %0 = bitcast %vec_t* %ptr53 to i8*
  tail call void @free(i8* %0)
  br label %merge55

merge55:                                          ; preds = %merge52, %del_nonzerosize
  ret void
}

define void @vecsum_job(i8* %input_data, i32 %input_leng, i8** %output_data, i32* %output
_leng) {
entry:
  %input_store = alloca i8*
  store i8* %input_data, i8** %input_store
  %ptr_val = load i8*, i8** %input_store
  %size = bitcast i8* %ptr_val to i32*
  %size_val = load i32, i32* %size
  %iter = alloca i32
  store i32 0, i32* %iter
```

```
  %new_ptr = getelementptr i8, i8* %ptr_val, i32 4
  store i8* %new_ptr, i8** %input_store
  %vec_ptr = alloca %vec_t
  %vec_ptr_p = getelementptr %vec_t, %vec_t* %vec_ptr, i32 0, i32 0
  %vec_ptr_sz = getelementptr %vec_t, %vec_t* %vec_ptr, i32 0, i32 1
  %vec_ptr_ln = getelementptr %vec_t, %vec_t* %vec_ptr, i32 0, i32 2
  %mallocsize = mul i32 %size_val, ptrtoint (i32* getelementptr (i32, i32* null, i32 1) t
o i32)
  %malloccall = tail call i8* @malloc(i32 %mallocsize)
  %buf = bitcast i8* %malloccall to i32*
  %buf_cast = bitcast i32* %buf to %vec_t*
  store %vec_t* %buf_cast, %vec_t** %vec_ptr_p
  store i32 %size_val, i32* %vec_ptr_sz
  store i32 %size_val, i32* %vec_ptr_ln
  br label %check_size

check_size:                                       ; preds = %entry, %del_vec_elem
  %iter_val = load i32, i32* %iter
  %keeplooping = icmp slt i32 %iter_val, %size_val
  br i1 %keeplooping, label %del_vec_elem, label %merge

del_vec_elem:                                     ; preds = %check_size
  %iter_val1 = load i32, i32* %iter
  %dest_ptr = getelementptr i32, i32* %buf, i32 %iter_val1
  %ptr_val2 = load i8*, i8** %input_store
  %src = bitcast i8* %ptr_val2 to i32*
  %new_ptr_val = getelementptr i8, i8* %ptr_val2, i32 ptrtoint (i32* getelementptr (i32,
i32* null, i32 1) to i32)
  store i8* %new_ptr_val, i8** %input_store
  %deserialized_typ = load i32, i32* %src
  store i32 %deserialized_typ, i32* %dest_ptr
  %new_iter_val = add i32 %iter_val1, 1
  store i32 %new_iter_val, i32* %iter
  br label %check_size

merge:                                            ; preds = %check_size
  %vec = load %vec_t, %vec_t* %vec_ptr
  %vecsum_result = call i32 @vecsum(%vec_t %vec)
  %size3 = alloca i32
  store i32 0, i32* %size3
  %size_val4 = load i32, i32* %size3
  %size_add = add i32 %size_val4, ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to
 i32)
  store i32 %size_add, i32* %size3
  %serializelen = load i32, i32* %size3
  %serializebuf = alloca i8*
  %mallocsize5 = mul i32 %serializelen, ptrtoint (i8* getelementptr (i8, i8* null, i32 1)
 to i32)
  %serializearg = tail call i8* @malloc(i32 %mallocsize5)
  store i8* %serializearg, i8** %serializebuf
  %target = load i8*, i8** %serializebuf
  %elem_nn = bitcast i8* %target to i32*
  store i32 %vecsum_result, i32* %elem_nn
  %newtarget = getelementptr i8, i8* %target, i32 ptrtoint (i32* getelementptr (i32, i32*
 null, i32 1) to i32)
  store i8* %newtarget, i8** %serializebuf
  store i8* %serializearg, i8** %output_data
  store i32 %serializelen, i32* %output_leng
  ret void
}

attributes #0 = { argmemonly nounwind }
```

```
master
{

    vector<struct example_inner_inner> test4; //1


    struct example_inner_inner a4;
    a4->e = 1;
    vector<int> tmp4;
    tmp4::1;
    a4->very_nasty = tmp4; //copied
    test4::a4; //copied

    vector<struct example_inner_inner> test14;
    test14 = test4; //copied
    tmp4[0]=2;
    test14[0]->very_nasty = tmp4;

    tmp4 = test14[0]->very_nasty;
    print(tmp4[0]); //2
    tmp4 = test4[0]->very_nasty;
    print(tmp4[0]); //1



    vector<struct example_inner> test; //1
    struct example_inner a1;
    struct example_inner_inner a;
    a->e = 1;
    vector<int> tmp; //2
    tmp::1;
    a->very_nasty = tmp;
    a1->inner = a;
    test::a1; //copied

    vector<struct example_inner> test1;
    test1 = test; //copied
    tmp[0] = 2;
    test1[0]->inner->very_nasty = tmp; //copied
    tmp = test1[0]->inner->very_nasty;
    print(tmp[0]);
    tmp = test[0]->inner->very_nasty;
    print(tmp[0]);

}

struct simple {
    int e;
};

struct veccy {
    int e;
    //vector<int> v;
    struct simple e;
};

struct example_inner_inner {
    int e;
    vector<int> very_nasty;
};

struct example_inner {
    int e;
    struct example_inner_inner inner;
};

struct example {
    int e;
    int c;
    struct example_inner nasty;
```

```
};
```

```
; ModuleID = 'M/s'
source_filename = "M/s"

%vec_t = type { %vec_t*, i32, i32 }
%example_inner_inner = type { i32, %vec_t }
%example_inner = type { i32, %example_inner_inner }

@fmt = private unnamed_addr constant [4 x i8] c"%d\0A\00"
@fmt.1 = private unnamed_addr constant [4 x i8] c"%f\0A\00"
@true_str = private unnamed_addr constant [6 x i8] c"true\0A\00"
@false_str = private unnamed_addr constant [7 x i8] c"false\0A\00"
@false_str.2 = private unnamed_addr constant [4 x i8] c"%s\0A\00"
@job_funcs = global [0 x void (i8*, i32, i8**, i32*)*] zeroinitializer
@num_jobs = global i32 0

declare i32 @printf(i8*, ...)

declare i32 @start_job(i32, i8*, i32)

declare void @reap_job(i32, i8**, i32*)

declare i32 @get_job_status(i32)

declare i32 @cancel_job(i32)

; Function Attrs: argmemonly nounwind
declare void @llvm.memcpy.p0i8.p0i8.i32(i8* nocapture writeonly, i8* nocapture readonly,
i32, i32, i1) #0

define i32 @master() {
entry:
  %retval = alloca i32
  store i32 0, i32* %retval
  %vec_alloc = alloca %vec_t
  %ptr = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  store %vec_t* null, %vec_t** %ptr
  %size = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 1
  store i32 0, i32* %size
  %len = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 2
  store i32 0, i32* %len
  %a4 = alloca %example_inner_inner
  store %example_inner_inner zeroinitializer, %example_inner_inner* %a4
  %struct_field = getelementptr inbounds %example_inner_inner, %example_inner_inner* %a4,
 i32 0, i32 0
  store i32 1, i32* %struct_field
  %vec_alloc1 = alloca %vec_t
  %ptr2 = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 0
  store %vec_t* null, %vec_t** %ptr2
  %size3 = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 1
  store i32 0, i32* %size3
  %len4 = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 2
  store i32 0, i32* %len4
  %size_p = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 1
  %size_var = load i32, i32* %size_p
  %len_p = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 2
  %leng_var = load i32, i32* %len_p
  %size_var_cmp_leng = icmp slt i32 %size_var, %leng_var
  br i1 %size_var_cmp_leng, label %pushback, label %resize

return_loc:                                       ; preds = %merge675
  %ptr_p690 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %size_p691 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 1
  %ptr692 = load %vec_t*, %vec_t** %ptr_p690
  %size693 = load i32, i32* %size_p691
  %iter694 = alloca i32
  store i32 0, i32* %iter694
  %iszerosize695 = icmp slt i32 %size693, 1
  br i1 %iszerosize695, label %mergeagain713, label %check_size696

resize:                                           ; preds = %entry
```

```
  %size_var_nonzero = icmp slt i32 %leng_var, 1
  br i1 %size_var_nonzero, label %resize_zero, label %resize_nonzero

resize_zero:                                      ; preds = %resize
  store i32 1, i32* %len_p
  br label %resize_merge

resize_nonzero:                                   ; preds = %resize
  %shl = shl i32 %leng_var, 1
  store i32 %shl, i32* %len_p
  br label %resize_merge

resize_merge:                                     ; preds = %resize_nonzero, %resize_zero
  %new_len_val = load i32, i32* %len_p
  %ptr_p = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 0
  %ptr_val = load %vec_t*, %vec_t** %ptr_p
  %mallocsize = mul i32 %new_len_val, ptrtoint (i32* getelementptr (i32, i32* null, i32 1
) to i32)
  %malloccall = tail call i8* @malloc(i32 %mallocsize)
  %vec_mem = bitcast i8* %malloccall to i32*
  %vec_mem_byte_cast = bitcast i32* %vec_mem to i8*
  %ptr_val_byte_cast = bitcast %vec_t* %ptr_val to i8*
  %sz_to_copy = mul i32 ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to i32), %si
ze_var
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast, i8* %ptr_val_byte_cast, i3
2 %sz_to_copy, i32 1, i1 true)
  %0 = bitcast %vec_t* %ptr_val to i8*
  tail call void @free(i8* %0)
  %vec_mem_cast = bitcast i32* %vec_mem to %vec_t*
  store %vec_t* %vec_mem_cast, %vec_t** %ptr_p
  br label %pushback

pushback:                                         ; preds = %resize_merge, %entry
  %elem_arr_p = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 0
  %elem_arr_load = load %vec_t*, %vec_t** %elem_arr_p
  %elem_arr_cast = bitcast %vec_t* %elem_arr_load to i32*
  %elem = getelementptr i32, i32* %elem_arr_cast, i32 %size_var
  store i32 1, i32* %elem
  %new_size = add i32 %size_var, 1
  store i32 %new_size, i32* %size_p
  %tmp4 = load %vec_t, %vec_t* %vec_alloc1
  %struct_field5 = getelementptr inbounds %example_inner_inner, %example_inner_inner* %a4
, i32 0, i32 1
  %ptr_p6 = getelementptr %vec_t, %vec_t* %struct_field5, i32 0, i32 0
  %size_p7 = getelementptr %vec_t, %vec_t* %struct_field5, i32 0, i32 1
  %ptr8 = load %vec_t*, %vec_t** %ptr_p6
  %size9 = load i32, i32* %size_p7
  %iszerosize = icmp slt i32 %size9, 1
  br i1 %iszerosize, label %merge, label %del_nonzerosize

del_nonzerosize:                                  ; preds = %pushback
  %1 = bitcast %vec_t* %ptr8 to i8*
  tail call void @free(i8* %1)
  br label %merge

merge:                                            ; preds = %pushback, %del_nonzerosize
  %temp = alloca %vec_t
  store %vec_t %tmp4, %vec_t* %temp
  %ptr_p10 = getelementptr %vec_t, %vec_t* %temp, i32 0, i32 0
  %size_p11 = getelementptr %vec_t, %vec_t* %temp, i32 0, i32 1
  %len_p12 = getelementptr %vec_t, %vec_t* %temp, i32 0, i32 2
  %ptr13 = load %vec_t*, %vec_t** %ptr_p10
  %size14 = load i32, i32* %size_p11
  %len15 = load i32, i32* %len_p12
  %out_vec = alloca %vec_t
  %out_ptr = getelementptr %vec_t, %vec_t* %out_vec, i32 0, i32 0
  %out_size = getelementptr %vec_t, %vec_t* %out_vec, i32 0, i32 1
  %out_len = getelementptr %vec_t, %vec_t* %out_vec, i32 0, i32 2
  store i32 %size14, i32* %out_size
  store i32 %len15, i32* %out_len
```

```
  %iszerosize16 = icmp slt i32 %size14, 1
  br i1 %iszerosize16, label %zerosize, label %nonzerosize

zerosize:                                         ; preds = %merge
  store %vec_t* null, %vec_t** %out_ptr
  br label %merge20

nonzerosize:                                      ; preds = %merge
  %mallocsize17 = mul i32 %len15, ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to
 i32)
  %malloccall18 = tail call i8* @malloc(i32 %mallocsize17)
  %newvec = bitcast i8* %malloccall18 to i32*
  %newvec_cast = bitcast i32* %newvec to %vec_t*
  store %vec_t* %newvec_cast, %vec_t** %out_ptr
  %src_cast = bitcast %vec_t* %ptr13 to i8*
  %dst_cast = bitcast i32* %newvec to i8*
  %sz_to_copy19 = mul i32 %size14, ptrtoint (i32* getelementptr (i32, i32* null, i32 1) t
o i32)
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast, i8* %src_cast, i32 %sz_to_copy19, i
32 1, i1 true)
  br label %merge20

merge20:                                          ; preds = %nonzerosize, %zerosize
  %out_vec_val = load %vec_t, %vec_t* %out_vec
  store %vec_t %out_vec_val, %vec_t* %struct_field5
  %new_struct = alloca %example_inner_inner
  %l_field_cp_assign = getelementptr inbounds %example_inner_inner, %example_inner_inner*
 %new_struct, i32 0, i32 0
  %r_field_cp_assign = getelementptr inbounds %example_inner_inner, %example_inner_inner*
 %a4, i32 0, i32 0
  %loaded_rptr = load i32, i32* %r_field_cp_assign
  store i32 %loaded_rptr, i32* %l_field_cp_assign
  %l_field_vec_cp_assign = getelementptr inbounds %example_inner_inner, %example_inner_in
ner* %new_struct, i32 0, i32 1
  %r_field_vec_cp_assign = getelementptr inbounds %example_inner_inner, %example_inner_in
ner* %a4, i32 0, i32 1
  %loaded_rptr21 = load %vec_t, %vec_t* %r_field_vec_cp_assign
  %temp22 = alloca %vec_t
  store %vec_t %loaded_rptr21, %vec_t* %temp22
  %ptr_p23 = getelementptr %vec_t, %vec_t* %temp22, i32 0, i32 0
  %size_p24 = getelementptr %vec_t, %vec_t* %temp22, i32 0, i32 1
  %len_p25 = getelementptr %vec_t, %vec_t* %temp22, i32 0, i32 2
  %ptr26 = load %vec_t*, %vec_t** %ptr_p23
  %size27 = load i32, i32* %size_p24
  %len28 = load i32, i32* %len_p25
  %out_vec29 = alloca %vec_t
  %out_ptr30 = getelementptr %vec_t, %vec_t* %out_vec29, i32 0, i32 0
  %out_size31 = getelementptr %vec_t, %vec_t* %out_vec29, i32 0, i32 1
  %out_len32 = getelementptr %vec_t, %vec_t* %out_vec29, i32 0, i32 2
  store i32 %size27, i32* %out_size31
  store i32 %len28, i32* %out_len32
  %iszerosize33 = icmp slt i32 %size27, 1
  br i1 %iszerosize33, label %zerosize34, label %nonzerosize35

zerosize34:                                       ; preds = %merge20
  store %vec_t* null, %vec_t** %out_ptr30
  br label %merge43

nonzerosize35:                                    ; preds = %merge20
  %mallocsize36 = mul i32 %len28, ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to
 i32)
  %malloccall37 = tail call i8* @malloc(i32 %mallocsize36)
  %newvec38 = bitcast i8* %malloccall37 to i32*
  %newvec_cast39 = bitcast i32* %newvec38 to %vec_t*
  store %vec_t* %newvec_cast39, %vec_t** %out_ptr30
  %src_cast40 = bitcast %vec_t* %ptr26 to i8*
  %dst_cast41 = bitcast i32* %newvec38 to i8*
  %sz_to_copy42 = mul i32 %size27, ptrtoint (i32* getelementptr (i32, i32* null, i32 1) t
o i32)
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast41, i8* %src_cast40, i32 %sz_to_copy4
```

```
2, i32 1, i1 true)
  br label %merge43

merge43:                                          ; preds = %nonzerosize35, %zerosize34
  %out_vec_val44 = load %vec_t, %vec_t* %out_vec29
  %temp45 = alloca %vec_t
  store %vec_t %out_vec_val44, %vec_t* %temp45
  %src_cast46 = bitcast %vec_t* %temp45 to i8*
  %dst_cast47 = bitcast %vec_t* %l_field_vec_cp_assign to i8*
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast47, i8* %src_cast46, i32 ptrtoint (%v
ec_t* getelementptr (%vec_t, %vec_t* null, i32 1) to i32), i32 1, i1 true)
  %new_struct_val = load %example_inner_inner, %example_inner_inner* %new_struct
  %size_p48 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 1
  %size_var49 = load i32, i32* %size_p48
  %len_p50 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 2
  %leng_var51 = load i32, i32* %len_p50
  %size_var_cmp_leng52 = icmp slt i32 %size_var49, %leng_var51
  br i1 %size_var_cmp_leng52, label %pushback69, label %resize53

resize53:                                         ; preds = %merge43
  %size_var_nonzero54 = icmp slt i32 %leng_var51, 1
  br i1 %size_var_nonzero54, label %resize_zero55, label %resize_nonzero56

resize_zero55:                                    ; preds = %resize53
  store i32 1, i32* %len_p50
  br label %resize_merge58

resize_nonzero56:                                 ; preds = %resize53
  %shl57 = shl i32 %leng_var51, 1
  store i32 %shl57, i32* %len_p50
  br label %resize_merge58

resize_merge58:                                   ; preds = %resize_nonzero56, %resize_ze
ro55
  %new_len_val59 = load i32, i32* %len_p50
  %ptr_p60 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %ptr_val61 = load %vec_t*, %vec_t** %ptr_p60
  %mallocsize62 = mul i32 %new_len_val59, ptrtoint (%example_inner_inner* getelementptr (
%example_inner_inner, %example_inner_inner* null, i32 1) to i32)
  %malloccall63 = tail call i8* @malloc(i32 %mallocsize62)
  %vec_mem64 = bitcast i8* %malloccall63 to %example_inner_inner*
  %vec_mem_byte_cast65 = bitcast %example_inner_inner* %vec_mem64 to i8*
  %ptr_val_byte_cast66 = bitcast %vec_t* %ptr_val61 to i8*
  %sz_to_copy67 = mul i32 ptrtoint (%example_inner_inner* getelementptr (%example_inner_i
nner, %example_inner_inner* null, i32 1) to i32), %size_var49
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast65, i8* %ptr_val_byte_cast66
, i32 %sz_to_copy67, i32 1, i1 true)
  %2 = bitcast %vec_t* %ptr_val61 to i8*
  tail call void @free(i8* %2)
  %vec_mem_cast68 = bitcast %example_inner_inner* %vec_mem64 to %vec_t*
  store %vec_t* %vec_mem_cast68, %vec_t** %ptr_p60
  br label %pushback69

pushback69:                                       ; preds = %resize_merge58, %merge43
  %elem_arr_p70 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %elem_arr_load71 = load %vec_t*, %vec_t** %elem_arr_p70
  %elem_arr_cast72 = bitcast %vec_t* %elem_arr_load71 to %example_inner_inner*
  %elem73 = getelementptr %example_inner_inner, %example_inner_inner* %elem_arr_cast72, i
32 %size_var49
  store %example_inner_inner %new_struct_val, %example_inner_inner* %elem73
  %new_size74 = add i32 %size_var49, 1
  store i32 %new_size74, i32* %size_p48
  %vec_alloc75 = alloca %vec_t
  %ptr76 = getelementptr %vec_t, %vec_t* %vec_alloc75, i32 0, i32 0
  store %vec_t* null, %vec_t** %ptr76
  %size77 = getelementptr %vec_t, %vec_t* %vec_alloc75, i32 0, i32 1
  store i32 0, i32* %size77
  %len78 = getelementptr %vec_t, %vec_t* %vec_alloc75, i32 0, i32 2
  store i32 0, i32* %len78
  %ptr_p79 = getelementptr %vec_t, %vec_t* %vec_alloc75, i32 0, i32 0
```

```
  %size_p80 = getelementptr %vec_t, %vec_t* %vec_alloc75, i32 0, i32 1
  %ptr81 = load %vec_t*, %vec_t** %ptr_p79
  %size82 = load i32, i32* %size_p80
  %iter = alloca i32
  store i32 0, i32* %iter
  %iszerosize83 = icmp slt i32 %size82, 1
  br i1 %iszerosize83, label %mergeagain, label %check_size

check_size:                                       ; preds = %merge91, %pushback69
  %iter_val = load i32, i32* %iter
  %keeplooping = icmp slt i32 %iter_val, %size82
  br i1 %keeplooping, label %del_vec_elem, label %merge92

del_vec_elem:                                     ; preds = %check_size
  %iter_val84 = load i32, i32* %iter
  %dest_ptr = getelementptr %vec_t, %vec_t* %ptr81, i32 %iter_val84
  %struct_p = bitcast %vec_t* %dest_ptr to %example_inner_inner*
  %vec_ptr_delete = getelementptr inbounds %example_inner_inner, %example_inner_inner* %s
truct_p, i32 0, i32 1
  %ptr_p85 = getelementptr %vec_t, %vec_t* %vec_ptr_delete, i32 0, i32 0
  %size_p86 = getelementptr %vec_t, %vec_t* %vec_ptr_delete, i32 0, i32 1
  %ptr87 = load %vec_t*, %vec_t** %ptr_p85
  %size88 = load i32, i32* %size_p86
  %iszerosize89 = icmp slt i32 %size88, 1
  br i1 %iszerosize89, label %merge91, label %del_nonzerosize90

del_nonzerosize90:                                ; preds = %del_vec_elem
  %3 = bitcast %vec_t* %ptr87 to i8*
  tail call void @free(i8* %3)
  br label %merge91

merge91:                                          ; preds = %del_vec_elem, %del_nonzerosi
ze90
  %new_iter_val = add i32 %iter_val84, 1
  store i32 %new_iter_val, i32* %iter
  br label %check_size

merge92:                                          ; preds = %check_size
  %4 = bitcast %vec_t* %ptr81 to i8*
  tail call void @free(i8* %4)
  br label %mergeagain

mergeagain:                                       ; preds = %merge92, %pushback69
  %ptr_p93 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %size_p94 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 1
  %len_p95 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 2
  %ptr96 = load %vec_t*, %vec_t** %ptr_p93
  %size97 = load i32, i32* %size_p94
  %len98 = load i32, i32* %len_p95
  %iter99 = alloca i32
  store i32 0, i32* %iter99
  %out_vec100 = alloca %vec_t
  %out_ptr101 = getelementptr %vec_t, %vec_t* %out_vec100, i32 0, i32 0
  %out_size102 = getelementptr %vec_t, %vec_t* %out_vec100, i32 0, i32 1
  %out_len103 = getelementptr %vec_t, %vec_t* %out_vec100, i32 0, i32 2
  store i32 %size97, i32* %out_size102
  store i32 %len98, i32* %out_len103
  %iszerosize104 = icmp slt i32 %size97, 1
  br i1 %iszerosize104, label %zerosize105, label %nonzerosize106

zerosize105:                                      ; preds = %mergeagain
  store %vec_t* null, %vec_t** %out_ptr101
  br label %check_size111

nonzerosize106:                                   ; preds = %mergeagain
  %mallocsize107 = mul i32 %len98, ptrtoint (%example_inner_inner* getelementptr (%exampl
e_inner_inner, %example_inner_inner* null, i32 1) to i32)
  %malloccall108 = tail call i8* @malloc(i32 %mallocsize107)
  %newvec109 = bitcast i8* %malloccall108 to %example_inner_inner*
  %newvec_cast110 = bitcast %example_inner_inner* %newvec109 to %vec_t*
```

```
  store %vec_t* %newvec_cast110, %vec_t** %out_ptr101
  br label %check_size111

check_size111:                                    ; preds = %merge144, %nonzerosize106, %
zerosize105
  %iter_val112 = load i32, i32* %iter99
  %keeplooping113 = icmp slt i32 %iter_val112, %size97
  br i1 %keeplooping113, label %copy_vec_elem, label %merge150

copy_vec_elem:                                    ; preds = %check_size111
  %iter_val114 = load i32, i32* %iter99
  %vec_ptr = load %vec_t*, %vec_t** %out_ptr101
  %struct_vec_ptr = bitcast %vec_t* %vec_ptr to %example_inner_inner*
  %struct_vec_ptr115 = bitcast %vec_t* %ptr96 to %example_inner_inner*
  %dest_ptr116 = getelementptr %example_inner_inner, %example_inner_inner* %struct_vec_pt
r, i32 %iter_val114
  %origin_ptr = getelementptr %example_inner_inner, %example_inner_inner* %struct_vec_ptr
115, i32 %iter_val114
  %l_field_cp_assign117 = getelementptr inbounds %example_inner_inner, %example_inner_inn
er* %dest_ptr116, i32 0, i32 0
  %r_field_cp_assign118 = getelementptr inbounds %example_inner_inner, %example_inner_inn
er* %origin_ptr, i32 0, i32 0
  %loaded_rptr119 = load i32, i32* %r_field_cp_assign118
  store i32 %loaded_rptr119, i32* %l_field_cp_assign117
  %l_field_vec_cp_assign120 = getelementptr inbounds %example_inner_inner, %example_inner
_inner* %dest_ptr116, i32 0, i32 1
  %r_field_vec_cp_assign121 = getelementptr inbounds %example_inner_inner, %example_inner
_inner* %origin_ptr, i32 0, i32 1
  %loaded_rptr122 = load %vec_t, %vec_t* %r_field_vec_cp_assign121
  %temp123 = alloca %vec_t
  store %vec_t %loaded_rptr122, %vec_t* %temp123
  %ptr_p124 = getelementptr %vec_t, %vec_t* %temp123, i32 0, i32 0
  %size_p125 = getelementptr %vec_t, %vec_t* %temp123, i32 0, i32 1
  %len_p126 = getelementptr %vec_t, %vec_t* %temp123, i32 0, i32 2
  %ptr127 = load %vec_t*, %vec_t** %ptr_p124
  %size128 = load i32, i32* %size_p125
  %len129 = load i32, i32* %len_p126
  %out_vec130 = alloca %vec_t
  %out_ptr131 = getelementptr %vec_t, %vec_t* %out_vec130, i32 0, i32 0
  %out_size132 = getelementptr %vec_t, %vec_t* %out_vec130, i32 0, i32 1
  %out_len133 = getelementptr %vec_t, %vec_t* %out_vec130, i32 0, i32 2
  store i32 %size128, i32* %out_size132
  store i32 %len129, i32* %out_len133
  %iszerosize134 = icmp slt i32 %size128, 1
  br i1 %iszerosize134, label %zerosize135, label %nonzerosize136

zerosize135:                                      ; preds = %copy_vec_elem
  store %vec_t* null, %vec_t** %out_ptr131
  br label %merge144

nonzerosize136:                                   ; preds = %copy_vec_elem
  %mallocsize137 = mul i32 %len129, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
to i32)
  %malloccall138 = tail call i8* @malloc(i32 %mallocsize137)
  %newvec139 = bitcast i8* %malloccall138 to i32*
  %newvec_cast140 = bitcast i32* %newvec139 to %vec_t*
  store %vec_t* %newvec_cast140, %vec_t** %out_ptr131
  %src_cast141 = bitcast %vec_t* %ptr127 to i8*
  %dst_cast142 = bitcast i32* %newvec139 to i8*
  %sz_to_copy143 = mul i32 %size128, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
 to i32)
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast142, i8* %src_cast141, i32 %sz_to_cop
y143, i32 1, i1 true)
  br label %merge144

merge144:                                         ; preds = %nonzerosize136, %zerosize135
  %out_vec_val145 = load %vec_t, %vec_t* %out_vec130
  %temp146 = alloca %vec_t
  store %vec_t %out_vec_val145, %vec_t* %temp146
  %src_cast147 = bitcast %vec_t* %temp146 to i8*
```

```
  %dst_cast148 = bitcast %vec_t* %l_field_vec_cp_assign120 to i8*
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast148, i8* %src_cast147, i32 ptrtoint (
%vec_t* getelementptr (%vec_t, %vec_t* null, i32 1) to i32), i32 1, i1 true)
  %new_iter_val149 = add i32 %iter_val114, 1
  store i32 %new_iter_val149, i32* %iter99
  br label %check_size111

merge150:                                         ; preds = %check_size111
  %out_vec_val151 = load %vec_t, %vec_t* %out_vec100
  store %vec_t %out_vec_val151, %vec_t* %vec_alloc75
  %test4 = load %vec_t, %vec_t* %vec_alloc
  %elem_arr_p152 = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 0
  %elem_arr_load153 = load %vec_t*, %vec_t** %elem_arr_p152
  %elem_arr_cast154 = bitcast %vec_t* %elem_arr_load153 to i32*
  %elem155 = getelementptr i32, i32* %elem_arr_cast154, i32 0
  store i32 2, i32* %elem155
  %tmp4156 = load %vec_t, %vec_t* %vec_alloc1
  %elem_arr_p157 = getelementptr %vec_t, %vec_t* %vec_alloc75, i32 0, i32 0
  %elem_arr_load158 = load %vec_t*, %vec_t** %elem_arr_p157
  %elem_arr_cast159 = bitcast %vec_t* %elem_arr_load158 to %example_inner_inner*
  %elem160 = getelementptr %example_inner_inner, %example_inner_inner* %elem_arr_cast159,
 i32 0
  %struct_field161 = getelementptr inbounds %example_inner_inner, %example_inner_inner* %
elem160, i32 0, i32 1
  %ptr_p162 = getelementptr %vec_t, %vec_t* %struct_field161, i32 0, i32 0
  %size_p163 = getelementptr %vec_t, %vec_t* %struct_field161, i32 0, i32 1
  %ptr164 = load %vec_t*, %vec_t** %ptr_p162
  %size165 = load i32, i32* %size_p163
  %iszerosize166 = icmp slt i32 %size165, 1
  br i1 %iszerosize166, label %merge168, label %del_nonzerosize167

del_nonzerosize167:                               ; preds = %merge150
  %5 = bitcast %vec_t* %ptr164 to i8*
  tail call void @free(i8* %5)
  br label %merge168

merge168:                                         ; preds = %merge150, %del_nonzerosize16
7
  %temp169 = alloca %vec_t
  store %vec_t %tmp4156, %vec_t* %temp169
  %ptr_p170 = getelementptr %vec_t, %vec_t* %temp169, i32 0, i32 0
  %size_p171 = getelementptr %vec_t, %vec_t* %temp169, i32 0, i32 1
  %len_p172 = getelementptr %vec_t, %vec_t* %temp169, i32 0, i32 2
  %ptr173 = load %vec_t*, %vec_t** %ptr_p170
  %size174 = load i32, i32* %size_p171
  %len175 = load i32, i32* %len_p172
  %out_vec176 = alloca %vec_t
  %out_ptr177 = getelementptr %vec_t, %vec_t* %out_vec176, i32 0, i32 0
  %out_size178 = getelementptr %vec_t, %vec_t* %out_vec176, i32 0, i32 1
  %out_len179 = getelementptr %vec_t, %vec_t* %out_vec176, i32 0, i32 2
  store i32 %size174, i32* %out_size178
  store i32 %len175, i32* %out_len179
  %iszerosize180 = icmp slt i32 %size174, 1
  br i1 %iszerosize180, label %zerosize181, label %nonzerosize182

zerosize181:                                      ; preds = %merge168
  store %vec_t* null, %vec_t** %out_ptr177
  br label %merge190

nonzerosize182:                                   ; preds = %merge168
  %mallocsize183 = mul i32 %len175, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
to i32)
  %malloccall184 = tail call i8* @malloc(i32 %mallocsize183)
  %newvec185 = bitcast i8* %malloccall184 to i32*
  %newvec_cast186 = bitcast i32* %newvec185 to %vec_t*
  store %vec_t* %newvec_cast186, %vec_t** %out_ptr177
  %src_cast187 = bitcast %vec_t* %ptr173 to i8*
  %dst_cast188 = bitcast i32* %newvec185 to i8*
  %sz_to_copy189 = mul i32 %size174, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
 to i32)
```

```
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast188, i8* %src_cast187, i32 %sz_to_cop
y189, i32 1, i1 true)
  br label %merge190

merge190:                                         ; preds = %nonzerosize182, %zerosize181
  %out_vec_val191 = load %vec_t, %vec_t* %out_vec176
  store %vec_t %out_vec_val191, %vec_t* %struct_field161
  %temp192 = alloca %vec_t
  %elem_arr_p193 = getelementptr %vec_t, %vec_t* %vec_alloc75, i32 0, i32 0
  %elem_arr_load194 = load %vec_t*, %vec_t** %elem_arr_p193
  %elem_arr_cast195 = bitcast %vec_t* %elem_arr_load194 to %example_inner_inner*
  %elem196 = getelementptr %example_inner_inner, %example_inner_inner* %elem_arr_cast195,
 i32 0
  %struct_field197 = getelementptr inbounds %example_inner_inner, %example_inner_inner* %
elem196, i32 0, i32 1
  %struct_field_load = load %vec_t, %vec_t* %struct_field197
  store %vec_t %struct_field_load, %vec_t* %temp192
  %ptr_p198 = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 0
  %size_p199 = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 1
  %ptr200 = load %vec_t*, %vec_t** %ptr_p198
  %size201 = load i32, i32* %size_p199
  %iszerosize202 = icmp slt i32 %size201, 1
  br i1 %iszerosize202, label %merge204, label %del_nonzerosize203

del_nonzerosize203:                               ; preds = %merge190
  %6 = bitcast %vec_t* %ptr200 to i8*
  tail call void @free(i8* %6)
  br label %merge204

merge204:                                         ; preds = %merge190, %del_nonzerosize20
3
  %ptr_p205 = getelementptr %vec_t, %vec_t* %temp192, i32 0, i32 0
  %size_p206 = getelementptr %vec_t, %vec_t* %temp192, i32 0, i32 1
  %len_p207 = getelementptr %vec_t, %vec_t* %temp192, i32 0, i32 2
  %ptr208 = load %vec_t*, %vec_t** %ptr_p205
  %size209 = load i32, i32* %size_p206
  %len210 = load i32, i32* %len_p207
  %out_vec211 = alloca %vec_t
  %out_ptr212 = getelementptr %vec_t, %vec_t* %out_vec211, i32 0, i32 0
  %out_size213 = getelementptr %vec_t, %vec_t* %out_vec211, i32 0, i32 1
  %out_len214 = getelementptr %vec_t, %vec_t* %out_vec211, i32 0, i32 2
  store i32 %size209, i32* %out_size213
  store i32 %len210, i32* %out_len214
  %iszerosize215 = icmp slt i32 %size209, 1
  br i1 %iszerosize215, label %zerosize216, label %nonzerosize217

zerosize216:                                      ; preds = %merge204
  store %vec_t* null, %vec_t** %out_ptr212
  br label %merge225

nonzerosize217:                                   ; preds = %merge204
  %mallocsize218 = mul i32 %len210, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
to i32)
  %malloccall219 = tail call i8* @malloc(i32 %mallocsize218)
  %newvec220 = bitcast i8* %malloccall219 to i32*
  %newvec_cast221 = bitcast i32* %newvec220 to %vec_t*
  store %vec_t* %newvec_cast221, %vec_t** %out_ptr212
  %src_cast222 = bitcast %vec_t* %ptr208 to i8*
  %dst_cast223 = bitcast i32* %newvec220 to i8*
  %sz_to_copy224 = mul i32 %size209, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
 to i32)
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast223, i8* %src_cast222, i32 %sz_to_cop
y224, i32 1, i1 true)
  br label %merge225

merge225:                                         ; preds = %nonzerosize217, %zerosize216
  %out_vec_val226 = load %vec_t, %vec_t* %out_vec211
  store %vec_t %out_vec_val226, %vec_t* %vec_alloc1
  %elem_arr_p227 = getelementptr %vec_t, %vec_t* %vec_alloc75, i32 0, i32 0
  %elem_arr_load228 = load %vec_t*, %vec_t** %elem_arr_p227
```

```
  %elem_arr_cast229 = bitcast %vec_t* %elem_arr_load228 to %example_inner_inner*
  %elem230 = getelementptr %example_inner_inner, %example_inner_inner* %elem_arr_cast229,
 i32 0
  %struct_field231 = getelementptr inbounds %example_inner_inner, %example_inner_inner* %
elem230, i32 0, i32 1
  %struct_field_load232 = load %vec_t, %vec_t* %struct_field231
  %elem_arr_p233 = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 0
  %elem_arr_load234 = load %vec_t*, %vec_t** %elem_arr_p233
  %elem_arr_cast235 = bitcast %vec_t* %elem_arr_load234 to i32*
  %elem236 = getelementptr i32, i32* %elem_arr_cast235, i32 0
  %element = load i32, i32* %elem236
  %printf = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4 x i8]* @
fmt, i32 0, i32 0), i32 %element)
  %temp237 = alloca %vec_t
  %elem_arr_p238 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %elem_arr_load239 = load %vec_t*, %vec_t** %elem_arr_p238
  %elem_arr_cast240 = bitcast %vec_t* %elem_arr_load239 to %example_inner_inner*
  %elem241 = getelementptr %example_inner_inner, %example_inner_inner* %elem_arr_cast240,
 i32 0
  %struct_field242 = getelementptr inbounds %example_inner_inner, %example_inner_inner* %
elem241, i32 0, i32 1
  %struct_field_load243 = load %vec_t, %vec_t* %struct_field242
  store %vec_t %struct_field_load243, %vec_t* %temp237
  %ptr_p244 = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 0
  %size_p245 = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 1
  %ptr246 = load %vec_t*, %vec_t** %ptr_p244
  %size247 = load i32, i32* %size_p245
  %iszerosize248 = icmp slt i32 %size247, 1
  br i1 %iszerosize248, label %merge250, label %del_nonzerosize249

del_nonzerosize249:                               ; preds = %merge225
  %7 = bitcast %vec_t* %ptr246 to i8*
  tail call void @free(i8* %7)
  br label %merge250

merge250:                                         ; preds = %merge225, %del_nonzerosize24
9
  %ptr_p251 = getelementptr %vec_t, %vec_t* %temp237, i32 0, i32 0
  %size_p252 = getelementptr %vec_t, %vec_t* %temp237, i32 0, i32 1
  %len_p253 = getelementptr %vec_t, %vec_t* %temp237, i32 0, i32 2
  %ptr254 = load %vec_t*, %vec_t** %ptr_p251
  %size255 = load i32, i32* %size_p252
  %len256 = load i32, i32* %len_p253
  %out_vec257 = alloca %vec_t
  %out_ptr258 = getelementptr %vec_t, %vec_t* %out_vec257, i32 0, i32 0
  %out_size259 = getelementptr %vec_t, %vec_t* %out_vec257, i32 0, i32 1
  %out_len260 = getelementptr %vec_t, %vec_t* %out_vec257, i32 0, i32 2
  store i32 %size255, i32* %out_size259
  store i32 %len256, i32* %out_len260
  %iszerosize261 = icmp slt i32 %size255, 1
  br i1 %iszerosize261, label %zerosize262, label %nonzerosize263

zerosize262:                                      ; preds = %merge250
  store %vec_t* null, %vec_t** %out_ptr258
  br label %merge271

nonzerosize263:                                   ; preds = %merge250
  %mallocsize264 = mul i32 %len256, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
to i32)
  %malloccall265 = tail call i8* @malloc(i32 %mallocsize264)
  %newvec266 = bitcast i8* %malloccall265 to i32*
  %newvec_cast267 = bitcast i32* %newvec266 to %vec_t*
  store %vec_t* %newvec_cast267, %vec_t** %out_ptr258
  %src_cast268 = bitcast %vec_t* %ptr254 to i8*
  %dst_cast269 = bitcast i32* %newvec266 to i8*
  %sz_to_copy270 = mul i32 %size255, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
 to i32)
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast269, i8* %src_cast268, i32 %sz_to_cop
y270, i32 1, i1 true)
  br label %merge271
```

```
merge271:                                         ; preds = %nonzerosize263, %zerosize262
  %out_vec_val272 = load %vec_t, %vec_t* %out_vec257
  store %vec_t %out_vec_val272, %vec_t* %vec_alloc1
  %elem_arr_p273 = getelementptr %vec_t, %vec_t* %vec_alloc, i32 0, i32 0
  %elem_arr_load274 = load %vec_t*, %vec_t** %elem_arr_p273
  %elem_arr_cast275 = bitcast %vec_t* %elem_arr_load274 to %example_inner_inner*
  %elem276 = getelementptr %example_inner_inner, %example_inner_inner* %elem_arr_cast275,
 i32 0
  %struct_field277 = getelementptr inbounds %example_inner_inner, %example_inner_inner* %
elem276, i32 0, i32 1
  %struct_field_load278 = load %vec_t, %vec_t* %struct_field277
  %elem_arr_p279 = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 0
  %elem_arr_load280 = load %vec_t*, %vec_t** %elem_arr_p279
  %elem_arr_cast281 = bitcast %vec_t* %elem_arr_load280 to i32*
  %elem282 = getelementptr i32, i32* %elem_arr_cast281, i32 0
  %element283 = load i32, i32* %elem282
  %printf284 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4 x i8]
* @fmt, i32 0, i32 0), i32 %element283)
  %vec_alloc285 = alloca %vec_t
  %ptr286 = getelementptr %vec_t, %vec_t* %vec_alloc285, i32 0, i32 0
  store %vec_t* null, %vec_t** %ptr286
  %size287 = getelementptr %vec_t, %vec_t* %vec_alloc285, i32 0, i32 1
  store i32 0, i32* %size287
  %len288 = getelementptr %vec_t, %vec_t* %vec_alloc285, i32 0, i32 2
  store i32 0, i32* %len288
  %a1 = alloca %example_inner
  store %example_inner zeroinitializer, %example_inner* %a1
  %a = alloca %example_inner_inner
  store %example_inner_inner zeroinitializer, %example_inner_inner* %a
  %struct_field289 = getelementptr inbounds %example_inner_inner, %example_inner_inner* %
a, i32 0, i32 0
  store i32 1, i32* %struct_field289
  %vec_alloc290 = alloca %vec_t
  %ptr291 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 0
  store %vec_t* null, %vec_t** %ptr291
  %size292 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 1
  store i32 0, i32* %size292
  %len293 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 2
  store i32 0, i32* %len293
  %size_p294 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 1
  %size_var295 = load i32, i32* %size_p294
  %len_p296 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 2
  %leng_var297 = load i32, i32* %len_p296
  %size_var_cmp_leng298 = icmp slt i32 %size_var295, %leng_var297
  br i1 %size_var_cmp_leng298, label %pushback315, label %resize299

resize299:                                        ; preds = %merge271
  %size_var_nonzero300 = icmp slt i32 %leng_var297, 1
  br i1 %size_var_nonzero300, label %resize_zero301, label %resize_nonzero302

resize_zero301:                                   ; preds = %resize299
  store i32 1, i32* %len_p296
  br label %resize_merge304

resize_nonzero302:                                ; preds = %resize299
  %shl303 = shl i32 %leng_var297, 1
  store i32 %shl303, i32* %len_p296
  br label %resize_merge304

resize_merge304:                                  ; preds = %resize_nonzero302, %resize_z
ero301
  %new_len_val305 = load i32, i32* %len_p296
  %ptr_p306 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 0
  %ptr_val307 = load %vec_t*, %vec_t** %ptr_p306
  %mallocsize308 = mul i32 %new_len_val305, ptrtoint (i32* getelementptr (i32, i32* null,
 i32 1) to i32)
  %malloccall309 = tail call i8* @malloc(i32 %mallocsize308)
  %vec_mem310 = bitcast i8* %malloccall309 to i32*
  %vec_mem_byte_cast311 = bitcast i32* %vec_mem310 to i8*
```

```
  %ptr_val_byte_cast312 = bitcast %vec_t* %ptr_val307 to i8*
  %sz_to_copy313 = mul i32 ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to i32),
%size_var295
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast311, i8* %ptr_val_byte_cast3
12, i32 %sz_to_copy313, i32 1, i1 true)
  %8 = bitcast %vec_t* %ptr_val307 to i8*
  tail call void @free(i8* %8)
  %vec_mem_cast314 = bitcast i32* %vec_mem310 to %vec_t*
  store %vec_t* %vec_mem_cast314, %vec_t** %ptr_p306
  br label %pushback315

pushback315:                                      ; preds = %resize_merge304, %merge271
  %elem_arr_p316 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 0
  %elem_arr_load317 = load %vec_t*, %vec_t** %elem_arr_p316
  %elem_arr_cast318 = bitcast %vec_t* %elem_arr_load317 to i32*
  %elem319 = getelementptr i32, i32* %elem_arr_cast318, i32 %size_var295
  store i32 1, i32* %elem319
  %new_size320 = add i32 %size_var295, 1
  store i32 %new_size320, i32* %size_p294
  %tmp = load %vec_t, %vec_t* %vec_alloc290
  %struct_field321 = getelementptr inbounds %example_inner_inner, %example_inner_inner* %
a, i32 0, i32 1
  %ptr_p322 = getelementptr %vec_t, %vec_t* %struct_field321, i32 0, i32 0
  %size_p323 = getelementptr %vec_t, %vec_t* %struct_field321, i32 0, i32 1
  %ptr324 = load %vec_t*, %vec_t** %ptr_p322
  %size325 = load i32, i32* %size_p323
  %iszerosize326 = icmp slt i32 %size325, 1
  br i1 %iszerosize326, label %merge328, label %del_nonzerosize327

del_nonzerosize327:                               ; preds = %pushback315
  %9 = bitcast %vec_t* %ptr324 to i8*
  tail call void @free(i8* %9)
  br label %merge328

merge328:                                         ; preds = %pushback315, %del_nonzerosiz
e327
  %temp329 = alloca %vec_t
  store %vec_t %tmp, %vec_t* %temp329
  %ptr_p330 = getelementptr %vec_t, %vec_t* %temp329, i32 0, i32 0
  %size_p331 = getelementptr %vec_t, %vec_t* %temp329, i32 0, i32 1
  %len_p332 = getelementptr %vec_t, %vec_t* %temp329, i32 0, i32 2
  %ptr333 = load %vec_t*, %vec_t** %ptr_p330
  %size334 = load i32, i32* %size_p331
  %len335 = load i32, i32* %len_p332
  %out_vec336 = alloca %vec_t
  %out_ptr337 = getelementptr %vec_t, %vec_t* %out_vec336, i32 0, i32 0
  %out_size338 = getelementptr %vec_t, %vec_t* %out_vec336, i32 0, i32 1
  %out_len339 = getelementptr %vec_t, %vec_t* %out_vec336, i32 0, i32 2
  store i32 %size334, i32* %out_size338
  store i32 %len335, i32* %out_len339
  %iszerosize340 = icmp slt i32 %size334, 1
  br i1 %iszerosize340, label %zerosize341, label %nonzerosize342

zerosize341:                                      ; preds = %merge328
  store %vec_t* null, %vec_t** %out_ptr337
  br label %merge350

nonzerosize342:                                   ; preds = %merge328
  %mallocsize343 = mul i32 %len335, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
to i32)
  %malloccall344 = tail call i8* @malloc(i32 %mallocsize343)
  %newvec345 = bitcast i8* %malloccall344 to i32*
  %newvec_cast346 = bitcast i32* %newvec345 to %vec_t*
  store %vec_t* %newvec_cast346, %vec_t** %out_ptr337
  %src_cast347 = bitcast %vec_t* %ptr333 to i8*
  %dst_cast348 = bitcast i32* %newvec345 to i8*
  %sz_to_copy349 = mul i32 %size334, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
 to i32)
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast348, i8* %src_cast347, i32 %sz_to_cop
y349, i32 1, i1 true)
```

```
  br label %merge350

merge350:                                         ; preds = %nonzerosize342, %zerosize341
  %out_vec_val351 = load %vec_t, %vec_t* %out_vec336
  store %vec_t %out_vec_val351, %vec_t* %struct_field321
  %struct_field352 = getelementptr inbounds %example_inner, %example_inner* %a1, i32 0, i
32 1
  %l_field_cp_assign353 = getelementptr inbounds %example_inner_inner, %example_inner_inn
er* %struct_field352, i32 0, i32 0
  %r_field_cp_assign354 = getelementptr inbounds %example_inner_inner, %example_inner_inn
er* %a, i32 0, i32 0
  %loaded_rptr355 = load i32, i32* %r_field_cp_assign354
  store i32 %loaded_rptr355, i32* %l_field_cp_assign353
  %l_field_vec_cp_assign356 = getelementptr inbounds %example_inner_inner, %example_inner
_inner* %struct_field352, i32 0, i32 1
  %r_field_vec_cp_assign357 = getelementptr inbounds %example_inner_inner, %example_inner
_inner* %a, i32 0, i32 1
  %loaded_rptr358 = load %vec_t, %vec_t* %r_field_vec_cp_assign357
  %ptr_p359 = getelementptr %vec_t, %vec_t* %l_field_vec_cp_assign356, i32 0, i32 0
  %size_p360 = getelementptr %vec_t, %vec_t* %l_field_vec_cp_assign356, i32 0, i32 1
  %ptr361 = load %vec_t*, %vec_t** %ptr_p359
  %size362 = load i32, i32* %size_p360
  %iszerosize363 = icmp slt i32 %size362, 1
  br i1 %iszerosize363, label %merge365, label %del_nonzerosize364

del_nonzerosize364:                               ; preds = %merge350
  %10 = bitcast %vec_t* %ptr361 to i8*
  tail call void @free(i8* %10)
  br label %merge365

merge365:                                         ; preds = %merge350, %del_nonzerosize36
4
  %temp366 = alloca %vec_t
  store %vec_t %loaded_rptr358, %vec_t* %temp366
  %ptr_p367 = getelementptr %vec_t, %vec_t* %temp366, i32 0, i32 0
  %size_p368 = getelementptr %vec_t, %vec_t* %temp366, i32 0, i32 1
  %len_p369 = getelementptr %vec_t, %vec_t* %temp366, i32 0, i32 2
  %ptr370 = load %vec_t*, %vec_t** %ptr_p367
  %size371 = load i32, i32* %size_p368
  %len372 = load i32, i32* %len_p369
  %out_vec373 = alloca %vec_t
  %out_ptr374 = getelementptr %vec_t, %vec_t* %out_vec373, i32 0, i32 0
  %out_size375 = getelementptr %vec_t, %vec_t* %out_vec373, i32 0, i32 1
  %out_len376 = getelementptr %vec_t, %vec_t* %out_vec373, i32 0, i32 2
  store i32 %size371, i32* %out_size375
  store i32 %len372, i32* %out_len376
  %iszerosize377 = icmp slt i32 %size371, 1
  br i1 %iszerosize377, label %zerosize378, label %nonzerosize379

zerosize378:                                      ; preds = %merge365
  store %vec_t* null, %vec_t** %out_ptr374
  br label %merge387

nonzerosize379:                                   ; preds = %merge365
  %mallocsize380 = mul i32 %len372, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
to i32)
  %malloccall381 = tail call i8* @malloc(i32 %mallocsize380)
  %newvec382 = bitcast i8* %malloccall381 to i32*
  %newvec_cast383 = bitcast i32* %newvec382 to %vec_t*
  store %vec_t* %newvec_cast383, %vec_t** %out_ptr374
  %src_cast384 = bitcast %vec_t* %ptr370 to i8*
  %dst_cast385 = bitcast i32* %newvec382 to i8*
  %sz_to_copy386 = mul i32 %size371, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
 to i32)
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast385, i8* %src_cast384, i32 %sz_to_cop
y386, i32 1, i1 true)
  br label %merge387

merge387:                                         ; preds = %nonzerosize379, %zerosize378
  %out_vec_val388 = load %vec_t, %vec_t* %out_vec373
```

```
  store %vec_t %out_vec_val388, %vec_t* %l_field_vec_cp_assign356
  %new_struct389 = alloca %example_inner
  %l_field_cp_assign390 = getelementptr inbounds %example_inner, %example_inner* %new_str
uct389, i32 0, i32 0
  %r_field_cp_assign391 = getelementptr inbounds %example_inner, %example_inner* %a1, i32
 0, i32 0
  %loaded_rptr392 = load i32, i32* %r_field_cp_assign391
  store i32 %loaded_rptr392, i32* %l_field_cp_assign390
  %l_inner_struct_field = getelementptr inbounds %example_inner, %example_inner* %new_str
uct389, i32 0, i32 1
  %r_inner_struct_field = getelementptr inbounds %example_inner, %example_inner* %a1, i32
 0, i32 1
  %l_field_cp_assign393 = getelementptr inbounds %example_inner_inner, %example_inner_inn
er* %l_inner_struct_field, i32 0, i32 0
  %r_field_cp_assign394 = getelementptr inbounds %example_inner_inner, %example_inner_inn
er* %r_inner_struct_field, i32 0, i32 0
  %loaded_rptr395 = load i32, i32* %r_field_cp_assign394
  store i32 %loaded_rptr395, i32* %l_field_cp_assign393
  %l_field_vec_cp_assign396 = getelementptr inbounds %example_inner_inner, %example_inner
_inner* %l_inner_struct_field, i32 0, i32 1
  %r_field_vec_cp_assign397 = getelementptr inbounds %example_inner_inner, %example_inner
_inner* %r_inner_struct_field, i32 0, i32 1
  %loaded_rptr398 = load %vec_t, %vec_t* %r_field_vec_cp_assign397
  %temp399 = alloca %vec_t
  store %vec_t %loaded_rptr398, %vec_t* %temp399
  %ptr_p400 = getelementptr %vec_t, %vec_t* %temp399, i32 0, i32 0
  %size_p401 = getelementptr %vec_t, %vec_t* %temp399, i32 0, i32 1
  %len_p402 = getelementptr %vec_t, %vec_t* %temp399, i32 0, i32 2
  %ptr403 = load %vec_t*, %vec_t** %ptr_p400
  %size404 = load i32, i32* %size_p401
  %len405 = load i32, i32* %len_p402
  %out_vec406 = alloca %vec_t
  %out_ptr407 = getelementptr %vec_t, %vec_t* %out_vec406, i32 0, i32 0
  %out_size408 = getelementptr %vec_t, %vec_t* %out_vec406, i32 0, i32 1
  %out_len409 = getelementptr %vec_t, %vec_t* %out_vec406, i32 0, i32 2
  store i32 %size404, i32* %out_size408
  store i32 %len405, i32* %out_len409
  %iszerosize410 = icmp slt i32 %size404, 1
  br i1 %iszerosize410, label %zerosize411, label %nonzerosize412

zerosize411:                                      ; preds = %merge387
  store %vec_t* null, %vec_t** %out_ptr407
  br label %merge420

nonzerosize412:                                   ; preds = %merge387
  %mallocsize413 = mul i32 %len405, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
to i32)
  %malloccall414 = tail call i8* @malloc(i32 %mallocsize413)
  %newvec415 = bitcast i8* %malloccall414 to i32*
  %newvec_cast416 = bitcast i32* %newvec415 to %vec_t*
  store %vec_t* %newvec_cast416, %vec_t** %out_ptr407
  %src_cast417 = bitcast %vec_t* %ptr403 to i8*
  %dst_cast418 = bitcast i32* %newvec415 to i8*
  %sz_to_copy419 = mul i32 %size404, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
 to i32)
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast418, i8* %src_cast417, i32 %sz_to_cop
y419, i32 1, i1 true)
  br label %merge420

merge420:                                         ; preds = %nonzerosize412, %zerosize411
  %out_vec_val421 = load %vec_t, %vec_t* %out_vec406
  %temp422 = alloca %vec_t
  store %vec_t %out_vec_val421, %vec_t* %temp422
  %src_cast423 = bitcast %vec_t* %temp422 to i8*
  %dst_cast424 = bitcast %vec_t* %l_field_vec_cp_assign396 to i8*
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast424, i8* %src_cast423, i32 ptrtoint (
%vec_t* getelementptr (%vec_t, %vec_t* null, i32 1) to i32), i32 1, i1 true)
  %new_struct_val425 = load %example_inner, %example_inner* %new_struct389
  %size_p426 = getelementptr %vec_t, %vec_t* %vec_alloc285, i32 0, i32 1
  %size_var427 = load i32, i32* %size_p426
```

```
  %len_p428 = getelementptr %vec_t, %vec_t* %vec_alloc285, i32 0, i32 2
  %leng_var429 = load i32, i32* %len_p428
  %size_var_cmp_leng430 = icmp slt i32 %size_var427, %leng_var429
  br i1 %size_var_cmp_leng430, label %pushback447, label %resize431

resize431:                                        ; preds = %merge420
  %size_var_nonzero432 = icmp slt i32 %leng_var429, 1
  br i1 %size_var_nonzero432, label %resize_zero433, label %resize_nonzero434

resize_zero433:                                   ; preds = %resize431
  store i32 1, i32* %len_p428
  br label %resize_merge436

resize_nonzero434:                                ; preds = %resize431
  %shl435 = shl i32 %leng_var429, 1
  store i32 %shl435, i32* %len_p428
  br label %resize_merge436

resize_merge436:                                  ; preds = %resize_nonzero434, %resize_z
ero433
  %new_len_val437 = load i32, i32* %len_p428
  %ptr_p438 = getelementptr %vec_t, %vec_t* %vec_alloc285, i32 0, i32 0
  %ptr_val439 = load %vec_t*, %vec_t** %ptr_p438
  %mallocsize440 = mul i32 %new_len_val437, ptrtoint (%example_inner* getelementptr (%exa
mple_inner, %example_inner* null, i32 1) to i32)
  %malloccall441 = tail call i8* @malloc(i32 %mallocsize440)
  %vec_mem442 = bitcast i8* %malloccall441 to %example_inner*
  %vec_mem_byte_cast443 = bitcast %example_inner* %vec_mem442 to i8*
  %ptr_val_byte_cast444 = bitcast %vec_t* %ptr_val439 to i8*
  %sz_to_copy445 = mul i32 ptrtoint (%example_inner* getelementptr (%example_inner, %exam
ple_inner* null, i32 1) to i32), %size_var427
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %vec_mem_byte_cast443, i8* %ptr_val_byte_cast4
44, i32 %sz_to_copy445, i32 1, i1 true)
  %11 = bitcast %vec_t* %ptr_val439 to i8*
  tail call void @free(i8* %11)
  %vec_mem_cast446 = bitcast %example_inner* %vec_mem442 to %vec_t*
  store %vec_t* %vec_mem_cast446, %vec_t** %ptr_p438
  br label %pushback447

pushback447:                                      ; preds = %resize_merge436, %merge420
  %elem_arr_p448 = getelementptr %vec_t, %vec_t* %vec_alloc285, i32 0, i32 0
  %elem_arr_load449 = load %vec_t*, %vec_t** %elem_arr_p448
  %elem_arr_cast450 = bitcast %vec_t* %elem_arr_load449 to %example_inner*
  %elem451 = getelementptr %example_inner, %example_inner* %elem_arr_cast450, i32 %size_v
ar427
  store %example_inner %new_struct_val425, %example_inner* %elem451
  %new_size452 = add i32 %size_var427, 1
  store i32 %new_size452, i32* %size_p426
  %vec_alloc453 = alloca %vec_t
  %ptr454 = getelementptr %vec_t, %vec_t* %vec_alloc453, i32 0, i32 0
  store %vec_t* null, %vec_t** %ptr454
  %size455 = getelementptr %vec_t, %vec_t* %vec_alloc453, i32 0, i32 1
  store i32 0, i32* %size455
  %len456 = getelementptr %vec_t, %vec_t* %vec_alloc453, i32 0, i32 2
  store i32 0, i32* %len456
  %ptr_p457 = getelementptr %vec_t, %vec_t* %vec_alloc453, i32 0, i32 0
  %size_p458 = getelementptr %vec_t, %vec_t* %vec_alloc453, i32 0, i32 1
  %ptr459 = load %vec_t*, %vec_t** %ptr_p457
  %size460 = load i32, i32* %size_p458
  %iter461 = alloca i32
  store i32 0, i32* %iter461
  %iszerosize462 = icmp slt i32 %size460, 1
  br i1 %iszerosize462, label %mergeagain480, label %check_size463

check_size463:                                    ; preds = %merge477, %pushback447
  %iter_val464 = load i32, i32* %iter461
  %keeplooping465 = icmp slt i32 %iter_val464, %size460
  br i1 %keeplooping465, label %del_vec_elem466, label %merge479

del_vec_elem466:                                  ; preds = %check_size463
```

```
  %iter_val467 = load i32, i32* %iter461
  %dest_ptr468 = getelementptr %vec_t, %vec_t* %ptr459, i32 %iter_val467
  %struct_p469 = bitcast %vec_t* %dest_ptr468 to %example_inner*
  %inner_struct_field_delete = getelementptr inbounds %example_inner, %example_inner* %st
ruct_p469, i32 0, i32 1
  %vec_ptr_delete470 = getelementptr inbounds %example_inner_inner, %example_inner_inner*
 %inner_struct_field_delete, i32 0, i32 1
  %ptr_p471 = getelementptr %vec_t, %vec_t* %vec_ptr_delete470, i32 0, i32 0
  %size_p472 = getelementptr %vec_t, %vec_t* %vec_ptr_delete470, i32 0, i32 1
  %ptr473 = load %vec_t*, %vec_t** %ptr_p471
  %size474 = load i32, i32* %size_p472
  %iszerosize475 = icmp slt i32 %size474, 1
  br i1 %iszerosize475, label %merge477, label %del_nonzerosize476

del_nonzerosize476:                               ; preds = %del_vec_elem466
  %12 = bitcast %vec_t* %ptr473 to i8*
  tail call void @free(i8* %12)
  br label %merge477

merge477:                                         ; preds = %del_vec_elem466, %del_nonzer
osize476
  %new_iter_val478 = add i32 %iter_val467, 1
  store i32 %new_iter_val478, i32* %iter461
  br label %check_size463

merge479:                                         ; preds = %check_size463
  %13 = bitcast %vec_t* %ptr459 to i8*
  tail call void @free(i8* %13)
  br label %mergeagain480

mergeagain480:                                    ; preds = %merge479, %pushback447
  %ptr_p481 = getelementptr %vec_t, %vec_t* %vec_alloc285, i32 0, i32 0
  %size_p482 = getelementptr %vec_t, %vec_t* %vec_alloc285, i32 0, i32 1
  %len_p483 = getelementptr %vec_t, %vec_t* %vec_alloc285, i32 0, i32 2
  %ptr484 = load %vec_t*, %vec_t** %ptr_p481
  %size485 = load i32, i32* %size_p482
  %len486 = load i32, i32* %len_p483
  %iter487 = alloca i32
  store i32 0, i32* %iter487
  %out_vec488 = alloca %vec_t
  %out_ptr489 = getelementptr %vec_t, %vec_t* %out_vec488, i32 0, i32 0
  %out_size490 = getelementptr %vec_t, %vec_t* %out_vec488, i32 0, i32 1
  %out_len491 = getelementptr %vec_t, %vec_t* %out_vec488, i32 0, i32 2
  store i32 %size485, i32* %out_size490
  store i32 %len486, i32* %out_len491
  %iszerosize492 = icmp slt i32 %size485, 1
  br i1 %iszerosize492, label %zerosize493, label %nonzerosize494

zerosize493:                                      ; preds = %mergeagain480
  store %vec_t* null, %vec_t** %out_ptr489
  br label %check_size499

nonzerosize494:                                   ; preds = %mergeagain480
  %mallocsize495 = mul i32 %len486, ptrtoint (%example_inner* getelementptr (%example_inn
er, %example_inner* null, i32 1) to i32)
  %malloccall496 = tail call i8* @malloc(i32 %mallocsize495)
  %newvec497 = bitcast i8* %malloccall496 to %example_inner*
  %newvec_cast498 = bitcast %example_inner* %newvec497 to %vec_t*
  store %vec_t* %newvec_cast498, %vec_t** %out_ptr489
  br label %check_size499

check_size499:                                    ; preds = %merge541, %nonzerosize494, %
zerosize493
  %iter_val500 = load i32, i32* %iter487
  %keeplooping501 = icmp slt i32 %iter_val500, %size485
  br i1 %keeplooping501, label %copy_vec_elem502, label %merge547

copy_vec_elem502:                                 ; preds = %check_size499
  %iter_val503 = load i32, i32* %iter487
  %vec_ptr504 = load %vec_t*, %vec_t** %out_ptr489
```

```
  %struct_vec_ptr505 = bitcast %vec_t* %vec_ptr504 to %example_inner*
  %struct_vec_ptr506 = bitcast %vec_t* %ptr484 to %example_inner*
  %dest_ptr507 = getelementptr %example_inner, %example_inner* %struct_vec_ptr505, i32 %i
ter_val503
  %origin_ptr508 = getelementptr %example_inner, %example_inner* %struct_vec_ptr506, i32
%iter_val503
  %l_field_cp_assign509 = getelementptr inbounds %example_inner, %example_inner* %dest_pt
r507, i32 0, i32 0
  %r_field_cp_assign510 = getelementptr inbounds %example_inner, %example_inner* %origin_
ptr508, i32 0, i32 0
  %loaded_rptr511 = load i32, i32* %r_field_cp_assign510
  store i32 %loaded_rptr511, i32* %l_field_cp_assign509
  %l_inner_struct_field512 = getelementptr inbounds %example_inner, %example_inner* %dest
_ptr507, i32 0, i32 1
  %r_inner_struct_field513 = getelementptr inbounds %example_inner, %example_inner* %orig
in_ptr508, i32 0, i32 1
  %l_field_cp_assign514 = getelementptr inbounds %example_inner_inner, %example_inner_inn
er* %l_inner_struct_field512, i32 0, i32 0
  %r_field_cp_assign515 = getelementptr inbounds %example_inner_inner, %example_inner_inn
er* %r_inner_struct_field513, i32 0, i32 0
  %loaded_rptr516 = load i32, i32* %r_field_cp_assign515
  store i32 %loaded_rptr516, i32* %l_field_cp_assign514
  %l_field_vec_cp_assign517 = getelementptr inbounds %example_inner_inner, %example_inner
_inner* %l_inner_struct_field512, i32 0, i32 1
  %r_field_vec_cp_assign518 = getelementptr inbounds %example_inner_inner, %example_inner
_inner* %r_inner_struct_field513, i32 0, i32 1
  %loaded_rptr519 = load %vec_t, %vec_t* %r_field_vec_cp_assign518
  %temp520 = alloca %vec_t
  store %vec_t %loaded_rptr519, %vec_t* %temp520
  %ptr_p521 = getelementptr %vec_t, %vec_t* %temp520, i32 0, i32 0
  %size_p522 = getelementptr %vec_t, %vec_t* %temp520, i32 0, i32 1
  %len_p523 = getelementptr %vec_t, %vec_t* %temp520, i32 0, i32 2
  %ptr524 = load %vec_t*, %vec_t** %ptr_p521
  %size525 = load i32, i32* %size_p522
  %len526 = load i32, i32* %len_p523
  %out_vec527 = alloca %vec_t
  %out_ptr528 = getelementptr %vec_t, %vec_t* %out_vec527, i32 0, i32 0
  %out_size529 = getelementptr %vec_t, %vec_t* %out_vec527, i32 0, i32 1
  %out_len530 = getelementptr %vec_t, %vec_t* %out_vec527, i32 0, i32 2
  store i32 %size525, i32* %out_size529
  store i32 %len526, i32* %out_len530
  %iszerosize531 = icmp slt i32 %size525, 1
  br i1 %iszerosize531, label %zerosize532, label %nonzerosize533

zerosize532:                                      ; preds = %copy_vec_elem502
  store %vec_t* null, %vec_t** %out_ptr528
  br label %merge541

nonzerosize533:                                   ; preds = %copy_vec_elem502
  %mallocsize534 = mul i32 %len526, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
to i32)
  %malloccall535 = tail call i8* @malloc(i32 %mallocsize534)
  %newvec536 = bitcast i8* %malloccall535 to i32*
  %newvec_cast537 = bitcast i32* %newvec536 to %vec_t*
  store %vec_t* %newvec_cast537, %vec_t** %out_ptr528
  %src_cast538 = bitcast %vec_t* %ptr524 to i8*
  %dst_cast539 = bitcast i32* %newvec536 to i8*
  %sz_to_copy540 = mul i32 %size525, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
 to i32)
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast539, i8* %src_cast538, i32 %sz_to_cop
y540, i32 1, i1 true)
  br label %merge541

merge541:                                         ; preds = %nonzerosize533, %zerosize532
  %out_vec_val542 = load %vec_t, %vec_t* %out_vec527
  %temp543 = alloca %vec_t
  store %vec_t %out_vec_val542, %vec_t* %temp543
  %src_cast544 = bitcast %vec_t* %temp543 to i8*
  %dst_cast545 = bitcast %vec_t* %l_field_vec_cp_assign517 to i8*
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast545, i8* %src_cast544, i32 ptrtoint (
```

```
%vec_t* getelementptr (%vec_t, %vec_t* null, i32 1) to i32), i32 1, i1 true)
  %new_iter_val546 = add i32 %iter_val503, 1
  store i32 %new_iter_val546, i32* %iter487
  br label %check_size499

merge547:                                         ; preds = %check_size499
  %out_vec_val548 = load %vec_t, %vec_t* %out_vec488
  store %vec_t %out_vec_val548, %vec_t* %vec_alloc453
  %test = load %vec_t, %vec_t* %vec_alloc285
  %elem_arr_p549 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 0
  %elem_arr_load550 = load %vec_t*, %vec_t** %elem_arr_p549
  %elem_arr_cast551 = bitcast %vec_t* %elem_arr_load550 to i32*
  %elem552 = getelementptr i32, i32* %elem_arr_cast551, i32 0
  store i32 2, i32* %elem552
  %tmp553 = load %vec_t, %vec_t* %vec_alloc290
  %elem_arr_p554 = getelementptr %vec_t, %vec_t* %vec_alloc453, i32 0, i32 0
  %elem_arr_load555 = load %vec_t*, %vec_t** %elem_arr_p554
  %elem_arr_cast556 = bitcast %vec_t* %elem_arr_load555 to %example_inner*
  %elem557 = getelementptr %example_inner, %example_inner* %elem_arr_cast556, i32 0
  %struct_field558 = getelementptr inbounds %example_inner, %example_inner* %elem557, i32
 0, i32 1
  %struct_field559 = getelementptr inbounds %example_inner_inner, %example_inner_inner* %
struct_field558, i32 0, i32 1
  %ptr_p560 = getelementptr %vec_t, %vec_t* %struct_field559, i32 0, i32 0
  %size_p561 = getelementptr %vec_t, %vec_t* %struct_field559, i32 0, i32 1
  %ptr562 = load %vec_t*, %vec_t** %ptr_p560
  %size563 = load i32, i32* %size_p561
  %iszerosize564 = icmp slt i32 %size563, 1
  br i1 %iszerosize564, label %merge566, label %del_nonzerosize565

del_nonzerosize565:                               ; preds = %merge547
  %14 = bitcast %vec_t* %ptr562 to i8*
  tail call void @free(i8* %14)
  br label %merge566

merge566:                                         ; preds = %merge547, %del_nonzerosize56
5
  %temp567 = alloca %vec_t
  store %vec_t %tmp553, %vec_t* %temp567
  %ptr_p568 = getelementptr %vec_t, %vec_t* %temp567, i32 0, i32 0
  %size_p569 = getelementptr %vec_t, %vec_t* %temp567, i32 0, i32 1
  %len_p570 = getelementptr %vec_t, %vec_t* %temp567, i32 0, i32 2
  %ptr571 = load %vec_t*, %vec_t** %ptr_p568
  %size572 = load i32, i32* %size_p569
  %len573 = load i32, i32* %len_p570
  %out_vec574 = alloca %vec_t
  %out_ptr575 = getelementptr %vec_t, %vec_t* %out_vec574, i32 0, i32 0
  %out_size576 = getelementptr %vec_t, %vec_t* %out_vec574, i32 0, i32 1
  %out_len577 = getelementptr %vec_t, %vec_t* %out_vec574, i32 0, i32 2
  store i32 %size572, i32* %out_size576
  store i32 %len573, i32* %out_len577
  %iszerosize578 = icmp slt i32 %size572, 1
  br i1 %iszerosize578, label %zerosize579, label %nonzerosize580

zerosize579:                                      ; preds = %merge566
  store %vec_t* null, %vec_t** %out_ptr575
  br label %merge588

nonzerosize580:                                   ; preds = %merge566
  %mallocsize581 = mul i32 %len573, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
to i32)
  %malloccall582 = tail call i8* @malloc(i32 %mallocsize581)
  %newvec583 = bitcast i8* %malloccall582 to i32*
  %newvec_cast584 = bitcast i32* %newvec583 to %vec_t*
  store %vec_t* %newvec_cast584, %vec_t** %out_ptr575
  %src_cast585 = bitcast %vec_t* %ptr571 to i8*
  %dst_cast586 = bitcast i32* %newvec583 to i8*
  %sz_to_copy587 = mul i32 %size572, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
 to i32)
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast586, i8* %src_cast585, i32 %sz_to_cop
```

```
y587, i32 1, i1 true)
  br label %merge588

merge588:                                         ; preds = %nonzerosize580, %zerosize579
  %out_vec_val589 = load %vec_t, %vec_t* %out_vec574
  store %vec_t %out_vec_val589, %vec_t* %struct_field559
  %temp590 = alloca %vec_t
  %elem_arr_p591 = getelementptr %vec_t, %vec_t* %vec_alloc453, i32 0, i32 0
  %elem_arr_load592 = load %vec_t*, %vec_t** %elem_arr_p591
  %elem_arr_cast593 = bitcast %vec_t* %elem_arr_load592 to %example_inner*
  %elem594 = getelementptr %example_inner, %example_inner* %elem_arr_cast593, i32 0
  %struct_field595 = getelementptr inbounds %example_inner, %example_inner* %elem594, i32
 0, i32 1
  %struct_field596 = getelementptr inbounds %example_inner_inner, %example_inner_inner* %
struct_field595, i32 0, i32 1
  %struct_field_load597 = load %vec_t, %vec_t* %struct_field596
  store %vec_t %struct_field_load597, %vec_t* %temp590
  %ptr_p598 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 0
  %size_p599 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 1
  %ptr600 = load %vec_t*, %vec_t** %ptr_p598
  %size601 = load i32, i32* %size_p599
  %iszerosize602 = icmp slt i32 %size601, 1
  br i1 %iszerosize602, label %merge604, label %del_nonzerosize603

del_nonzerosize603:                               ; preds = %merge588
  %15 = bitcast %vec_t* %ptr600 to i8*
  tail call void @free(i8* %15)
  br label %merge604

merge604:                                         ; preds = %merge588, %del_nonzerosize60
3
  %ptr_p605 = getelementptr %vec_t, %vec_t* %temp590, i32 0, i32 0
  %size_p606 = getelementptr %vec_t, %vec_t* %temp590, i32 0, i32 1
  %len_p607 = getelementptr %vec_t, %vec_t* %temp590, i32 0, i32 2
  %ptr608 = load %vec_t*, %vec_t** %ptr_p605
  %size609 = load i32, i32* %size_p606
  %len610 = load i32, i32* %len_p607
  %out_vec611 = alloca %vec_t
  %out_ptr612 = getelementptr %vec_t, %vec_t* %out_vec611, i32 0, i32 0
  %out_size613 = getelementptr %vec_t, %vec_t* %out_vec611, i32 0, i32 1
  %out_len614 = getelementptr %vec_t, %vec_t* %out_vec611, i32 0, i32 2
  store i32 %size609, i32* %out_size613
  store i32 %len610, i32* %out_len614
  %iszerosize615 = icmp slt i32 %size609, 1
  br i1 %iszerosize615, label %zerosize616, label %nonzerosize617

zerosize616:                                      ; preds = %merge604
  store %vec_t* null, %vec_t** %out_ptr612
  br label %merge625

nonzerosize617:                                   ; preds = %merge604
  %mallocsize618 = mul i32 %len610, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
to i32)
  %malloccall619 = tail call i8* @malloc(i32 %mallocsize618)
  %newvec620 = bitcast i8* %malloccall619 to i32*
  %newvec_cast621 = bitcast i32* %newvec620 to %vec_t*
  store %vec_t* %newvec_cast621, %vec_t** %out_ptr612
  %src_cast622 = bitcast %vec_t* %ptr608 to i8*
  %dst_cast623 = bitcast i32* %newvec620 to i8*
  %sz_to_copy624 = mul i32 %size609, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
 to i32)
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast623, i8* %src_cast622, i32 %sz_to_cop
y624, i32 1, i1 true)
  br label %merge625

merge625:                                         ; preds = %nonzerosize617, %zerosize616
  %out_vec_val626 = load %vec_t, %vec_t* %out_vec611
  store %vec_t %out_vec_val626, %vec_t* %vec_alloc290
  %elem_arr_p627 = getelementptr %vec_t, %vec_t* %vec_alloc453, i32 0, i32 0
  %elem_arr_load628 = load %vec_t*, %vec_t** %elem_arr_p627
```

```
  %elem_arr_cast629 = bitcast %vec_t* %elem_arr_load628 to %example_inner*
  %elem630 = getelementptr %example_inner, %example_inner* %elem_arr_cast629, i32 0
  %struct_field631 = getelementptr inbounds %example_inner, %example_inner* %elem630, i32
 0, i32 1
  %struct_field632 = getelementptr inbounds %example_inner_inner, %example_inner_inner* %
struct_field631, i32 0, i32 1
  %struct_field_load633 = load %vec_t, %vec_t* %struct_field632
  %elem_arr_p634 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 0
  %elem_arr_load635 = load %vec_t*, %vec_t** %elem_arr_p634
  %elem_arr_cast636 = bitcast %vec_t* %elem_arr_load635 to i32*
  %elem637 = getelementptr i32, i32* %elem_arr_cast636, i32 0
  %element638 = load i32, i32* %elem637
  %printf639 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4 x i8]
* @fmt, i32 0, i32 0), i32 %element638)
  %temp640 = alloca %vec_t
  %elem_arr_p641 = getelementptr %vec_t, %vec_t* %vec_alloc285, i32 0, i32 0
  %elem_arr_load642 = load %vec_t*, %vec_t** %elem_arr_p641
  %elem_arr_cast643 = bitcast %vec_t* %elem_arr_load642 to %example_inner*
  %elem644 = getelementptr %example_inner, %example_inner* %elem_arr_cast643, i32 0
  %struct_field645 = getelementptr inbounds %example_inner, %example_inner* %elem644, i32
 0, i32 1
  %struct_field646 = getelementptr inbounds %example_inner_inner, %example_inner_inner* %
struct_field645, i32 0, i32 1
  %struct_field_load647 = load %vec_t, %vec_t* %struct_field646
  store %vec_t %struct_field_load647, %vec_t* %temp640
  %ptr_p648 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 0
  %size_p649 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 1
  %ptr650 = load %vec_t*, %vec_t** %ptr_p648
  %size651 = load i32, i32* %size_p649
  %iszerosize652 = icmp slt i32 %size651, 1
  br i1 %iszerosize652, label %merge654, label %del_nonzerosize653

del_nonzerosize653:                               ; preds = %merge625
  %16 = bitcast %vec_t* %ptr650 to i8*
  tail call void @free(i8* %16)
  br label %merge654

merge654:                                         ; preds = %merge625, %del_nonzerosize65
3
  %ptr_p655 = getelementptr %vec_t, %vec_t* %temp640, i32 0, i32 0
  %size_p656 = getelementptr %vec_t, %vec_t* %temp640, i32 0, i32 1
  %len_p657 = getelementptr %vec_t, %vec_t* %temp640, i32 0, i32 2
  %ptr658 = load %vec_t*, %vec_t** %ptr_p655
  %size659 = load i32, i32* %size_p656
  %len660 = load i32, i32* %len_p657
  %out_vec661 = alloca %vec_t
  %out_ptr662 = getelementptr %vec_t, %vec_t* %out_vec661, i32 0, i32 0
  %out_size663 = getelementptr %vec_t, %vec_t* %out_vec661, i32 0, i32 1
  %out_len664 = getelementptr %vec_t, %vec_t* %out_vec661, i32 0, i32 2
  store i32 %size659, i32* %out_size663
  store i32 %len660, i32* %out_len664
  %iszerosize665 = icmp slt i32 %size659, 1
  br i1 %iszerosize665, label %zerosize666, label %nonzerosize667

zerosize666:                                      ; preds = %merge654
  store %vec_t* null, %vec_t** %out_ptr662
  br label %merge675

nonzerosize667:                                   ; preds = %merge654
  %mallocsize668 = mul i32 %len660, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
to i32)
  %malloccall669 = tail call i8* @malloc(i32 %mallocsize668)
  %newvec670 = bitcast i8* %malloccall669 to i32*
  %newvec_cast671 = bitcast i32* %newvec670 to %vec_t*
  store %vec_t* %newvec_cast671, %vec_t** %out_ptr662
  %src_cast672 = bitcast %vec_t* %ptr658 to i8*
  %dst_cast673 = bitcast i32* %newvec670 to i8*
  %sz_to_copy674 = mul i32 %size659, ptrtoint (i32* getelementptr (i32, i32* null, i32 1)
 to i32)
  call void @llvm.memcpy.p0i8.p0i8.i32(i8* %dst_cast673, i8* %src_cast672, i32 %sz_to_cop
```

```
y674, i32 1, i1 true)
  br label %merge675

merge675:                                         ; preds = %nonzerosize667, %zerosize666
  %out_vec_val676 = load %vec_t, %vec_t* %out_vec661
  store %vec_t %out_vec_val676, %vec_t* %vec_alloc290
  %elem_arr_p677 = getelementptr %vec_t, %vec_t* %vec_alloc285, i32 0, i32 0
  %elem_arr_load678 = load %vec_t*, %vec_t** %elem_arr_p677
  %elem_arr_cast679 = bitcast %vec_t* %elem_arr_load678 to %example_inner*
  %elem680 = getelementptr %example_inner, %example_inner* %elem_arr_cast679, i32 0
  %struct_field681 = getelementptr inbounds %example_inner, %example_inner* %elem680, i32
 0, i32 1
  %struct_field682 = getelementptr inbounds %example_inner_inner, %example_inner_inner* %
struct_field681, i32 0, i32 1
  %struct_field_load683 = load %vec_t, %vec_t* %struct_field682
  %elem_arr_p684 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 0
  %elem_arr_load685 = load %vec_t*, %vec_t** %elem_arr_p684
  %elem_arr_cast686 = bitcast %vec_t* %elem_arr_load685 to i32*
  %elem687 = getelementptr i32, i32* %elem_arr_cast686, i32 0
  %element688 = load i32, i32* %elem687
  %printf689 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4 x i8]
* @fmt, i32 0, i32 0), i32 %element688)
  br label %return_loc

check_size696:                                    ; preds = %merge710, %return_loc
  %iter_val697 = load i32, i32* %iter694
  %keeplooping698 = icmp slt i32 %iter_val697, %size693
  br i1 %keeplooping698, label %del_vec_elem699, label %merge712

del_vec_elem699:                                  ; preds = %check_size696
  %iter_val700 = load i32, i32* %iter694
  %dest_ptr701 = getelementptr %vec_t, %vec_t* %ptr692, i32 %iter_val700
  %struct_p702 = bitcast %vec_t* %dest_ptr701 to %example_inner_inner*
  %vec_ptr_delete703 = getelementptr inbounds %example_inner_inner, %example_inner_inner*
 %struct_p702, i32 0, i32 1
  %ptr_p704 = getelementptr %vec_t, %vec_t* %vec_ptr_delete703, i32 0, i32 0
  %size_p705 = getelementptr %vec_t, %vec_t* %vec_ptr_delete703, i32 0, i32 1
  %ptr706 = load %vec_t*, %vec_t** %ptr_p704
  %size707 = load i32, i32* %size_p705
  %iszerosize708 = icmp slt i32 %size707, 1
  br i1 %iszerosize708, label %merge710, label %del_nonzerosize709

del_nonzerosize709:                               ; preds = %del_vec_elem699
  %17 = bitcast %vec_t* %ptr706 to i8*
  tail call void @free(i8* %17)
  br label %merge710

merge710:                                         ; preds = %del_vec_elem699, %del_nonzer
osize709
  %new_iter_val711 = add i32 %iter_val700, 1
  store i32 %new_iter_val711, i32* %iter694
  br label %check_size696

merge712:                                         ; preds = %check_size696
  %18 = bitcast %vec_t* %ptr692 to i8*
  tail call void @free(i8* %18)
  br label %mergeagain713

mergeagain713:                                    ; preds = %merge712, %return_loc
  %ptr_p714 = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 0
  %size_p715 = getelementptr %vec_t, %vec_t* %vec_alloc1, i32 0, i32 1
  %ptr716 = load %vec_t*, %vec_t** %ptr_p714
  %size717 = load i32, i32* %size_p715
  %iszerosize718 = icmp slt i32 %size717, 1
  br i1 %iszerosize718, label %merge720, label %del_nonzerosize719

del_nonzerosize719:                               ; preds = %mergeagain713
  %19 = bitcast %vec_t* %ptr716 to i8*
  tail call void @free(i8* %19)
  br label %merge720
```

```
merge720:                                         ; preds = %mergeagain713, %del_nonzeros
ize719
  %ptr_p721 = getelementptr %vec_t, %vec_t* %vec_alloc453, i32 0, i32 0
  %size_p722 = getelementptr %vec_t, %vec_t* %vec_alloc453, i32 0, i32 1
  %ptr723 = load %vec_t*, %vec_t** %ptr_p721
  %size724 = load i32, i32* %size_p722
  %iter725 = alloca i32
  store i32 0, i32* %iter725
  %iszerosize726 = icmp slt i32 %size724, 1
  br i1 %iszerosize726, label %mergeagain745, label %check_size727

check_size727:                                    ; preds = %merge742, %merge720
  %iter_val728 = load i32, i32* %iter725
  %keeplooping729 = icmp slt i32 %iter_val728, %size724
  br i1 %keeplooping729, label %del_vec_elem730, label %merge744

del_vec_elem730:                                  ; preds = %check_size727
  %iter_val731 = load i32, i32* %iter725
  %dest_ptr732 = getelementptr %vec_t, %vec_t* %ptr723, i32 %iter_val731
  %struct_p733 = bitcast %vec_t* %dest_ptr732 to %example_inner*
  %inner_struct_field_delete734 = getelementptr inbounds %example_inner, %example_inner*
%struct_p733, i32 0, i32 1
  %vec_ptr_delete735 = getelementptr inbounds %example_inner_inner, %example_inner_inner*
 %inner_struct_field_delete734, i32 0, i32 1
  %ptr_p736 = getelementptr %vec_t, %vec_t* %vec_ptr_delete735, i32 0, i32 0
  %size_p737 = getelementptr %vec_t, %vec_t* %vec_ptr_delete735, i32 0, i32 1
  %ptr738 = load %vec_t*, %vec_t** %ptr_p736
  %size739 = load i32, i32* %size_p737
  %iszerosize740 = icmp slt i32 %size739, 1
  br i1 %iszerosize740, label %merge742, label %del_nonzerosize741

del_nonzerosize741:                               ; preds = %del_vec_elem730
  %20 = bitcast %vec_t* %ptr738 to i8*
  tail call void @free(i8* %20)
  br label %merge742

merge742:                                         ; preds = %del_vec_elem730, %del_nonzer
osize741
  %new_iter_val743 = add i32 %iter_val731, 1
  store i32 %new_iter_val743, i32* %iter725
  br label %check_size727

merge744:                                         ; preds = %check_size727
  %21 = bitcast %vec_t* %ptr723 to i8*
  tail call void @free(i8* %21)
  br label %mergeagain745

mergeagain745:                                    ; preds = %merge744, %merge720
  %ptr_p746 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 0
  %size_p747 = getelementptr %vec_t, %vec_t* %vec_alloc290, i32 0, i32 1
  %ptr748 = load %vec_t*, %vec_t** %ptr_p746
  %size749 = load i32, i32* %size_p747
  %iszerosize750 = icmp slt i32 %size749, 1
  br i1 %iszerosize750, label %merge752, label %del_nonzerosize751

del_nonzerosize751:                               ; preds = %mergeagain745
  %22 = bitcast %vec_t* %ptr748 to i8*
  tail call void @free(i8* %22)
  br label %merge752

merge752:                                         ; preds = %mergeagain745, %del_nonzeros
ize751
  %ptr_p753 = getelementptr %vec_t, %vec_t* %vec_alloc285, i32 0, i32 0
  %size_p754 = getelementptr %vec_t, %vec_t* %vec_alloc285, i32 0, i32 1
  %ptr755 = load %vec_t*, %vec_t** %ptr_p753
  %size756 = load i32, i32* %size_p754
  %iter757 = alloca i32
  store i32 0, i32* %iter757
  %iszerosize758 = icmp slt i32 %size756, 1
```

```
  br i1 %iszerosize758, label %mergeagain777, label %check_size759

check_size759:                                    ; preds = %merge774, %merge752
  %iter_val760 = load i32, i32* %iter757
  %keeplooping761 = icmp slt i32 %iter_val760, %size756
  br i1 %keeplooping761, label %del_vec_elem762, label %merge776

del_vec_elem762:                                  ; preds = %check_size759
  %iter_val763 = load i32, i32* %iter757
  %dest_ptr764 = getelementptr %vec_t, %vec_t* %ptr755, i32 %iter_val763
  %struct_p765 = bitcast %vec_t* %dest_ptr764 to %example_inner*
  %inner_struct_field_delete766 = getelementptr inbounds %example_inner, %example_inner*
%struct_p765, i32 0, i32 1
  %vec_ptr_delete767 = getelementptr inbounds %example_inner_inner, %example_inner_inner*
 %inner_struct_field_delete766, i32 0, i32 1
  %ptr_p768 = getelementptr %vec_t, %vec_t* %vec_ptr_delete767, i32 0, i32 0
  %size_p769 = getelementptr %vec_t, %vec_t* %vec_ptr_delete767, i32 0, i32 1
  %ptr770 = load %vec_t*, %vec_t** %ptr_p768
  %size771 = load i32, i32* %size_p769
  %iszerosize772 = icmp slt i32 %size771, 1
  br i1 %iszerosize772, label %merge774, label %del_nonzerosize773

del_nonzerosize773:                               ; preds = %del_vec_elem762
  %23 = bitcast %vec_t* %ptr770 to i8*
  tail call void @free(i8* %23)
  br label %merge774

merge774:                                         ; preds = %del_vec_elem762, %del_nonzer
osize773
  %new_iter_val775 = add i32 %iter_val763, 1
  store i32 %new_iter_val775, i32* %iter757
  br label %check_size759

merge776:                                         ; preds = %check_size759
  %24 = bitcast %vec_t* %ptr755 to i8*
  tail call void @free(i8* %24)
  br label %mergeagain777

mergeagain777:                                    ; preds = %merge776, %merge752
  %ptr_p778 = getelementptr %vec_t, %vec_t* %vec_alloc75, i32 0, i32 0
  %size_p779 = getelementptr %vec_t, %vec_t* %vec_alloc75, i32 0, i32 1
  %ptr780 = load %vec_t*, %vec_t** %ptr_p778
  %size781 = load i32, i32* %size_p779
  %iter782 = alloca i32
  store i32 0, i32* %iter782
  %iszerosize783 = icmp slt i32 %size781, 1
  br i1 %iszerosize783, label %mergeagain801, label %check_size784

check_size784:                                    ; preds = %merge798, %mergeagain777
  %iter_val785 = load i32, i32* %iter782
  %keeplooping786 = icmp slt i32 %iter_val785, %size781
  br i1 %keeplooping786, label %del_vec_elem787, label %merge800

del_vec_elem787:                                  ; preds = %check_size784
  %iter_val788 = load i32, i32* %iter782
  %dest_ptr789 = getelementptr %vec_t, %vec_t* %ptr780, i32 %iter_val788
  %struct_p790 = bitcast %vec_t* %dest_ptr789 to %example_inner_inner*
  %vec_ptr_delete791 = getelementptr inbounds %example_inner_inner, %example_inner_inner*
 %struct_p790, i32 0, i32 1
  %ptr_p792 = getelementptr %vec_t, %vec_t* %vec_ptr_delete791, i32 0, i32 0
  %size_p793 = getelementptr %vec_t, %vec_t* %vec_ptr_delete791, i32 0, i32 1
  %ptr794 = load %vec_t*, %vec_t** %ptr_p792
  %size795 = load i32, i32* %size_p793
  %iszerosize796 = icmp slt i32 %size795, 1
  br i1 %iszerosize796, label %merge798, label %del_nonzerosize797

del_nonzerosize797:                               ; preds = %del_vec_elem787
  %25 = bitcast %vec_t* %ptr794 to i8*
  tail call void @free(i8* %25)
  br label %merge798
```

```
merge798:                                         ; preds = %del_vec_elem787, %del_nonzer
osize797
  %new_iter_val799 = add i32 %iter_val788, 1
  store i32 %new_iter_val799, i32* %iter782
  br label %check_size784

merge800:                                         ; preds = %check_size784
  %26 = bitcast %vec_t* %ptr780 to i8*
  tail call void @free(i8* %26)
  br label %mergeagain801

mergeagain801:                                    ; preds = %merge800, %mergeagain777
  %inner_struct_field = getelementptr inbounds %example_inner, %example_inner* %a1, i32 0
, i32 1
  %vec = getelementptr inbounds %example_inner_inner, %example_inner_inner* %inner_struct
_field, i32 0, i32 1
  %ptr_p802 = getelementptr %vec_t, %vec_t* %vec, i32 0, i32 0
  %size_p803 = getelementptr %vec_t, %vec_t* %vec, i32 0, i32 1
  %ptr804 = load %vec_t*, %vec_t** %ptr_p802
  %size805 = load i32, i32* %size_p803
  %iszerosize806 = icmp slt i32 %size805, 1
  br i1 %iszerosize806, label %merge808, label %del_nonzerosize807

del_nonzerosize807:                               ; preds = %mergeagain801
  %27 = bitcast %vec_t* %ptr804 to i8*
  tail call void @free(i8* %27)
  br label %merge808

merge808:                                         ; preds = %mergeagain801, %del_nonzeros
ize807
  %vec809 = getelementptr inbounds %example_inner_inner, %example_inner_inner* %a4, i32 0
, i32 1
  %ptr_p810 = getelementptr %vec_t, %vec_t* %vec809, i32 0, i32 0
  %size_p811 = getelementptr %vec_t, %vec_t* %vec809, i32 0, i32 1
  %ptr812 = load %vec_t*, %vec_t** %ptr_p810
  %size813 = load i32, i32* %size_p811
  %iszerosize814 = icmp slt i32 %size813, 1
  br i1 %iszerosize814, label %merge816, label %del_nonzerosize815

del_nonzerosize815:                               ; preds = %merge808
  %28 = bitcast %vec_t* %ptr812 to i8*
  tail call void @free(i8* %28)
  br label %merge816

merge816:                                         ; preds = %merge808, %del_nonzerosize81
5
  %vec817 = getelementptr inbounds %example_inner_inner, %example_inner_inner* %a, i32 0,
 i32 1
  %ptr_p818 = getelementptr %vec_t, %vec_t* %vec817, i32 0, i32 0
  %size_p819 = getelementptr %vec_t, %vec_t* %vec817, i32 0, i32 1
  %ptr820 = load %vec_t*, %vec_t** %ptr_p818
  %size821 = load i32, i32* %size_p819
  %iszerosize822 = icmp slt i32 %size821, 1
  br i1 %iszerosize822, label %merge824, label %del_nonzerosize823

del_nonzerosize823:                               ; preds = %merge816
  %29 = bitcast %vec_t* %ptr820 to i8*
  tail call void @free(i8* %29)
  br label %merge824

merge824:                                         ; preds = %merge816, %del_nonzerosize82
3
  %final_retval = load i32, i32* %retval
  ret i32 %final_retval
}

declare noalias i8* @malloc(i32)

declare void @free(i8*)
```

```
attributes #0 = { argmemonly nounwind }
```

```
COMPILER_DIR = compiler
RUNTIME_DIR = runtime

all : compile cplibms

.PHONY : compile
compile:
	$(MAKE) -C $(COMPILER_DIR)
	$(MAKE) -C $(RUNTIME_DIR)


.PHONY : cplibms
cplibms:
	cp $(RUNTIME_DIR)/libms* $(COMPILER_DIR)

.PHONY : clean
clean:
	$(MAKE) -C $(COMPILER_DIR) clean
	$(MAKE) -C $(RUNTIME_DIR) clean
```

```
(* Abstract Syntax Tree and functions for printing it *)

type op = Add | Sub | Mult | Div | Equal | Neq | Less | Leq | Greater | Geq |
          And | Or

type uop = Neg | Not

type typ =
    Int
  | Double
  | Char
  | Bool
  | Void
  | Vector of typ
  | Job of typ
  | Struct of string
  | NoPtrStruct of string

type bind = typ * string

type expr =
    Literal of int
  | DoubleLit of float
  | BoolLit of bool
  | StringLit of string
  | Id of string
  | Binop of expr * op * expr
  | Unop of uop * expr
  | Assign of string * expr
  | Call of string * expr list
  | RemoteCall of string * expr list
  | Get of string
  | Cancel of string
  | Running of string
  | Finished of string
  | Failed of string
  | ListLiteral of expr list
  | VectorAccess of string * expr list
  | VectorAssign of string * expr list * expr
  | VectorRangeAccess of string * expr * expr
  | VectorSize of expr
  | StructFieldAccess of expr * string
  | StructFieldAssign of expr * string * expr
  | Concat of expr * expr
  | Noexpr

type stmt =
    Block of stmt list
  | Expr of expr
  | Return of expr
  | If of expr * stmt * stmt
  | For of expr * expr * expr * stmt
  | While of expr * stmt
  | VarDecl of typ * string
  | VarDeclAssign of typ * string * expr
  | VectorPushBack of expr * expr

type func_decl = {
    typ : typ;
    fname : string;
    formals : bind list;
    body : stmt list;
}

type struct_type_decl = {
    sname : string;
    blist : bind list;
}

type program = func_decl list * struct_type_decl list
```

```
(* Pretty-printing functions *)

let string_of_op = function
    Add -> "+"
  | Sub -> "-"
  | Mult -> "*"
  | Div -> "/"
  | Equal -> "=="
  | Neq -> "!="
  | Less -> "<"
  | Leq -> "<="
  | Greater -> ">"
  | Geq -> ">="
  | And -> "&&"
  | Or -> "||"

let string_of_uop = function
    Neg -> "-"
  | Not -> "!"

let rec string_of_typ = function
    Int -> "int"
  | Double -> "double"
  | Bool -> "bool"
  | Void -> "void"
  | Job(x) -> "job" ^ "<" ^ string_of_typ x ^ ">"
  | Vector(Char) -> "string"
  | Vector(x) -> "vector" ^ "<" ^ string_of_typ x ^ ">"
  | Struct(id) -> "struct " ^ id
  | _ -> "unknown type"

let map_and_concat f list = String.concat ";\n" (List.map f list)

let string_of_vdecl (t, id) = string_of_typ t ^ " " ^ id ^ ";\n"
let string_of_bind (t, id) = string_of_typ t ^ " " ^ id
let string_of_bind_list list = map_and_concat string_of_bind list

let rec string_of_expr = function
    Literal(l) -> string_of_int l
  | DoubleLit(f) ->  string_of_float f
  | StringLit(s) -> s
  | BoolLit(true) -> "true"
  | BoolLit(false) -> "false"
  | Id(s) -> s
  | Binop(e1, o, e2) ->
      string_of_expr e1 ^ " " ^ string_of_op o ^ " " ^ string_of_expr e2
  | Unop(o, e) -> string_of_uop o ^ string_of_expr e
  | Assign(v, e) -> v ^ " = " ^ string_of_expr e
  | ListLiteral(el) -> "[" ^ String.concat ", " (List.map string_of_expr el) ^ "]"
  | Call(f, el) ->
      f ^ "(" ^ String.concat ", " (List.map string_of_expr el) ^ ")"
  | RemoteCall(f, el) ->
      "remote " ^ f ^ "(" ^ String.concat ", " (List.map string_of_expr el) ^ ")"
  | Get(j) -> "get" ^ j
  | Cancel(j) -> "cancel" ^ j
  | Running(j) -> j ^ ".running"
  | Finished(j) -> j ^ ".finished"
  | Failed(j) -> j ^ ".failed"
  | VectorAccess(id, el) -> id ^ String.concat "" (List.map (fun el -> "["^(string_of_exp
r el)^"]") el)
  | VectorRangeAccess(id, e1, e2) -> id ^ "[" ^ string_of_expr e1 ^ ":" ^ string_of_expr
e2 ^ "]"
  | VectorAssign(id, e1_list, e2) -> id ^ String.concat "" (List.map (fun el -> "["^(stri
ng_of_expr el)^"]") e1_list)^ "=" ^ string_of_expr e2
  | StructFieldAccess(s, f) -> string_of_expr s ^ "->" ^ f
  | StructFieldAssign(s, f, v) -> string_of_expr s ^ "->" ^ f ^ " = " ^ string_of_expr v
  | VectorSize(e) -> "size " ^ string_of_expr e
  | Concat(e1, e2) -> string_of_expr e1 ^ " << " ^ string_of_expr e2
```

```
  | Noexpr -> ""

let rec string_of_stmt = function
    Block(stmts) ->
      "{\n" ^ String.concat "" (List.map string_of_stmt stmts) ^ "}\n"
  | Expr(expr) -> string_of_expr expr ^ ";\n";
  | Return(expr) -> "return " ^ string_of_expr expr ^ ";\n";
  | If(e, s, Block([])) -> "if (" ^ string_of_expr e ^ ")\n" ^ string_of_stmt s
  | If(e, s1, s2) ->  "if (" ^ string_of_expr e ^ ")\n" ^
      string_of_stmt s1 ^ "else\n" ^ string_of_stmt s2
  | For(e1, e2, e3, s) ->
      "for (" ^ string_of_expr e1  ^ " ; " ^ string_of_expr e2 ^ " ; " ^
      string_of_expr e3  ^ ") " ^ string_of_stmt s
  | While(e, s) -> "while (" ^ string_of_expr e ^ ") " ^ string_of_stmt s
  | VarDecl(typ, id) -> string_of_vdecl(typ, id)
  | VarDeclAssign(typ, id, e) -> string_of_vdecl(typ, id) ^ "=" ^ string_of_expr e
  | VectorPushBack(e1, e) -> string_of_expr e1 ^ "@" ^ string_of_expr e

let string_of_fdecl fdecl =
  string_of_typ fdecl.typ ^ " " ^
  fdecl.fname ^ "(" ^ String.concat ", " (List.map snd fdecl.formals) ^
  ")\n{\n" ^
  String.concat "" (List.map string_of_stmt fdecl.body) ^
  "}\n"

let string_of_struct_type_decl sdecl =
  "struct " ^ sdecl.sname ^ " {\n"^ string_of_bind_list sdecl.blist ^ "\n}"

let string_of_program (funcs, structs) =
  String.concat "\n" (List.map string_of_fdecl funcs) ^ String.concat "\n" (List.map stri
ng_of_struct_type_decl structs)
```

```
(* Code generation: translate takes a semantically checked AST and
produces LLVM IR

LLVM tutorial: Make sure to read the OCaml version of the tutorial

http://llvm.org/docs/tutorial/index.html

Detailed documentation on the OCaml LLVM library:

http://llvm.moe/
http://llvm.moe/ocaml/

*)

module S = String
module L = Llvm
module A = Ast

module StringMap = Map.Make(String)

let translate (functions, struct_decls) =
        let context = L.global_context () in
        let the_module = L.create_module context "M/s"
        and double_type = L.double_type context
  and char_t = L.i8_type context
        and i32_t = L.i32_type context
        and i8_t = L.i8_type context
        and i1_t = L.i1_type context
        and void_t = L.void_type context in
  let vec_t = L.named_struct_type context "vec_t" in
  ignore (L.struct_set_body vec_t [|L.pointer_type vec_t ; i32_t ; i32_t|] false) ;



  let struct_defs = Hashtbl.create 5000 in
  let rec create_struct_map map cur_idx = function
      [] -> StringMap.empty
    | [(typ, name)] -> StringMap.add name (typ, cur_idx) map
    | h::t -> let m = create_struct_map map cur_idx [h] in create_struct_map m (cur_idx+1
) t
  in
  let rec create_global_map global_map = function
      [] -> StringMap.empty
    |  [x] -> StringMap.add x.A.sname (create_struct_map StringMap.empty 0 x.A.blist) glo
bal_map
    | h::t -> let m = create_global_map global_map [h] in create_global_map m t
  in

  let global_map = create_global_map StringMap.empty struct_decls in

  let (*rec*) ltype_of_typ = function
      A.Int -> i32_t
    | A.Char -> char_t
    | A.Double -> double_type
    | A.Bool -> i1_t
    | A.Void -> void_t
    | A.Job(_) -> i32_t
    | A.Vector(_) -> vec_t (*ltype_of_typ t*)
        | A.Struct(name) -> Hashtbl.find struct_defs name
        | _ -> raise(Failure("unrecognized type")) in

  let struct_def struct_decl =
      let struct_body = List.fold_left (fun a (typ, _(*name*)) -> Array.append a [|ltype_
of_typ typ|]) [||] struct_decl.A.blist
      in let struct_name = L.named_struct_type context struct_decl.A.sname
      in ignore(L.struct_set_body struct_name struct_body false); struct_name;
    in ignore (List.map (fun struct_decl ->
        let llvm_def = struct_def struct_decl in Hashtbl.add struct_defs struct_decl.A.sn
ame llvm_def)
        struct_decls) ;
```

```
         (* Declare printf(), which the print built-in function will call *)
        let printf_t = L.var_arg_function_type i32_t [| L.pointer_type i8_t |] in
        let printf_func = L.declare_function "printf" printf_t the_module in

        (* declare master library functions *)
    let start_job_t = L.function_type i32_t [| i32_t ; L.pointer_type i8_t ;
        i32_t |] in
    let start_job_func = L.declare_function "start_job" start_job_t the_module in

    let reap_job_t = L.function_type void_t [| i32_t ;
        L.pointer_type (L.pointer_type i8_t) ; L.pointer_type i32_t |] in
    let reap_job_func = L.declare_function "reap_job" reap_job_t the_module in

    let job_status_t = L.function_type i32_t [| i32_t |] in
    let get_job_status_func = L.declare_function "get_job_status" job_status_t the_module
 in

    let cancel_job_t = L.function_type i32_t [| i32_t |] in
    let cancel_job_func = L.declare_function "cancel_job" cancel_job_t the_module in

  let memcpy_t = L.function_type void_t [|L.pointer_type i8_t; L.pointer_type i8_t; i32_t
; i32_t; i1_t|] in
  let memcpy_func = L.declare_function "llvm.memcpy.p0i8.p0i8.i32" memcpy_t the_module in

        (* Define each function (arguments and return type) so we can call it *)
        (*master_decls is a map that stores key=main, value=(llvm's def of main, AST of m
aster to map *)

  let get_ordinal f = let rec get_index e n = function
    | [] -> -1
    | x::_ when (x.A.fname = e) -> n
    | x::rest when (x.A.fname = "master") -> get_index e n rest
    | _::rest -> get_index e (n+1) rest in
  L.const_int i32_t (get_index f 0 functions + 1) in

        let zeroth = L.const_int i32_t 0 in
        let first = L.const_int i32_t 1 in
        let second = L.const_int i32_t 2 in

        (* serialization code *)
        let serialize builder evals the_function typs =
                let rec compute_size (b, size) e typ = (match typ with
                    A.Vector(t) ->
                        let e_store = L.build_alloca vec_t "e_store" b in
                        ignore (L.build_store e e_store b) ;
                        let iter = L.build_alloca i32_t "vec_sz_var" b in
                        ignore (L.build_store (L.const_null i32_t) iter b);
                        let vec_ptr = L.build_gep e_store [|zeroth; zeroth|] "vec_ptr" b
in
                        let ptr = L.build_load vec_ptr "ptr" b in
                        let vec_sz_p = L.build_gep e_store [|zeroth; first|] "vec_sz_p" b
 in
                        let vec_sz = L.build_load vec_sz_p "vec_sz" b in
                        let size_val = L.build_load size "size_val" b in
                        let size_add = L.build_add size_val (L.const_int i32_t 4) "size_p
lus4" b in
                        ignore (L.build_store size_add size b);
                        let pred_bb = L.append_block context "check_size" the_function in
                                let pred_builder = L.builder_at_end context pred_bb in
                        let iter_val = L.build_load iter "iter_val" pred_builder in
                        let keeplooping = L.build_icmp L.Icmp.Slt iter_val vec_sz "keeplo
oping" pred_builder in
                                let sz_vec_elem_bb = L.append_block context "del_vec_elem
" the_function in
                        let sz_vec_elem_builder = L.builder_at_end context sz_vec_elem_bb
 in
                        let iter_val = L.build_load iter "iter_val" sz_vec_elem_builder i
n
                        let origin_ptr = L.build_gep ptr [|iter_val|] "dest_ptr" sz_vec_e
```

```
lem_builder in
                        let origin_ptr_cast = L.build_bitcast origin_ptr (L.pointer_type
(ltype_of_typ t)) "origin_ptr_cast" sz_vec_elem_builder in
                        let inner_vec = L.build_load origin_ptr_cast "inner_vec" sz_vec_e
lem_builder in
                        let (newbuilder, _) = compute_size (sz_vec_elem_builder, size) in
ner_vec t in
                        let new_iter_val = L.build_add iter_val (L.const_int i32_t 1) "ne
w_iter_val" newbuilder in
                        ignore (L.build_store new_iter_val iter newbuilder);
                                let merge_bb = L.append_block context "merge" the_functio
n in
                        let merge_builder = L.builder_at_end context merge_bb in
                                ignore (L.build_cond_br keeplooping sz_vec_elem_bb merge_
bb pred_builder);
                                    ignore (L.build_br pred_bb newbuilder);
                        ignore (L.build_br pred_bb b);
                        (merge_builder, size)
                  | A.Struct(n) ->
                        let e_store = L.build_alloca (ltype_of_typ (A.Struct(n))) "store_
struct" b in
                        ignore (L.build_store e e_store b) ;
                        let struct_def_map = StringMap.find n global_map in
                        let bindings = StringMap.bindings struct_def_map in
                        let size_elem (b, size) binding =
                            let (_, (fieldtyp, idx)) = binding in
                            let field_ptr = L.build_struct_gep e_store idx "field_ptr" b
in
                            let field = L.build_load field_ptr "field" b in
                            let (newbuilder, _) = compute_size (b, size) field fieldtyp i
n
                            (newbuilder, size) in
                        List.fold_left size_elem (b, size) bindings
                  | t -> let size_val = L.build_load size "size_val" b in
                                let size_add = L.build_add size_val (L.build_trunc (L.s
ize_of (ltype_of_typ t)) i32_t "trunc" b) "size_add" b in
                                ignore (L.build_store size_add size b) ;
                                (b, size)) in
                let size = L.build_alloca i32_t "size" builder in
                ignore (L.build_store (L.const_null i32_t) size builder) ;
                let (nnbuilder, serializelen_p) = List.fold_left2 compute_size (builder,
size) evals typs in
                let serializelen = L.build_load serializelen_p "serializelen" nnbuilder i
n
                let serializebuf = L.build_alloca (L.pointer_type i8_t) "serializebuf" nn
builder in
                let serializebuf_val = L.build_array_malloc i8_t serializelen "serializea
rg" nnbuilder in
                ignore (L.build_store serializebuf_val serializebuf nnbuilder);

                let rec serialize_next (ptr, b) e typ =
                  let target = L.build_load ptr "target" b in
                  (match typ with
                    A.Vector(typ) ->
                  let target_size_p = L.build_bitcast target (L.pointer_type i32_t) "size
_32" b in
                  let e_store = L.build_alloca vec_t "e_store" b in
                  ignore (L.build_store e e_store b) ;
                  let vec_ptr = L.build_gep e_store [|zeroth; zeroth|] "vec_ptr" b in
                  let ptr_v = L.build_load vec_ptr "ptr" b in
                  let vec_sz_p = L.build_gep e_store [|zeroth; first|] "vec_sz_p" b in
                  let vec_sz = L.build_load vec_sz_p "vec_sz" b in
                  ignore (L.build_store vec_sz target_size_p b) ;
                  ignore (L.build_store (L.build_gep target [|L.const_int i32_t 4|] "next
_target" b) ptr b);
                  let iter = L.build_alloca i32_t "vec_sz_var" b in
                  ignore (L.build_store (L.const_null i32_t) iter b);
                  let pred_bb = L.append_block context "check_size" the_function in
                          let pred_builder = L.builder_at_end context pred_bb in
                  let iter_val = L.build_load iter "iter_val" pred_builder in
```

```
                  let keeplooping = L.build_icmp L.Icmp.Slt iter_val vec_sz "keeplooping"
 pred_builder in
                        let sz_vec_elem_bb = L.append_block context "del_vec_elem" the_
function in
                  let sz_vec_elem_builder = L.builder_at_end context sz_vec_elem_bb in
                  let iter_val = L.build_load iter "iter_val" sz_vec_elem_builder in
                  let ptr_v_cast = L.build_bitcast ptr_v (L.pointer_type (ltype_of_typ ty
p)) "ptr_v_cast" sz_vec_elem_builder in
                  let origin_ptr = L.build_gep ptr_v_cast [|iter_val|] "dest_ptr" sz_vec_
elem_builder in
                  let inner_elem = L.build_load origin_ptr "inner_elem" sz_vec_elem_build
er in
                  let (_, newbuilder) = serialize_next (ptr, sz_vec_elem_builder) inner_e
lem typ  in
                  let new_iter_val = L.build_add iter_val (L.const_int i32_t 1) "new_iter
_val" newbuilder in
                  ignore (L.build_store new_iter_val iter newbuilder);
                          let merge_bb = L.append_block context "merge" the_function in
                  let merge_builder = L.builder_at_end context merge_bb in
                          ignore (L.build_cond_br keeplooping sz_vec_elem_bb merge_bb pre
d_builder);
                              ignore (L.build_br pred_bb newbuilder);
                  ignore (L.build_br pred_bb b);
                  (ptr, merge_builder)
                  | A.Struct(n) ->
                        let e_store = L.build_alloca (ltype_of_typ (A.Struct(n))) "store_
struct" b in
                        ignore (L.build_store e e_store b);
                        let struct_def_map = StringMap.find n global_map in
                        let bindings = StringMap.bindings struct_def_map in
                        let serialize_elem b binding =
                            let (_, (fieldtyp, idx)) = binding in
                            let field_ptr = L.build_struct_gep e_store idx "field_ptr" b
in
                            let field = L.build_load field_ptr "field" b in
                            let (_, newbuilder) = serialize_next (ptr, b) field fieldtyp
in
                            newbuilder in
                        (ptr, List.fold_left serialize_elem b bindings)
                    | typ -> let target_typ_p = L.build_bitcast target (L.pointer_type (l
type_of_typ typ)) "elem_nn" b in
                             ignore (L.build_store e target_typ_p b) ;
                             ignore (L.build_store (L.build_gep target [|L.build_trunc (L
.size_of (ltype_of_typ typ)) i32_t "trunc" b|] "newtarget" b) ptr b) ;
                             (ptr, b)) in
                (* serialize each argument into serializebuf using list fold magic *)
                let (_, finalbuilder) = List.fold_left2 serialize_next (serializebuf, nnb
uilder) evals typs in
                (*returns serialize buf*)
                (serializebuf_val, serializelen, finalbuilder) in

        let rec deserialize (lst, ptr, the_function, builder) typ =
            let ptr_val = L.build_load ptr "ptr_val" builder in
            (match typ with
                A.Vector(t) -> let size = L.build_bitcast ptr_val (L.pointer_type i32_t)
"size" builder in
                               let size_val = L.build_load size "size_val" builder in
                               let iter = L.build_alloca i32_t "iter" builder in
                               ignore (L.build_store (L.const_null i32_t) iter builder);
                               let new_ptr = L.build_gep ptr_val [|L.const_int i32_t 4|]
"new_ptr" builder in
                               ignore (L.build_store new_ptr ptr builder);
                               let vec_ptr = L.build_alloca vec_t "vec_ptr" builder in
                               let vec_ptr_p = L.build_gep vec_ptr [|zeroth; zeroth|] "ve
c_ptr_p" builder in
                               let vec_ptr_sz = L.build_gep vec_ptr [|zeroth; first|] "ve
c_ptr_sz" builder in
                               let vec_ptr_ln = L.build_gep vec_ptr [|zeroth; second|] "v
ec_ptr_ln" builder in
                               let buf = L.build_array_malloc (ltype_of_typ t) size_val "
```

```
buf" builder in
                                let buf_cast = L.build_bitcast buf (L.pointer_type vec_t)
"buf_cast" builder in
                                ignore (L.build_store buf_cast vec_ptr_p builder);
                                ignore (L.build_store size_val vec_ptr_sz builder);
                                ignore (L.build_store size_val vec_ptr_ln builder);
                                let pred_bb = L.append_block context "check_size" the_func
tion in
                                        let pred_builder = L.builder_at_end context pred_b
b in
                                let iter_val = L.build_load iter "iter_val" pred_builder i
n
                                let keeplooping = L.build_icmp L.Icmp.Slt iter_val size_va
l "keeplooping" pred_builder in
                                        let sz_vec_elem_bb = L.append_block context "del_v
ec_elem" the_function in
                                let sz_vec_elem_builder = L.builder_at_end context sz_vec_
elem_bb in
                                let iter_val = L.build_load iter "iter_val" sz_vec_elem_bu
ilder in
                                let ptr_v_cast = L.build_bitcast buf (L.pointer_type (ltyp
e_of_typ t)) "ptr_v_cast" sz_vec_elem_builder in
                                let origin_ptr = L.build_gep ptr_v_cast [|iter_val|] "dest
_ptr" sz_vec_elem_builder in
                                let (res_l, _, _, newbuilder) = deserialize ([], ptr, the_
function, sz_vec_elem_builder) t in
                                let res = (match res_l with [r] -> r | _ -> raise(Failure(
"expected to only deserialize one thing")) ) in
                                ignore (L.build_store res origin_ptr newbuilder);
                                let new_iter_val = L.build_add iter_val (L.const_int i32_t
 1) "new_iter_val" newbuilder in
                                ignore (L.build_store new_iter_val iter newbuilder);
                                        let merge_bb = L.append_block context "merge" the_
function in
                                let merge_builder = L.builder_at_end context merge_bb in
                                        ignore (L.build_cond_br keeplooping sz_vec_elem_bb
 merge_bb pred_builder);
                                            ignore (L.build_br pred_bb newbuilder);
                                ignore (L.build_br pred_bb builder);
                                ((L.build_load vec_ptr "vec" merge_builder)::lst, ptr, the
_function, merge_builder)
                  | A.Struct(n) ->
                        let e_store = L.build_alloca (ltype_of_typ (A.Struct(n))) "store_
struct" builder in
                        let struct_def_map = StringMap.find n global_map in
                        let bindings = StringMap.bindings struct_def_map in
                        let serialize_field b binding =
                            let (_, (fieldtyp, idx)) = binding in
                            let field_ptr = L.build_struct_gep e_store idx "field_ptr" b
in
                            let (field, _, _, newbuilder) = deserialize ([], ptr, the_fun
ction, b) fieldtyp in
                            let res = (match field with [r] -> r | _ -> raise(Failure("ex
pected to only deserialize one thing"))) in
                            ignore (L.build_store res field_ptr newbuilder) ;
                            newbuilder in
                        let nbuilder = List.fold_left serialize_field builder bindings in
                        ((L.build_load e_store "struct" nbuilder)::lst, ptr, the_function
, nbuilder)
                | typ -> let src = L.build_bitcast ptr_val (L.pointer_type (ltype_of_typ ty
p)) "src" builder in
                       let new_ptr_val = L.build_gep ptr_val [|L.build_trunc (L.size_of (
ltype_of_typ typ)) i32_t "trunc" builder|] "new_ptr_val" builder in
                       ignore (L.build_store new_ptr_val ptr builder) ;
                       ((L.build_load src "deserialized_typ" builder)::lst, ptr, the_func
tion, builder) ) in

              let rec struct_delete ptr struct_def_map builder the_function  =
                    let bindings = StringMap.bindings struct_def_map in
                    let struct_field_delete_helper builder binding  =
```

```
                    let (_, (fieldtyp, idx)) = binding in
                        (match fieldtyp with
                            A.Vector(in_typ) -> let vec_ptr = L.build_struct_gep ptr id
x "vec_ptr_delete" builder in
                            (*raise(Failure (L.string_of_llvalue((vec_ptr)) ^ "; "))*)

                                                vector_delete vec_ptr builder the_functio
n in_typ
                          | A.Struct(in_typ) ->
                            let field_val = L.build_struct_gep ptr idx "inner_struct_
field_delete" builder in
                            (*raise(Failure (L.string_of_llvalue((field_val)) ^ "; ")
)*)

                            let field_struct_def_map = StringMap.find in_typ global_m
ap  in
                            struct_delete field_val field_struct_def_map builder the_
function

                          | _ -> builder) in
                    List.fold_left struct_field_delete_helper builder bindings;
    and vector_delete vec builder the_function = function
        A.Vector(typ) -> let ptr_p = L.build_gep vec [|zeroth; zeroth|] "ptr_p" builder i
n
            let size_p = L.build_gep vec [|zeroth; first|] "size_p" builder in
            let ptr = L.build_load ptr_p "ptr" builder in
            let size = L.build_load size_p "size" builder in
            let iter = L.build_alloca i32_t "iter" builder in
            ignore (L.build_store (L.const_int i32_t 0) iter builder) ;
            let zerosize = L.build_icmp L.Icmp.Slt size (L.const_int i32_t 1) "iszerosize
" builder in
            let pred_bb = L.append_block context "check_size" the_function in
                      let pred_builder = L.builder_at_end context pred_bb in
              let iter_val = L.build_load iter "iter_val" pred_builder in
              let keeplooping = L.build_icmp L.Icmp.Slt iter_val size "keeplooping" pred_
builder in
                    let del_vec_elem_bb = L.append_block context "del_vec_elem" the_funct
ion in
              let del_vec_elem_builder = L.builder_at_end context del_vec_elem_bb in
              let iter_val = L.build_load iter "iter_val" del_vec_elem_builder in
              let origin_ptr = L.build_gep ptr [|iter_val|] "dest_ptr" del_vec_elem_build
er in
              let newbuilder = vector_delete origin_ptr del_vec_elem_builder the_function
 typ in
                  let new_iter_val = L.build_add iter_val (L.const_int i32_t 1) "new_iter
_val" newbuilder in
                  ignore (L.build_store new_iter_val iter newbuilder);
                  let merge_bb = L.append_block context "merge" the_function in
              let merge_builder = L.builder_at_end context merge_bb in
              ignore(L.build_free ptr merge_builder) ;
            let mergeagain_bb = L.append_block context "mergeagain" the_function in
              let mergeagain_builder = L.builder_at_end context mergeagain_bb in
          ignore (L.build_cond_br zerosize mergeagain_bb pred_bb builder);
          ignore (L.build_cond_br keeplooping del_vec_elem_bb merge_bb pred_builder);
        ignore (L.build_br pred_bb newbuilder);
            ignore (L.build_br mergeagain_bb merge_builder);
            mergeagain_builder

      | A.Struct(inner_t) -> let ptr_p = L.build_gep vec [|zeroth; zeroth|] "ptr_p" build
er in
            let size_p = L.build_gep vec [|zeroth; first|] "size_p" builder in
            let ptr = L.build_load ptr_p "ptr" builder in
            let size = L.build_load size_p "size" builder in
            let iter = L.build_alloca i32_t "iter" builder in
            ignore (L.build_store (L.const_int i32_t 0) iter builder) ;
            let zerosize = L.build_icmp L.Icmp.Slt size (L.const_int i32_t 1) "iszerosize
" builder in
            let pred_bb = L.append_block context "check_size" the_function in
          let pred_builder = L.builder_at_end context pred_bb in
              let iter_val = L.build_load iter "iter_val" pred_builder in
```

```
              let keeplooping = L.build_icmp L.Icmp.Slt iter_val size "keeplooping" pred_
builder in
        let del_vec_elem_bb = L.append_block context "del_vec_elem" the_function in
              let del_vec_elem_builder = L.builder_at_end context del_vec_elem_bb in
              let iter_val = L.build_load iter "iter_val" del_vec_elem_builder in
              let origin_ptr = L.build_gep ptr [|iter_val|] "dest_ptr" del_vec_elem_build
er in
              let struct_def_map = StringMap.find inner_t global_map in
                  let llvm_struct_type = ltype_of_typ(A.Struct(inner_t)) in
                  let struct_p = L.build_bitcast origin_ptr (L.pointer_type llvm_struct_t
ype) "struct_p" del_vec_elem_builder in
                  let newbuilder = struct_delete struct_p struct_def_map del_vec_elem_bui
lder the_function in
                  let new_iter_val = L.build_add iter_val (L.const_int i32_t 1) "new_iter
_val" newbuilder in
                  ignore (L.build_store new_iter_val iter newbuilder);
                  let merge_bb = L.append_block context "merge" the_function in
              let merge_builder = L.builder_at_end context merge_bb in
              ignore(L.build_free ptr merge_builder) ;
            let mergeagain_bb = L.append_block context "mergeagain" the_function in
              let mergeagain_builder = L.builder_at_end context mergeagain_bb in
        ignore (L.build_cond_br zerosize mergeagain_bb pred_bb builder);
        ignore (L.build_cond_br keeplooping del_vec_elem_bb merge_bb pred_builder);
      ignore (L.build_br pred_bb newbuilder);
            ignore (L.build_br mergeagain_bb merge_builder);
            mergeagain_builder
        | _ -> let ptr_p = L.build_gep vec [|zeroth; zeroth|] "ptr_p" builder in
            let size_p = L.build_gep vec [|zeroth; first|] "size_p" builder in
            let ptr = L.build_load ptr_p "ptr" builder in
            let size = L.build_load size_p "size" builder in
            let iszerosize = L.build_icmp L.Icmp.Slt size (L.const_int i32_t 1) "iszerosi
ze" builder in
            let nonzerosize_bb = L.append_block context "del_nonzerosize" the_function in
              let nonzerosize_builder = L.builder_at_end context nonzerosize_bb in
              ignore (L.build_free ptr nonzerosize_builder);
            let merge_bb = L.append_block context "merge" the_function in
              let merge_builder = L.builder_at_end context merge_bb in
            ignore(L.build_br merge_bb nonzerosize_builder);
            ignore(L.build_cond_br iszerosize merge_bb nonzerosize_bb builder);
            merge_builder
        in
        let function_decls =
                let function_decl m fdecl =
                        let name = fdecl.A.fname
                        (* formal_type should be an empty array list*)
                        and formal_type = Array.of_list (List.map (fun (t,_) -> ltype_of_
typ t) fdecl.A.formals)
                        (*ftype is int*)
                in let ftype = L.function_type (ltype_of_typ fdecl.A.typ) formal_type in
                (* stores key=main, value=(llvm's def of main, AST of master to map *)
                        StringMap.add name (L.define_function name ftype the_module, fdec
l) m in
                List.fold_left function_decl StringMap.empty functions in

        (* Fill in the body of the given function *)
        let build_function_body fdecl =
                let (the_function, _) = StringMap.find fdecl.A.fname function_decls in
                (* builder used always points to end of block*)
                let builder = L.builder_at_end context (L.entry_block the_function) in

                let int_format_str = L.build_global_stringptr "%d\n" "fmt" builder in
                let double_format_str = L.build_global_stringptr "%f\n" "fmt" builder in

                let true_str = L.build_global_stringptr "true\n" "true_str" builder in
                let false_str = L.build_global_stringptr "false\n" "false_str" builder in
    let string_str = L.build_global_stringptr "%s\n" "false_str" builder in

        let vector_type_map = Hashtbl.create 5000 in
        let add_vector_type s typ = Hashtbl.add vector_type_map s typ in
        let is_vector n = Hashtbl.mem vector_type_map n in
```

```
        let get_vector_type n = Hashtbl.find vector_type_map n in
        let string_lit_map = Hashtbl.create 5000 in
        let add_to_string_lit_cleanup ptr =
          Hashtbl.add string_lit_map ptr ""
        in

    let struct_type_map = Hashtbl.create 5000 in
    let add_struct_type s typ = Hashtbl.add struct_type_map s typ in
    let get_struct_type n = try (Hashtbl.find struct_type_map n) with Not_found -> raise(
Failure("fooooo")) in
    let is_struct id = Hashtbl.mem struct_type_map id in
                (* adds formals and local vars to local_var_map
                        set user var name to llvm's param pointer
                *)
                let local_var_map = Hashtbl.create 5000 in
    let llval_to_id_map = Hashtbl.create 5000 in
                let add_formal (t, n) p = (match t with
                A.Vector(typ) -> ignore (add_vector_type n typ)
              | A.Struct(typ) -> ignore (add_struct_type n typ)
              | _ -> ()) ; L.set_value_name n p;
                let local = L.build_alloca (ltype_of_typ t) n builder in
                        ignore (L.build_store p local builder); Hashtbl.replace local_var
_map n local in
                ignore (List.iter2 add_formal fdecl.A.formals (Array.to_list (L.params th
e_function))) ;

                let add_var_init id llvalue =
                        let _ = Hashtbl.add llval_to_id_map llvalue id in Hashtbl.add loc
al_var_map id llvalue
                in
                let lookup s =
      if Hashtbl.mem local_var_map s then Hashtbl.find local_var_map s
    else raise (Failure("cannot find symbol in local_var_map"))
                in
        let job_type_map = Hashtbl.create 5000 in
        let add_job_type s typ = Hashtbl.add job_type_map s typ in

    let vector_decl(id, typ, builder) =
          let vec_alloc = L.build_alloca (vec_t) "vec_alloc" builder in
          let ptr = L.build_gep vec_alloc [|zeroth; zeroth|] "ptr" builder in
          ignore (L.build_store (L.const_pointer_null (L.pointer_type vec_t))  ptr builde
r) ;
          let size = L.build_gep vec_alloc [|zeroth; first|] "size" builder in
          ignore (L.build_store (L.const_null i32_t) size builder) ;
          let len = L.build_gep vec_alloc [|zeroth; second|] "len" builder in
          ignore (L.build_store (L.const_null i32_t) len builder) ;
          add_vector_type id typ ; vec_alloc
    in

    let rec string_store s i n vec builder =
          (match i with
            | i when (i == n) ->
              let ith = L.const_int i32_t i in
              let index = L.build_gep vec [|ith|] "index" builder in
              ignore (L.build_store (L.const_null i8_t) index builder) ;
            | _ ->
              let c = S.get s i in
              let c1 = L.const_int i8_t (int_of_char c) in
              let ith = L.const_int i32_t i in
              let index = L.build_gep vec [|ith|] "index" builder in
              ignore (L.build_store c1 index builder) ;
              string_store s (i+1) n vec builder

          )
    in

    let string_decl s builder =
          let s_length = S.length s in
          let vec_alloc = L.build_alloca vec_t "vec_alloc" builder in
          let ptr = L.build_gep vec_alloc [|zeroth; zeroth|] "ptr" builder in
```

```
          let s_length_null = L.const_int i32_t (s_length + 1) in
          let oldvec = L.build_array_malloc char_t s_length_null "oldvec" builder in
          let newvec = L.build_bitcast oldvec (L.pointer_type vec_t) "newvec" builder in
          ignore (L.build_store newvec ptr builder) ;
          let size = L.build_gep vec_alloc [|zeroth; first|] "size" builder in
          ignore (L.build_store s_length_null size builder) ;
          let len = L.build_gep vec_alloc [|zeroth; second|] "len" builder in
          ignore (L.build_store s_length_null len builder) ;
          ignore (string_store s 0 s_length oldvec builder) ;
          L.build_load vec_alloc "vec" builder

    in

    let string_concat vect1_orig vect2_orig builder =
        let vect1 = L.build_alloca vec_t "tmp_vect1" builder in
        let vect2 = L.build_alloca vec_t "tmp_vect1" builder in
        let _ = L.build_store vect1_orig vect1 builder in
        let _ = L.build_store vect2_orig vect2 builder in
        let size1 = L.build_gep vect1 [|zeroth; first|] "size_1" builder in
        let size1 = L.build_load size1 "loaded_size_1" builder in
        let ptr1 = L.build_gep vect1 [|zeroth; zeroth|] "ptr_1" builder in
        (*let len1 = L.build_gep vect1 [|zeroth; second|] "len_1" builder in*)
        let size2 = L.build_gep vect2 [|zeroth; first|] "size_2" builder in
        let size2 = L.build_load size2 "loaded_size_2" builder in
        let ptr2 = L.build_gep vect2 [|zeroth; zeroth|] "ptr_2" builder in
        (*let len2 = L.build_gep vect2 [|zeroth; second|] "len_2" builder in*)
        let newsize = L.build_add size1 size2 "new_size" builder in
        let newsize = L.build_sub newsize (L.const_int i32_t 1) "new_size_sub" builder in
        let newvec_alloc = L.build_alloca vec_t "new_vec_alloc" builder in
        let new_ptr = L.build_gep newvec_alloc [|zeroth; zeroth|] "new_ptr" builder in
        let new_size_var = L.build_gep newvec_alloc [|zeroth; first|] "new_size" builder
in
        let new_leng_var = L.build_gep newvec_alloc [|zeroth; second|] "new_leng" builder
 in
        let malloc_inner = L.build_array_malloc char_t newsize "malloc_string" builder in
        let malloc_inner_cast = L.build_bitcast malloc_inner (L.pointer_type vec_t) "mall
oc_inner_cast" builder in
        ignore (L.build_store malloc_inner_cast new_ptr builder);
        let cast_vect1 = L.build_bitcast ptr1 (L.pointer_type (L.pointer_type i8_t)) "cas
t_vect1" builder in
        let cast_vect2 = L.build_bitcast ptr2 (L.pointer_type (L.pointer_type i8_t)) "cas
t_vect2" builder in
        let cast_newvec = L.build_bitcast new_ptr (L.pointer_type (L.pointer_type i8_t))
"cast_newvec" builder in
        let cast_vect1 = L.build_load cast_vect1 "cast_vect1" builder in
        let cast_vect2 = L.build_load cast_vect2 "cast_vect1" builder in
        let cast_newvec = L.build_load cast_newvec "cast_vect1" builder in
        let leng_to_cp = L.build_sub size1 (L.const_int i32_t 1) "copy_leng_first" builde
r in
        let _ = L.build_call memcpy_func [|cast_newvec; cast_vect1; leng_to_cp; L.const_i
nt i32_t 1; L.const_int i1_t 1|] "" builder in
        let move_cp_ptr = L.build_gep cast_newvec [|leng_to_cp|] "move_cp_ptr" builder in

        let _ = L.build_call memcpy_func [|move_cp_ptr; cast_vect2; size2; L.const_int i3
2_t 1; L.const_int i1_t 1|] "" builder in
       (*
        raise(Failure (L.string_of_llvalue((move_cp_ptr)) ^ "; " ^ L.string_of_llvalue((v
ect1))))
        *)
        ignore(L.build_store newsize new_leng_var builder);
        ignore(L.build_store newsize new_size_var builder);
        L.build_load newvec_alloc "newvec" builder

      in
```

```
              let rec struct_copy lptr rptr struct_def_map builder =
                let bindings = StringMap.bindings struct_def_map in
                let struct_field_assign_helper builder binding  =
                  let (_, (fieldtyp, idx)) = binding in
                    (match fieldtyp with
                        A.Vector(in_typ) ->
                  let lvec_ptr = L.build_struct_gep lptr idx "l_field_vec_cp_assign" buil
der in
                  let rvec_ptr = L.build_struct_gep rptr idx "r_field_vec_cp_assign" buil
der in
                  let load_rptr = L.build_load rvec_ptr "loaded_rptr" builder in
                  let builder3 = builder in
                  let temp = L.build_alloca vec_t "temp" builder3 in
                  let _ = L.build_store load_rptr temp builder3 in
                    let (vec_cpy, builder4) = vector_copy temp builder3 in_typ in
                    let sizeof = L.build_trunc (L.size_of (ltype_of_typ fieldtyp)) i32_t
"sizeof" builder4 in
                    let temp = L.build_alloca vec_t "temp" builder4 in
                    let _ = L.build_store vec_cpy temp builder4 in
                    let src_cast = L.build_bitcast temp (L.pointer_type i8_t) "src_cast"
builder4 in
                    let dst_cast = L.build_bitcast lvec_ptr (L.pointer_type i8_t) "dst_ca
st" builder4 in
                    let _ = L.build_call memcpy_func [|dst_cast; src_cast; sizeof; L.cons
t_int i32_t 1; L.const_int i1_t 1|] "" builder4
                  in builder4
                | A.Struct(in_typ) ->
                    let l_field_val = L.build_struct_gep lptr idx "l_inner_struct_field"
builder in
                    let r_field_val = L.build_struct_gep rptr idx "r_inner_struct_field"
builder in
                    let field_struct_def_map = StringMap.find in_typ global_map  in
                    let builder = struct_copy l_field_val r_field_val field_struct_def_ma
p builder in builder;
                | _ ->
                    let l_field_ptr = L.build_struct_gep lptr idx "l_field_cp_assign" bui
lder in
                    let r_field_ptr = L.build_struct_gep rptr idx "r_field_cp_assign" bui
lder in
                    let load_rptr = L.build_load r_field_ptr "loaded_rptr" builder in
                    ignore(L.build_store load_rptr l_field_ptr builder); builder)
                  in
              let builder = List.fold_left struct_field_assign_helper builder bindings in b
uilder;

      and vector_copy vec builder outer_type = match outer_type with
          A.Vector(typ) -> let ptr_p = L.build_gep vec [|zeroth; zeroth|] "ptr_p" builder i
n
              let size_p = L.build_gep vec [|zeroth; first|] "size_p" builder in
              let len_p = L.build_gep vec [|zeroth; second|] "len_p" builder in
              let ptr = L.build_load ptr_p "ptr" builder in
              let size = L.build_load size_p "size" builder in
              let len = L.build_load len_p "len" builder in
              let iter = L.build_alloca i32_t "iter" builder in
              ignore (L.build_store (L.const_int i32_t 0) iter builder) ;
              let out_vec = L.build_alloca vec_t "out_vec" builder in
              let out_ptr = L.build_gep out_vec [|zeroth; zeroth|] "out_ptr" builder in
              let out_size = L.build_gep out_vec [|zeroth; first|] "out_size" builder in
              let out_len = L.build_gep out_vec [|zeroth; second|] "out_len" builder in
              ignore (L.build_store size out_size builder) ;
              ignore (L.build_store len out_len builder) ;
              let zerosize = L.build_icmp L.Icmp.Slt size (L.const_int i32_t 1) "iszerosize
" builder in
              let zerosize_bb = L.append_block context "zerosize" the_function in
                let zerosize_builder = L.builder_at_end context zerosize_bb in
                ignore (L.build_store (L.const_pointer_null (L.pointer_type vec_t)) out_ptr
 zerosize_builder) ;
              let nonzerosize_bb = L.append_block context "nonzerosize" the_function in
```

```
              let nonzerosize_builder = L.builder_at_end context nonzerosize_bb in
              let newvec = L.build_array_malloc vec_t len "newvec" nonzerosize_builder in
              ignore (L.build_store newvec out_ptr nonzerosize_builder) ;
            let pred_bb = L.append_block context "check_size" the_function in
                      let pred_builder = L.builder_at_end context pred_bb in
              let iter_val = L.build_load iter "iter_val" pred_builder in
              let keeplooping = L.build_icmp L.Icmp.Slt iter_val size "keeplooping" pred_
builder in
                    let copy_vec_elem_bb = L.append_block context "copy_vec_elem" the_fun
ction in
              let copy_vec_elem_builder = L.builder_at_end context copy_vec_elem_bb in
              let iter_val = L.build_load iter "iter_val" copy_vec_elem_builder in
              let vec_ptr = L.build_load out_ptr "vec_ptr" copy_vec_elem_builder in
              let dest_ptr = L.build_gep vec_ptr [|iter_val|] "dest_ptr" copy_vec_elem_bu
ilder in
              let origin_ptr = L.build_gep ptr [|iter_val|] "origin_ptr" copy_vec_elem_bu
ilder in
              let (newvec, newbuilder) = vector_copy origin_ptr copy_vec_elem_builder typ
 in
                    ignore (L.build_store newvec dest_ptr newbuilder) ;
                    let new_iter_val = L.build_add iter_val (L.const_int i32_t 1) "new_it
er_val" newbuilder in
                    ignore (L.build_store new_iter_val iter newbuilder);
                    let merge_bb = L.append_block context "merge" the_function in
                      let merge_builder = L.builder_at_end context merge_bb in
                      ignore (L.build_cond_br zerosize zerosize_bb nonzerosize_bb builder
) ;
                      ignore (L.build_br pred_bb zerosize_builder);
                      ignore (L.build_br pred_bb nonzerosize_builder);
                      ignore (L.build_cond_br keeplooping copy_vec_elem_bb merge_bb pred_
builder);
                      ignore (L.build_br pred_bb newbuilder);
                      (L.build_load out_vec "out_vec_val" merge_builder, merge_builder)
      | A.Struct(typ) -> let ptr_p = L.build_gep vec [|zeroth; zeroth|] "ptr_p" builder i
n
            let size_p = L.build_gep vec [|zeroth; first|] "size_p" builder in
            let len_p = L.build_gep vec [|zeroth; second|] "len_p" builder in
            let ptr = L.build_load ptr_p "ptr" builder in
            let size = L.build_load size_p "size" builder in
            let len = L.build_load len_p "len" builder in
            let iter = L.build_alloca i32_t "iter" builder in
            ignore (L.build_store (L.const_int i32_t 0) iter builder) ;
            let out_vec = L.build_alloca vec_t "out_vec" builder in
            let out_ptr = L.build_gep out_vec [|zeroth; zeroth|] "out_ptr" builder in
            let out_size = L.build_gep out_vec [|zeroth; first|] "out_size" builder in
            let out_len = L.build_gep out_vec [|zeroth; second|] "out_len" builder in
            ignore (L.build_store size out_size builder) ;
            ignore (L.build_store len out_len builder) ;
            let zerosize = L.build_icmp L.Icmp.Slt size (L.const_int i32_t 1) "iszerosize
" builder in
            let zerosize_bb = L.append_block context "zerosize" the_function in
              let zerosize_builder = L.builder_at_end context zerosize_bb in
              ignore (L.build_store (L.const_pointer_null (L.pointer_type vec_t)) out_ptr
 zerosize_builder) ;
            let nonzerosize_bb = L.append_block context "nonzerosize" the_function in
              let nonzerosize_builder = L.builder_at_end context nonzerosize_bb in


              let newvec = L.build_array_malloc (ltype_of_typ (A.Struct(typ))) len "newve
c" nonzerosize_builder in
              (*vec_t **)
              let newvec_cast = L.build_bitcast newvec (L.pointer_type vec_t) "newvec_cas
t" nonzerosize_builder in
              (*output_ptr -> vect_t** *)
              ignore (L.build_store newvec_cast out_ptr nonzerosize_builder) ;
            let pred_bb = L.append_block context "check_size" the_function in
            let pred_builder = L.builder_at_end context pred_bb in
              let iter_val = L.build_load iter "iter_val" pred_builder in
              let keeplooping = L.build_icmp L.Icmp.Slt iter_val size "keeplooping" pred_
builder in
```

```
               let copy_vec_elem_bb = L.append_block context "copy_vec_elem" the_function
 in
              let copy_vec_elem_builder = L.builder_at_end context copy_vec_elem_bb in
              let iter_val = L.build_load iter "iter_val" copy_vec_elem_builder in

              (* vec_t* *)
              let vec_ptr = L.build_load out_ptr "vec_ptr" copy_vec_elem_builder in
              let struct_vec_ptr = L.build_bitcast vec_ptr (L.pointer_type (ltype_of_typ
(A.Struct(typ)))) "struct_vec_ptr" copy_vec_elem_builder in
              let struct_ptr = L.build_bitcast ptr (L.pointer_type (ltype_of_typ (A.Struc
t(typ)))) "struct_vec_ptr" copy_vec_elem_builder in

              (* struct xyz* *)
              let dest_ptr = L.build_gep struct_vec_ptr [|iter_val|] "dest_ptr" copy_vec_
elem_builder in
              let origin_ptr = L.build_gep struct_ptr [|iter_val|] "origin_ptr" copy_vec_
elem_builder in
        let struct_def_map = StringMap.find typ global_map in
        let newbuilder = struct_copy dest_ptr origin_ptr struct_def_map copy_vec_elem_bui
lder in
        let new_iter_val = L.build_add iter_val (L.const_int i32_t 1) "new_iter_val" newb
uilder in
                    ignore (L.build_store new_iter_val iter newbuilder);
                    let merge_bb = L.append_block context "merge" the_function in
                      let merge_builder = L.builder_at_end context merge_bb in
                      ignore (L.build_cond_br zerosize zerosize_bb nonzerosize_bb builder
) ;
                      ignore (L.build_br pred_bb zerosize_builder);
                      ignore (L.build_br pred_bb nonzerosize_builder);
                      ignore (L.build_cond_br keeplooping copy_vec_elem_bb merge_bb pred_
builder);
                      ignore (L.build_br pred_bb newbuilder);
                      (L.build_load out_vec "out_vec_val" merge_builder, merge_builder)



      | typ -> let ptr_p = L.build_gep vec [|zeroth; zeroth|] "ptr_p" builder in
            let size_p = L.build_gep vec [|zeroth; first|] "size_p" builder in
            let len_p = L.build_gep vec [|zeroth; second|] "len_p" builder in
            let ptr = L.build_load ptr_p "ptr" builder in
            let size = L.build_load size_p "size" builder in
            let len = L.build_load len_p "len" builder in
            let out_vec = L.build_alloca vec_t "out_vec" builder in
            let out_ptr = L.build_gep out_vec [|zeroth; zeroth|] "out_ptr" builder in
            let out_size = L.build_gep out_vec [|zeroth; first|] "out_size" builder in
            let out_len = L.build_gep out_vec [|zeroth; second|] "out_len" builder in
            ignore (L.build_store size out_size builder) ;
            ignore (L.build_store len out_len builder) ;
            let iszerosize = L.build_icmp L.Icmp.Slt size (L.const_int i32_t 1) "iszerosi
ze" builder in
            let zerosize_bb = L.append_block context "zerosize" the_function in
              let zerosize_builder = L.builder_at_end context zerosize_bb in
              ignore (L.build_store (L.const_pointer_null (L.pointer_type vec_t)) out_ptr
 zerosize_builder) ;
            let nonzerosize_bb = L.append_block context "nonzerosize" the_function in
              let nonzerosize_builder = L.builder_at_end context nonzerosize_bb in
              let newvec = L.build_array_malloc (ltype_of_typ typ) len "newvec" nonzerosi
ze_builder in
              let newvec_cast = L.build_bitcast newvec (L.pointer_type vec_t) "newvec_cas
t" nonzerosize_builder in
              ignore (L.build_store newvec_cast out_ptr nonzerosize_builder) ;
              let src_cast = L.build_bitcast ptr (L.pointer_type i8_t) "src_cast" nonzero
size_builder in
              let dst_cast = L.build_bitcast newvec (L.pointer_type i8_t) "dst_cast" nonz
erosize_builder in
              let sizeof = L.build_trunc (L.size_of (ltype_of_typ typ)) i32_t "sizeof" no
nzerosize_builder in
              let sz_to_copy = L.build_mul size sizeof "sz_to_copy" nonzerosize_builder i
n
              ignore (L.build_call memcpy_func [|dst_cast; src_cast; sz_to_copy; L.const_
```

```
int i32_t 1; L.const_int i1_t 1|] "" nonzerosize_builder) ;
            let merge_bb = L.append_block context "merge" the_function in
              let merge_builder = L.builder_at_end context merge_bb in
            ignore(L.build_br merge_bb zerosize_builder);
            ignore(L.build_br merge_bb nonzerosize_builder);
            ignore(L.build_cond_br iszerosize zerosize_bb nonzerosize_bb builder);
            (L.build_load out_vec "out_vec_val" merge_builder, merge_builder) in

        (* takes a builder and a function:
        if current block is terminator, do not go to merge block
        else go to merge block
        if we have a "return" in a if block, we have to know if we want to
         add a terminator block.
       *)
   let add_terminal builder f =
      match L.block_terminator (L.insertion_block builder) with
        Some _ -> ()
      | None -> ignore (f builder) in
    let vector_access need_to_load ptr typ indices builder =
      let rec vector_access_helper ptr typ indices = (match (typ, indices) with
            (_, []) -> ptr
          | (A.Vector(in_typ), i::tl) -> let new_vec_p = L.build_gep ptr [|zeroth;zeroth|
] "vect_gep" builder in
                                         let new_vec = L.build_load new_vec_p "vec_load"
builder in
                                         let new_vec_elem = L.build_gep new_vec [|i|] "el
em" builder in
                                         vector_access_helper new_vec_elem in_typ tl
          | (final_type, i::_) -> let elem_arr_p = L.build_gep ptr [|zeroth;zeroth|] "ele
m_arr_p" builder in
                                  let elem_arr = L.build_load elem_arr_p "elem_arr_load"
 builder in
                                  let elem_arr_cast = L.build_bitcast elem_arr (L.pointe
r_type (ltype_of_typ final_type)) "elem_arr_cast" builder in
                                  L.build_gep elem_arr_cast [|i|] "elem" builder) in
        let target_ptr = vector_access_helper ptr typ indices in
      if need_to_load then L.build_load target_ptr "element" builder
      else target_ptr in

     (* Construct code for an expression; return its value
      Always attach the expression to the latest builder.
      Do not create a new block. Block creation happens at
      statement gen for Block type.
   *)
    let rec get_struct_access_field_info = function
        A.Id s ->  (A.Struct(get_struct_type s), -1)
      | A.StructFieldAccess(e, field_name) ->
        let struct_type_name = (match (get_struct_access_field_info e) with
                | A.Struct(name), _ -> name
                | _ -> raise(Failure("accessing field of something not a struct"))) in
        let struct_def_map = StringMap.find struct_type_name global_map  in
        StringMap.find field_name struct_def_map (* returns (typ, idx) of field_name*)
      | A.VectorAccess(id, _) -> let rec walk_back_type = function
                A.Vector(typ) -> walk_back_type typ
              | typ -> typ in
            (walk_back_type (get_vector_type id), -1)
      | _ -> raise(Failure("accessing field of something not a struct")) in


      let rec struct_copy_assign lptr rptr struct_def_map builder delete_vec =
        let bindings = StringMap.bindings struct_def_map in
        let struct_field_assign_helper builder binding  =
          let (_, (fieldtyp, idx)) = binding in
            (match fieldtyp with
              A.Vector(in_typ) ->
                  let lvec_ptr = L.build_struct_gep lptr idx "l_field_vec_cp_assign" buil
der in
                  let rvec_ptr = L.build_struct_gep rptr idx "r_field_vec_cp_assign" buil
der in
                  let load_rptr = L.build_load rvec_ptr "loaded_rptr" builder in
```

```
                  let builder3 = vector_delete lvec_ptr builder the_function in_typ in
                  let temp = L.build_alloca vec_t "temp" builder3 in
                  let _ = L.build_store load_rptr temp builder3 in
                  if (delete_vec)
                  then

                    let (vec_cpy, builder4) = vector_copy temp builder3 in_typ in
                    let builder5 = vector_delete rvec_ptr builder4 the_function in_typ in
                      ignore(L.build_store vec_cpy lvec_ptr builder5); builder5
                  else
                  (*
                    raise(Failure (L.string_of_llvalue((lvec_ptr)) ^ "; " ^ L.string_of_l
lvalue((rvec_ptr))))
                      *)
                    let (vec_cpy, builder4) = vector_copy temp builder3 in_typ in
                      ignore(L.build_store vec_cpy lvec_ptr builder4); builder4

                | A.Struct(in_typ) ->
                  (* %struct_field = getelementptr inbounds %example* %b, i32 0, i32 2;
                     %struct_field1 = getelementptr inbounds %example* %a, i32 0, i32 2
                     raise(Failure (L.string_of_lltype((ltype_of_typ fieldtyp)) ^ " " ^ f
ield_name))
                   *)


                    let l_field_val = L.build_struct_gep lptr idx "l_inner_struct_field"
builder in

                    let r_field_val = L.build_struct_gep rptr idx "r_inner_struct_field"
builder in

                    let field_struct_def_map = StringMap.find in_typ global_map  in
                    let (_, builder) = struct_copy_assign l_field_val r_field_val field_s
truct_def_map builder delete_vec in builder;

                | _ ->
                    let l_field_ptr = L.build_struct_gep lptr idx "l_field_cp_assign" bui
lder in
                    let r_field_ptr = L.build_struct_gep rptr idx "r_field_cp_assign" bui
lder in
                    let load_rptr = L.build_load r_field_ptr "loaded_rptr" builder in
                    ignore(L.build_store load_rptr l_field_ptr builder); builder) in
        let builder = List.fold_left struct_field_assign_helper builder bindings in (lptr
, builder);
    in

                let rec expr builder = function
                        A.Literal i -> (L.const_int i32_t i, builder)
                      | A.DoubleLit f -> (L.const_float double_type f, builder)
      | A.StringLit s -> let result = string_decl s builder in
                let vect1 = L.build_alloca vec_t "gc_tmp_vect1" builder in
                let _ = L.build_store result vect1 builder in
                let _ = add_to_string_lit_cleanup vect1 in (result, builder)
                      | A.BoolLit b -> (L.const_int i1_t (if b then 1 else 0), builder)
                      | A.Noexpr -> (L.const_int i32_t 0, builder)
            | A.Id s -> ((if (Hashtbl.mem struct_type_map s) then lookup s else L.build_l
oad (lookup s) s builder), builder)
                      | A.Binop (e1, op, e2) ->
                             let (e1', builder') = expr builder e1 in
                             let (e2', builder'') = expr builder' e2 in
                             let exp_type1 = L.classify_type(L.type_of e1') in
                             (*let exp_type2 =*) ignore (L.classify_type(L.type_of e2'
));
                             ((match exp_type1 with
                                     | L.TypeKind.Double ->
                            (match op with
                              A.Add     -> L.build_fadd
                            | A.Sub     -> L.build_fsub
                            | A.Mult    -> L.build_fmul
                            | A.Div     -> L.build_fdiv
```

```
                                | A.And     -> L.build_and
                                | A.Or      -> L.build_or
                                | A.Equal   -> L.build_fcmp L.Fcmp.Oeq
                                | A.Neq     -> L.build_fcmp L.Fcmp.One
                                | A.Less    -> L.build_fcmp L.Fcmp.Ult
                                | A.Leq     -> L.build_fcmp L.Fcmp.Ole
                                | A.Greater -> L.build_fcmp L.Fcmp.Ogt
                                | A.Geq     -> L.build_fcmp L.Fcmp.Oge
                               ) e1' e2' "tmp" builder''
                                        | _ ->
                                             (match op with
                                               A.Add                -> L.build_add
                                             | A.Sub                -> L.build_sub
                                             | A.Mult               -> L.build_mul
                                             | A.Div                -> L.build_sdiv
                                             | A.And                -> L.build_and
                                             | A.Or                 -> L.build_or
                                             | A.Equal              -> L.build_icmp L
.Icmp.Eq
                                             | A.Neq                -> L.build_icmp L
.Icmp.Ne
                                             | A.Less               -> L.build_icmp L
.Icmp.Slt
                                             | A.Leq                -> L.build_icmp L
.Icmp.Sle
                                             | A.Greater     -> L.build_icmp L.Icmp.Sg
t
                                             | A.Geq                -> L.build_icmp L
.Icmp.Sge
                                             ) e1' e2' "tmp" builder''
                                  ), builder'')

                          | A.Unop(op, e) ->
                                 let (e', builder') = expr builder e in
                                 ((match op with
                                        A.Neg                   -> L.build_neg
                                   | A.Not                      -> L.build_not) e' "tmp"
builder', builder')
                          | A.Assign (s, e) -> if ((is_vector s) && (match e with A.Call(_,
_) -> false | A.Get(_) -> false | _ -> true))
                                  then (let (vec_ptr, vec_typ, builder'') = (match e with
                                          A.Id(id) -> (lookup id, get_vector_type id, buil
der)
                                        | e -> let temp = L.build_alloca vec_t "temp" buil
der in
                                               let (e', b') = expr builder e in
                                               ignore (L.build_store e' temp b');
                                               (temp, get_vector_type s, b')) in
                                  let builder3 = vector_delete (lookup s) builder'' the_fu
nction (get_vector_type s) in
                                  let (vec_cpy, builder4) = vector_copy vec_ptr builder3 v
ec_typ in
                                  ignore (L.build_store vec_cpy (lookup s) builder4);
                                  expr builder4 e)
                             else (if (is_vector s)
                                 then (let (e', builder') = expr builder e in
                                 let builder'' = vector_delete (lookup s) builder' the_funct
ion (get_vector_type s) in
                                                                                 ignor
e (L.build_store e' (lookup s) builder''); (e', builder''))
                               else (if ((is_struct s)  && (match e with A.StringLit(_) -> fal
se | A.Call(_,_) -> false | A.Get(_) -> false | _ -> true))
                                 then let (e', builder') = expr builder e in
                                 let struct_type = get_struct_type s in
                                 let struct_def_map = StringMap.find struct_type global_map
 in
                                 let lptr = lookup s in
                                 (*raise(Failure (L.string_of_llvalue(lptr) ^ ";" ^ L.string
_of_llvalue(e')))*)
                                 struct_copy_assign lptr e' struct_def_map builder' false
```

```
                          else (if (is_struct s) then (let (e', builder') = expr builder
e in
                                  let struct_type = get_struct_type s in
                                  let field_struct_def_map = StringMap.find struct_type g
lobal_map  in
                                  let builder'' = struct_delete (lookup s) field_struct_d
ef_map builder' the_function in
                                  ignore (L.build_store e' (lookup s) builder''); (e', bu
ilder''))

                            else (let (e', builder') = expr builder e in
                            ignore (L.build_store e' (lookup s) builder'); (e', builder')
))))
      | A.VectorAccess(id, list_e) -> let (idx_list, builder') =
          List.fold_left (fun (accumulate, b) arg -> let (e', b') = (expr b arg) in (e'::
accumulate, b')) ([], builder) list_e in
          let ptr = lookup id in
          let typ = get_vector_type id in
          (match typ with
            A.Struct(_) -> (vector_access false ptr typ idx_list builder', builder')
          | _ -> (vector_access true ptr typ idx_list builder', builder'))


      | A.VectorAssign(id, e1_list, e2) -> let (idx_list_rev, builder') =
          (List.fold_left (fun (accumulate, b) e -> let (e', b') = (expr b e) in (e'::acc
umulate, b')) ([], builder) e1_list) in
          let idx_list = List.rev idx_list_rev in
          let typ = get_vector_type id in
                let rec walk_back_type = function
              (0, typ) -> typ
            | (n, A.Vector(typ)) -> walk_back_type (n - 1, typ)
            | _ -> raise(Failure("Walking back type of something not a vector")) in
          let innertyp = walk_back_type ((List.length idx_list) - 1, typ) in
          (match innertyp with
              A.Vector(_) when (match e2 with A.Call(_,_) -> false | A.Get(_) -> false |
_ -> true) ->
                    let (vec_ptr, vec_typ, builder'') = (match e2 with
                         A.Id(id2) -> (lookup id2, get_vector_type id2, builder)
                       | e -> let temp = L.build_alloca vec_t "temp" builder in
                              let (e', b') = expr builder e in
                              ignore (L.build_store e' temp b');
                              (temp, get_vector_type id, b')) in
                    let element = vector_access false (lookup id) typ idx_list builder''
in
                    let builder3 = vector_delete element builder'' the_function vec_typ i
n
                    let (vec_cpy, builder4) = vector_copy vec_ptr builder3 vec_typ in
                    ignore (L.build_store vec_cpy element builder4);
                    expr builder4 e2
            | A.Vector(_)  -> let element = vector_access false (lookup id) typ idx_list
builder' in
                  let builder'' = vector_delete element builder' the_function innertyp i
n
                  let (right_hand_val, builder3) = expr builder'' e2 in
                  ignore (L.build_store right_hand_val element builder3);
                  (right_hand_val, builder3)
            | A.Struct(in_typ) when (match e2 with A.Call(_,_) -> false | A.Get(_) -> fal
se | _ -> true) ->
                  let element = vector_access false (lookup id) typ idx_list builder' in
                  let (right_hand_val, builder'') = expr builder' e2 in
                  let field_struct_def_map = StringMap.find in_typ global_map  in
                  let builder3 = struct_delete element field_struct_def_map builder'' th
e_function in
                  let builder4 = struct_copy element right_hand_val field_struct_def_map
 builder3 in
                  (right_hand_val, builder4)
            | A.Struct(in_typ) ->
                  let element = vector_access false (lookup id) typ idx_list builder' in
                  let (right_hand_val, builder'') = expr builder' e2 in
                  let field_struct_def_map = StringMap.find in_typ global_map  in
```

```
                    let builder3 = struct_delete element field_struct_def_map builder'' th
e_function in
                    ignore (L.build_store right_hand_val element builder3);
                    (right_hand_val, builder3)
            | _ -> let element = vector_access false (lookup id) typ idx_list builder' in
                    let (right_hand_val, builder'') = expr builder' e2 in
                    ignore (L.build_store right_hand_val element builder'');
                    (right_hand_val, builder''))
      | A.VectorSize(e) ->
          let (vector_val, builder'') = (match e with
                  A.Id(id) -> (lookup id, builder)
                | A.VectorAccess(id, list_e) -> let (idx_list, builder') =
                        List.fold_left (fun (accumulate, b) arg -> let (e', b') = (expr b
 arg) in (e'::accumulate, b')) ([], builder) list_e in
                    let ptr = lookup id in
                    let typ = get_vector_type id in
                    (vector_access false ptr typ idx_list builder', builder')
                | _ -> raise(Failure("trying to take the size of something not a vector")
)) in
          let size_p = L.build_gep vector_val [|zeroth;first|] "size_p" builder'' in
          (L.build_load size_p "size" builder'', builder'')
      | A.Concat(e1, e2) -> let (result2, builder) = expr builder e2 in let (result1, bui
lder) = expr builder e1 in

            let result = string_concat result1 result2 builder in
            let vect1 = L.build_alloca vec_t "gc_tmp_vect1" builder in
            let _ = L.build_store result vect1 builder in
            let _ = add_to_string_lit_cleanup vect1 in (result, builder)
      | A.Get(job) -> let typ = Hashtbl.find job_type_map job in
        let jid        = L.build_load (lookup job) job builder in
        let out_data   = L.build_alloca (L.pointer_type i8_t) "out_data" builder in
        let out_len    = L.build_alloca (i32_t) "out_len" builder in
        ignore(L.build_call reap_job_func [| jid; out_data; out_len |] "" builder) ;
        let out_data_val = L.build_load out_data "out_data_val" builder in
        let (lst, _, _, newbuilder) = deserialize ([], out_data, the_function, builder) t
yp in
        let retval = (match lst with [r] -> r | _ -> raise(Failure("multiple returns from
 job")) ) in
        ignore(L.build_free out_data_val  newbuilder) ;
        (retval, newbuilder)

      | A.Cancel(job) -> let jid = L.build_load (lookup job) job builder in
        (L.build_call cancel_job_func [| jid |] "cancel_job" builder, builder)

      | A.Running(job) -> let jid = L.build_load (lookup job) job builder in
        let status = L.build_call get_job_status_func [| jid |] "get_job_status" builder
in
        (L.build_icmp L.Icmp.Sle (L.const_int i32_t 0) status "running" builder, builder)
      | A.Finished(job) -> let jid = L.build_load (lookup job) job builder in
        let status = L.build_call get_job_status_func [| jid |] "get_job_status" builder
in
        (L.build_icmp L.Icmp.Eq (L.const_int i32_t (-1)) status "finished" builder, build
er)
      | A.Failed(job) -> let jid = L.build_load (lookup job) job builder in
        let status = L.build_call get_job_status_func [| jid |] "get_job_status" builder
in
        (L.build_icmp L.Icmp.Eq (L.const_int i32_t (-2)) status "failed" builder, builder
)

      | A.RemoteCall (f, e) -> let (_, fdecl) = StringMap.find f function_decls in
        let (evals, nbuilder) = List.fold_left (fun (acc, b) e ->
                (match e with
                 A.Id(s) when (is_struct(s)) -> ((L.build_load (lookup s) "struct" b)::ac
c, b)
                | _ -> let (e', b') = expr b e in (e'::acc, b')))
             ([], builder) (List.rev e) in
        let typs = List.fold_left (fun acc formal -> let (typ, _) = formal in typ::acc) [
] (List.rev fdecl.A.formals) in
        let (arg, len, builder') = serialize nbuilder evals the_function typs in
        let ord = get_ordinal f in
```

```
        (L.build_call start_job_func [| ord ; arg ; len |] "start_job" builder', builder'
)

      | A.StructFieldAccess(e, field_name) ->
        let struct_type_name = (match (get_struct_access_field_info e) with
                | A.Struct(name), _ -> name
                | _ -> raise(Failure("assigning to field of something not a struct"))) in

        let (struct_val, builder'') = (match e with
                    A.VectorAccess(id, list_e) -> let (idx_list, builder') =
                        List.fold_left (fun (accumulate, b) arg -> let (e', b') = (expr b
 arg) in (e'::accumulate, b')) ([], builder) list_e in
                        let ptr = lookup id in
                        let typ = get_vector_type id in
                        (vector_access false ptr typ idx_list builder', builder')
                  | e -> expr builder e) in

        let struct_def_map = StringMap.find struct_type_name global_map  in
        let (typ, field_idx) = StringMap.find field_name struct_def_map  in
        (*raise(Failure (L.string_of_llvalue (struct_val)))*)

          let ptr =  L.build_struct_gep struct_val field_idx "struct_field" builder'' in
          (( match typ with
              A.Struct(_) -> ptr
            | _ -> L.build_load ptr "struct_field_load" builder''
          ), builder'')
      | A.StructFieldAssign(e1, field_name, e) ->
        let (result, builder') = expr builder e  in
        let (struct_val, builder'') = expr builder' e1 in
        let struct_type_name = (match (get_struct_access_field_info e1) with
                | A.Struct(name), _ -> name
                | _ -> raise(Failure("assigning to field of something not a struct"))) in
        let struct_def_map = StringMap.find struct_type_name global_map  in
        let (typ, field_idx) = StringMap.find field_name struct_def_map  in
         let ptr = L.build_struct_gep struct_val field_idx "struct_field" builder'' in (*
          raise(Failure (L.string_of_llvalue (struct_val) ^ L.string_of_llvalue ptr))) *)
          (match typ with
          A.Struct(inner_type) ->

            (*
            let struct_val_cast = L.build_bitcast result (L.pointer_type i8_t) "struct_sr
c_cast" builder'' in
            let ptr_cast = L.build_bitcast ptr (L.pointer_type i8_t) "struct_dest_cast" b
uilder'' in
            let struct_size = L.size_of (ltype_of_typ typ) in
            (L.build_call memcpy64_func [|ptr_cast; struct_val_cast; struct_size; L.const
_int i32_t 1; L.const_int i1_t 1|] "" builder'', builder'')
          *)
          let struct_def_map = StringMap.find inner_type global_map in
            struct_copy_assign ptr result struct_def_map builder'' false
          | A.Vector(inner_t) -> let builder3 = vector_delete ptr builder'' the_function
inner_t in
              let temp = L.build_alloca vec_t "temp" builder3 in
              ignore (L.build_store result temp builder3) ;
              let (vec_cpy, builder4) = vector_copy temp builder3 inner_t in
              (L.build_store vec_cpy ptr builder4, builder4)

          | _ -> (L.build_store result ptr builder'', builder''))


      | A.Call ("print", [e])  -> let (e', b') = expr builder e in
                                  (L.build_call printf_func [| int_format_str ; e' |] "prin
tf" b', b')

                          | A.Call ("printb", [e]) -> let (e', b') = expr builder e in
                                 let bstring = L.build_select e' true_str false_str "selec
t" b' in
                                  (L.build_call printf_func [| bstring |] "printf" b', b')

                          | A.Call ("printd", [e]) -> let (e', b') = expr builder e in
```

```
                                (L.build_call printf_func [| double_format_str ; e' |] "p
rintf" b', b')
      | A.Call ("prints", [e]) -> let (e', b') = expr builder e in
        let vect1 = L.build_alloca vec_t "tmp_vect1" builder in
        let _ = L.build_store e' vect1 builder in
        let ptr1 = L.build_gep vect1 [|zeroth; zeroth|] "ptr_1" builder in
        let cast_vect1 = L.build_bitcast ptr1 (L.pointer_type (L.pointer_type i8_t)) "cas
t_vect1" builder in
        let load_cast_vect1 = L.build_load cast_vect1 "load_cast_vect1" builder in
        (L.build_call printf_func [| string_str ; load_cast_vect1 |] "printf" b', b')

                        | A.Call (f, act) ->
                                let (fdef, fdecl) = StringMap.find f function_decls in
                                        (* needs to evaluate expr from right to left. int
 a = b = (a+1) *)

                                        let (actuals, _, builder') = List.fold_left (fun
(acc, fmls, b) e -> let (e', b') = expr b e in
                                       (match fmls with
                                        (A.Vector(typ), _)::tl -> let tmp = L.build_alloc
a vec_t "tmp" b' in
                                                                  ignore (L.build_store e
' tmp b');
                                                                  let (v_cpy, b'') = (mat
ch e with
                                                                      A.Call(_,_) | A.Rem
oteCall(_,_) -> (L.build_load tmp "vec" b', b')
                                                                    | _ -> vector_copy tm
p b' typ) in
                                                                  (v_cpy::acc, tl, b'')
                                        | (A.Struct(name), _)::tl -> let dst = L.build_al
loca (ltype_of_typ (A.Struct(name))) "dst" b' in
                                                                     let field_struct_def
_map = StringMap.find name global_map  in
                                                                     let nbuilder = (matc
h e with
                                                                           A.Call(_,_) |
A.RemoteCall(_,_) ->
                                                                                ignore (L
.build_store e' dst b'); b'
                                                                         | _ -> struct_co
py dst e' field_struct_def_map b') in
                                                                     ((L.build_load dst "
struct_copy" nbuilder)::acc, tl, nbuilder)
                                        | _::tl -> (e'::acc, tl, b')
                                        | [] -> raise(Failure("too many arguments!"))))
                                ([], (List.rev fdecl.A.formals), builder) (List.rev act)  in
                                        let result = (match fdecl.A.typ with A.Void -> ""
 | _ -> f ^ "_result") in
                                (L.build_call fdef (Array.of_list actuals) result builde
r', builder')
            | _ -> raise(Failure("unrecognized expr")) in

     (* make space for the return value *)
     let retval = (match fdecl.A.typ with
             A.Void -> L.const_null i32_t
           | typ -> let retval = L.build_alloca (ltype_of_typ typ) "retval" builder in
                    ignore (L.build_store (L.const_null (ltype_of_typ fdecl.A.typ)) retva
l builder) ;
                    retval ) in

     (* where to go right before returning *)
     let return_bb = L.append_block context "return_loc" the_function in
       let return_builder = L.builder_at_end context return_bb in

        (* Build the code for the given statement; return the builder for
                        the statement's successor *)
              let rec stmt builder = function
             A.Block sl -> List.fold_left stmt builder sl
                        | A.Expr e -> let (_, b') = (expr builder e) in b'
```

```
      | A.VarDecl(typ,id) ->
          (match typ with
            A.Vector(inner_t) -> ignore(let vector_result = vector_decl(id, inner_t, buil
der) in
                                        (add_var_init id vector_result)); builder
          | A.Job(innertyp) -> ignore(add_var_init id (L.build_alloca (ltype_of_typ typ
) id builder));
                                        ignore(add_job_type id innertyp); builder
          | A.Struct(struct_name) -> let _ = add_struct_type id struct_name in
               let buf = L.build_alloca (ltype_of_typ typ) id builder in
               ignore(L.build_store (L.const_null (ltype_of_typ typ)) buf builder);
               ignore(add_var_init id buf); builder
          | _ -> ignore(add_var_init id (L.build_alloca (ltype_of_typ typ) id builder))
; builder)
                      | A.Return e -> let rec walk_back_type = function
                              (0, typ) -> typ
                            | (n, A.Vector(typ)) -> walk_back_type (n - 1, typ)
                            | _ -> raise(Failure("Walking back type of something not a
 vector")) in
                (match fdecl.A.typ with
                                                    A.Void -> ignore(L.build_br retur
n_bb builder); builder
                          | A.Struct(in_typ) ->
                                 let (r_field_val, nbuilder) = expr builder e in
                                 let field_struct_def_map = StringMap.find in_typ global
_map  in
                                 struct_copy retval r_field_val field_struct_def_map nbu
ilder
                          | A.Vector(typ) -> ( match e with
                               A.VectorAccess(id, lst) -> let (idx_list, builder') =
                                  List.fold_left (fun (accumulate, b) arg -> let (e', b
') = (expr b arg) in (e'::accumulate, b')) ([], builder) lst in
                                 let ptr = lookup id in
                                 let innertyp = walk_back_type ((List.length lst), (ge
t_vector_type id)) in
                                 let vec = vector_access false ptr (get_vector_type id
) idx_list builder' in
                                 let (cpy, newbuilder) = vector_copy vec builder' inne
rtyp in
                                 ignore (L.build_store cpy retval newbuilder) ; newbui
lder
                             | A.Id(id) -> let (cpy, newbuilder) = vector_copy (lookup i
d) builder typ in
                                 ignore (L.build_store cpy retval newbuilder); newbuil
der
                             | _ -> raise(Failure("trying to return a vector but expr do
esn't seem to be a vector")))
                                                | _ -> let (e', b') = expr builder e in
                                ignore (L.build_store e' retval b');
                                ignore (L.build_br return_bb b'); b');

      | A.VectorPushBack(e1, e) ->
          let (right_hand_val, builder'') = (match e with
              A.VectorAccess(id, lst) -> let (idx_list, builder') =
                  List.fold_left (fun (accumulate, b) arg -> let (e', b') = (expr b arg)
in (e'::accumulate, b')) ([], builder) lst in
                let ptr = lookup id in
                let typ = get_vector_type id in
                (vector_access true ptr typ idx_list builder', builder')
            | A.Id(id) when (is_vector id) -> (lookup id, builder)
            | A.Id(id) when (Hashtbl.mem struct_type_map id) -> let in_typ = get_struct_t
ype id in
                 let l_field_val = L.build_alloca (ltype_of_typ (A.Struct(in_typ))) "n
ew_struct" builder in
                 let r_field_val = lookup id in
                 let field_struct_def_map = StringMap.find in_typ global_map  in
                 let builder' = struct_copy l_field_val r_field_val field_struct_def_m
ap builder in
                 (L.build_load l_field_val "new_struct_val" builder', builder')
            | _ -> expr builder e ) in
```

```
          (* we need to get typ of vector somehow -- how to do this with expr? *)
          let rec walk_back_type = function
              (0, typ) -> typ
            | (n, A.Vector(typ)) -> walk_back_type (n - 1, typ)
            | _ -> raise(Failure("Walking back type of something not a vector")) in
          let (vector_val, vector_typ, builder4) = (match e1 with
              A.VectorAccess(id, lst) -> let (idx_list, builder3) =
                  List.fold_left (fun (accumulate, b) arg -> let (e', b') = (expr b arg)
in (e'::accumulate, b')) ([], builder'') lst in
                  let ptr = lookup id in
                  let typ = get_vector_type id in
                  ((vector_access false ptr typ idx_list builder3), (walk_back_type ((Lis
t.length lst), (get_vector_type id))), builder3)
            | A.Id(id) -> (lookup id, get_vector_type id, builder'')
            | _ -> raise(Failure("VectorPushBack on something not a vector")) ) in
          let size_p = L.build_gep vector_val [|zeroth; first|] "size_p" builder4 in
          let size_var = L.build_load size_p "size_var" builder4 in
          let len_p = L.build_gep vector_val [|zeroth; second|] "len_p" builder4 in
          let len_var = L.build_load len_p "leng_var" builder4 in
          let should_resize = L.build_icmp L.Icmp.Slt size_var len_var "size_var_cmp_leng
" builder4 in

          let resize_bb = L.append_block context "resize" the_function in
            let resize_builder = L.builder_at_end context resize_bb in
            let resize_zero = L.build_icmp L.Icmp.Slt len_var (L.const_int i32_t 1) "size
_var_nonzero" resize_builder in
          let resize_zero_bb = L.append_block context "resize_zero" the_function in
            let resize_zero_builder = L.builder_at_end context resize_zero_bb in
            ignore(L.build_store (L.const_int i32_t 1) len_p resize_zero_builder) ;
          let resize_nonzero_bb = L.append_block context "resize_nonzero" the_function in
            let resize_nonzero_builder = L.builder_at_end context resize_nonzero_bb in
            let newlen = L.build_shl len_var (L.const_int i32_t 1) "shl" resize_nonzero_b
uilder in
            ignore(L.build_store newlen len_p resize_nonzero_builder) ;
          let resize_merge_bb = L.append_block context "resize_merge" the_function in
            let resize_merge_builder = L.builder_at_end context resize_merge_bb in
            let len_val = L.build_load len_p "new_len_val" resize_merge_builder in
            let ptr_p = L.build_gep vector_val [|zeroth;zeroth|] "ptr_p" resize_merge_bui
lder in
            let ptr_val = L.build_load ptr_p "ptr_val" resize_merge_builder in
            let vec_mem = L.build_array_malloc (ltype_of_typ vector_typ) len_val "vec_mem
" resize_merge_builder in
            let vec_mem_byte_cast = L.build_bitcast vec_mem (L.pointer_type i8_t) "vec_me
m_byte_cast" resize_merge_builder in
            let ptr_val_byte_cast = L.build_bitcast ptr_val (L.pointer_type i8_t) "ptr_va
l_byte_cast" resize_merge_builder in
            let sizeof = L.build_trunc (L.size_of (ltype_of_typ vector_typ)) i32_t "sizeo
f" resize_merge_builder in
            let sz_to_copy = L.build_mul sizeof size_var "sz_to_copy" resize_merge_builde
r in
            ignore (L.build_call memcpy_func [|vec_mem_byte_cast; ptr_val_byte_cast; sz_t
o_copy; L.const_int i32_t 1; L.const_int i1_t 1|] "" resize_merge_builder );
            ignore (L.build_free ptr_val resize_merge_builder) ;
            let vec_mem_cast = L.build_bitcast vec_mem (L.pointer_type vec_t) "vec_mem_ca
st" resize_merge_builder in
            ignore (L.build_store vec_mem_cast ptr_p resize_merge_builder) ;
          let merge_bb = L.append_block context "pushback" the_function in
            let merge_builder = L.builder_at_end context merge_bb in
            let element = vector_access false vector_val vector_typ [size_var] merge_buil
der in
            let new_merge_builder = (match vector_typ with
                    A.Vector(intyp) -> let (cpy, new_merge_builder) = vector_copy right_h
and_val merge_builder intyp in
                                       ignore (L.build_store cpy element new_merge_builde
r) ;
                                       new_merge_builder
                  | _ -> ignore (L.build_store right_hand_val element merge_builder) ;
                           merge_builder ) in
          let new_size = L.build_add size_var (L.const_int i32_t 1) "new_size" new_merge_
builder in
```

```
          ignore (L.build_store new_size size_p new_merge_builder) ;
          ignore(L.build_cond_br should_resize merge_bb resize_bb builder4);
          ignore(L.build_cond_br resize_zero resize_zero_bb resize_nonzero_bb resize_buil
der);
          add_terminal resize_merge_builder (L.build_br merge_bb);
          add_terminal resize_zero_builder (L.build_br resize_merge_bb);
          add_terminal resize_nonzero_builder (L.build_br resize_merge_bb);
          new_merge_builder
      | A.If (predicate, then_stmt, else_stmt) ->
                                let (bool_val, builder') = expr builder predicate in
        let merge_bb = L.append_block context "merge" the_function in

        let then_bb = L.append_block context "then" the_function in
        add_terminal (stmt (L.builder_at_end context then_bb) then_stmt)
                (L.build_br merge_bb);

        let else_bb = L.append_block context "else" the_function in
        add_terminal (stmt (L.builder_at_end context else_bb) else_stmt)
                (L.build_br merge_bb);

        ignore (L.build_cond_br bool_val then_bb else_bb builder');
        L.builder_at_end context merge_bb

                        | A.While (predicate, body) ->
              let pred_bb = L.append_block context "while" the_function in
              ignore (L.build_br pred_bb builder);

              let body_bb = L.append_block context "while_body" the_function in
              add_terminal (stmt (L.builder_at_end context body_bb) body)
                      (L.build_br pred_bb);

              let pred_builder = L.builder_at_end context pred_bb in
              let (bool_val, pred_builder') = expr pred_builder predicate in

              let merge_bb = L.append_block context "merge" the_function in
              ignore (L.build_cond_br bool_val body_bb merge_bb pred_builder');
              L.builder_at_end context merge_bb

                        | A.For (e1, e2, e3, body) -> stmt builder
                        ( A.Block [A.Expr e1 ; A.While (e2, A.Block [body ; A.Expr e3]) ]
 )
          | A.VarDeclAssign(typ, id, e) -> let b' = stmt builder (A.VarDecl(typ, id)) i
n
              let (_, b'') = expr b' (A.Assign(id, e)) in b''
              in

              (* Build the code for each statement in the function *)
              let builder = stmt builder (A.Block fdecl.A.body) in

              (* Add a "return" (that is, branch) if the last block falls off the end *
)
              add_terminal builder (L.build_br return_bb) ;

       (* destroy all vectors *)
      let vects = Hashtbl.fold (fun s t l -> (s,t)::l) vector_type_map [] in
      let finalbuilder = List.fold_left (fun b (s,t) ->
       (*
       raise(Failure (L.string_of_llvalue((lookup s)) ^ "; " ^ L.string_of_lltype(ltype_
of_typ t) ^ "; "))
         *)
         vector_delete (lookup s) b the_function t

       ) return_builder vects in

      let literals = Hashtbl.fold (fun s _ l -> s::l) string_lit_map [] in
      let reallyfinalbuilder = List.fold_left (fun b s ->
         (*
         raise(Failure (L.string_of_llvalue(s) ^ "; "))
         *)
```

```
          vector_delete s b the_function A.Char

        ) finalbuilder literals in


       let structs = Hashtbl.fold (fun s t l -> (s,t)::l) struct_type_map [] in

       let rec delete_vecs_of_fields ptr struct_def_map builder the_function =
              let bindings = StringMap.bindings struct_def_map in
              let delete_vecs_of_fields_helper builder binding =
                  let (_, (fieldtyp, idx)) = binding in
                  (match fieldtyp with
                    A.Vector(in_typ) -> let target_ptr = L.build_struct_gep ptr idx "vec"
 builder in
                        vector_delete target_ptr builder the_function in_typ
                    | A.Struct(in_typ) -> let target_ptr = L.build_struct_gep ptr idx "in
ner_struct_field" builder in
                        let field_struct_def_map = StringMap.find in_typ global_map  in
                        delete_vecs_of_fields target_ptr field_struct_def_map builder the
_function
                    | _ -> builder)
              in
              List.fold_left delete_vecs_of_fields_helper builder bindings

        in
        let delete_vecs_of_struct builder strct =
          let (id, typ) = strct in
           let ptr = lookup id in
           let struct_def_map = StringMap.find typ global_map in
           delete_vecs_of_fields ptr struct_def_map builder the_function
        in
        let reallyfinalbuilder = List.fold_left delete_vecs_of_struct reallyfinalbuilder
structs in

       (* execute the actual return *)
       add_terminal reallyfinalbuilder (if (fdecl.A.typ = A.Void) then L.build_ret_void
                                  else (L.build_ret (L.build_load retval "final_retval"
 reallyfinalbuilder))) ;

        in
        let _ = List.iter build_function_body functions in

        let job_decls =
                let job_decl m jobdecl =
                match jobdecl.A.fname with
                "master" -> m
                | _ ->
                        let job_name = jobdecl.A.fname ^ "_job"
                        in let void_pt = L.pointer_type i8_t
                        in let void_ppt = L.pointer_type void_pt
                        in let job_formal_type = [|void_pt; i32_t; void_ppt; L.pointer_ty
pe i32_t|]
                in let jobtype = L.function_type void_t job_formal_type in
                        StringMap.add job_name (L.define_function job_name jobtype the_mo
dule, jobdecl) m in
                List.fold_left job_decl StringMap.empty functions in

                let build_job_body jobdecl =
                        match jobdecl.A.fname with
                        "master" -> ()
                        | _ ->
                        let (the_job_function, _) = StringMap.find (jobdecl.A.fname ^ "_j
ob") job_decls in
                        let (the_function, _) = StringMap.find jobdecl.A.fname function_d
ecls in
                let builder = L.builder_at_end context (L.entry_block the_job_function) i
n

                (*set_value_name n p -> sets name in AST to equal to the name in llvm bui
lder*)
```

```
                let add_formal n p = L.set_value_name n p in
                ignore (List.iter2 add_formal ["input_data"; "input_leng"; "output_data";
 "output_leng"]
                                          (Array.to_list (L.params the_job_function))) ;
                let input_buf = L.param the_job_function 0 in
                let input_store = L.build_alloca (L.pointer_type i8_t) "input_store" buil
der in
                ignore (L.build_store input_buf input_store builder);
                let typs = List.map (fun (typ, _) -> typ) jobdecl.A.formals in
                let (arg_ints, _, _, newbuilder) = List.fold_left deserialize ([], input_
store, the_job_function, builder) typs in
                                let return_result = (match jobdecl.A.typ with A.Void -> "
" | _ -> jobdecl.A.fname ^ "_result") in
                                let tmp_call = L.build_call the_function (Array.of_list (
List.rev arg_ints)) return_result newbuilder in
                                let last_builder = (match jobdecl.A.typ with
                          A.Void -> ignore (L.build_store (L.const_null (L.pointer_type i8_
t)) (L.param the_job_function 2) newbuilder) ;
                                    ignore (L.build_store (L.const_null i32_t) (L.param the
_job_function 3) newbuilder);
                                    newbuilder
                        | typ -> let (buf, len, nnbuilder) = serialize newbuilder [tmp_call
] the_job_function [typ] in
                                                ignore (L.build_store buf (L.param the_job
_function 2) nnbuilder) ;
                                                ignore (L.build_store len (L.param the_job
_function 3) nnbuilder);
                                    (match typ with A.Vector(t) -> let vec_store = L.build_all
oca vec_t "vec_store" nnbuilder in
                                                                   ignore (L.build_store tmp_c
all vec_store nnbuilder) ;
                                                                   vector_delete vec_store nnb
uilder the_job_function t
                                                | A.Struct(n) -> let e_store = L.build_all
oca (ltype_of_typ (A.Struct(n))) "e_store" nnbuilder in
                                                                   ignore (L.build_store tmp_c
all e_store nnbuilder);
                                                                   let field_struct_def_map =
StringMap.find n global_map  in
                                                                   struct_delete e_store field
_struct_def_map nnbuilder the_job_function
                                                 | _ -> nnbuilder)) in

                                ignore(L.build_ret_void last_builder) in

            List.iter build_job_body functions ;

  let jobdecl_of_fdecl out_list = function
      x when (x.A.fname = "master") -> out_list
    | x -> let (a,_) = (StringMap.find (x.A.fname ^ "_job") job_decls) in
           a::out_list in
  let job_ptrs = List.fold_left jobdecl_of_fdecl [] (List.rev functions) in
  let job_func_type = L.function_type void_t [| (L.pointer_type i8_t) ; i32_t ;
    (L.pointer_type (L.pointer_type i8_t)); (L.pointer_type i32_t) |] in
  let job_ptr_arr = L.const_array (L.pointer_type job_func_type) (Array.of_list job_ptrs)
 in
  ignore (L.define_global "job_funcs" job_ptr_arr the_module) ;

  ignore (L.define_global "num_jobs" (L.const_int i32_t (List.length (StringMap.bindings
job_decls))) the_module) ;

the_module
```

```
# Make sure ocamlbuild can find opam-managed packages: first run
#
# eval `opam config env`

# Easiest way to build: using ocamlbuild, which in turn uses ocamlfind

all : mscompile.native

mscompile.native :
	ocamlbuild -use-ocamlfind -pkgs llvm,llvm.analysis -cflags -w,+a-4 \
		mscompile.native

# "make clean" removes all generated files

.PHONY : clean
clean :
	ocamlbuild -clean
	rm -rf testall.log *.diff mscompile scanner.ml parser.ml parser.mli
	rm -rf *.cmx *.cmi *.cmo *.ll *.cmx *.o *.s *.out *.exe *.a *.err *~
	rm -rf */master.dSYM/ */slave.dSYM/
	rm -rf *master *slave

# "make clean_tests" removes all generated files from testall.sh

.PHONY : clean_tests
clean_tests :
	rm -rf testall.log *.diff *master *slave ./tests/*.ll *.err *~ *.ll .ll *.out *.s

# More detailed: build using ocamlc/ocamlopt + ocamlfind to locate LLVM

OBJS = ast.cmx codegen.cmx parser.cmx scanner.cmx semant.cmx mscompile.cmx

mscompile : $(OBJS)
	ocamlfind ocamlopt -linkpkg -package llvm -package llvm.analysis $(OBJS) -o mscom
pile

scanner.ml : scanner.mll
	ocamllex scanner.mll

parser.ml parser.mli : parser.mly
	ocamlyacc parser.mly

%.cmo : %.ml
	ocamlc -c $<

%.cmi : %.mli
	ocamlc -c $<

%.cmx : %.ml
	ocamlfind ocamlopt -c -package llvm $<

### Generated by "ocamldep *.ml *.mli" after building scanner.ml and parser.ml
ast.cmo :
ast.cmx :
codegen.cmo : ast.cmo
codegen.cmx : ast.cmx
mscompile.cmo : semant.cmo scanner.cmo parser.cmi codegen.cmo ast.cmo
mscompile.cmx : semant.cmx scanner.cmx parser.cmx codegen.cmx ast.cmx
parser.cmo : ast.cmo parser.cmi
parser.cmx : ast.cmx parser.cmi
scanner.cmo : parser.cmi
scanner.cmx : parser.cmx
semant.cmo : ast.cmo
semant.cmx : ast.cmx
parser.cmi : ast.cmo

# Building the tarball

TESTS = add1 arith1 arith2 arith3 fib for1 for2 func1 func2 func3       \
    func4 func5 func6 func8 gcd2 gcd
```

```
\
    double1     double-add vector
                \
    hello if1 if2 if3 if4 if5 local1 local2 ops1 ops2 var1                    \
    while1 while2                                                        \
    remote-int remote-many-ints remote-doubles                                    \
    remote-job-states remote-cancel
        \


FAILS = assign1 assign2 assign3 dead1 dead2 expr1 expr2 for1 for2       \
    for3 for4 for5 func1 func2 func3 func4 func5 func6 func7 func8      \
    func9 if1 if2 if3 nomaster return1 return2 while1 while2

TESTFILES = $(TESTS:%=test-%.mc) $(TESTS:%=test-%.out) \
            $(FAILS:%=fail-%.mc) $(FAILS:%=fail-%.err)

TARFILES = ast.ml codegen.ml Makefile mscompile.ml parser.mly README scanner.mll \
        semant.ml testall.sh $(TESTFILES:%=tests/%) arcade-font.pbm \
        font2c

mscompile-llvm.tar.gz : $(TARFILES)
        cd .. && tar czf mscompile-llvm/mscompile-llvm.tar.gz \
                $(TARFILES:%=mscompile-llvm/%)
```

```
(* Top-level of the M/s compiler: scan & parse the input,
   check the resulting AST, generate LLVM IR, and dump the module *)

type action = Ast | LLVM_IR | Compile

let _ =
  let action = if Array.length Sys.argv > 1 then
    List.assoc Sys.argv.(1) [ ("-a", Ast);      (* Print the AST only *)
                              ("-l", LLVM_IR);  (* Generate LLVM, don't check *)
                              ("-c", Compile) ] (* Generate, check LLVM IR *)
  else Compile in
  let lexbuf = Lexing.from_channel stdin in
  let ast = Parser.program Scanner.token lexbuf in
  Semant.check ast;
  match action with
    Ast -> print_string (Ast.string_of_program ast)
  | LLVM_IR -> print_string (Llvm.string_of_llmodule (Codegen.translate ast))
  | Compile -> let m = Codegen.translate ast in
    Llvm_analysis.assert_valid_module m;
    print_string (Llvm.string_of_llmodule m)
```

```
/* Ocamlyacc parser for M/s */

%{
open Ast
%}

%token SEMI LPAREN RPAREN LCURL RCURL LBRACE RBRACE COMMA COLON DOT
%token PLUS MINUS TIMES DIVIDE ASSIGN
%token EQ NEQ LT LEQ GT GEQ TRUE FALSE AND OR NOT
%token RETURN IF ELSE FOR WHILE INT DOUBLE BOOL STRING VOID CONCAT
%token VECTOR
%token STRUCT FIELD
%token JOB MASTER REMOTE GET CANCEL RUNNING FINISHED FAILED

%token <int> INT_LITERAL
%token <float> DOUBLE_LITERAL
%token <string> STRING_LITERAL
%token <string> ID
%token EOF

%token SIZE PUSHBACK

%nonassoc NOELSE
%nonassoc ELSE
%right ASSIGN
%right SIZE
%left OR
%left AND
%left EQ NEQ
%left LT GT LEQ GEQ
%left PLUS MINUS
%left TIMES DIVIDE
%left CONCAT
%right NOT NEG
%right RBRACE
%left LBRACE
%right FIELD

%start program
%type <Ast.program> program

%%

program:
  struct_type_decl program { let (f,s) = $2 in f, $1 :: s }
| fdecl program { let (f,s) = $2 in $1 :: f, s }
| master program_terminable { let (f,s) = $2 in $1::f, s }

program_terminable:
  struct_type_decl program_terminable { let (f,s) = $2 in f, $1 :: s }
| fdecl program_terminable { let (f,s) = $2 in $1 :: f, s }
| EOF { [], [] }

master:
  MASTER LCURL stmt_list RCURL {
    {
      fname = "master";
      typ = Int;
      formals = [];
      body = List.rev  $3;
    }
  }

fdecl:
   JOB typ ID LPAREN formals_opt RPAREN LCURL stmt_list RCURL
     { { typ = $2;
         fname = $3;
         formals = $5;
         body = List.rev $8 } }
```

```
struct_type_decl:
  STRUCT ID LCURL bind_list RCURL SEMI { {sname = $2; blist = List.rev $4} }

formals_opt:
    /* nothing */ { [] }
  | formal_list   { List.rev $1 }

formal_list:
    typ ID                   { [($1,$2)] }
  | formal_list COMMA typ ID { ($3,$4) :: $1 }

bind:
    typ ID SEMI { ($1, $2) }

bind_list:
    /* nothing */ { [] }
  | bind_list bind { $2 :: $1 }

typ:
    INT { Int }
  | BOOL { Bool }
  | DOUBLE { Double }
  | STRING { Vector(Char) }
  | VOID { Void }
  | VECTOR LT typ GT { Vector($3) }
  | JOB LT typ GT {Job($3)}
  | STRUCT ID { Struct($2) }

vdecl:
  typ ID SEMI {VarDecl($1,$2)}
  | typ ID ASSIGN expr SEMI { VarDeclAssign($1, $2, $4) }

/* first list is vdecl list, second list is statement list */
stmt_list:
    /* nothing */  { [] }
  | stmt_list stmt { $2 :: $1 }

stmt:
    expr SEMI { Expr $1 }
  | vdecl { $1 }
  | RETURN SEMI { Return Noexpr }
  | RETURN expr SEMI { Return $2 }
  | LCURL stmt_list RCURL { Block(List.rev $2) }
  | IF LPAREN expr RPAREN stmt %prec NOELSE { If($3, $5, Block([])) }
  | IF LPAREN expr RPAREN stmt ELSE stmt { If($3, $5, $7) }
  | FOR LPAREN expr_opt SEMI expr SEMI expr_opt RPAREN stmt { For($3, $5, $7, $9) }
  | WHILE LPAREN expr RPAREN stmt { While($3, $5) }
  | expr PUSHBACK expr SEMI {VectorPushBack($1, $3)}

expr_opt:
    /* nothing */ { Noexpr }
  | expr          { $1 }

expr:
    INT_LITERAL      { Literal($1) }
  | DOUBLE_LITERAL   { DoubleLit($1) }
  | STRING_LITERAL   { StringLit($1) }
  | TRUE             { BoolLit(true) }
  | FALSE            { BoolLit(false) }
  | ID               { Id($1) }
  | LBRACE actuals_opt RBRACE { ListLiteral($2) }
  | expr PLUS   expr { Binop($1, Add,   $3) }
  | expr MINUS  expr { Binop($1, Sub,   $3) }
  | expr TIMES  expr { Binop($1, Mult,  $3) }
  | expr DIVIDE expr { Binop($1, Div,   $3) }
  | expr EQ     expr { Binop($1, Equal, $3) }
  | expr NEQ    expr { Binop($1, Neq,   $3) }
  | expr LT     expr { Binop($1, Less,  $3) }
  | expr LEQ    expr { Binop($1, Leq,   $3) }
  | expr GT     expr { Binop($1, Greater, $3) }
```

```
  | expr GEQ    expr { Binop($1, Geq,   $3) }
  | expr AND    expr { Binop($1, And,   $3) }
  | expr OR     expr { Binop($1, Or,    $3) }
  | MINUS expr %prec NEG { Unop(Neg, $2) }
  | NOT expr         { Unop(Not, $2) }
  | ID ASSIGN expr   { Assign($1, $3) }
  | ID LPAREN actuals_opt RPAREN { Call($1, $3) }
  | REMOTE ID LPAREN actuals_opt RPAREN { RemoteCall($2, $4) }
  | GET ID { Get($2) }
  | CANCEL ID { Cancel($2) }
  | ID DOT RUNNING { Running($1) }
  | ID DOT FINISHED { Finished($1) }
  | ID DOT FAILED { Failed($1) }
  | LPAREN expr RPAREN { $2 }
  | ID vector_indices {VectorAccess($1, $2)}
  | ID vector_indices ASSIGN expr {VectorAssign($1, $2, $4)}
  | ID LBRACE expr COLON expr RBRACE {VectorRangeAccess($1, $3, $5)}
  | expr FIELD ID               { StructFieldAccess($1, $3) }
  | expr FIELD ID ASSIGN expr   { StructFieldAssign($1, $3, $5)}
  | SIZE expr {VectorSize($2)}
  | expr CONCAT expr {Concat($1, $3)}

vector_indices:
    LBRACE expr RBRACE {[$2]}
  | vector_indices LBRACE expr RBRACE {$3 :: $1}

actuals_opt:
    /* nothing */ { [] }
  | actuals_list  { List.rev $1 }

actuals_list:
    expr                    { [$1] }
  | actuals_list COMMA expr { $3 :: $1 }
```

```
#!/bin/bash

input_ms=$1
echo $1
./mscompile.native < $1 > test.ll
llc test.ll
gcc -g -L. test.s -lmsslave -pthread -lm -o slave
gcc -g -L. test.s -lmsmaster -pthread -lm -o master
```

```
(* Ocamllex scanner for M/s *)

{ open Parser }

let whitespace = [' ' '\t' '\r' '\n']
let alpha = ['a'-'z' 'A'-'Z']
let digit = ['0'-'9']
let ascii = ([' '-'!' '#'-'[' ']'-'~'])
let escape = '\\' ['\\' ''' '"' 'n' 'r' 't']
let escape_char = ''' (escape) '''
let id = (alpha | '_')(alpha | digit | '_')*

rule token = parse
  whitespace { token lexbuf }  (* Whitespace *)
| "/*"           { comment lexbuf }  (* Comments *)
| "//"           { line_comment lexbuf }

(* Operators and separators *)
| '('      { LPAREN }
| ')'      { RPAREN }
| '{'      { LCURL }
| '}'      { RCURL }
| '['      { LBRACE }
| ']'      { RBRACE }
| '-''>'   { FIELD }
| ';'      { SEMI }
| ','      { COMMA }
| '.'      { DOT }
| '+'      { PLUS }
| '-'      { MINUS }
| '*'      { TIMES }
| '/'      { DIVIDE }
| '='      { ASSIGN }
| ':'      { COLON }
| "=="     { EQ }
| "!="     { NEQ }
| '<'      { LT }
| "<="     { LEQ }
| ">"      { GT }
| ">="     { GEQ }
| "&&"     { AND }
| "||"     { OR }
| '!'      { NOT }

(* Branching control *)
| "if"     { IF }
| "else"   { ELSE }
| "for"    { FOR }
| "while"  { WHILE }
| "return" { RETURN }

(* Types *)
| "int"    { INT }
| "double" { DOUBLE }
| "string" { STRING }
| "bool"   { BOOL }
| "void"   { VOID }
| "true"   { TRUE }
| "false"  { FALSE }
| "vector" { VECTOR }
| "struct" { STRUCT }

(* Job related *)
| "job"    { JOB }
| "master" { MASTER }
| "remote" { REMOTE }
| "get"    { GET }
| "cancel" { CANCEL }
| "running" { RUNNING }
| "finished" { FINISHED }
```

```
| "failed" { FAILED }

(* Vector related *)
| "size"   {SIZE}
| "::"     {PUSHBACK}

| "<<" {CONCAT}
(* Alternative float literal? *)
(*| ['-''+']?digit*('.')digit+(['e''E']['-''+']?digit+)? as lxm { DOUBLE_LITERAL(float_of
_string lxm) } *)
| (digit+)'.'(digit+) as lxm { DOUBLE_LITERAL(float_of_string lxm)}
| digit+ as lxm         { INT_LITERAL(int_of_string lxm) }
| '"'((ascii|escape|alpha|digit)* as lxm)'"'  { STRING_LITERAL(lxm) }
| id as lxm             { ID(lxm) }
| eof                          { EOF }
| _ as char             { raise (Failure("illegal character " ^ Char.escaped char)) }

and comment = parse
  "*/" { token lexbuf }
| _    { comment lexbuf }

and line_comment = parse
  '\n' { token lexbuf }
| _    { line_comment lexbuf }
```

```
(* Semantic checking for the M/s compiler *)

open Ast

module StringMap = Map.Make(String)


let check (functions, struct_decls) =

  (* Raise an exception if the given list has a duplicate *)
  let report_duplicate exceptf list =
    let rec helper = function
        n1 :: n2 :: _ when n1 = n2 -> raise (Failure (exceptf n1))
      | _ :: t -> helper t
      | [] -> ()
    in helper (List.sort compare list)
  in

  (*let get_vector_element_type = function
      Int -> Int
    | Double -> Double
    | Bool -> Bool
    | Void -> Void
    | Job(x) -> Job(x)
    | Vector(t) -> Vector(t)
  in *)

let rec create_struct_map map  = function
      [] -> StringMap.empty
    | [(typ, name)] -> StringMap.add name typ map
    | h::t -> let m = create_struct_map map  [h] in create_struct_map m t
  in
  let rec create_global_map global_map = function
      [] -> StringMap.empty
    |  [x] -> StringMap.add x.sname (create_struct_map StringMap.empty x.blist) global_ma
p
    | h::t -> let m = create_global_map global_map [h] in create_global_map m t
  in

  let global_map = create_global_map StringMap.empty struct_decls
in

  (* Raise an exception if a given binding is to a void type *)
  let check_not_void exceptf = function
      (Void, n) -> raise (Failure (exceptf n))
    | _ -> ()
  in

  (* Raise an exception of the given rvalue type cannot be assigned to
     the given lvalue type *)
  let check_assign lvaluet rvaluet err =
     if lvaluet = rvaluet then lvaluet else raise err
  in
  if List.mem "print" (List.map (fun fd -> fd.fname) functions)
  then raise (Failure ("function print may not be defined")) else
  if List.mem "printd" (List.map (fun fd -> fd.fname) functions)
  then raise (Failure ("function printd may not be defined")) else
  if List.mem "printb" (List.map (fun fd -> fd.fname) functions)
  then raise (Failure ("function printb may not be defined")) else
  if List.mem "prints" (List.map (fun fd -> fd.fname) functions)
  then raise (Failure ("function prints may not be defined")) else ();

  report_duplicate (fun n -> "duplicate function " ^ n)
    (List.map (fun fd -> fd.fname) functions);


  (* Function declaration for a named function *)

  let built_in_decls = List.fold_left
   (fun map (name, t) -> StringMap.add name t map)
```

```
   StringMap.empty
    [ ("print", { typ = Void; fname = "print"; formals = [(Int, "x")]; body = [] });
      ("printb", { typ = Void; fname = "printb"; formals = [(Bool, "x")]; body = [] });
      ("printd", { typ = Void; fname = "printd"; formals = [(Double, "x")]; body = [] });
      ("prints", { typ = Void; fname = "prints"; formals = [(Vector(Char), "x")]; body =
[] });
    ]
  in

  (* adds (fname, AST.fdecl) to map function_decls *)
  let function_decls = List.fold_left (fun m fd -> StringMap.add fd.fname fd m)
                       built_in_decls functions
  in

  let function_decl s = try StringMap.find s function_decls
       with Not_found -> raise (Failure ("unrecognized function " ^ s))
  in

  let _ = function_decl "master" in (*Ensure "master" is defined*)
  (*let expr_type = Hashtbl.create 5000 in*)
  let check_function func =
    List.iter (check_not_void (fun n -> "illegal void formal " ^ n ^
      " in " ^ func.fname)) func.formals;

    report_duplicate (fun n -> "duplicate formal " ^ n ^ " in " ^ func.fname)
      (List.map snd func.formals);

  (* Type of each variable (global, formal, or local *)
  let symbols = Hashtbl.create 5000 in
  let _ = List.iter (fun (t, n) -> Hashtbl.add symbols n t)
    func.formals in

  let type_of_identifier s =
    try Hashtbl.find symbols s
    with Not_found -> raise (Failure ("undeclared identifier " ^ s))
  in

  let struct_decl s =
    try StringMap.find s global_map
    with Not_found -> raise (Failure ("undeclared struct " ^ s))
  in

  (* Return the type of an expression or throw an exception *)
  let rec expr e = match e with
      Literal _ -> Int
    | DoubleLit _ -> Double
    | StringLit _ -> Vector(Char)
    | BoolLit _ -> Bool
    | Id s -> type_of_identifier s
    | Binop(e1, op, e2) as e -> let t1 = expr e1 and t2 = expr e2 in
        (match op with
            Add | Sub | Mult | Div when t1 = Int && t2 = Int -> Int
          | Add | Sub | Mult | Div when t1 = Double && t2 = Double -> Double
          | Equal | Neq when t1 = t2 -> Bool
          | Less | Leq | Greater | Geq when t1 = Int && t2 = Int || t1 = Double && t2 = D
ouble -> Bool
          | And | Or  when t1 = Bool && t2 = Bool -> Bool
          | _ -> raise (Failure ("illegal binary operator " ^
            string_of_typ t1 ^ " " ^ string_of_op op ^ " " ^
          string_of_typ t2 ^ " in " ^ string_of_expr e))
        )
    | Unop(op, e) as ex -> let t = expr e in
       (match op with
         Neg when t = Int -> Int
       | Neg when t = Double -> Double
       | Not when t = Bool -> Bool
            | _ -> raise (Failure ("illegal unary operator " ^ string_of_uop op ^
              string_of_typ t ^ " in " ^ string_of_expr ex)))
    | Noexpr -> Void
    | Assign(var, e) as ex -> let lt = type_of_identifier var
```

```
                              and rt = expr e in
        check_assign lt rt (Failure ("illegal assignment " ^ string_of_typ lt ^
           " = " ^ string_of_typ rt ^ " in " ^
           string_of_expr ex))
    | RemoteCall (fname, actuals) as call-> let fd = function_decl fname in
      if List.length actuals != List.length fd.formals then
        raise (Failure ("expecting " ^ string_of_int
          (List.length fd.formals) ^ " arguments in " ^ string_of_expr call))
      else
        List.iter2 (fun (ft, _) e -> let et = expr e in
           ignore (check_assign ft et
             (Failure ("illegal actual argument found " ^ string_of_typ et ^
             " expected " ^ string_of_typ ft ^ " in " ^ string_of_expr e))))
          fd.formals actuals;
          Job((function_decl fname).typ)
    | Call(fname, actuals) as call -> let fd = function_decl fname in
      if List.length actuals != List.length fd.formals then
        raise (Failure ("expecting " ^ string_of_int
          (List.length fd.formals) ^ " arguments in " ^ string_of_expr call))
      else
        List.iter2 (fun (ft, _) e -> let et = expr e in
           ignore (check_assign ft et
             (Failure ("illegal actual argument found " ^ string_of_typ et ^
             " expected " ^ string_of_typ ft ^ " in " ^ string_of_expr e))))
          fd.formals actuals;
        fd.typ

    | Get(job) -> (match (type_of_identifier job) with
          Job(typ) -> typ
          | _ -> raise(Failure("Getting something not a job")) )

    | Cancel(job) -> (match (type_of_identifier job) with
          Job(_) -> Void
          | _ -> raise(Failure("Cancelling something not a job")) )
    | Running(job) -> (match (type_of_identifier job) with
          Job(_) -> Bool
          | _ -> raise(Failure("Checking running of something not a job")) )
    | Finished(job) -> (match (type_of_identifier job) with
          Job(_) -> Bool
          | _ -> raise(Failure("Checking finished of something not a job")) )
    | Failed(job) -> (match (type_of_identifier job) with
          Job(_) -> Bool
          | _ -> raise(Failure("Checking failed of something not a job")) )

    | VectorAccess(e, e1) -> let rec step = function
                               (typ, []) -> typ
                             | (Vector(typ), _::l) -> step (typ, l)
                             | _ -> raise(Failure("Vector access of something not a vecto
r")) in
                         step ((type_of_identifier e), e1)
    | VectorAssign(id, e1, e2) -> let ltyp = expr (VectorAccess(id, e1)) in
                                  let rtyp = expr e2 in
                                  ignore (check_assign ltyp rtyp
                                         (Failure ("illegal vector assignment found "
                                                  ^ string_of_typ rtyp ^ " expected "
                                                  ^ string_of_typ ltyp))) ; ltyp
    | VectorSize(_) -> Int
    | StructFieldAccess(e, fieldname) -> (match (expr e) with
      | Struct(struct_name) ->
          let struct_def_map = struct_decl struct_name in
          if StringMap.mem fieldname struct_def_map
            then StringMap.find fieldname struct_def_map
          else raise(Failure("cannot find fieldname: " ^ fieldname))
      | _ -> raise(Failure("accessing field of something not a struct")) )

    | StructFieldAssign(e1, fieldname, e) -> let ltyp = expr (StructFieldAccess(e1, field
name)) in
      let rtyp = expr e in ignore(check_assign ltyp rtyp (Failure ("illegal struct field
assignment"))); ltyp
    | Concat(e1, e2) -> let ltyp = expr e1 in let rtyp = expr e2 in ignore(check_assign l
```

```
typ rtyp (Failure ("illegal string concat"))); ltyp
    | _ -> raise(Failure("unrecognized expr"))

  in

  let check_bool_expr e = if expr e != Bool
     then raise (Failure ("expected Boolean expression in " ^ string_of_expr e))
     else ()
  in

  (* Verify a statement or throw an exception *)
  let rec stmt inside_loop = function
    | Block sl -> let rec check_block = function
        [Return _ as s] -> stmt inside_loop s
      | Return _ :: _ -> raise (Failure "nothing may follow a return")
      | Block sl :: ss -> check_block (sl @ ss)
      | s :: ss -> stmt inside_loop s ; check_block ss
      | [] -> ()
      in check_block sl

    | VarDecl(typ, id) ->
          if inside_loop then (match typ with
            Vector(_) -> raise (Failure "cannot declare vectors or strings inside for loo
p")
            | _ -> ();
          );
          (match typ with
            | Struct(name) -> ignore(struct_decl name); ignore(Hashtbl.add symbols id typ
);
            check_not_void (fun n -> "illegal void local " ^ n ^ " in " ^ func.fname) (ty
p, id)
            | _ -> if Hashtbl.mem symbols id then raise (Failure "duplicate local variabl
e declarations") else ignore(Hashtbl.add symbols id typ) ;
            check_not_void (fun n -> "illegal void local " ^ n ^ " in " ^ func.fname) (ty
p, id)
          )
    | VarDeclAssign(typ, id, e) ->
          if inside_loop then (match typ with
            Vector(_) -> raise (Failure "cannot declare vectors or strings inside for loo
p")
            | _ -> ();
          );
          if Hashtbl.mem symbols id then raise (Failure "duplicate local variable declara
tions") else  ignore(Hashtbl.add symbols id typ) ;
          check_not_void (fun n -> "illegal void local " ^ n ^ " in " ^ func.fname) (typ,
 id) ;
          let rt = expr e in
          ignore (check_assign typ rt (Failure ("illegal assignment " ^ string_of_typ typ
 ^ " = " ^ string_of_typ rt)))

    | Expr e -> ignore (expr e)
    | Return e -> let t = expr e in if t = func.typ then () else
       raise (Failure ("return gives " ^ string_of_typ t ^ " expected " ^
                       string_of_typ func.typ ^ " in " ^ string_of_expr e))

    | If(p, b1, b2) -> check_bool_expr p; stmt inside_loop b1; stmt inside_loop b2
    | For(e1, e2, e3, st) -> ignore (expr e1); check_bool_expr e2;
                             ignore (expr e3); stmt true st
    | While(p, s) -> check_bool_expr p; stmt true s
    | VectorPushBack(e1, e) -> (match (expr e1) with
                Vector(ltyp) -> let rtyp = expr e in
                    ignore (check_assign ltyp rtyp (Failure ("Illegal vector pushback fou
nd "
                    ^ string_of_typ rtyp^ " expected " ^ string_of_typ ltyp)))
              | _ -> raise(Failure("pushing back to something not a vector")))
  in

  stmt false (Block func.body)

  in
```

```
  List.iter check_function functions
```

```
#!/bin/sh

# Regression testing script for M/s
# Step through a list of files
#  Compile, run, and check the output of each expected-to-work test
#  Compile and check the error of each expected-to-fail test

# Path to the LLVM interpreter
LLI="lli"
#LLI="/usr/local/opt/llvm/bin/lli"

# Path to the ms compiler.  Usually "./mscompile.native"
# Try "_build/mscompile.native" if ocamlbuild was unable to create a symbolic link.
MSCOMPILE="./mscompile.native"
#MSCOMPILE="_build/mscompile.native"

# Set a port number to run
PORT="8888"

# Set time limit for all operations
ulimit -t 30

globallog=testall.log
rm -f $globallog
error=0
globalerror=0

keep=0

Usage() {
    echo "Usage: testall.sh [options] [.ms files]"
    echo "-k    Keep intermediate files"
    echo "-h    Print this help"
    exit 1
}

SignalError() {
    if [ $error -eq 0 ] ; then
        echo "FAILED"
        error=1
    fi
    echo "  $1"
}

# Compare <outfile> <reffile> <difffile>
# Compares the outfile with reffile.  Differences, if any, written to difffile
Compare() {
    generatedfiles="$generatedfiles $3"
    echo diff -b $1 $2 ">" $3 1>&2
    diff -b "$1" "$2" > "$3" 2>&1 || {
        SignalError "$1 differs"
        echo "FAILED $1 differs from $2" 1>&2
    cat "$3"
    }
}

# Run <args>
# Report the command, run it, and report any errors
Run() {
    echo $* 1>&2
    eval $* || {
        SignalError "$1 failed on $*"
        return 1
    }
}

# RunFail <args>
# Report the command, run it, and expect an error
RunFail() {
    echo $* 1>&2
```

```
    eval $* && {
        SignalError "failed: $* did not report an error"
        return 1
    }
    return 0
}

Check() {
    error=0
    basename=`echo $1 | sed 's/.*\\///
                             s/.ms//'`
    reffile=`echo $1 | sed 's/.ms$//'`
    basedir="`echo $1 | sed 's/\/[^\/]*$//'`/."

    echo -n "$basename..."

    echo 1>&2
    echo "###### Testing $basename" 1>&2

    generatedfiles=""

    # Note: libmaster.a and libslave.a must be present in the compilers folder already
    generatedfiles="$generatedfiles ${basename}.ll ${basename}.out" &&
    Run "$MSCOMPILE" "<" $1 ">" "${basename}.ll" &&
    Run "llc ${basename}.ll" &&
    Run "gcc -L. ${basename}.s -lmsmaster -pthread -lm -o ${basename}-master" &&
    Run "gcc -L. ${basename}.s -lmsslave -pthread -lm -o ${basename}-slave" &&
    # Change port number if tests are freezing
    Run "./${basename}-master $PORT > ${basename}.out" &&
    Compare ${basename}.out ${reffile}.out ${basename}.diff

    # Report the status and clean up the generated files

    if [ $error -eq 0 ] ; then
    if [ $keep -eq 0 ] ; then
        rm -f $generatedfiles
    fi
    echo "OK"
    echo "###### SUCCESS" 1>&2
    else
    echo "###### FAILED" 1>&2
    globalerror=$error
    fi
}

CheckRemote() {
    error=0
    basename=`echo $1 | sed 's/.*\\///
                             s/.ms//'`
    reffile=`echo $1 | sed 's/.ms$//'`
    basedir="`echo $1 | sed 's/\/[^\/]*$//'`/."

    echo -n "$basename..."

    echo 1>&2
    echo "###### Testing $basename" 1>&2

    generatedfiles=""

    # Note: libmaster.a and libslave.a must be present in the compilers folder already
    generatedfiles="$generatedfiles ${basename}.ll ${basename}.out" &&
    Run "$MSCOMPILE" "<" $1 ">" "${basename}.ll" &&
    Run "llc ${basename}.ll" &&
    Run "gcc -L. ${basename}.s -lmsmaster -pthread -lm -o ${basename}-master" &&
    Run "gcc -L. ${basename}.s -lmsslave -pthread -lm -o ${basename}-slave" &&
    # Change port number if tests are freezing
    Run './${basename}-master $PORT > ${basename}.out & PID=$! ; while [ -z "`netstat -an
 | grep $PORT`" ] ; do : ; done ; ./${basename}-slave $PORT ; wait $PID' &&
    Compare ${basename}.out ${reffile}.out ${basename}.diff
```

```
    # Report the status and clean up the generated files

    if [ $error -eq 0 ] ; then
    if [ $keep -eq 0 ] ; then
        rm -f $generatedfiles
    fi
    echo "OK"
    echo "###### SUCCESS" 1>&2
    else
    echo "###### FAILED" 1>&2
    globalerror=$error
    fi
}

CheckFail() {
    error=0
    basename=`echo $1 | sed 's/.*\\///
                            s/.ms//'`
    reffile=`echo $1 | sed 's/.ms$//'`
    basedir="`echo $1 | sed 's/\/[^\/]*$//'`/."

    echo -n "$basename..."

    echo 1>&2
    echo "###### Testing $basename" 1>&2

    generatedfiles=""

    generatedfiles="$generatedfiles ${basename}.err ${basename}.diff" &&
    RunFail "$MSCOMPILE" "<" $1 "2>" "${basename}.err" ">>" $globallog &&
    Compare ${basename}.err ${reffile}.err ${basename}.diff

    # Report the status and clean up the generated files

    if [ $error -eq 0 ] ; then
        if [ $keep -eq 0 ] ; then
            rm -f $generatedfiles
        fi
        echo "OK"
        echo "###### SUCCESS" 1>&2
    else
        echo "###### FAILED" 1>&2
        globalerror=$error
    fi
}

while getopts kdpsh c; do
    case $c in
        k) # Keep intermediate files
            keep=1
            ;;
        h) # Help
            Usage
            ;;
    esac
done

shift `expr $OPTIND - 1`

LLIFail() {
  echo "Could not find the LLVM interpreter \"$LLI\"."
  echo "Check your LLVM installation and/or modify the LLI variable in testall.sh"
  exit 1
}

which "$LLI" >> $globallog || LLIFail


if [ $# -ge 1 ]
then
```

```
    files=$@
else
    files="tests/test-*.ms tests/fail-*.ms"
fi

for file in $files
do
    case $file in
        *test-remote-*)
            CheckRemote $file 2>> $globallog
            ;;
    *test-*)
        Check $file 2>> $globallog
        ;;
        *fail-*)
            CheckFail $file 2>> $globallog
            ;;
        *)
            echo "unknown file type $file"
            globalerror=1
            ;;
    esac
done

exit $globalerror
```

```
#include "job_list.h"
#include <stdio.h>

static struct job *job_lookup(struct job *beg, struct job *end, int jid)
{
    //printf("search with beg=%p, end=%p, jid=%d\n", beg, end, jid);
    struct job *mid = beg + (end-beg)/2;
    if ((mid == end) || (mid->jid > jid && mid == beg))
        return NULL;
    if (mid->jid > jid)
        return job_lookup(beg, mid, jid);
    if (mid->jid < jid)
        return job_lookup(mid +1, end, jid);
    return mid;
}

static void job_list_resize(struct job_list *jl, int dif) /* dif = +/-1 */
{
    int oldlen = jl->len;
    jl->len += dif;
    if ((!(jl->len & (jl->len - 1)) && (dif < 0)) ||
            (!((jl->len - 1) & (jl->len - 2)) && (dif > 0))) {
        struct job *old_start = jl->jobs;
        int newsize;
        if ((jl->len < 2) || (dif < 0))
            newsize = jl->len * sizeof (struct job);
        else
            newsize = (jl->len - 1) * 2 * sizeof(struct job);
        jl->jobs = realloc(jl->jobs, newsize);
        for (int i = 0; i < ((dif > 0) ? oldlen : jl->len); i++) {
            if (jl->jobs[i].prev)
                jl->jobs[i].prev = (struct job *) (((uint8_t *) jl->jobs[i].prev) +
                    (((uint8_t *) jl->jobs) - ((uint8_t *) old_start)));
            if (jl->jobs[i].next)
                jl->jobs[i].next = (struct job *) (((uint8_t *) jl->jobs[i].next) +
                    (((uint8_t *) jl->jobs) - ((uint8_t *) old_start)));
        }
        if (jl->first)
            jl->first = (struct job *) (((uint8_t *) jl->first) +
                    (((uint8_t *) jl->jobs) - ((uint8_t *) old_start)));
        if (jl->last)
            jl->last = (struct job *) (((uint8_t *) jl->last) +
                    (((uint8_t *) jl->jobs) - ((uint8_t *) old_start)));
        if (dif > 0)
            memset(jl->jobs + oldlen, 0, dif * sizeof(struct job));
    }
}

int job_add(struct job_list *jl, int ord, uint8_t *data, uint32_t len,
        int sock, int restart)
{
    pthread_mutex_lock(&jl->lock);
    job_list_resize(jl, +1);
    jl->jobs[jl->len - 1] = (struct job){ .jid = jl->next_jid++, .ord = ord,
        .len = len, .data = data, .sock = sock, .attempts = restart };
    int jid = jl->jobs[jl->len - 1].jid;
    //printf("data at %p\n", jl->jobs[jl->len - 1].data);
    pthread_mutex_unlock(&jl->lock);
    return jid;
}

void job_enqueue(struct job_list *jl, int jid)
{
    pthread_mutex_lock(&jl->lock);
    struct job *j = job_lookup(jl->jobs, jl->jobs + jl->len, jid);
    //printf("enqueue job %d at %p\n", jid, j);
    //printf("current first is %p\n", jl->first);
    if (j != NULL) {
        j->prev = jl->last;
        if (j->prev)
```

```
            j->prev->next = j;
        if (jl->first == NULL)
            jl->first = j;
        jl->last = j;
        j->next = NULL;
    }
    //printf("new first is %p\n", jl->first);
    //printf("job's prev is %p\n", j->prev);
    //printf("job's next is %p\n", j->next);
    pthread_mutex_unlock(&jl->lock);
}

static void job_enqueue_front_nolock(struct job_list *jl, int jid)
{
    struct job *j = job_lookup(jl->jobs, jl->jobs + jl->len, jid);
    if (j != NULL) {
        j->next = jl->first;
        if (j->next)
            j->next->prev = j;
        if (jl->last == NULL)
            jl->last = j;
        jl->first = j;
        j->prev = NULL;
    }
}

void job_enqueue_front(struct job_list *jl, int jid)
{
    pthread_mutex_lock(&jl->lock);
    job_enqueue_front_nolock(jl, jid);
    pthread_mutex_unlock(&jl->lock);
}

int job_dequeue(struct job_list *jl, struct pak_header *hdr, uint8_t **data,
        int sock)
{
    pthread_mutex_lock(&jl->lock);
    struct job *j = jl->first;
    //printf("dequeueing job at %p\n", j);
    if (j != NULL && ((j->sock == 0) || (j->sock == sock))) {
        jl->first = j->next;
        if (jl->first)
            jl->first->prev = NULL;
        j->prev = NULL;
        j->next = NULL;
        hdr->ord = j->ord;
        hdr->jid = j->jid;
        *data = j->data;
        hdr->len = j->len;
    } else {
        hdr->ord = 0;
        *data = NULL;
        hdr->len = 0;
        hdr->jid = 0;
    }
    pthread_mutex_unlock(&jl->lock);
    return -1 * (hdr->ord == 0);
}

void job_set_socket(struct job_list *jl, int jid, int sock)
{
    pthread_mutex_lock(&jl->lock);
    struct job *j = job_lookup(jl->jobs, jl->jobs + jl->len, jid);
    if (j != NULL)
        j->sock = sock;
    pthread_mutex_unlock(&jl->lock);
}

void job_socket_failed(struct job_list *jl, int sock)
{
```

```
    pthread_mutex_lock(&jl->lock);
    for (struct job *j = jl->jobs + jl->len - 1; j != jl->jobs - 1; j--)
        if (j->sock == sock) {
            if (j->attempts-- > 0) {
                j->sock = JOB_UNASSIGNED;
                job_enqueue_front_nolock(jl, j->jid);
            } else
                j->sock = JOB_FAILED;
        }
    pthread_mutex_unlock(&jl->lock);
}

int job_status(struct job_list *jl, int jid)
{
    pthread_mutex_lock(&jl->lock);
    struct job *j = job_lookup(jl->jobs, jl->jobs + jl->len, jid);
    int stat = (j == NULL) ? JOB_LOST : j->sock;
    pthread_mutex_unlock(&jl->lock);
    return stat;
}

void job_add_data(struct job_list *jl, int jid, uint8_t *data, uint32_t len)
{
    job_status(jl, jid);
    pthread_mutex_lock(&jl->lock);
    struct job *j = job_lookup(jl->jobs, jl->jobs + jl->len, jid);
    if (j != NULL) {
        free(j->data);
        j->len = len;
        j->data = data;
        j->sock = JOB_FINISHED;
    } else if (data)
        free(data);
    pthread_mutex_unlock(&jl->lock);
}

int job_remove(struct job_list *jl, int jid, uint8_t **data, uint32_t *len)
{
    pthread_mutex_lock(&jl->lock);
    struct job *j = job_lookup(jl->jobs, jl->jobs + jl->len, jid);
    //printf("found job %d at %p\n", jid, j);
    //printf("data at %p\n", j->data);
    int stat = (j == NULL) ? JOB_LOST : j->sock;
    if ((data != NULL) && (len != NULL)) {
        if (j != NULL && j->sock == JOB_FINISHED) {
            *data = j->data;
            *len = j->len;
        } else {
            if (j->data)
                free(j->data);
            *data = NULL;
            *len = 0;
        }
    }
    if (j != NULL) {
        int off = j - jl->jobs;
        if (jl->first == j)
            jl->first = j->next;
        if (jl->last == j)
            jl->last = j->prev;
        if (j->prev)
            j->prev->next = j->next;
        if (j->next)
            j->next->prev = j->prev;
        memmove(j, j + 1, (jl->len - off - 1) * sizeof(struct job));
        job_list_resize(jl, -1);
        for (int i = 0; i < jl->len; i++) {
            if ((jl->jobs[i].next) && (jl->jobs[i].next > jl->jobs + off))
                jl->jobs[i].next--;
            if (jl->jobs[i].prev && (jl->jobs[i].prev > jl->jobs + off))
```

```
                jl->jobs[i].prev--;
        }
        if (jl->first > jl->jobs + off)
            jl->first--;
        if (jl->last > jl->jobs + off)
            jl->last--;

    }
    pthread_mutex_unlock(&jl->lock);
    return stat;
}

void job_list_destroy(struct job_list *jl)
{
    for (struct job *j = jl->jobs; j != jl->jobs + jl->len; j++)
        if (j->data)
            free(j->data);
    free(jl->jobs);
    pthread_mutex_destroy(&jl->lock);
    *jl = (struct job_list) {0};
}
```

```
#ifndef __JOB_LIST_H_
#define __JOB_LIST_H_

#include <pthread.h>
#include <stdint.h>
#include <stdlib.h>
#include <string.h>

#include "util.h"

#define JOB_LIST_INITIALIZER { .lock = PTHREAD_MUTEX_INITIALIZER }
#define JOB_UNASSIGNED 0
#define JOB_FINISHED -1
#define JOB_FAILED -2
#define JOB_LOST -3

struct job {
    int jid;
    int ord;
    uint32_t len;
    uint8_t *data;
    int sock;
    struct job *next;
    struct job *prev;
    int attempts;
};

struct job_list {
    struct job *jobs;
    struct job *first;
    struct job *last;
    int len;
    int next_jid;
    pthread_mutex_t lock;
};

int job_add(struct job_list *jl, int ord, uint8_t *data, uint32_t len,
        int sock, int restart);
void job_enqueue(struct job_list *jl, int jid);
void job_enqueue_front(struct job_list *jl, int jid);
int job_dequeue(struct job_list *jl, struct pak_header *hdr, uint8_t **data,
        int sock);
void job_set_socket(struct job_list *jl, int jid, int sock);
void job_socket_failed(struct job_list *jl, int sock);
void job_add_data(struct job_list *jl, int jid, uint8_t *data,
        uint32_t len);
int job_status(struct job_list *jl, int jid);
int job_remove(struct job_list *jl, int jid, uint8_t **data,
        uint32_t *len);
void job_list_destroy(struct job_list *jl);

#endif /* __JOB_LIST_H_ */
```

```
#include <stdint.h>
#include "job_list.h"

int main(int argc, char**argv)
{
    struct job_list jl = JOB_LIST_INITIALIZER;
    uint8_t *m1 = malloc(16);
    uint32_t j1 = job_add(&jl, 1, m1, 16, 0, 0);
    job_enqueue(&jl, j1);
    uint8_t *m2 = malloc(16);
    uint32_t j2 = job_add(&jl, 1, m2, 16, 0, 0);
    job_enqueue(&jl, j2);
    uint8_t *m3 = malloc(16);
    uint32_t j3 = job_add(&jl, 1, m3, 16, 0, 0);
    job_enqueue(&jl, j3);
    uint8_t *m4 = malloc(16);
    uint32_t j4 = job_add(&jl, 1, m4, 16, 0, 0);
    job_enqueue(&jl, j4);
    uint8_t *m5 = malloc(16);
    uint32_t j5 = job_add(&jl, 1, m5, 16, 0, 0);
    job_enqueue(&jl, j5);
    struct pak_header ph;
    uint8_t *data;
    uint32_t len;
    job_dequeue(&jl, &ph, &data, 0);
    job_dequeue(&jl, &ph, &data, 0);
    job_dequeue(&jl, &ph, &data, 0);
    job_dequeue(&jl, &ph, &data, 0);
    job_dequeue(&jl, &ph, &data, 0);
    job_remove(&jl, j1, &data, &len);
    job_remove(&jl, j2, &data, &len);
    job_remove(&jl, j3, &data, &len);
    job_remove(&jl, j4, &data, &len);
    job_remove(&jl, j5, &data, &len);
    return 0;
}
```

```
CC=gcc
CFLAGS=-g -Wall
LDFLAGS=-g
LDLIBS=-pthread

.PHONY: default
default: libmsmaster.a libmsslave.a

libmsmaster.a: master.o job_list.o slave_queue.o util.o
	ar rc libmsmaster.a master.o job_list.o slave_queue.o util.o
	ranlib libmsmaster.a

libmsslave.a: slave.o util.o
	ar rc libmsslave.a slave.o util.o
	ranlib libmsslave.a

job_list.o: job_list.c job_list.h

master.o: master.c master.h util.h slave_queue.h job_list.h

slave.o: slave.c slave.h util.h

slave_queue.o: slave_queue.c slave_queue.h

util.o: util.c util.h

.PHONY: clean
clean:
	rm -f *.o *.a

.PHONY: all
all: clean default
```

```
#include "master.h"

static volatile int should_die;
static struct slave_queue slave_q = SLAVE_QUEUE_INITIALIZER;
static struct job_list job_l = JOB_LIST_INITIALIZER;
static int s_sock;

int start_job(uint32_t ord, uint8_t *data, uint32_t len)
{
    int jid = job_add(&job_l, ord, data, len, 0, 2);
    job_enqueue(&job_l, jid);
    return jid;
}

void reap_job(int jid, uint8_t **data, uint32_t *len)
{
    while (job_status(&job_l, jid) >= 0)
        usleep(10000);
    job_remove(&job_l, jid, data, len);
}

int get_job_status(int jid) { return job_status(&job_l, jid); }

void cancel_job(int jid)
{
    int sock = job_remove(&job_l, jid, NULL, NULL);
    if (sock > 0) {
        int newjid = job_add(&job_l, -jid, NULL, 0, sock, 0);
        job_enqueue_front(&job_l, newjid);
    }
}

void *slave_feeder(void *arg)
{
    int sock = *((int *)arg);
    struct pak_header hdr;
    uint8_t *data;
    while (*((int *)arg)) {
        if (job_dequeue(&job_l, &hdr, &data, sock) < 0) {
            usleep(10000);
            continue;
        }
        job_set_socket(&job_l, hdr.jid, sock);
        if (write_msg(sock, hdr, data) < 0)
            break;
    }
    return NULL;
}

void *slave_return_collector(void *arg) {
    int sock = *((int *)arg);
    pthread_t feeder;
    pthread_create(&feeder, NULL, slave_feeder, arg);
    struct pak_header header;
    uint8_t *data;
    while (!should_die) {
        if (poll(&(struct pollfd){.fd=sock, .events=POLLIN}, 1, 0) <= 0) {
            if (should_die)
                break;
            else
                usleep(10000);
        } else {
            if (should_die || (read_msg(sock, &header, &data) < 0))
                break;
            job_add_data(&job_l, header.jid, data, header.len);
        }
    }
    *((int *)arg) = 0;
    pthread_join(feeder, NULL);
    job_socket_failed(&job_l, sock);
```

```
    slave_queue_remove(&slave_q, sock);
    free(arg);
    return NULL;
}

void *slave_acceptor(void *arg)
{
    while (!should_die) {
        if (poll(&(struct pollfd){.fd=s_sock, .events=POLLIN}, 1, 0) <= 0)
            usleep(10000);
        else {
            int *p_sock = malloc(sizeof(int));
            *p_sock = accept(s_sock, &(struct sockaddr){0},
                    &(socklen_t) { sizeof(struct sockaddr)});
            slave_queue_add(&slave_q, *p_sock);
            pthread_t th;
            pthread_create(&th, NULL, slave_return_collector,
                p_sock);
            slave_queue_set_thread(&slave_q, *p_sock, th);
        }
    }
    return NULL;
}

int main(int argc, char **argv)
{
    signal(SIGPIPE, SIG_IGN);
    if (start_server(&s_sock, (argc > 1) ? atoi(argv[1]) : 8888) < 0) {
        perror("could not start server");
        exit(1);
    }
    pthread_t accept_th;
    pthread_create(&accept_th, NULL, slave_acceptor, NULL);
    master();
    should_die = 1;
    pthread_join(accept_th, NULL);
    slave_queue_destroy(&slave_q);
    job_list_destroy(&job_l);
    close(s_sock);
    return 0;
}
```

```
#ifndef __MASTER_H_
#define __MASTER_H_

#include <poll.h>
#include <pthread.h>
#include <signal.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>

#include "job_list.h"
#include "slave_queue.h"
#include "util.h"

extern void master(void);

#endif /* __MASTER_H_ */
```

M/s runtime library
Benjamin Hanser (bwh2124)
==========

To build and run the test, run test/test_all.sh.

To compile manually, first run make in this directory, and then in the test directory. This builds the runtime libraries and tests, and performs linking. To test, run test/master [PORT] and test/slave [[HOST] PORT].

The runtime consists of two libraries, libmsmaster.a and libmsslave.a, against which user code will be linked.

The code that links against libmaster.a must have no main and must instead supply void master(). Master may currently call two functions:

```
int start_job(uint32_t ord, uint8_t *data, uint32_t len)
```
Starts a job given by ordinal ord, with serialized data of length len, and returns the jid of the created job. This function does not block, but the job may not be sent to the slave right away.

```
void reap_job(int jid, uint8_t **data, uint32_t *len)
```
Blocks until the job given by jid is terminated. Returns serialized return data in *data, and returns its length in *len. On error, *data is set to NULL and *len is set to 0.

The code that links against libslave.a must supply code for the jobs and must supply the job table (called job_funcs).

Jobs have the following signature:
```
void j(uint8_t *in_d, uint32_t in_l, uint8_t **out_d, uint32_t *out_l)
```
Takes in serialized arguments pointed by in_d, of length in_l, and returns its result in *out_d, of length *out_l.

The slave must declare the job table (job_funcs) as follows:
```
void (*job_funcs [])(uint8_t *, uint32_t, uint8_t **, uint32_t *)
    = {job_1, job_2, job_3, job_4};
```
Each entry in the initializer list is the name of a job, ie a function pointer to that job, and the order in which they are given determines the ordinal of the job (starting at 1).

The slave must finally provide the number of jobs as follows:
```
int num_jobs =  (sizeof(job_funcs) / sizeof(*job_funcs));
```

Note that the user is ENTIRELY responsible for all serialization code. This is because serialization of data is intimately connected to actually gathering it from its various locations in memory, and so it makes sense for the user compiler to generate this code.

```c
#include "slave.h"

static pthread_mutex_t write_mutex = PTHREAD_MUTEX_INITIALIZER;
static int sock;

void *run_job(void *arg)
{
    pthread_detach(pthread_self());
    struct pak_header *hdr = (struct pak_header *) arg;
    uint8_t *out_data, *in_data = ((uint8_t *)arg) + sizeof *hdr;
    if (hdr->ord >= 1 && hdr->ord <= num_jobs) {
        job_funcs[hdr->ord - 1](in_data, hdr->len, &out_data, &hdr->len);
        pthread_mutex_lock(&write_mutex);
        write_msg(sock, *hdr, out_data);
        pthread_mutex_unlock(&write_mutex);
        if (out_data)
            free(out_data);
    }
    if (arg)
        free(arg);
    return NULL;
}

void slave()
{
    uint8_t *data;
    while (read_msg_combined(sock, &data) == 0)
        pthread_create(&(pthread_t){0}, NULL, &run_job, data);
}

int main(int argc, char **argv)
{   /* usage: ./slave [[HOST] PORT] */
    client_connect(&sock, (argc > 2) ? argv[1]              : "127.0.0.1",
                          (argc > 1) ? argv[1 + (argc > 2)] : "8888");
    slave();
    close(sock);
    return 0;
}
```

```
#ifndef __SLAVE_H_
#define __SLAVE_H_

#include <pthread.h>
#include <stdlib.h>
#include <unistd.h>

#include "util.h"

extern void (*job_funcs [])(uint8_t *, uint32_t, uint8_t **, uint32_t *);
extern int num_jobs;

/* dummy functions to keep linker happy */

int start_job(uint32_t ord, uint8_t *data, uint32_t len) { return 0; }

void reap_job(int jid, uint8_t **data, uint32_t *len) { }

int get_job_status(int jid) { return 0; }

void cancel_job(int jid) { }


#endif /* __SLAVE_H */
```

```
#include "slave_queue.h"
#include <stdio.h>

struct slave *slave_lookup(struct slave *beg, struct slave *end, int sock)
{
    struct slave *mid = beg + (end-beg)/2;
    if (mid == end)
        return mid;
    if (mid->sock > sock && mid > beg)
        return slave_lookup(beg, mid, sock);
    if (mid->sock < sock && mid + 1 == end)
        return mid + 1;
    if (mid->sock < sock)
        return slave_lookup(mid + 1, end, sock);
    return mid;
}

void slave_queue_add(struct slave_queue *sq, int sock)
{
    if (sock < 0)
        return;
    pthread_mutex_lock(&sq->lock);
    struct slave *loc = slave_lookup(sq->slaves, sq->slaves+sq->len, sock);
    if ((loc != sq->slaves + sq->len) && (loc->sock == sock)) {
        pthread_mutex_unlock(&sq->lock);
        return;
    }
    if ((sq->len == 0) || !(sq->len & (sq->len - 1))) { /* power of 2 */
        int offset = loc - sq->slaves;
        sq->slaves = realloc(sq->slaves, ((sq->len) ? sq->len*2 : 1)
                * sizeof(struct slave));
        loc = sq->slaves + offset;
        memmove(loc + 1, loc, (sq->slaves + sq->len - loc) *
                sizeof(struct slave));
    }
    loc->sock = sock;
    sq->len++;
    pthread_mutex_unlock(&sq->lock);
}
void slave_queue_set_thread(struct slave_queue *sq, int sock, pthread_t th)
{
    pthread_mutex_lock(&sq->lock);
    sq->threads = realloc(sq->threads, ++sq->threadlen * sizeof(pthread_t));
    sq->threads[sq->threadlen-1] = th;
    pthread_mutex_unlock(&sq->lock);
}

void slave_queue_remove(struct slave_queue *sq, int sock)
{
    while ( pthread_mutex_trylock(&sq->lock) != 0) {
        if (sq->being_destroyed)
            return;
        else
            usleep(10000);
    }
    struct slave *loc = slave_lookup(sq->slaves, sq->slaves+sq->len, sock);
    if ((loc != sq->slaves + sq->len) && (loc->sock == sock)) {
        close(loc->sock);
        memmove(loc, loc + 1, (--sq->len - (loc - sq->slaves)) *
                sizeof (struct slave));
        if (!(sq->len & (sq->len - 1)))
            sq->slaves=realloc(sq->slaves, sq->len * sizeof(struct slave));
    }
    pthread_mutex_unlock(&sq->lock);
}

void slave_queue_destroy(struct slave_queue *sq)
{
    pthread_mutex_lock(&sq->lock);
    sq->being_destroyed = 1;
```

```
    for (struct slave *s = sq->slaves; s != sq->slaves + sq->len; s++)
        close(s->sock);
    for (int i = 0; i < sq->threadlen; i++)
        pthread_join(sq->threads[i], NULL);
    if (sq->slaves)
        free(sq->slaves);
    if (sq->threads)
        free(sq->threads);
    *sq = (struct slave_queue) {0};
}
```

```
#ifndef __SLAVE_QUEUE_H_
#define __SLAVE_QUEUE_H_

#include <pthread.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

#define SLAVE_QUEUE_INITIALIZER { .lock = PTHREAD_MUTEX_INITIALIZER }

struct slave {
    int sock;
};

struct slave_queue {
    struct slave *slaves;
    pthread_t *threads;
    int len;
    int threadlen;
    int being_destroyed;
    pthread_mutex_t lock;
};

void slave_queue_add(struct slave_queue *sq, int sock);
void slave_queue_set_thread(struct slave_queue *sq, int sock, pthread_t th);
void slave_queue_remove(struct slave_queue *sq, int sock);
/*int slave_queue_assign_next(struct slave_queue *sq);
void slave_lock(struct slave_queue *sq, int sock);
void slave_unlock(struct slave_queue *sq, int sock);*/
void slave_queue_destroy(struct slave_queue *sq);

#endif /* __SLAVE_QUEUE_H_ */
```

```
#include "util.h"

void die(const char * s)
{
    (errno) ? perror(s) : fprintf(stderr, "%s.\n", s);
    exit(1);
}

int op_blk(int s, void *b, size_t l, ssize_t(*f)(int, void *, size_t, int))
{
    while (l) {
        ssize_t bytes = f(s, b, l, 0);
        if (bytes <= 0)
            return -1;
        l -= bytes;
        b += bytes;
    }
    return 0;
}

int read_blk(int s, void *b, size_t l) { return op_blk(s, b, l, recv); }

int write_blk(int s, void *b, size_t l)
{
    return op_blk(s, b, l, (ssize_t (*) (int, void *, size_t, int)) send);
}

int read_msg(int sock, struct pak_header *hdr, uint8_t **data)
{
    return (((       read_blk(sock, hdr, sizeof *hdr)) < 0) ||
           ((*data = malloc(hdr->len)) == NULL) ||
           (((       read_blk(sock, *data, hdr->len)) < 0 ) &&
           (             free(*data), 1))) * -1;
}

int read_msg_combined(int sock, uint8_t **data)
{
    struct pak_header hdr;
    return (((       read_blk(sock, &hdr, sizeof hdr)) < 0) ||
           ((*data = malloc(hdr.len + sizeof hdr)) == NULL) ||
           (((       read_blk(sock, *data + sizeof hdr, hdr.len)) < 0 ) &&
           (             free(*data), 1)) ||
           (         *((struct pak_header *) *data) = hdr, 0)) * -1;
}

int write_msg(int sock, struct pak_header hdr, uint8_t *data)
{
    return ((write_blk(sock, &hdr, sizeof hdr) < 0) ||
           ( write_blk(sock, data, hdr.len) < 0)) * -1;
}

int client_connect(int *sock, const char *host, const char *port)
{
    struct addrinfo *r;
    return ((     getaddrinfo(host, port, &(struct addrinfo) {
                        .ai_family=AF_UNSPEC,
                        .ai_socktype=SOCK_STREAM}, &r) != 0) ||
        ((*sock = socket(r->ai_family, r->ai_socktype, r->ai_protocol)) < 0)
        || ((     connect(*sock, r->ai_addr, r->ai_addrlen) < 0) &&
        (             close(*sock), freeaddrinfo(r), 1)) ||
        (         freeaddrinfo(r), 0));
}


int start_server(int *s_sock, short port)
{
    return (((*s_sock = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP)) < 0) ||
           (            setsockopt(*s_sock, SOL_SOCKET, SO_REUSEADDR,
                                &(int){1}, sizeof(int)) < 0) ||
           (            bind(*s_sock, (struct sockaddr *)
```

```
                                &(struct sockaddr_in) {
                                    .sin_family = AF_INET,
                                    .sin_addr.s_addr = htonl(INADDR_ANY),
                                    .sin_port = htons(port) },
                                sizeof(struct sockaddr_in)) < 0) ||
            (               listen(*s_sock, 0) < 0)) * -1;
}
```

```
#ifndef __UTIL_H_
#define __UTIL_H_

#include <arpa/inet.h>
#include <errno.h>
#include <netdb.h>
#include <stdio.h>
#include <stdint.h>
#include <stdlib.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <unistd.h>

struct pak_header {
    uint32_t ord;
    int jid;
    uint32_t len;
};

void die(const char * s);
int read_blk(int sock, void *b, size_t size);
int write_blk(int sock, void *b, size_t size);
int read_msg(int sock, struct pak_header *hdr, uint8_t **data);
int read_msg_combined(int sock, uint8_t **data);
int write_msg(int sock, struct pak_header hdr, uint8_t *data);
int client_connect(int *sock, const char *host, const char *port);
int start_server(int *s_sock, short port);
#endif /* __UTIL_H */
```

```
Fatal error: exception Failure("illegal assignment int = bool in i = false")
```

```
master
{
  int i;
  bool b;

  i = 42;
  i = 10;
  b = true;
  b = false;
  i = false; /* Fail: assigning a bool to an integer */
}
```

```
Fatal error: exception Failure("illegal assignment bool = int in b = 48")
```

```
master
{
  int i;
  bool b;

  b = 48; /* Fail: assigning an integer to a bool */
}
```

```
Fatal error: exception Failure("illegal assignment int = void in i = myvoid()")
```

```
master
{
  int i;

  i = myvoid(); /* Fail: assigning a void to an integer */
}

job void myvoid()
{
  return;
}
```

```
Fatal error: exception Failure("illegal assignment double = void in d = myvoid()")
```

```
master
{
  double d;

  d = myvoid(); /* Fail: assigning a void to a double */
}

job void myvoid()
{
  return;
}
```

```
Fatal error: exception Failure("illegal assignment string = double in s = d")
```

```
master
{
  string s;
  double d = 3.4;
  s = d;
}
```

```
Fatal error: exception Failure("illegal assignment string = int in s = 48")
```

```
master
{
  string s;

  s = 48; /* Fail: assigning an integer to a string*/
}
```

```
Fatal error: exception Failure("nothing may follow a return")
```

```
master
{
  int i;

  i = 15;
  return i;
  i = 32; /* Error: code after a return */
}
```

```
Fatal error: exception Failure("nothing may follow a return")
```

```
master
{
  int i;

  {
    i = 15;
    return i;
  }
  i = 32; /* Error: code after a return */
}
```

```
Fatal error: exception Failure("illegal binary operator bool + int in d + a")
```

```
master
{
  return 0;
}

job void foo(int c, bool d)
{
  int a;
  a + c;
  c - a;
  a * 3;
  c / 2;
  d + a; /* Error: bool + int */
}
```

```
Fatal error: exception Failure("illegal binary operator bool + int in b + a")
```

```
master
{
  return 0;
}

job void foo(int c, bool f)
{
  int a;
  bool b;
  int d;
  bool e;
  b + a; /* Error: bool + int */
}
```

```
Fatal error: exception Failure("undeclared identifier j")
```

```
master
{
  int i;
  for ( ; true ; ) {} /* OK: Forever */

  for (i = 0 ; i < 10 ; i = i + 1) {
    if (i == 3) return 42;
  }

  for (j = 0; i < 10 ; i = i + 1) {} /* j undefined */

  return 0;
}
```

```
Fatal error: exception Failure("undeclared identifier j")
```

```
master
{
  int i;

  for (i = 0; j < 10 ; i = i + 1) {} /* j undefined */

  return 0;
}
```

```
Fatal error: exception Failure("expected Boolean expression in i")
```

```
master
{
  int i;

  for (i = 0; i ; i = i + 1) {} /* i is an integer, not Boolean */

  return 0;
}
```

```
Fatal error: exception Failure("undeclared identifier j")
```

```
master
{
  int i;

  for (i = 0; i < 10 ; i = j + 1) {} /* j undefined */

  return 0;
}
```

```
Fatal error: exception Failure("unrecognized function foo")
```

```
master
{
  int i;

  for (i = 0; i < 10 ; i = i + 1) {
    foo(); /* Error: no function foo */
  }

  return 0;
}
```

```
Fatal error: exception Failure("function printd may not be defined")
```

```
master
{
}

job void baz() {}

job int printd() {} /* Should not be able to define printd */
```

```
Fatal error: exception Failure("function prints may not be defined")
```

```
master
{
}

job void baz() {}

job int prints() {} /* Should not be able to define prints */
```

```
Fatal error: exception Failure("duplicate function bar")
```

```
master
{
  return 0;
}

job int foo() {}

job int bar() {}

job int baz() {}

job void bar() {} /* Error: duplicate function bar */
```

```
Fatal error: exception Failure("duplicate formal a in bar")
```

```
master
{
  return 0;
}

job int foo(int a, bool b, int c) { }

job void bar(int a, bool b, int a) {} /* Error: duplicate formal a in bar */
```

```
Fatal error: exception Failure("illegal void formal b in bar")
```

```
master
{
  return 0;
}

job int foo(int a, bool b, int c) { }

job void bar(int a, void b, int c) {} /* Error: illegal void formal b */
```

```
Fatal error: exception Failure("function print may not be defined")
```

```
master
{
  return 0;
}

job int foo() {}

job void bar() {}

job int print() {} /* Should not be able to define print */

job void baz() {}
```

```
Fatal error: exception Failure("illegal void local b in bar")
```

```
master
{
  return 0;
}

job int foo() {}

job int bar() {
  int a;
  void b; /* Error: illegal void local b */
  bool c;

  return 0;
}
```

```
Fatal error: exception Failure("expecting 2 arguments in foo(42)")
```

```
master
{
  foo(42, true);
  foo(42); /* Wrong number of arguments */
}

job void foo(int a, bool b)
{
}
```

```
Fatal error: exception Failure("expecting 2 arguments in foo(42, true, false)")
```

```
master
{
  foo(42, true);
  foo(42, true, false); /* Wrong number of arguments */
}

job void foo(int a, bool b)
{
}
```

```
Fatal error: exception Failure("illegal actual argument found void expected bool in bar()
")
```

```
master
{
  foo(42, true);
  foo(42, bar()); /* int and void, not int and bool */
}

job void foo(int a, bool b)
{
}

job void bar()
{
}
```

```
Fatal error: exception Failure("illegal actual argument found int expected bool in 42")
```

```
master
{
  foo(42, true);
  foo(42, 42); /* Fail: int, not bool */
}

job void foo(int a, bool b)
{
}
```

```
Fatal error: exception Failure("expected Boolean expression in 42")
```

```
master
{
  if (true) {}
  if (false) {} else {}
  if (42) {} /* Error: non-bool predicate */
}
```

```
Fatal error: exception Failure("undeclared identifier foo")
```

```
master
{
  if (true) {
    foo; /* Error: undeclared variable */
  }
}
```

```
Fatal error: exception Failure("undeclared identifier bar")
```

```
master
{
  if (true) {
    42;
  } else {
    bar; /* Error: undeclared variable */
  }
}
```

```
Fatal error: exception Failure("Cancelling something not a job")
```

```
master {
  int a;
  a = 10;
  cancel a;
}
```

```
Fatal error: exception Failure("Getting something not a job")
```

```
master {
  int a;
  a = 10;
  job<int> j1 = remote f(a);
  bool b;
  b = get a;
}

job int f(int a) {
  int i = 0;
  return i;
}
```

```
Fatal error: exception Failure("illegal assignment bool = int in b = getj1")
```

```
master {
  int a;
  a = 10;
  job<int> j1 = remote f(a);
  bool b;
  b = get j1;
}

job int f(int a) {
  int i = 0;
  return i;
}
```

```
Fatal error: exception Failure("Checking running of something not a job")
```

```
master {
  int a;
  a = 10;
  bool b;
  b = a.running;
}
```

```
Fatal error: exception Failure("illegal assignment int = bool in a = j1.running")
```

```
master {
  int a;
  a = 10;
  job<int> j1;

  j1 = remote f(a);
  a = j1.running;
}

job int f(int a) {
  int i = 0;
  return i;
}
```

```
Fatal error: exception Parsing.Parse_error
```

```
Fatal error: exception Failure("unrecognized function a")
```

```
master {
  int a;
  a = 10;
  job<int> j1 = remote a();
}

job int f(int a) {
  int i = 0;
  return i;
}
```

```
Fatal error: exception Failure("expecting 1 arguments in remote f()")
```

```
master {
  int a;
  a = 10;
  job<int> j1 = remote f();
}

job int f(int a) {
  int i = 0;
  return i;
}
```

```
Fatal error: exception Failure("return gives bool expected int in true")
```

```
master
{
  return true; /* Should return int */
}
```

```
Fatal error: exception Failure("return gives int expected void in 42")
```

```
master
{
  return 42;
}

job void foo()
{
  if (true) return 42; /* Should return void */
  else return;
}
```

```
Fatal error: exception Failure("illegal string concat")
```

```
master
{
  string s = "hello";
  string t;
  double d = 3.3;

  s << d;
}
```

```
Fatal error: exception Failure("cannot find fieldname: c")
```

```
master
{
    struct example s1;
    s1 -> c = 2;
}

struct example {
    int a;
    int b;
    struct example_inner inner1;
};
```

```
Fatal error: exception Failure("illegal assignment struct example = struct example_inner
in s1 = s2")
```

```
master
{
    struct example s1;
    struct example_inner s2;
    s1 = s2;
}

struct example_inner {
    int c;
    struct example_inner_inner very_inner;
};

struct example {
    int a;
    int b;
    struct example_inner inner1;
};
```

```
Fatal error: exception Failure("illegal struct field assignment")
```

```
master
{
    struct example s1;
    struct example_inner s2;
    s1 -> a = s2->very_inner->nasty;
}

struct example_inner_inner {
    int d;
    vector<int> nasty;
};

struct example_inner {
    int c;
    struct example_inner_inner very_inner;
};

struct example {
    int a;
    int b;
    struct example_inner inner1;
};
```

```
Fatal error: exception Failure("illegal struct field assignment")
```

```
master
{
    struct example s1;
    struct example_inner s2;
    s1 -> a = s2;
}

struct example_inner {
    int c;
    struct example_inner_inner very_inner;
};

struct example {
    int a;
    int b;
    struct example_inner inner1;
};
```

```
Fatal error: exception Failure("accessing field of something not a struct")
```

```
master
{
    int a;
    a->b = c;
}
```

```
Fatal error: exception Failure("undeclared struct non_existent")
```

```
master
{
    struct example s1;
    struct non_existent s2;

}

struct example {
    int a;
    int b;
};
```

```
Fatal error: exception Failure("illegal assignment vector<string> = vector<int> in a = b"
)
```

```
master {
    vector<string> a;
    vector<int> b;
    b::1;
    b::2;
    a = b;
}
```

```
Fatal error: exception Failure("pushing back to something not a vector")
```

```
master {
    vector<vector<int>> a;
    int b;
        b::2;
}
```

```
Fatal error: exception Failure("Vector access of something not a vector")
```

```
master {
    vector<int> b;
        b::2;
        int a;
        b::a[3];
}
```

```
Fatal error: exception Failure("Illegal vector pushback found string expected int")
```

```
master {
    vector<int> a;
    a::"hello";
}
```

```
Fatal error: exception Failure("expected Boolean expression in 42")
```

```
master
{
  int i;

  while (true) {
    i = i + 1;
  }

  while (42) { /* Should be boolean */
    i = i + 1;
  }

}
```

```
Fatal error: exception Failure("unrecognized function foo")
```

```
master
{
  int i;

  while (true) {
    i = i + 1;
  }

  while (true) {
    foo(); /* foo undefined */
  }

}
```

```
master
{
  print( add(17, 25) );
  return 0;
}

job int add(int x, int y)
{
  return x + y;
}
```

42

```
master
{
  print(39 + 3);
  return 0;
}
```

```
42
```

```
master
{
  print(1 + 2 * 3 + 4);
  return 0;
}
```

```
11
```

```
master
{
  int a;
  a = 42;
  a = a + 5;
  print(a);
  return 0;
}

job int foo(int a)
{
  return a;
}
```

47

```
master {
  int a;
  a = 10;
  job<int> j1;
  bool run;
  bool fin;
  bool fail;

  j1 = remote f(a);
  cancel j1;

  run = j1.running;
  fin = j1.finished;
  fail = j1.failed;

  printb(run);
  printb(fin);
  printb(fail);

  return 1;
}

job int f(int a) {
  int i;
  i = 0;
  while (i < 100) {
    i = i + 1;
  }
  return i;
}
```

```
false
false
false
```

```
master {
    int a = 5;
    print(a);
}
```

5

```
master
{
  double d;
  d = 3.4;
  printd(d);
}
```

```
3.400000
```

```
master
{
  double d;
  d = 3.4;
  double e;
  e = d + 1.0;
  printd(e);
}
```

```
4.400000
```

```
master
{
  print(fib(0));
  print(fib(1));
  print(fib(2));
  print(fib(3));
  print(fib(4));
  print(fib(5));
  return 0;
}

job int fib(int x)
{
  if (x < 2) return 1;
  return fib(x-1) + fib(x-2);
}
```

```
1
1
2
3
5
8
```

```
master
{
  int i;
  for (i = 0 ; i < 5 ; i = i + 1) {
    print(i);
  }
  print(42);
  return 0;
}
```

```
0
1
2
3
4
42
```

```
master
{
  int i;
  i = 0;
  for ( ; i < 5; ) {
    print(i);
    i = i + 1;
  }
  print(42);
  return 0;
}
```

```
0
1
2
3
4
42
```

```
master
{
  int a;
  a = add(39, 3);
  print(a);
  return 0;
}

job int add(int a, int b)
{
  return a + b;
}
```

```
42
```

```
master
{
  int i;
  i = 1;

  fun(i = 2, i = i+1);

  print(i);
  return 0;
}

job int fun(int x, int y)
{
  return 0;
}
```

2

```
master
{
  printem(42,17,192,8);
  return 0;
}

job void printem(int a, int b, int c, int d)
{
  print(a);
  print(b);
  print(c);
  print(d);
}
```

```
42
17
192
8
```

```
master
{
  int d;
  d = add(52, 10);
  print(d);
  return 0;
}

job int add(int a, int b)
{
  int c;
  c = a + b;
  return c;
}
```

62

```
master
{
  return 0;
}

job int foo(int a)
{
  return a;
}
```

```
master
{
  print(bar(17, false, 25));
  return 0;
}

job void foo() {}

job int bar(int a, bool b, int c) { return a + c; }
```

```
42
```

```
master
{
  foo(40);
  return 0;
}

job void foo(int a)
{
  print(a + 3);
}
```

43

```
master
{
  print(gcd(14,21));
  print(gcd(8,36));
  print(gcd(99,121));
  return 0;
}

job int gcd(int a, int b) {
  while (a != b)
    if (a > b) a = a - b;
    else b = b - a;
  return a;
}
```

```
7
4
11
```

```
master
{
  print(gcd(2,14));
  print(gcd(3,15));
  print(gcd(99,121));
  return 0;
}

job int gcd(int a, int b) {
  while (a != b) {
    if (a > b) a = a - b;
    else b = b - a;
  }
  return a;
}
```

```
2
3
11
```

```
master
{
  print(42);
  print(71);
  print(1);
  return 0;
}
```

```
42
71
1
```

```
master
{
  if (true) print(42);
  print(17);
  return 0;
}
```

```
42
17
```

```
master
{
  if (true) print(42); else print(8);
  print(17);
  return 0;
}
```

```
42
17
```

```
master
{
  if (false) print(42);
  print(17);
  return 0;
}
```

17

```
master
{
  if (false) print(42); else print(8);
  print(17);
  return 0;
}
```

```
8
17
```

```
master
{
 print(cond(true));
 print(cond(false));
 return 0;
}

job int cond(bool b)
{
  int x;
  if (b)
    x = 42;
  else
    x = 17;
  return x;
}
```

```
42
17
```

```
master
{
  foo(true);
  return 0;
}

job void foo(bool j)
{
  int i;

  i = 42;
  print(i + i);
}
```

84

```
master {
 print(foo(37, false));
 return 0;
}

job int foo(int a, bool b)
{
  int c;
  bool d;

  c = a;

  return c + 10;
}
```

47

```
master
{
  print(1 + 2);
  print(1 - 2);
  print(1 * 2);
  print(100 / 2);
  print(99);
  printb(1 == 2);
  printb(1 == 1);
  print(99);
  printb(1 != 2);
  printb(1 != 1);
  print(99);
  printb(1 < 2);
  printb(2 < 1);
  print(99);
  printb(1 <= 2);
  printb(1 <= 1);
  printb(2 <= 1);
  print(99);
  printb(1 > 2);
  printb(2 > 1);
  print(99);
  printb(1 >= 2);
  printb(1 >= 1);
  printb(2 >= 1);
  return 0;
}
```

```
3
-1
2
50
99
false
true
99
true
false
99
true
false
99
true
true
false
99
false
true
99
false
true
true
```

```
master
{
  printb(true);
  printb(false);
  printb(true && true);
  printb(true && false);
  printb(false && true);
  printb(false && false);
  printb(true || true);
  printb(true || false);
  printb(false || true);
  printb(false || false);
  printb(!false);
  printb(!true);
  print(-10);
  print(--42);
}
```

```
true
false
true
false
false
false
true
true
true
false
true
false
-10
42
```

```
master {
  int b;
  b = 11;
  int a;
  a = 10;
  job<int> j1;
  j1 = remote five_prod(a, b, a+b, a*b, b-a);
  job<int> j2;
  j2 = remote gcd(168, 216);
  job<int> j3;
  j3 = remote sum_or_diff(true, a+b, a-b);
  job<int> j4;
  j4 = remote sum_or_diff(false, a+b, a-b);
  job<int> j5;
  j5 = remote sum_with_bools(true, false, 5, true, b);
  double d1;
  d1 = 5.0;
  double d2;
  d2 = 5.5;
  job<int> j6;
  job<int> j7;
  j6 = remote cmpBoolDoubles(true, d1, d2);
  j7 = remote cmpBoolDoubles(false, d1, d2);
  job<double> j8;
  job<bool> j9;
  j8 = remote dbldbl(2.33);
  j9 = remote islessthan(2.33, 3.44);
  print(get j1);
  print(get j2);
  print(get j3);
  print(get j4);
  print(get j5);
  print(get j6);
  print(get j7);
  printd(get j8);
  if (get j9)
      print(1);
  else
      print(0);
  return 1;
}

job bool islessthan(double a, double b) { return a < b; }

job int five_prod(int a, int b, int c, int d, int e) { return a*b*c*d*e; }

job int gcd (int a, int b)
{
    if (a < b) return gcd(b-a, a);
    if (b < a) return gcd(b, a-b);
    return a;
}

job double dbldbl (double d) { return 2.0 * d; }

job int cmpBoolDoubles(bool lt, double a, double b)
{
    if (lt) {
        if (a < b)
            return 1;
        else
            return 0;
    } else {
        if (a > b)
            return 1;
        else
            return 0;
    }
}
```

```
job int sum_or_diff (bool sum, int a, int b)
{
    if (sum) return a + b;
    else return a - b;
}

job int sum_with_bools (bool a, bool b, int c, bool d, int e)
{
    int sum;
    sum = 0;
    if (a) sum = sum + 1;
    if (b) sum = sum + 1;
    if (d) sum = sum + 1;
    sum = sum + c;
    sum = sum + e;
    return sum;
}
```

```
254100
24
20
22
18
1
0
4.660000
1
```

```
master {
  int b;
  b = 11;
  int a;
  a = 10;
  print(f(a));
  job<int> j1;
  j1 = remote f(a+b);
  int result;
  result = get j1;
  print(result);
  return 1;
}

job int f(int a) {
  return 2*a;
}
```

```
20
42
```

```
master {
  int a;
  int b;
  a = 10;
  job<int> j1 = remote f(a);
  a = get j1;
  print(a);
  if(j1.finished) {
        // j1 should be "lost" now, so this should not run
        b = get j1;
  }
}

job int f(int a) {
  int i = 0;
  return i;
}
```

0

```
master {
  int a;
  a = 10;
  job<int> j1;

  bool fin;
  bool run;
  bool fail;

  j1 = remote f(a);
  run = j1.running;
  fin = j1.finished;
  fail = j1.failed;

  printb(run);
  printb(fin);
  printb(fail);

/* For testing fail and finished
  job<int> j2;
  j2 = remote g(a);
  int i;
  i = 0;

  while(i < 10000000) {
    i = i + 1;
    print(i);
  }

  run = j2.running;
  fin = j2.finished;
  fail = j2.failed;
  printb(run);
  printb(fin);
  printb(fail);
*/

  int result;
  result = get j1;
  print(result);

  return 1;
}

job int f(int a) {
  return a*2;
}
job int g(int a) {
  return a / 0;
}
```

```
true
false
false
20
```

```
master {
  int b;
  b = 11;
  int a;
  a = 10;
  job<int> j1;
  j1 = remote five_prod(a, b, a+b, a*b, b-a);
  job<int> j2;
  j2 = remote gcd(168, 216);
  job<int> j3;
  j3 = remote sum_or_diff(true, a+b, a-b);
  job<int> j4;
  j4 = remote sum_or_diff(false, a+b, a-b);
  job<int> j5;
  j5 = remote sum_with_bools(true, false, 5, true, b);
  print(get j1);
  print(get j2);
  print(get j3);
  print(get j4);
  print(get j5);
  return 1;
}

job int five_prod(int a, int b, int c, int d, int e) { return a*b*c*d*e; }

job int gcd (int a, int b)
{
    if (a < b) return gcd(b-a, a);
    if (b < a) return gcd(b, a-b);
    return a;
}

job int sum_or_diff (bool sum, int a, int b)
{
    if (sum) return a + b;
    else return a - b;
}

job int sum_with_bools (bool a, bool b, int c, bool d, int e)
{
    int sum;
    sum = 0;
    if (a) sum = sum + 1;
    if (b) sum = sum + 1;
    if (d) sum = sum + 1;
    sum = sum + c;
    sum = sum + e;
    return sum;
}
```

```
254100
24
20
22
18
```

```
job int j_gcd(int a, int b)
{
    if (a > b)
        return j_gcd(a-b, b);
    if (a < b)
        return j_gcd(a, b-a);
    return a;
}
job int foo(int a) {
  return a;
}

job int j_prime(int a)
{
    if (a < 2)
        return 0;
    int s = a-1;
    int i = 2;
    for (i; i <= s; i=i+1) {
        if (a - a/i*i == 0)
            return 0;
    }
    return 1;
}

job vector<int> j_prime_mul(vector<int> b) {
    int i = 0;
    vector<int> results;
    for (i; i < size b; i=i+1) {
        int result = j_prime(b[i]);
        if (result == 1) {
            results::result;
        }

    }
    return results;
}


master {
  vector<job<int>> vector_test1;

    int i = 100;
    int j;
    for (j = 1; j < 100; j=j+1) {
        //print(j_gcd(j, j+2));
    }

    for (j = 0; j < 100; j=j+1) {
        job<int> test_tmp1 = remote j_gcd(j+1, j+2);
        vector_test1::test_tmp1;
    }
    for (j = 0; j < 100; j=j+1) {
        job<int> test_tmp2 = vector_test1[j];
        get test_tmp2;
    }

    vector<vector<int>> vector_input;
    vector<int> vector_input_part;
    vector<int> empty;
    int k;
    for (j = 0; j < 100; j=j+1) {
        vector_input_part = empty;
        for (k = 0; k < 1000; k=k+1) {
            vector_input_part::(j * 1000 + k);
        }
        vector_input::vector_input_part;
    }

    vector<job<vector<int>>> vector_test2;
```

```
    for (j = 0; j < 100; j=j+1) {

        vector_input_part = vector_input[j];
        job<vector<int>> test_tmp3 = remote j_prime_mul(vector_input_part);
        vector_test2::test_tmp3;
    }

    vector<int> result;
    int result_size = 0;
    string s = "job failed";
    for (j = 0; j < 100; j=j+1) {
        job<vector<int>> test_tmp4 = vector_test2[j];
        while (!test_tmp4.finished && !test_tmp4.failed) { }
        if (!test_tmp4.failed) {
            result = get test_tmp4;
            result_size  = (size result) + result_size;
            if ((j == 0) || (j == 9))
                print(result_size);
        } else
            prints(s);
    }
    print(result_size);
}
```

```
168
1229
9592
```

```
struct foo {
    int a;
    vector<int> b;
    bool c;
};

job struct foo foo77(struct foo arg)
{
    vector<int> b;
    b = arg->b;
    b::77;
    arg->b = b;
    vector<int> c;
    c = arg->b;
    return arg;
}

master
{
    struct foo bar;
    vector<int> b;
    b::66;
    bar->a = 17;
    bar->b = b;
    bar->c = true;
    vector<int> e =  bar->b;
    print(size e);

    job<struct foo> j1 = remote foo77(bar);

    bar = get j1;
    printb(bar->c);
    print(bar->a);
    int i;
    b = bar->b;
    for (i = 0; i < size b; i = i + 1)
        print(b[i]);

    struct foo bar2 = foo77(bar);
    b = bar2->b;
    for (i = 0; i < size b; i = i + 1)
        print(b[i]);


}
```

```
1
true
17
66
77
66
77
77
```

```
job int foo (vector<int> vec)
{
    return vec[4];
}

job int bar (vector<double> vec)
{
    return size vec;
}

job vector<int> baz (vector<vector<int>> vec)
{
    return vec[1];
}

job int vecsum (vector<int> vec)
{
    int i;
    int sum = 0;
    for (i = 0; i < size vec; i = i + 1)
        sum = sum + vec[i];
    return sum;
}

master
{
    vector<int> a;
    a::4;
    a::6;
    a::8;
    a::10;
    a::12;
    vector<double> d;
    d::2.2;
    d::3.3;
    d::4.4;
    vector<vector<int>> aa;
    aa::a;
    vector<int> b;
    b::3;
    b::5;
    aa::b;
    job<int> j = remote foo(a);
    job<int> k = remote bar(d);
    job<vector<int>> l = remote baz(aa);
    print(get j);
    print(get k);
    vector<int> zz = get l;
    print(zz[1]);
    job<int> sj = remote vecsum(a);
    print(get sj);
}
```

```
12
3
5
40
```

```
master{
        string s;
        s = "hello";
}
```

```
master{
        string s;
        string t;
        string u;
        s = "hello";
        t = " goodbye";

        /* need to implement print vector
        prints(s);
        prints(t); */
}
```

```
master {
    vector<int> x;
    x::1;
    "blah";
    string a = "blah";
    prints(a);

    prints(a << "ldasjlkj djaskljd");

    string b = a << "ldasjlkj djaskljd";

    string c = b << a;
    prints(b);
    prints(c);

    return 0;

}
```

```
blah
blahldasjlkj djaskljd
blahldasjlkj djaskljd
blahldasjlkj djaskljdblah
```

```
master
{

    struct example a;
    struct example b;

    b->nasty->x = 1;
    b->nasty->inner->x = 2;


    vector<int> v;
    v = b->nasty->inner->very_nasty;
    v::1;
    b->nasty->inner->very_nasty = v;
    a->nasty = b->nasty;


    b->nasty->inner->x = 1;
    print(a->nasty->inner->x); //2
    print(a->nasty->x); //1

    v = b->nasty->inner->very_nasty;
    v[0]=2;
    b->nasty->inner->very_nasty = v;
    vector<int> e;
    e = a->nasty->inner->very_nasty;
    print(e[0]); //1
    print(v[0]); //2

    a = b;
    vector<int> xyz;
    xyz = a->nasty->inner->very_nasty;
    print(xyz[0]); //2



}
//%l_inner_struct_field = getelementptr inbounds %example_inner* %struct_field68, i32 0,
i32 1 ;
//%r_inner_struct_field = getelementptr inbounds %example_inner* %struct_field67, i32 0,
i32 1 "
struct veccy {
    int sz;
    vector<int> v;
};

struct example_inner_inner {
    int x;
    vector<int> very_nasty;
};

struct example_inner {
    int x;
    struct example_inner_inner inner;
};

struct example {
    int e;
    int c;
    struct example_inner nasty;
};
```

```
2
1
1
2
2
```

```
master
{
    vector<struct Books2> bookies;
    struct Books2 book;
    book->b->book_id = 99;
    bookies::book;
    struct Books2 outbook;
    outbook = bookies[0];
    print(outbook->b->book_id);
    print(bookies[0]->b->book_id);

    struct veccy vy;
//    vy->v::5;
    vector<int> v;
    v::5;
    vy->v = v;
    v[0] = 6;
    vy->sz = 1;
//    print(vy->v[0]);
    vector<int> vv;
    vv::778;
    vv = vy->v;
    print(vv[0]);
}

struct Books {
   int book_id;
   int d;
};
struct Books2 {
   int book_id;
   int d;
   struct Books b;
};
struct veccy {
    int sz;
    vector<int> v;
};
```

```
99
99
5
```

```
struct foo {
    vector<int> vec;
};

job struct foo fun (struct foo bar, int i)
{
    vector<int> v = bar->vec;
    v::i;
    bar->vec = v;
    return bar;
}

job vector<int> ugly (vector<int> a, int i)
{
    a::i;
    return a;
}

master
{
    struct foo bar;
    bar = fun(fun(bar, 1), 2);
    vector<int> v = bar->vec;
    v = ugly(ugly(v, 3), 4);
    print(v[0]);
    print(v[1]);
    print(v[2]);
    print(v[3]);
}
```

1
2
3
4

```
struct foo {
    vector<struct bar> vec;
};

struct bar {
    bool b;
    vector<vector<struct baz>> vec;
    int a;
};

struct baz {
    int a;
};

job struct bar fun()
{
    struct bar br;
    struct baz bz;
    bz->a = 3;
    vector<struct baz> v;
    v::bz;
    vector<vector<struct baz>> vv;
    vv::v;
    br->vec = vv;
    br->b = false;
    br->a = 4;
    return br;
}

master {
    struct foo myfoo;
    struct bar mybar;
    struct baz mybaz;
    vector<struct baz> myvec;
    vector<vector<struct baz>> temp;
    for (mybaz->a = 0; mybaz->a < 10; mybaz->a = mybaz-> a + 1) {
        myvec::mybaz;
        temp::myvec;
    }
    mybar->vec = temp;
    mybar->b = true;
    mybar->a = 1002;
    vector<struct bar> temp2;
    temp2::mybar;
    myfoo->vec = temp2;
    mybar->b = false;
    temp2[0] = mybar;
    myfoo->vec = temp2;
        //print(size myfoo->vec);
        //print(size myfoo->vec[0]->vec);
        //print(size myfoo->vec[0]->vec[0]);
    vector<struct bar> v = myfoo->vec;
    vector<vector<struct baz>> w = v[0]->vec;
    print(w[5][5]->a);
        //print(myfoo->vec[0]->vec[0][5]->a);
    struct bar br = fun();
    print(br->a);
}
```

```
5
4
```

```
master
{
  int a;
  a = 42;
  print(a);
  return 0;
}
```

42

```
job vector<int> getidx (vector<vector<int>> vec, int idx)
{
    return vec[idx];
}

job double sum (vector<double> vec)
{
    double sum = 0.0;
    int i;
    for (i = 0; i < size vec; i = i + 1 )
        sum = sum + vec[i];
    return sum;
}

job vector<int> first_of_three(vector<int> a, vector<int> b, vector<int> c)
{
    vector<int> ret;
    ret::a[0];
    ret::b[0];
    ret::c[0];
    return ret;
}

master
{
    vector<vector<int>> vv;
    vector<int> v;
    v::5;
    v::7;
    v::9;
    vv::v;
    v::11;
    vv::v;
    vector<int> zeroth;
    zeroth = getidx(vv, 0);
    vector<int> first;
    first = getidx(vv, 1);
    print(size zeroth);
    print(size first);
    print(zeroth[2]);
    vector<double> ds;
    ds::2.3;
    ds::4.5;
    ds::7.7;
    vector<vector<double>> dd;
    dd::ds;
    printd(sum(dd[0]));
    vector<int> w;
    w::3;
    vector<int> z;
    z::88;
    vector<int> res;
    res = first_of_three(v, w, z);
    print(res[0]);
    print(res[1]);
    print(res[2]);
}
```

```
3
4
9
14.500000
5
3
88
```

```
master {
    vector<int> a;
    a::1;
    a::2;
    vector<int> b;
    b::0;
    b::77;
    print(b[0]);
    print(b[1]);
    b = a;
    a[0] = 5;
    a[1] = 6;
    print(a[0]);
    print(a[1]);
    print(b[0]);
    print(b[1]);

    vector<vector<int>>aa;
    vector<int>bb;
    bb::1;
    bb::2;
    bb::3;
    vector<int>cc;
    aa::bb;
    aa::cc;
    print(aa[0][0]);
    bb::4;
    cc::7;
    aa[0] = bb;
    aa[1] = cc;
    bb::5;
    cc::8;
    print(size aa[0]);
    print(size aa[1]);

}
```

```
0
77
5
6
1
2
1
4
1
```

```
master {

    vector<int> a;
    vector<int> aa;
    vector<vector<int>>b;
    a::1;
    aa::100;
    b::a;
    b[0]::2;
    b::aa;
    b[1]::200;
    b[1]::300;
    print(b[0][0]);
    print(b[0][1]);
    print(b[1][0]);
    print(b[1][1]);
    print(b[1][2]);
    vector<vector<vector<vector<double>>>> d4v;
    vector<vector<vector<double>>> d3v;
    vector<vector<double>> d2v;
    vector<double> d1v;
    d1v::4.5;
    d2v::d1v;
    d3v::d2v;
    d4v::d3v;
    vector<vector<double>> e2v;
    vector<double>e1v;
    e1v::5.4;
    e1v::6.7;
    e2v::e1v;
    d4v[0]::e2v;
    vector<double>lastv;
    lastv::4.2;
    d4v[0][0]::lastv;
    printd(d4v[0][0][0][0]);
    printd(d4v[0][1][0][0]);
    printd(d4v[0][1][0][1]);
    printd(d4v[0][0][1][0]);


  vector<int> aaa;
  vector<vector<int>> bbb;
  vector<vector<vector<int>>> ccc;
  vector<double> d;
  d::2.33;
  d::3.2;
  printd(d[1]);

  int i;
  for (i = 0; i < 10; i=i+1) {
    aaa::i;
    bbb::aaa;
    bbb[i]::10*i; //this works now!
    print(size aaa);
    print(i);
    print(aaa[i]);
    print(size bbb);
    print(bbb[i][i]);
    print(bbb[i][i+1]);
    ccc::bbb;
    print(size ccc);
    print(ccc[i][i][i]);
  }

  vector<int> foo;
  foo::5;
  foo::6;
  foo = seq (7);
  vector<int> bar;
  bar = seq2 (7);
```

```
  for (i = 0; i < size foo ; i = i + 1) {
    print(foo[i]);
    print(bar[i]);
  }
    int bazz;
    bazz = baz (5);
    print(bazz);
}

job vector<int> seq (int b)
{
    vector<int> a;
    int i;
    for (i = 0; i < b; i = i + 1) {
        a::i;
    }
    return a;
}

job vector<int> seq2 (int b)
{
    vector<vector<int>> a;
    vector<int> dummy;
    vector<int> dummy2;
    a::dummy;
    a::dummy2;
    int i;
    for (i = 0; i < b; i = i + 1) {
        a[1]::(2*i);
    }
    return a[1];
}

job int baz (int b)
{
    vector<int> a;
    a::b;
    return a[0];
}
```

```
1
2
100
200
300
4.500000
5.400000
6.700000
4.200000
3.200000
1
0
0
1
0
0
1
0
2
1
1
2
1
10
2
1
3
2
2
3
2
20
3
2
4
3
3
4
3
30
4
3
5
4
4
5
4
40
5
4
6
5
5
6
5
50
6
5
7
6
6
7
6
60
7
6
8
7
7
8
```

```
7
70
8
7
9
8
8
9
8
80
9
8
10
9
9
10
9
90
10
9
0
0
1
2
2
4
3
6
4
8
5
10
6
12
5
```

```
master
{
    vector<struct example> test;
    struct example a;
    a->e = 1;
    test::a;

    vector<struct example> test1;

    test1 = test;
    struct example tmp;
    tmp->e = 0;
    test[0] = tmp;
    print((test1[0])->e);
    print((test[0])->e);



}

struct veccy {
    int sz;
    vector<int> v;
};

struct example_inner_inner {
    int x;
    vector<int> very_nasty;
};

struct example_inner {
    int x;
    struct example_inner_inner inner;
};

struct example {
    int e;
    int c;
    struct example_inner nasty;
};
```

```
1
0
```

```
master
{

    vector<struct example_inner_inner> test4; //1


    struct example_inner_inner a4;
    a4->e = 1;
    vector<int> tmp4;
    tmp4::1;
    a4->very_nasty = tmp4; //copied
    test4::a4; //copied

    vector<struct example_inner_inner> test14;
    test14 = test4; //copied
    tmp4[0]=2;
    test14[0]->very_nasty = tmp4;

    tmp4 = test14[0]->very_nasty;
    print(tmp4[0]); //2
    tmp4 = test4[0]->very_nasty;
    print(tmp4[0]); //1



    vector<struct example_inner> test; //1
    struct example_inner a1;
    struct example_inner_inner a;
    a->e = 1;
    vector<int> tmp; //2
    tmp::1;
    a->very_nasty = tmp;
    a1->inner = a;
    test::a1; //copied

    vector<struct example_inner> test1;
    test1 = test; //copied
    tmp[0] = 2;
    test1[0]->inner->very_nasty = tmp; //copied
    tmp = test1[0]->inner->very_nasty;
    print(tmp[0]);
    tmp = test[0]->inner->very_nasty;
    print(tmp[0]);

}

struct simple {
    int e;
};

struct veccy {
    int e;
    //vector<int> v;
    struct simple e;
};

struct example_inner_inner {
    int e;
    vector<int> very_nasty;
};

struct example_inner {
    int e;
    struct example_inner_inner inner;
};

struct example {
    int e;
    int c;
    struct example_inner nasty;
```

```
};
```

```
2
1
2
1
```

```
master
{
  int i;
  i = 5;
  while (i > 0) {
    print(i);
    i = i - 1;
  }
  print(42);
  return 0;
}
```

```
5
4
3
2
1
42
```

```
master
{
  print(foo(7));
  return 0;
}

job int foo(int a)
{
  int j;
  j = 0;
  while (a > 0) {
    j = j + 2;
    a = a - 1;
  }
  return j;
}
```

```
14
```

```
CC=gcc
CFLAGS=-g -Wall
LDFLAGS=-g -L../

.PHONY: default
default: master slave

master: test$(TEST)_master.o ../libmsmaster.a
	$(CC) $(LDFLAGS) test$(TEST)_master.o -o master -lmsmaster -pthread -lm

slave: test$(TEST)_slave.o ../libmsslave.a
	$(CC) $(LDFLAGS) test$(TEST)_slave.o -o slave -lmsslave -pthread -lm

test$(TEST)_master.o: test$(TEST)_master.c test_master.h

test$(TEST)_slave.o: test$(TEST)_slave.c

.PHONY: clean
clean:
	rm -f *.o master slave

.PHONY: all
all: clean default
```

```
#include "test_master.h"

void master()
{
    int a = 4;
    int *p_a = malloc(sizeof(int));
    *p_a = a;
    int jid = start_job(1, (uint8_t *) p_a, sizeof *p_a);
    uint8_t *buf;
    uint32_t len;
    reap_job(jid, &buf, &len);
    if (len != 4) {
        fprintf(stderr, "bad length!\n");
    } else {
        int b = *((int *) buf);
        printf("%d->%d\n", a, b);
    }
    if (buf)
        free(buf);
}
```

```
#include <stdint.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

void job1(uint8_t *in_d, uint32_t in_l, uint8_t **out_d, uint32_t *out_l)
{
    if (in_l == sizeof(int)) {
        *out_d = malloc(sizeof(int));
        *((int *) *out_d) = *((int *) in_d) * 4;
        *out_l = sizeof(int);
    } else {
        *out_d = NULL;
        *out_l = 0;
    }
}

void (*job_funcs [])(uint8_t *, uint32_t, uint8_t **, uint32_t *) = {job1};
int num_jobs =  (sizeof(job_funcs) / sizeof(*job_funcs));
```

```
#include "test_master.h"

void master()
{
    int *a = malloc(1000 * sizeof(int));
    for (int i = 0; i < 1000; i++)
        a[i] = i;
    long long *b = malloc(100000 * sizeof(long long));
    for (int i = 0; i < 100000; i++)
        b[i] = i;
    int j1 = start_job(1, (uint8_t *) a, 1000 * sizeof(int));
    int j2 = start_job(2, (uint8_t *) b, 100000 * sizeof(long long));
    uint8_t *buf;
    uint32_t len;
    reap_job(j2, &buf, &len);
    if (len != sizeof(long long))
        fprintf(stderr, "Job 2: bad len\n");
    else
        printf("sum of 100k: %lli\n", *((long long *) buf));
    if (buf)
        free(buf);
    reap_job(j1, &buf, &len);
    if (len != sizeof(int))
        fprintf(stderr, "Job 1: bad len %d\n", len);
    else
        printf("sum of 1k: %d\n", *((int *) buf));
    if (buf)
        free(buf);
}
```

```
#include <stdint.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <unistd.h>

void addi(uint8_t *in_d, uint32_t in_l, uint8_t **out_d, uint32_t *out_l)
{
    if ((in_l % sizeof(int)) == 0) {
        int sum = 0;
        for (int i = 0; i < in_l / sizeof(int); i++)
            sum += ((int *) in_d)[i];
        int *out = (int *) malloc(sizeof(int));
        out[0] = sum;
        *out_l = sizeof(int);
        *out_d = (uint8_t *) out;
    } else {
        *out_d = NULL;
        *out_l = 0;
    }
}

void addl(uint8_t *in_d, uint32_t in_l, uint8_t **out_d, uint32_t *out_l)
{
    sleep(1);
    if ((in_l % sizeof(long long)) == 0) {
        long long sum = 0;
        for (int i = 0; i < in_l / sizeof(long long); i++)
            sum += ((long long*) in_d)[i];
        long long *out = (long long *) malloc(sizeof(long long));
        out[0] = sum;
        *out_l = sizeof(long long);
        *out_d = (uint8_t *) out;
    } else {
        *out_d = NULL;
        *out_l = 0;
    }
}

void (*job_funcs [])(uint8_t *, uint32_t, uint8_t **, uint32_t *) = {
    addi, addl };
int num_jobs =  (sizeof(job_funcs) / sizeof(*job_funcs));
```

```
#include "test_master.h"
#include <string.h>

#define NUMJOBS 1000
#define JOBSIZE 10000

int sum_ints(int *list, int num)
{
    int res = 0;
    for (int i = 0; i < num; i++)
        res += list[i];
    return res;
}

void master()
{
    int i1[] = {332114, 3562749};
    int *m_i1 = malloc(sizeof i1);
    memcpy(m_i1, i1, sizeof i1);
    int j1 = start_job(1, (uint8_t *) m_i1, sizeof i1);
    int primejobs[NUMJOBS];
    int *numsets[NUMJOBS];
    for (int i = 0; i < NUMJOBS; i++) {
        numsets[i] = malloc( JOBSIZE * sizeof(int));
        for (int j = 0; j < JOBSIZE; j++)
            numsets[i][j] = JOBSIZE * i + j;
        primejobs[i] = start_job(2, (uint8_t *) numsets[i],
                JOBSIZE * sizeof(int));
    }
    int *primesets[NUMJOBS];
    uint32_t primelens[NUMJOBS];
    for (int i = 0; i < NUMJOBS; i++) {
        reap_job(primejobs[i], (uint8_t **) &primesets[i], &primelens[i]);
        /*if (i == 200)
            cancel_job(primejobs[NUMJOBS-1]);*/
        if (primesets[i]) {
            free(primesets[i]);
        } else
            printf("lost job %d\n", primejobs[i]);
    }
    for (int numsets = 1; numsets <= NUMJOBS; numsets *= 10)
        printf("Number of primes up to %d: %ld\n", numsets * JOBSIZE,
            sum_ints((int *)primelens, numsets) / sizeof(int));
    int *gcd;
    uint32_t len;
    reap_job(j1, (uint8_t **)&gcd, &len);
    if (len == sizeof(int))
        printf("and the gcd of %d and %d is %d\n", i1[0], i1[1], *gcd);
    else
        printf("bad len from gcd\n");
    if (gcd)
        free(gcd);
}
```

```
#include <stdint.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <math.h>

int j_gcd(int a, int b)
{
    if (a > b)
        return j_gcd(a-b, b);
    if (a < b)
        return j_gcd(a, b-a);
    return a;
}

int j_prime(int a)
{
    if (a < 2)
        return 0;
    int s = (int) sqrt(a);
    for (int i = 2; i <= s; i++)
        if (a % i == 0)
            return 0;
    return 1;
}

void s_gcd(uint8_t *in_d, uint32_t in_l, uint8_t **out_d, uint32_t *out_l)
{
    if (in_l != 2 * sizeof(int)) {
        *out_d = NULL;
        *out_l = 0;
        return;
    }
    *out_d = malloc(sizeof(int));
    *((int *)*out_d) = j_gcd(((int *)in_d)[0], ((int *)in_d)[1]);
    *out_l = sizeof(int);
}

void s_prime(uint8_t *in_d, uint32_t in_l, uint8_t **out_d, uint32_t *out_l)
{
    if (in_l % sizeof(int)) {
        *out_d = NULL;
        *out_l = 0;
        return;
    }
    *out_d = malloc(in_l);
    int num_primes = 0;
    for (int i = 0; i < in_l / sizeof(int); i++)
        if (j_prime(*(((int *)in_d) + i)))
            *(((int *)*out_d) + num_primes++) = *(((int *)in_d) + i);
    *out_d = realloc(*out_d, num_primes * sizeof(int));
    *out_l = num_primes * sizeof(int);
}

void (*job_funcs [])(uint8_t *, uint32_t, uint8_t **, uint32_t *) = {
    s_gcd, s_prime };
int num_jobs =  (sizeof(job_funcs) / sizeof(*job_funcs));
```

```
#!/bin/bash

RESULT1="4->16"
RESULT2=$'sum of 100k: 4999950000\nsum of 1k: 499500'
RESULT3=$'Number of primes up to 10000: 1229\nNumber of primes up to 100000: 9592\nNumber
 of primes up to 1000000: 78498\nNumber of primes up to 10000000: 664579\nand the gcd of
332114 and 3562749 is 787'

cd ..
make all > /dev/null
cd test

for i in {1..3}
do
    make all TEST=$i > /dev/null
    S=`./master & ./slave`
    VAR="RESULT$i"
    if [[ "$S" == "${!VAR}" ]]
    then
        echo "test $i pass"
    else
        echo "test $i fail"
    fi
done
```

```
#ifndef __TEST_MASTER_H_
#define __TEST_MASTER_H_

#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>

#define JOB_UNASSIGNED 0
#define JOB_FINISHED -1
#define JOB_FAILED -2
#define JOB_LOST -3

int start_job(uint32_t ord, uint8_t *data, uint32_t len);
void reap_job(int32_t jid, uint8_t **data, uint32_t *len);
int get_job_status(int jid);
int cancel_job(int jid);

#endif /* __TEST_MASTER_H_ */
```