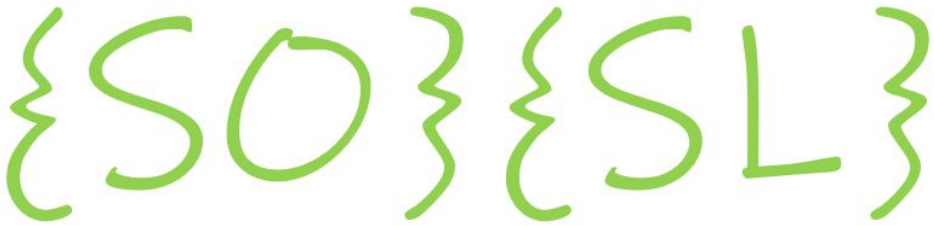


SOSL: Set Operation Simplification Language

Ryan Koning: Manager

Garrison Grogan: Language Guru

Ryan Chun: System Architect

Trisha Maniar: Tester

TABLE OF CONTENTS

1. Introduction & Proposal

SOSL (Set Operation Simplification Language) is a language that focuses on simplifying formal set theory operations in order to make it easy to manipulate sets and create functions that act on them, their elements, and/or subsets. Rather than rely on the user to create functions for basic set operations, SOSL will provide special operators for these functions (intersection, union, complement, etc.).

SOSL is meant to allow the programmer to write programs that evaluate set theoretic equations and functions without having to go through the cumbersome process of defining the functions long hand in a given language. The power of our language is in the operator set, because the programmer can forego calling a cumbersome standard library function and simply use SOSL's special operators. This makes things concise and simplifies, relative to other languages like Java, the notion of working with sets in a programming environment.

2. Language Tutorial

SOSL has been developed through OCaml; all related libraries and packages can easily be downloaded through opam, which is the OCaml Package Manager. The version of the OCaml llvm library should match the version of the LLVM system installed on your system.

2.1 To Run and Test SOSL

To run: To run, inside the SOSL directory, use “make.”

```
opam config exec -- \
      ocamlbuild -use-ocamlfind sosl.native
Finished, 25 targets (0 cached) in 00:00:04.
cc    -c -o printlib.o printlib.c
cc    -c -o linkedlist.o linkedlist.c
cc    -c -o setlib.o setlib.c
```

To test: To test all available tests, run ./testall.sh inside the SOSL directory. To run individual tests, run ./sosl.native ./tests/test-hello.sl, where “test-hello.sl” can be subsituted for any of the tests available in the test directory. ./sosl.native takes a SOSL file, which has an .sl extension, and outputs LLVM assembly. (screenshot of ./sosl.native ./tests/test-hello.sl).

```
; ModuleID = 'SOSL'
source_filename = "SOSL"

@fmt = private unnamed_addr constant [4 x i8] c"%d\0A\00"
@fmt.1 = private unnamed_addr constant [4 x i8] c"%g\0A\00"
@fmt.2 = private unnamed_addr constant [4 x i8] c"%s\0A\00"
@fmt.3 = private unnamed_addr constant [4 x i8] c"%c\0A\00"
@string = private unnamed_addr constant [13 x i8] c"Hello World!\00"

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

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

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

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

declare void @destroy(i8*, ...)

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

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

declare i8* @complement(i8*, i8*, ...)

declare i8* @set_union(i8*, i8*, ...)
```

```
declare i8* @intersect(i8*, i8*, ...)

define i32 @main() {
entry:
  %str = alloca i8*
  store i8* getelementptr inbounds ([13 x i8], [13 x i8]* @string, i32 0, i32
0), i8** %str
  %str1 = load i8*, i8** %str
  %prints = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8],
[4 x i8]* @fmt.2, i32 0, i32 0), i8* %str1)
  ret i32 0
}
```

2.2 Example Program

```
1 int main()
2 {
3         int a;
4         a = 42;
5         print(a);
6         return 0;
7 }
```

As can be seen in the screenshot above, a main method is mandatory in SOSL. All variables have to be initialized at the beginning of each method before any values are stored. There is also a built-in print function.

3. Language Manual

3.1 Data Types

int
A series of digits [0-9] to be read in decimal form that can range from −2,147,483,648 to 2,147,483,647 (32-bit signed integer).

- Example: int num;
  num = 32;

char
A single character (digit, letter, etc.) enclosed in single quotes (‘x’).

- Example: char c;
  c = ‘a’;

boolean
A binary variable that has two possible values: true and false.

- Example: boolean flag;
  flag = true;

void
An empty data type used for methods that have no return value.

set
A sequence of integers or characters separated by commas within colon-braces (:{x, y, z}:). A set can also be a sequence of subsets which hold integers or characters.

- Example: s:{int}: a;
  a = :{1, 2, 3, 4}:;

3.2 Operators

3.2.1 Basic Operators

+ (plus):

- Binary operator defined as addition on integers
- Valid types: int, set
- Examples: a + b, (a + b) + c
- Returns: Integer if left and right hand sides are integers. If the left and right hand sides are sets, returns a set.
- When sets are given, the operator returns a set whose elements are the additions of all the elements of the left and right hand side sets. Sets and integers cannot be mixed and will result in an error. Sets of different cardinalities will also return an error if they are added together. All the elements of the sets must have the operator defined for their type. Adding two types that do not have addition defined for each other will also return an error.
- 5 + 3 = 8

- :{1,2}: + :{3,4}: = :{4,6}:
- :{:{1,2}:,1}: + :{:{3,4}:,7}: = :{:{4,6}:,8}:
- :{1,:{1}:}: + :{:{1}:,1}: returns an error

- (minus):

- Binary operator defined as subtraction on integers
- Valid types: int, set
- Examples: a - b, (a - b) - c
- When a set is given, the operator returns a set whose elements are the subtractions of all the elements of the left and right hand side sets. Sets and Integers cannot be mixed and will return an error. Sets of different cardinalities will also return an error if they are subtracted together. All the elements of the sets must have the operator defined for their type. Subtracting two types that do not have - defined for each other will also return an error
- 5 - 6 = -1
- :{1,2}: - :{3,4}: = :{-2,-2}:
- :{:{5,6}:,1:} - :{:{3,4}:,7:} = :{:{2,2}:,-6}:
- :{1,:{5,6}:}: -: {7,:{3,4}:}: returns an error

*** (star):**

- Binary operator defined as multiplication on integers as well as sets
- Valid types: int, set
- Examples: a * b, (a * b) * c
- When a set is given, the operator returns a set whose elements are the product of all the elements of the left and right hand side sets. Sets and Integers cannot be mixed and will return an error. Sets of different cardinalities will also return an error if they are multiplied together. All the elements of the sets must have the operator defined for their type. Multiplying two types that do not have * defined for each other will also return an error.
- 5 * 6 = 30
- :{1,2}: * B=:{3,4}: = :{3,8}:
- :{:{5,6}:,1}: * B = :{:{3,4}:,7}: = :{:{15,12}:,7}:
- :{1,:{5,6}:}: * :{7,:{3,4}:}: returns an error

/ (divide):

- Binary operator defined as division on integers and sets
- Valid types: int, set
- Examples: a / b, (a / b) / c
- When a set is given, the operator returns a set whose elements are the quotient of all the elements of the left and right hand side sets. Sets and Integers cannot be mixed and will return an error. Sets of different cardinalities will also return an error if they are divided together. All the elements of the sets must have the operator defined for their type. Multiplying two types that do not have / defined for each other will also return an error. Integers are rounded down always.
- 5 / 6 = 0

- :{3,4}: / B=:{2,2}: = :{1,2}:
- :{:{5,9}:,7}: / :{:{3,4}:,2}: = :{:{1,2}:,3}:
- :{1,:{5,6}:}: / :{7,:{3,4}:}: returns an error

% (modulus):

- Binary operator defined as modulus on integers and sets.
- Valid types: int, set
- Examples: a % b, (a % b) % c
- When a set is given, the operator returns a set whose elements are the modulus of all the elements of the left and right hand side sets. Sets and Integers cannot be mixed and will return an error. Sets of different cardinalities will also return an error if they are moduloed together. All the elements of the sets must have the operator defined for their type. Taking the mod of two types that do not have / defined for each other will also return an error.
- 23 % 6 = 5
- :{3,4}: % :{2,2}: = :{1,0}:
- :{:{5,9}:,13}: % :{:{3,4}:,7} = :{:{2,1}:,6}:
- :{1,:{5,6}:}: % :{7,:{3,4}:}: returns an error

:u (union):

- Binary operator defined as the union of two sets
- Valid types: set
- Examples: A:u B, A :u (B :u C)
- When a set is given, the operator returns a set whose elements are the union of the left and right hand side sets. If a type is given on either side that is not a set, an error is returned.
- :{1,2}: :u :{1,3,4}: = :{1,2,3,4}:
- :{:{1,4,5}:,6,:{7,8}:}: :u :{5}: = :{:{1,4,5}:,6,:{7,8}:,5}:

:n (intersection):

- Binary operator defined as the intersection of two sets
- Valid types: set
- Examples: A :n B, A :n (B :n C)
- When a set is given, the operator returns a set whose elements are the intersection of the left and right hand side sets. If a type is given on either side that is not a set, an error is
  Returned.
- :{1,2}: :n :{1,3,4}: = :{1}:
- :{:{1,4,5}:,6,:{7,8}:}: :n :{5}: = :{}:
- :{:{1,2}:,5,6}: :n :{:{1,2}:,6,7}: = :{:{1,2}:,6}:

:i (in):

- Binary operator that checks if an element is in a particular set.
- Valid types: set valid on LHS and RHS. int valid only as LHS
- Examples: A :i B, A :i (B :u C),6 :i C

- The operator returns a boolean telling whether or not the LHS is in the RHS. Putting an integer as the RHS returns an error.
- :{1,2}: :i :{1,3,4}: is false
- 1 :i :{1,3,4}: is true
- :{2,3}: :i 6 returns an error

:c (complement):
- Binary operator that returns the complement of a set given a universe.
- Valid types: sets only, both the LHS and RHS
- Examples: A :c B, A :c (B :u C)
- The operator returns a Set that is the complement of the LHS argument, given a universe of the RHS argument. If LHS contains elements not in RHS, those are not returned in the resultant set.
- :{1,2}: :c :{1,3,4}: = :{3,4}:
- 1 :c :{1,3,4}: returns error
- :{2,3}: :c :{1,6,7}: = :{1,6,7}:
- :{1,5,6}: :c :{1,2,3,4,5,6}: = :{2,3,4}:

|| (cardinality):
- A unary operator, also a delimiter. When a set is placed between the two bars, an integer representing the cardinality of the set. Returns an error if not used with a set.
- Ex: A = :{1,2,3,4,5}:; |A| returns 5.

3.2.2 Comparison Operators

In SOSL, comparison operators allow the user to compare the quantity of elements in a set on the left side of an expression to the quantity of elements in a set on the right side of an expression. Comparison operators return either true or false and are of type boolean.

> (greater than):
- Binary operator that returns a boolean from a left to right comparison:
- Ex. :{2, 3}: > :{4, 5}: is false
- :{2, 3}: > :{4, 5, 6}: is false
- :{2, 3, 4}: > :{4, 5}: is true

>= (greater than or equal to):
- Binary operator that returns a boolean from a left to right comparison:
- Ex. :{2, 3}: >= :{4, 5}: is true
- :{2, 3} >= {4, 5, 6}: is false
- :{2, 3, 4} >= {4, 5}: is true

< (less than):
- Binary operator that returns a boolean from a left to right comparison:
- Ex. :{2, 3}: < :{4, 5}: is false
- :{2, 3}: < :{4, 5, 6}: is true
- :{2, 3, 4}: < :{4, 5}: is false

<= (less than or equal to):

- Binary operator that returns a boolean from a left to right comparison:
- Ex. :{2, 3}: <= :{4, 5}: is true
- :{2, 3}: <= :{4, 5, 6}: is true
- :{2, 3, 4}: <= :{4, 5}: is false

== (equal to):

- Binary operator that returns a boolean from a left to right comparison. Only applies to sets and compares them element wise.
- Ex. :{2,3}: == :{4,5}: is false
- :{4,5,6}: == :{4,5,6}: is true
- To check equality regarding the cardinality of two sets, use the | | delimiter and ==.

3.2.3 Logical Operators:

!:

- Unary operator that returns a boolean for the logical inverse of its operand.

OR:

- Binary operator that returns a boolean for the logical or of its operands.

AND:

- Binary operator that returns a boolean for the logical and of its operands.

3.2.4 Order of Operations:

Order of Operations: Mathematical expressions involving integers behave with normal PEMDAS order of operations. Set operations are evaluated left to right in following the following hierarchy: (),:u = :n,:c,:i. :i has the lowest since the left and right sides of an :i expression must be completely evaluated before :i can. Since :u and :n have equal order, they will be evaluated left to right.

- Ex: A :c C :u A :n B is equivalent to A:c ((C :u A) :n B).
- Ex: A :c C :n D :i B :u C is equivalent to (A: c (C :n D)) :i (B :u C)

3.3 Control Flow

3.3.1 Scoping

A scope is a logical region of program where name binding is valid. Inspired by C language, a SOSL program consists of several scopes defined by the nestable curly brackets. Name bindings declared outside of explicit brackets will be defined as global variables and be accessed anywhere in the program.

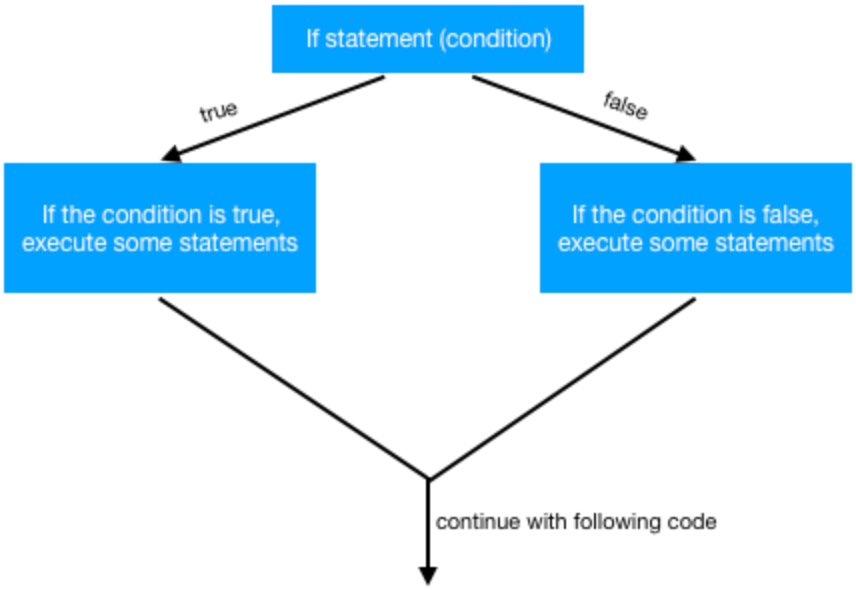
```
1 int main()
2 {
3         set:{int}: x;
4         int y;
```

```
 5
 6          x = :{1,2,3}:;
 7          if(|x| > 0)
 8          {
 9                  y = y + 10;
10          }
11          return y;
12  }
```

In the above example, two variables, x and y, are defined in the main function scope. Since the if-condition scope is nested in the main function scope, variable x and y can be accessible inside the if scope.

3.3.2 Conditional Logic

Execution of statements based on condition met:




```
 1 set:{int}: s;
 2 set:{int}: t;
 3 s = :(1,2,3,4}:;
 4 t = :{2,3,4,5}:;
 5
 6 if (s == t)
 7 {
 8          s = s + :{1,1,1,1}:;
 9 }
10 if else (t = s + :{1, 1, 1, 1}:)
11 {
12 t = t + :{1, 1, 1, 1}:;
13 }
14 else
```

```
15 {
16         s = s + :{2, 2, 2, 2}:;
17 }
18
19 print_set(s);
20 print_set(t);
```

After this, s = :{1, 2, 3, 4}: and t = :{3, 4, 5, 6}: because the “if else” statement is true. Therefore, the statements within those brackets gets executed. If neither the “if” or the “if else” statement were true, then the statements within the “else” brackets would get executed.

Execution of statements for each element in set

- The forEach method works by parsing through each element in a given set
- Example:

```
1 set :{int}: s;
2 s = :{1, 2, 3, 4}:;
3 forEach (int e in s)
4 {
5         e = e + 1;
6 }
7 print_set(s);
```

- After this, s = :{2, 3, 4, 5}:

Execution of statements based on iterating systematically through elements

- The for method works by iterating given 3 specific statements
- The first statement will initialize a variable
- The second statement will be a condition that the variable must meet
- The third statement will modify the variable in some way (usually by incrementing or decrementing)

3.3.3 Jump Statements

Break: A call to “break” in the middle of a “forEach” loop will terminate that loop and proceed to the next statement outside of the forEach loop.

3.4 Error Handling

SOSL will throw errors for a variety of reasons, including:

- **Unrecognized functions**: functions that have not been declared correctly
- **Type mismatch**: setting incorrect value types to variables
- **Undeclared identifier**: calling an identifier without initializing it first
- **Incorrect number of arguments**: expected number of arguments is different from given number of arguments

- **Code following a return statement**: there should not be code to run after the return statement

3.6 Grammar

Files in Appendix

3.7 Standard Library & Sample Program

3.4.1 Standard Library

The SOSL Standard Library provides commonly used set operations. It also provides functions like print. These functions can referenced as built in. Below are some example programs implemented in the standard library.

3.4.1.1 cartP - Cartesian Product

```
1  set:{int}: cartP(set:{int}: x, set:{int}: y)
2  {
3          set:{int}: temp;
4          forEach(i in x)
5          {
6                  forEach(j in y)
7                  {
8                          temp = temp :u :{x, y}:;
9                  }
10         }
11         return temp;
12 }
```

3.4.2 Sample Program

```
1  set subtract(set:{int}: a, set:{int}: y)
2  {
3          return a :n (b :c a);
4  }
5
6  boolean isSubset(set:{int}: x, set:{int}: y)
7  {
8          if (|x| < |y|)
9          {
10                 return void;
11         }
12
13         forEach (element in y)
14         {
15                 if(!(element :i x))
16                 {
17                         return false;
```

```
18                  }
19          }
20          return true;
21  }
22
23  int main()
24  {
25          set:{int}: a;
26          set:{int}: b;
27          set:{int}: c;
28          boolean subset;
29
30          a = :{1,2,3,4}:;
31          b = :{3,4}:;
32          c = subtract(a, b);
33          print_set(c);
34          subset = isSubset(a, c);
35          printb(subset);
36          return 0;
37  }
```

4. Project Plan

4.1 Planning Process

In general, SOSL group members met every week; the day of the week varied based on availabilities, but most often we met on Thursdays. As deliverable deadlines approached, we met multiple times a week to complete everything in a timely manner.

We used Slack mostly to communicate with each other, and any deadlines or responsibilities that we decided on at meetings were put in the Slack by our manager, Ryan K. When writing our Language Reference Manual, we split up the different operators and control flow amongst the 4 of us.

For testing, Trisha set up the test suite. Trisha wrote a majority of the tests, and the rest of the group members assisted along the way. Towards the end of the project, Trisha went back and reviewed all tests to make sure that all parts of the language were tested and made sure to include robust tests that failed and passed.

4.2 Timeline

Deliverable	**Due Date**
Deciding on language	September 11th
Initial proposal written	September 19th
Split up work on Language Reference Manual	October 4th
LRM, and initial Scanner, Parser and Ast completed	October 14th
Working on Makefile, Ast	November 1st
Working on Ast, Codegen, Semant, Sast, SOSL files	November 13th
Hello World	November 14th
Discussed deliverables setup by SOSL team	November 17th
Tests are beginning to run successfully	November 27th

Conditional logic in codegen edited, debugging for operator tests	November 30th
set.c begins to work	December 11th
Working on set functionality in SOSL, writing more tests, completion of language	December 12-18th

4.3 Programming Style Guide

The following styling and conventions were used in the SOSL language:

- OCaml was used for all compiler architecture
- C was used for creation of set and for some operators
- SOSL files end in extension .sl
- Variable and functions identifiers begin with lowercases and use camelcase
- Test files were written using "-" punctuation

In general for collaboration we used the following guidelines:

- Each team member, except Garrison who pushed directly to master, created separate branches to be used for pushing requests
- We tried not to push code that would break the compiler, but sometimes it was unavoidable
- If changes were made in files, we signed them with our initials in comments, so anyone else going into the file could know who made what changes in order for follow-up
- Tests for features were written as the features were implemented to confirm their functionality

Environments used:

- Operating Systems: Mac OS, Linux (Ubuntu and Arch), Windows Ubuntu Subsystem
- Text Editor: Vim, Visual Studio
- Version Control: Git

4.4 Roles and Responsibilities

Ryan K - Manager:

My responsibilities consisted of scheduling regular meetings, team correspondence to TA's, disseminating information about compiler progress to remaining teammates, determining the scope of deliverables and bringing them to the team to distribute work. Regularly posting relevant documentation. Stitching together the mess of code snips and reverse engineered all

files of code to compile early version of SOSL. Debugged said early version of all files until basic tests passed. Built set type declaration syntax, SetLiteral syntax and integrated them into all files. Typed all built in function decls in codegen. Added necessary changes to all pretty-printing functions in ast and sast. An awful lot more I can't remember.

Ryan C - System Architect:

I was in charge of designing system architecture, including implementation of set type and functions and corresponding semant checking. I also was responsible for linking set.c and llvm through void pointers.

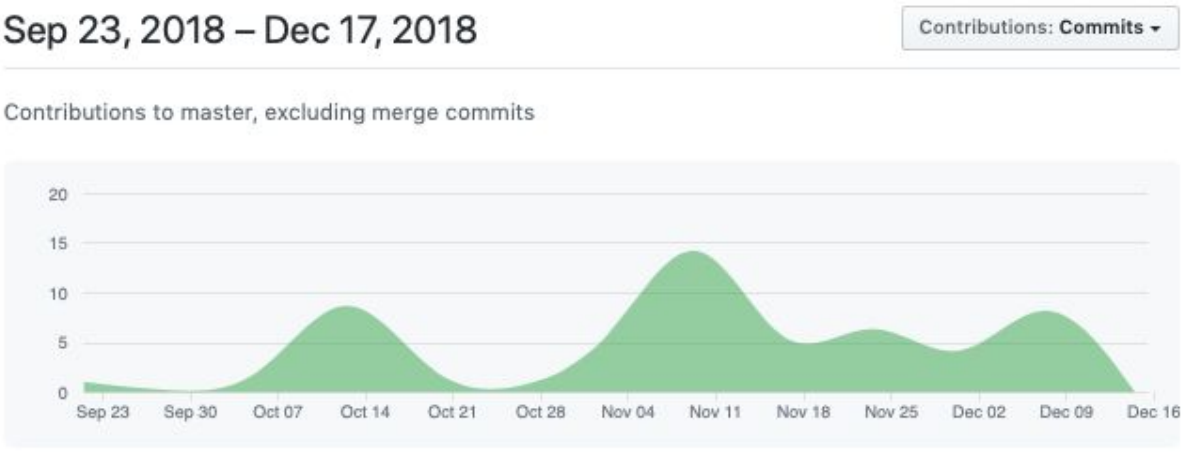
Garrison - Language Guru:

I was responsible for the language design. I made most final syntax choices. I also wrote most of the set library functions. I also wrote bits and pieces of the AST, Semantic checker, and Parser.

Trisha - Tester:

I wrote the testing suite and a majority of the tests for our language. I helped write pieces of the AST and Parser, and helped in removing pattern matching warnings across all the files. I wrote a majority of the final report and put together our presentation slides.

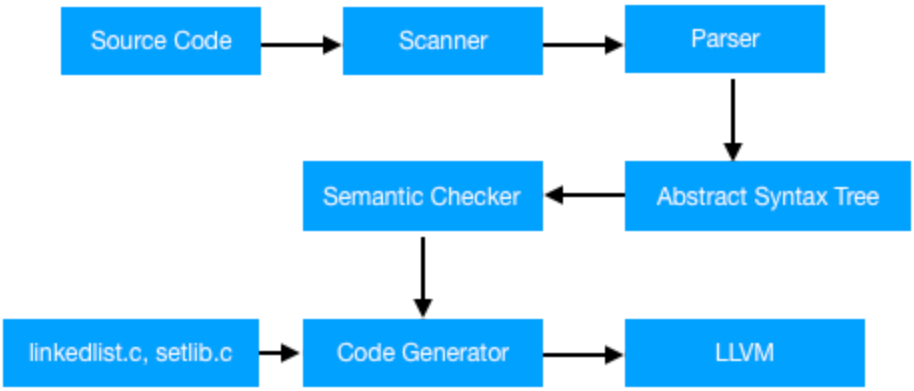
4.5 Project Log



Commit log is in Appendix.

5. Architectural Design

5.1 Block Diagram



5.2: Roles

Scanner: Garrison, Ryan K. (mostly), Trisha (sometimes)

Parser: Ryan C., Ryan K. (mostly), Garrison (sometimes)

Ast: Garrison, Trisha, Ryan K.

Semantic Checker: Ryan C., Ryan K.

Code Generator: Ryan C., Ryan K. (mostly)

Test files: Trisha (mostly), Ryan C., Ryan K. (sometimes)

.c, .h files: Ryan C., Garrison, Ryan K. (linkedlist.c)

Language Reference Manual: Trisha, Ryan C, Ryan K, Garrison

Final Report and Project Slides: Trisha

The scanner (scanner.mll) takes a file with the extension .sl as input and generates tokens as specified by SOSL's conventions. If something in the input file is not recognizable by the scanner, it will throw an error. The tokens then generated are used by the parser.

The parser (parser.mly) takes the tokens generated by the scanner and generates an Abstract Syntax Tree (AST - ast.ml) based on the grammar. If the code can be parsed, then it is syntactically correct.

The semantic checker verifies the AST generated by the parser and translates it into the Semantically-checked Abstract Synax Tree (SAST - sast.ml).

The code generator (codegen.ml) traverses through the AST to generate LLVM IR. Through the code generator we also added set.c, which creates our set struct in C. The functions and set types are also defined in codegen.ml in relation to the C file.

6. Test Plan

Running ./testall.sh in the SOSL directory will run all of the tests in the tests directory and say whether each test passed with “OK” or failed with “FAILED”. Trisha worked on testall.sh, which was the shell script used for automation, and most of the demos with everyone’s help.

6.1 Example programs and Target Languages

Example program: test-func1.sl

```
 1 int add()
 2 {
 3         int a;
 4         int b;
 5         int c;
 6         a = 1;
 7         b = 2;
 8         c = a+b;
 9         return c;
10 }
11
12 int main()
13 {
14         int i;
15         i = add();
16         print(i);
17         return 0;
18 }
```

LLVM:

```
; ModuleID = 'SOSL'
source_filename = "SOSL"

@fmt = private unnamed_addr constant [4 x i8] c"%d\0A\00"
@fmt.1 = private unnamed_addr constant [4 x i8] c"%g\0A\00"
@fmt.2 = private unnamed_addr constant [4 x i8] c"%s\0A\00"
@fmt.3 = private unnamed_addr constant [4 x i8] c"%c\0A\00"
@fmt.4 = private unnamed_addr constant [4 x i8] c"%d\0A\00"
@fmt.5 = private unnamed_addr constant [4 x i8] c"%g\0A\00"
@fmt.6 = private unnamed_addr constant [4 x i8] c"%s\0A\00"
@fmt.7 = private unnamed_addr constant [4 x i8] c"%c\0A\00"

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

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

declare i8* @create_set(i32, ...)
```

```
declare i8* @adds(i8*, i32, ...)

declare void @destroy(i8*, ...)

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

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

declare i8* @complement(i8*, i8*, ...)

declare i8* @set_union(i8*, i8*, ...)

declare i8* @intersect(i8*, i8*, ...)

define i32 @main() {
entry:
  %i = alloca i32
  %add_result = call i32 @add()
  store i32 %add_result, i32* %i
  %i1 = load i32, i32* %i
  %print = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4
x i8]* @fmt, i32 0, i32 0), i32 %i1)
  ret i32 0
}

define i32 @add() {
entry:
  %a = alloca i32
  %b = alloca i32
  %c = alloca i32
  store i32 1, i32* %a
  store i32 2, i32* %b
  %a1 = load i32, i32* %a
  %b2 = load i32, i32* %b
  %tmp = add i32 %a1, %b2
  store i32 %tmp, i32* %c
  %c3 = load i32, i32* %c
  ret i32 %c3
}
```

Example program: test-if2.sl

```
1 int main()
2 {
3     int variable;
4     boolean flag;
5     flag = true;
```

```
 6     if(flag)
 7     {
 8         variable = 200;
 9     }
10
11     print(variable);
12
13         return 0;
14 }
```

LLVM:

```
; ModuleID = 'SOSL'
source_filename = "SOSL"

@fmt = private unnamed_addr constant [4 x i8] c"%d\0A\00"
@fmt.1 = private unnamed_addr constant [4 x i8] c"%g\0A\00"
@fmt.2 = private unnamed_addr constant [4 x i8] c"%s\0A\00"
@fmt.3 = private unnamed_addr constant [4 x i8] c"%c\0A\00"

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

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

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

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

declare void @destroy(i8*, ...)

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

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

declare i8* @complement(i8*, i8*, ...)

declare i8* @set_union(i8*, i8*, ...)

declare i8* @intersect(i8*, i8*, ...)

define i32 @main() {
entry:
  %variable = alloca i32
  %flag = alloca i1
  store i1 true, i1* %flag
  %flag1 = load i1, i1* %flag
  br i1 %flag1, label %then, label %else
```

```
merge:                                            ; preds = %else, %then
  %variable2 = load i32, i32* %variable
  %print = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4
x i8]* @fmt, i32 0, i32 0), i32 %variable2)
  ret i32 0

then:                                             ; preds = %entry
  store i32 200, i32* %variable
  br label %merge

else:                                             ; preds = %entry
  br label %merge
}
```

6.2 Test Suites

6.2.1 Variable assignments: We tested the variables individually to make sure that declaring a variable worked because that is a building block for all our code.

Test:

```
1 int main()
2 {
3         int a;
4         a = 42;
5         print(a);
6         return 0;
7 }
```

Output:

```
1 42
```

Test:

```
1 int main()
2 {
3         char c;
4         c = 'a';
5         printc(c);
6         return 0;
7 }
```

Output:

```
1 a
```

Test:

```
1 int main()
```

```
{
    boolean fact;
    fact = true;
    printb(fact);
        return 0;
}
```

Output:

```
1
```

Test:

```
int main()
{
    boolean fact;
    fact = false;
    printb(fact);
        return 0;
}
```

Output:

```
0
```

Test:

```
int g;
int main()
{
        int a;
        int b;
        a = b = 42;
        g = 72;
        print(a = a+b+g);
        print(a);
        return 0;
}
```

Output:

```
156
156
```

6.2.2 Simple Arithmetic: We tested simple arithmetic (addition, subtraction, multiplication and division) because these were 4 major operators for integers in our language.

Test:

```
int main()
```

```
 2 {
 3         int a;
 4     int b;
 5     int c;
 6     a = 10;
 7     b = 30;
 8     c = a+b;
 9
10     print(c);
11         return 0;
12 }
```

Output:

```
1 40
```

Test:

```
int main()
{
    int a;
    int b;
    int c;
    a = 10;
    b = 30;
    c = b-a;

    print(c);
        return 0;
}
```

Output:

```
1 20
```

Test:

```
 1 int main()
 2 {
 3     int a;
 4     int b;
 5     int c;
 6     a = 10;
 7     b = 30;
 8     c = a*b;
 9
10     print(c);
11         return 0;
12 }
```

Output:

```
300
```

Test:

```
nt main()
{
    int a;
    int b;
    int c;
    a = 10;
    b = 30;
    c = b/a;

    print(c);
        return 0;
}
```

Output:

```
3
```

6.2.3 If statements: We wanted to test if statements because this is a part of control flow, which is important in our language in relation to what code is read and in what order.

Test:

```
int main()
{
        boolean flag;
        int a;
        int b;
        flag = true;
        if (true)
        {
                a = 1;
                print(a);
        }

        else
        {
                b = 2;
                print(b);
        }

        return 0;
}
```

Output:

```
1
```

Test:

```
int main()
{
    int variable;
    boolean flag;
    flag = true;
    if(flag)
    {
        variable = 200;
    }

    print(variable);

        return 0;
}
```

Output:

```
200
```

Test:

```
int main()
{
        int a;
        int b;
        a = 5;
        b = 5;
        if (a==b)
        {
                print(a);
        }
        else
        {
                print(0);
        }

        return 0;
}
```

Output:

```
5
```

6.2.4 For statements: We wanted to test for statements because this is a part of control flow, which is important in our language in relation to what code is read and in what order.

Test:

```
int main()
{
        int i;
        for (i = 0; i < 5; i = i+1)
        {
                print(i);
        }

        return 0;
}
```

Output:

```
0
1
2
3
4
```

Test:

```
int main()
{
        int out;
        int i;
        out = 0;
        for (i = 0; i < 5; i=i+1)
        {
                out = out + 5;
        }

        print(out);

        return 0;
}
```

Output:

```
25
```

6.2.5 forEach statement: We wanted to test forEach statements because this is a part of control flow, which is important in our language in relation to what code is read and in what order.

Test

```
int main()
{
    set:{int}: a;
    int sum;

    a = :{1,2,3}:;
    sum = 0;

    foreach (int k in a) {
        sum = sum + k;
    }

    print(sum);
        return 0;
}
```

Output

```
6
```

6.2.6 Basic sets: We wanted to test basic sets because sets are also a building block of our language.

Test:

```
int main()
{
    set:{int}: a;

    prints("OK");
        return 0;
}
```

Output:

```
OK
```

Test:

```
int main()
{
        set:{set:{int}:}: a;
        a = :{:{1}:}:;
}
```

Output:

```
{1,2,3,4}
```

Test:

```
int main()
{
        set:{string}: a;

        a = :{"hi","bye","sye"}:;
        prints("OK");
        return 0;
}
```

Output:

```
OK
```

6.2.6 Union statements: We wanted to test the union of sets because this is what makes our language unique, and is part of most programs having to do with sets.

Test:

```
int main()
{
    set:{int}: a;
    set:{int}: b;
    set:{int}: c;
    int result;

    a = :{1,2,3}:;
    b = :{4,5,6}:;

    c = a :u b;

    result = (c :i 1) + 10000;
    print(result);
        return 0;
}
```

Output:

```
10001
```

6.2.7 Intersection statements: We wanted to test the intersection of sets because this is what makes our language unique, and is part of most programs having to do with sets.

Test:

```
int main()
{
    set:{int}: a;
```

```
 4     set:{int}: b;
 5     set:{int}: c;
 6
 7     a = :{1,2,3}:;
 8     b = :{3,4,5}:;
 9
10     c = a :n b;
11
12         return 0;
13 }
```

Output:

```
1
```

6.2.8 Complement statements: We wanted to test the complement of sets because this is what makes our language unique, and is part of most programs having to do with sets.

Test:

```
 1 int main()
 2 {
 3     set:{int}: a;
 4     set:{int}: b;
 5     set:{int}: c;
 6
 7     a = :{1,2,3}:;
 8     b = :{3,4,5}:;
 9
10     c = a :c b;
11
12         return 0;
13 }
```

Output:

```
1
```

6.2.9 Element of statements: We wanted to test the element of function because this is what makes our language unique, and is part of most programs having to do with sets.

Test:

```
1 int main()
2 {
3     set:{int}: a;
4     int result;
5
6     a = :{1,2,3}:;
```

```
 7      result = 100;
 8
 9      if (a :i 3){
10          result = result + 100;
11      }
12
13      print(result);
14
15          return 0;
16  }
```

Output:

```
1 200
```

6.2.10 Adding characters to set: We wanted to test adding characters to sets because it's important to be able to change the length of the set because users would want to manipulate that.

Test:

```
1 nt main()
2 {
3      set:{int}: a;
4      adds(a, 7);
5      print_set(a);
6          return 0;
7 }
```

Output:

```
1 :{7}:
```

6.2.11 Note:

While we were able to add many of the sets needed for our language, we would have wanted to add more to test more of the intricacies and more complicated programs. Because, however, debugging took us quite a while, especially in relation to all the functions in setlib.c, these were the only tests we have. Furthermore, because we weren't able to print sets, we couldn't add further tests because they didn't prove to be helpful in verifying whether certain parts of our code worked. If print_set did work, we would have added more tests having to do with tests in relation to union, intersection, complement, etc. in relation to different types of control flow and with different types of elements.

7. Lessons Learned

Trisha: One of the biggest things I learned was the importance of testing, especially in creating tests along the way. Having the automated script was a huge help because otherwise we would have needed to test each individual function again after making any changes. I also learned how important it was to not push commits that had errors in them because of the confusion it can cause within the team. In relation to collaboration style, I learned why this was so important. Advice I would have for future teams is to start as early as possible because you won't even realize how quick the deadlines come up!

Ryan K: I learned that it is always better start small and build outward to further complexity. Trying to start in the middle of complexity and building to more complexity creates quagmires.

Ryan C: Sometimes it is better to ask for help than struggling to solve a problem for hours by yourself.

Garrison: I learned it is easier to work piecemeal on lots of separate parts, then attempt to completely finish one before moving on to the next. It was/is difficult to integrate codegen with our C code without running into lots of gotchas. I would advise other teams to schedule more meetings with your group TA, as general office hours can fill up/TAs not attend.

8. Appendix

Commit Log:

```
commit 796d4bbcebc739621f7c4b8d6ee9311d1e07c778 (HEAD -> master, origin/master,
origin/HEAD)
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 18:32:14 2018 -0500

    prob in has

commit 2a673815b780182937db4a06ef5a4861783b52e3
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 18:26:26 2018 -0500

    removed all no used code, only version warnings left

commit 26fc91552155db18bc1af533751f409c6360d988
Merge: c448267 fe1af2b
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 18:12:52 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit c44826786e2140ca500b67c0967608af5b32984a
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 18:12:49 2018 -0500

    more fixes for tests and semant

commit fe1af2b08ef4cbecf48a64847f642ce0814f5bcb
Merge: 725f353 41b1d58
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Dec 19 18:09:58 2018 -0500

    Merge pull request #78 from gammison/ryan-n

    uncomment

commit 41b1d58120ae148036b5b2a431106f964e8d6596 (origin/ryan-n)
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 19 18:09:24 2018 -0500

    uncomments

commit 725f35336ef4390b5725d0b7e214965d22fd4b3e
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 18:04:33 2018 -0500

    more name dumb
```

```
commit 943f11bffcfa26cc689805666c2639b90d202e4e
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 18:01:13 2018 -0500

    typo in create name

commit af7be588b8bc8ae025856ab2a518e9385665d5f1
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 17:58:39 2018 -0500

    add fix

commit 6cc59685719b3048e89e4c0c6951fba09c2ccdb1
Merge: 3455d0e 204869f
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Dec 19 17:00:18 2018 -0500

    Merge pull request #77 from gammison/ryan-n

    no add function

commit 204869f1c25fe716bb6c5264a32a61f051fa1687
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 19 16:59:48 2018 -0500

    no add function

commit 3455d0e23cd916cb6255e6e5b962bc0dae855bb2
Merge: 65cd4dc cf97d4e
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Dec 19 16:46:47 2018 -0500

    Merge pull request #76 from gammison/ryan-n

    ? tests

commit cf97d4eb54fccbacf0c433f80ecdbfed106902e2
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 19 16:46:18 2018 -0500

    ? tests

commit 65cd4dcd72c567ba902a03cb50218620d64f81fd
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 16:39:57 2018 -0500

    has const kinda works, setlit not working on strings
```

```
commit be2f6a8fbab71da1047078a4d7b5f101321646a9
Merge: c151f8e 8d0562e
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 16:23:42 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit c151f8e25db10da74ee7d34fcae581aa640d8e3d
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 16:23:26 2018 -0500

    has and has const

commit 8d0562eb0e93d1ce0cff48aae6650011ea8479e7
Merge: c3ab5eb 8246398
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Dec 19 16:22:22 2018 -0500

    Merge pull request #75 from gammison/ryan-n

    Ryan n

commit 8246398ba12b374dfea57f6c2b2090a227fc616b
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 19 16:21:51 2018 -0500

    setlit modification

commit 44433f34f7b8769f5b3c0559f6cc7c2aee361bcb
Merge: 9cdd032 c3ab5eb
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 15:57:07 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit 6f818cb15e465899432bc5219c57bde0dc0c101e
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 19 15:55:44 2018 -0500

    more tests

commit c3ab5eb113ee477f2cf32542ed0fad704369a129
Merge: dba3f3e 8c402ca
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Dec 19 15:52:59 2018 -0500

    Merge pull request #74 from gammison/ryan-n
```

```
    new test and has

commit 8c402ca10fe10f7b4c6b46b25c80a772df06cacf
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 19 15:52:17 2018 -0500

    new test and has

commit 9cdd0323e151957349e2687f906b5473452e2518
Merge: b7d24f0 dba3f3e
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 15:38:18 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit b7d24f0149605129e4f54bff093e577e09a22dd7
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 15:38:15 2018 -0500

    ignore objs

commit dba3f3e2852162ee0683872e63f18ca1d08b1236
Merge: a2fd996 6ffde2c
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Dec 19 15:37:57 2018 -0500

    Merge pull request #73 from gammison/ryan-n

    test

commit 6ffde2ca509a223c9438c8d29f80e902f1039bf0
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 19 15:37:30 2018 -0500

    test

commit a2fd996d5921b11e538340626e72c4260eafc452
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 15:26:28 2018 -0500

    forgot to push

commit 840799f4f186e05ee2fc7afd6c5ae452f62e2386
Merge: 7bb1853 cad1e3e
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 15:23:21 2018 -0500
```

```
    Merge branch 'ryan-n' of https://github.com/gammison/SOSL

commit cad1e3e8283093381e18505fce8743e0791ec54f
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 19 15:22:53 2018 -0500

    tests

commit 7bb1853d9b486eab61bf0544da359f44cdf0f72a
Merge: 7c87695 4ffc73d
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Dec 19 15:17:05 2018 -0500

    Merge pull request #72 from gammison/ryan-test

    set.h change

commit 4ffc73d0fa95b15367c83039e8c2bd9718e909f4 (origin/ryan-test)
Merge: 4f7a623 7c87695
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Dec 19 15:16:58 2018 -0500

    Merge branch 'master' into ryan-test

commit 4f7a623d7f0dc195f563f02b47fc1e6df8beafe5
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 19 15:15:46 2018 -0500

    set.h change

commit 7c876957e6e23bd6a8820d3229d37fdd713469fe
Merge: e90e53e 59d8f63
Author: Garrison Grogan <gammison@users.noreply.github.com>
Date:   Wed Dec 19 15:00:26 2018 -0500

    Merge pull request #71 from gammison/ryan-test

    Ryan test

commit 59d8f6390b1924becd22b7b487f8019ff231dc02
Merge: 0800115 c0be4dc
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 19 14:59:42 2018 -0500

    lots of stuff

commit e90e53e278d9fd382ed4b91db9dc1aee8cdf2f86
Author: Garrison Grogan <garrison.grogan@columbia.edu>
```

```
Date:   Wed Dec 19 14:52:39 2018 -0500

    calls for print add and remove, still a semant error

commit 08001152105ccdfcf228f09e8f30fb8aa4fd34b2
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 19 13:42:58 2018 -0500

    codegen fix

commit c0be4dc5f4ae0a0ffd771d1b0859d13f74e19375
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 13:39:39 2018 -0500

    adds still broken, as is using set

commit 09364345b007b6f05fffaa9a741124a46978fc00
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 12:28:32 2018 -0500

    almost, wierd type error and literals still need done

commit 9ec65bc7b4a0a30aca4c8ebae86f81650c877a50
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 04:48:56 2018 -0500

    printc fix, function semant checking still broke

commit 4d3f46be2eb391824b902e22a5201a2a82da0cfc
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 04:09:19 2018 -0500

    should all be lists, but its still expecting an elmtype

commit 7a279dfefbcdc380daac913860b8d715accc6f77
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 04:06:28 2018 -0500

    expects an elmtype but we need to do lists, what do

commit e164e9cce55ebb99b869d9bbf54002c4bca05878
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 03:41:24 2018 -0500

    semant errors

commit ba39ea6e74b4b91e61388558f90a70556852d7a2
Author: Garrison Grogan <garrison.grogan@columbia.edu>
```

```
Date:   Wed Dec 19 03:17:19 2018 -0500

    more print fixing

commit 8344f878448febed419e989ccc0bec6aec75bcdf
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 03:05:01 2018 -0500

    print_set and literals broken, need be fixed

commit 0e62abec59d04cc79c96b3fbaa490576b22f2caf
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 02:51:58 2018 -0500

    test add change f call

commit 6f0872af60fb3d63efb6c84a589f166732684c29
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 02:49:47 2018 -0500

    changed right print, changed set add to adds avoid function ambiguity

commit 001670b0b605ebd17903691e1c4b07746559a989
Merge: 38f6f9c 7ab5232
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Dec 19 02:43:33 2018 -0500

    Merge pull request #70 from gammison/ryan-test

    print_set codegen

commit 7ab5232f3e2515fbaa0c57aca7308155c92fee11
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 19 02:42:54 2018 -0500

    print_set codegen

commit 38f6f9cc2e7abfd9613e409d7a6bf1e30bb81b6a
Merge: abd7055 16bd05c
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 02:41:19 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit abd7055dab6cdd890067302d5d8a7b92bd4eab4b
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 02:41:16 2018 -0500
```

```
    fixed casting

commit 16bd05c90e0bf69126d782e92b74ad5d8aaf5136
Merge: ad251d4 ba1c36f
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Dec 19 02:30:01 2018 -0500

    Merge pull request #69 from gammison/ryan-test

    scall add

commit ba1c36f65f0f27c4704e4653c1f1f5575fe75fb2
Merge: 8deb4be ad251d4
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Dec 19 02:29:53 2018 -0500

    Merge branch 'master' into ryan-test

commit 8deb4be4a700a6c7faa503c0bd79903e041d641c
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 19 02:26:55 2018 -0500

    scall add

commit ad251d47b905817f3604947251db0940a04873ec
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 19 02:20:28 2018 -0500

    printing set added, changed remove behavior

commit 53ed5df460114b975a138691296ee1450c521180
Merge: 778f211 6d70d46
Author: Ryan <kryanchun@gmail.com>
Date:   Tue Dec 18 23:58:25 2018 -0500

    Merge pull request #68 from gammison/ryan-test

    Ryan test

commit 6d70d467db1700a547127aa97f303d844250ca62
Merge: 0e106e8 d3215e9
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Tue Dec 18 23:54:18 2018 -0500

    set operators and semant change

commit 778f2115115e8be2ee48564109f630a731d56152
Author: Garrison Grogan <garrison.grogan@columbia.edu>
```

```
Date:   Tue Dec 18 23:05:51 2018 -0500

    removed unused fns

commit d3215e9370c0685b6a3d25ce7ca5e9f551b3ada8
Merge: 7bb91f1 415a7d1
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Dec 18 22:36:37 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit 7bb91f15544a5a8a2591ab375516850621cb5f60
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Dec 18 22:36:21 2018 -0500

    extra cases on compare

commit 415a7d19fea374d68cc1e9df8d6df902a4a768ab
Merge: 5dc1768 8d5d575
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Tue Dec 18 22:34:42 2018 -0500

    Merge pull request #67 from gammison/RKbranch2

    forgot to add the correct return values and args for a couple of the …

commit 8d5d5753c67fd5d0e767935f74bf1eee0bd1227b (origin/RKbranch2)
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Tue Dec 18 22:33:12 2018 -0500

    forgot to add the correct return values and args for a couple of the
function declarations from set.c in codegen.

commit 5dc17680e540dab778fe9c52d16e65e822895436
Merge: eaa4aba bf1764b
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Dec 18 22:32:39 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit eaa4aba318258a928aaf3843544038a48f5427de
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Dec 18 22:32:28 2018 -0500

    intersection rewrite and h file rename for consitency

commit 0e106e8ff01a6bccdcaab70d122633a0741d1eb1
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
```

```
Date:   Tue Dec 18 22:14:34 2018 -0500

    setlit test on codegen

commit bf1764bbca3e9944d3a303979ebe7c91e7e636c7
Merge: 64f8261 8c3c39a
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Tue Dec 18 22:12:46 2018 -0500

    Merge pull request #66 from gammison/RKbranch2

    added the rest of the functions from set.c that were added, added the…

commit 8c3c39aa224a12627f77dd4c95e07e39aef283c0
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Tue Dec 18 22:12:11 2018 -0500

    added the rest of the functions from set.c that were added, added the
correct struct type and pointer for set in llvm data types, and changed the
set.c functions to the correct return and argument types to reflect the change.

commit 64f8261186d08b64dc0164866732cbec5b86e261
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Dec 18 21:03:38 2018 -0500

    change set.c to setlib.c

commit 07c486cc857b7136aa5ea8bedba737123b4faf98
Merge: 77079e3 a0585f9
Author: Ryan <kryanchun@gmail.com>
Date:   Tue Dec 18 20:08:01 2018 -0500

    Merge pull request #65 from gammison/ryan-test

    more functions in set.c

commit a0585f96caa0d88ab7683a67f95c086eb0335567
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Tue Dec 18 20:06:54 2018 -0500

    more functions in set.c

commit 77079e3e494486aa60af01f61b0a4af2e6db93b4
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Dec 18 19:55:06 2018 -0500

    added create set and head data/access
```

```
commit f879b3586c5a99d7353b237f1d06887ad3b08673
Merge: c273b17 a4e7688
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Dec 18 19:45:20 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit c273b17f02ec8a0e0e1e8f45059acfbeda6e5ca1
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Dec 18 19:45:15 2018 -0500

    will have to fix merge

commit a4e768851378bcc93283ba30c79e6e89958e8364
Merge: 375a0cb b35515a
Author: Ryan <kryanchun@gmail.com>
Date:   Tue Dec 18 19:44:38 2018 -0500

    Merge pull request #64 from gammison/ryan-test

    getCard and some progress on forEach

commit b35515ade9ff72dafee165ca7843541789268013
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Tue Dec 18 19:43:39 2018 -0500

    getCard and some progress on forEach

commit 375a0cb5bba2fcb8bc419e1684f0a1bcc153dfdb
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Dec 18 18:51:13 2018 -0500

    forgot the h file

commit a341f65e91baff3164d54438ead981bc5e380428
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Dec 18 18:49:30 2018 -0500

    set.c finished hope, no more errors and just have to write create function

commit 3dfa0f6fb1470f28da4612028f1d770e4719aaef
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Dec 18 18:15:51 2018 -0500

    took out edwards code, set.c almost done, fixing has method and need to
write create_set function for llvm

commit 3ee503cfb84503780f33adbd83d0cf96375efbfa
```

```
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Dec 18 14:17:40 2018 -0500

    fixed linking the list library and set library

commit 7b5136eadc64e726aff019ec2ff3f440a524ac9d
Merge: 0eb15ab 6d87e21
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Tue Dec 18 00:58:17 2018 -0500

    Merge pull request #63 from gammison/RKbranch2

    added the rest of the functions from our standard library to codegen

commit 6d87e21853dc40e21307b7e8463807c0db1d1378
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Tue Dec 18 00:57:20 2018 -0500

    added the rest of the functions from our standard library to codegen

commit 0eb15ab2c2205eea4a70447500c3ca008cbe3a53
Merge: a23e48b 3122e5c
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Mon Dec 17 23:28:33 2018 -0500

    Merge pull request #62 from gammison/RKbranch2

    added some of the set.c functions in codegen, we still need to figure…

commit 3122e5c6910b5f7ec98930d2bd9717c894843f40
Merge: 6fc7ab1 a23e48b
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Mon Dec 17 23:28:26 2018 -0500

    Merge branch 'master' into RKbranch2

commit 6fc7ab1b56076776f2f0680e4c067e19976302f5
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Mon Dec 17 23:27:06 2018 -0500

    added some of the set.c functions in codegen, we still need to figure out
what we want to do with create.

commit a23e48bc5f83b6d4dd297f8d62c2c2f9f0d28c15
Merge: 7cc41e1 67fbb77
Author: Ryan <kryanchun@gmail.com>
Date:   Mon Dec 17 23:25:48 2018 -0500
```

```
    Merge pull request #61 from gammison/ryan-test

    setlit in semant

commit 67fbb77dc12b10c5693b553b2b3a9269e1265c70
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Mon Dec 17 23:24:35 2018 -0500

    ast fix

commit e568db92641971f5807ae3eb11cf20f001d57eb9
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Mon Dec 17 23:01:19 2018 -0500

    setlit in semant

commit 7cc41e18cf66911c279f222583e397d9bd4e186a
Merge: a6b4616 9f8bdea
Author: Ryan <kryanchun@gmail.com>
Date:   Mon Dec 17 21:49:24 2018 -0500

    Merge pull request #60 from gammison/ryan-test

    Ryan test

commit 9f8bdeae8fc45032a2b56d37dedc6407d5dbb516
Merge: 9a40d7c a6b4616
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Mon Dec 17 21:48:34 2018 -0500

    formatting

commit a6b461673d11868c645199aa8acaa9f60f5e07fa
Merge: d6a1c31 43953f7
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Mon Dec 17 21:46:34 2018 -0500

    Merge pull request #59 from gammison/RKbranch2

    updated codegen for set type to handle void pointer. Fixed parsing er…

commit 43953f74230cd9eb45ed2c2b5c31d2427cd97c0f
Merge: 4da4e0b d6a1c31
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Mon Dec 17 21:46:28 2018 -0500

    Merge branch 'master' into RKbranch2
```

```
commit 4da4e0b1a5eedeb6e56a383e6a9886b3127e7200
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Mon Dec 17 21:42:47 2018 -0500

    updated codegen for set type to handle void pointer. Fixed parsing errors
in set type, updated semant.

commit 9a40d7ccdc446f0f9529d30dcc856874814ec25e
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Mon Dec 17 21:40:20 2018 -0500

    structure

commit 096a36c0575d572ee680917bd8de7c28f0430a81
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Mon Dec 17 20:46:03 2018 -0500

    typo in set.c

commit d6a1c311657353baa8d6e6439e48a4db3b6c04ce
Merge: 1aeefb6 d105def
Author: Ryan <kryanchun@gmail.com>
Date:   Mon Dec 17 20:44:52 2018 -0500

    Merge pull request #58 from gammison/ryan-test

    set.c create

commit d105def18474040f3ebf99fa51e95770cb918cc5
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Mon Dec 17 20:44:05 2018 -0500

    set.c create

commit 1aeefb6f84545442807c3ab4135ae3361be235c4
Merge: 91a8316 8346997
Author: trishamaniar1 <trm2144@barnard.edu>
Date:   Sun Dec 16 16:10:18 2018 -0500

    Merge pull request #57 from gammison/TrishaBranch

    edits to test files

commit 8346997763fb756e98e10a9742a6f87182271971 (origin/TrishaBranch)
Author: Yesha Maniar <yeshamaniar@Yeshas-Air.lan1>
Date:   Sun Dec 16 16:07:43 2018 -0500

    edits to test files
```

```
commit 91a8316ee8d6a9dc3fd6d0ebb6762bc8c33145b6
Merge: 282222a 9ff35e5
Author: Ryan <kryanchun@gmail.com>
Date:   Sun Dec 16 14:42:30 2018 -0500

    Merge pull request #56 from gammison/ryan-test

    tests: set operators

commit 9ff35e5c1a966f9c9365657dea5d32093e0572e2
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Sun Dec 16 14:41:41 2018 -0500

    set operators

commit 282222aa0372dbe1a34d6fb708923802426838d3
Merge: 789f6ab aa89700
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Sun Dec 16 13:00:02 2018 -0500

    Merge pull request #55 from gammison/RKbranch2

    needed to change Set type to return a pointer, not the type itself.

commit aa89700acd66dc3bf9cf999143fc001785b20e33
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Sun Dec 16 12:59:22 2018 -0500

    needed to change Set type to return a pointer, not the type itself.

commit 789f6aba792cd8adfa7d586583119be5e9c4121d
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sat Dec 15 22:49:37 2018 -0500

    comment out set and arr elm assign, break semant, added a test

commit a44972be87ff8e6468e0d70913d0d15cd58499b8
Merge: c491da2 9fed115
Author: Ryan <kryanchun@gmail.com>
Date:   Sat Dec 15 22:47:23 2018 -0500

    Merge pull request #54 from gammison/ryan-test

    commented out set operators

commit 9fed11544c2b274594fe1aaee58e064fb3972c2b
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
```

```
Date:   Sat Dec 15 22:46:26 2018 -0500

    commented out set operators

commit c491da27151f26a0cd319c5bb42d0272b130e715
Merge: 6a616f2 bee29eb
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sat Dec 15 22:37:02 2018 -0500

    merge resolved

commit 6a616f233f1e202c668ab0d66beda0af20dd0a5c
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sat Dec 15 22:36:06 2018 -0500

    pretty indentation needed

commit bee29eb9241707ce047d8e22bd58b50c4ccc0df1
Merge: 407870a b7d8d7d
Author: Ryan <kryanchun@gmail.com>
Date:   Sat Dec 15 22:29:20 2018 -0500

    Merge pull request #53 from gammison/ryan-test

    Ryan test

commit b7d8d7df2163969ba0859ea2aa754d6da53e8781
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Sat Dec 15 22:28:42 2018 -0500

    foreach

commit 0534e7866b406e14b94f0aa64e395292a7b57997
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Sat Dec 15 22:18:08 2018 -0500

    elof

commit 30c78bd01899a8c5437360844bfb046aa9f3aa87
Merge: 93831a5 407870a
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Sat Dec 15 22:15:52 2018 -0500

    change

commit 407870a0dd2059f3ff430923c36395e7d4cf09bc
Merge: dec0f09 b24e92e
Author: Garrison Grogan <garrison.grogan@columbia.edu>
```

```
Date:   Sat Dec 15 22:11:15 2018 -0500

    merge resolved, comment space

commit dec0f090323bd12d29f95a405f9ab9f10b34230a
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sat Dec 15 22:10:06 2018 -0500

    naming and setlit sexpr

commit b24e92eb6db5a3dbfc937bd91fa6db879d468757
Merge: ddddd81 72ca48e
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Sat Dec 15 22:05:21 2018 -0500

    Merge pull request #52 from gammison/RKbranch2

    Uncommented and fixed Set type declaration in codegen, changed test-s…

commit 72ca48e2987b402c2414d70dc7e656fc34dc07a6
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Sat Dec 15 22:04:31 2018 -0500

    Uncommented and fixed Set type declaration in codegen, changed test-set3 to
char set and verified through compiler that set type declaration returns the
correct type in the back end and commented out set_access for time being so
compilation is successful.

commit 93831a591463fe668d908df95da63ec48a83b5b7
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Sat Dec 15 21:40:25 2018 -0500

    semant for set checking

commit ddddd8189b658f8d2450c6ca869288772cd9e289
Merge: 363ef33 bcd0951
Author: trishamaniar1 <trm2144@barnard.edu>
Date:   Fri Dec 14 16:54:31 2018 -0500

    Merge pull request #51 from gammison/TrishaBranch

    more function tests

commit bcd095174ed5b48b69aced3e9ec548f1b469c0ec
Author: Yesha Maniar <yeshamaniar@Yeshas-MacBook-Air.local>
Date:   Fri Dec 14 16:52:49 2018 -0500

    more function tests
```

```
commit 363ef3311636c986bbde3783a70fa4deea3429a1
Merge: 6408bc5 c155667
Author: trishamaniar1 <trm2144@barnard.edu>
Date:   Fri Dec 14 15:59:39 2018 -0500

    Merge pull request #50 from gammison/TrishaBranch

    function tests

commit c155667dab07dcc37262131926f0c1f0a1cd7909
Author: Yesha Maniar <yeshamaniar@Yeshas-MacBook-Air.local>
Date:   Fri Dec 14 15:59:16 2018 -0500

    function tests

commit 6408bc582aac5cd705b55e6c9dcc1ff7a693c7ef
Merge: e222a67 9a3a02d
Author: trishamaniar1 <trm2144@barnard.edu>
Date:   Fri Dec 14 15:08:35 2018 -0500

    Merge pull request #49 from gammison/TrishaBranch

    new fail tests

commit 9a3a02ddf2b3f38509abc7e9b4768eef5101e90c
Author: Yesha Maniar <yeshamaniar@Yeshas-MacBook-Air.local>
Date:   Fri Dec 14 15:07:49 2018 -0500

    new fail tests

commit e222a672c795bf562cc5553c3eb3c3831e7d2bd5
Merge: 840e464 3bd5a5b
Author: trishamaniar1 <trm2144@barnard.edu>
Date:   Fri Dec 14 14:24:40 2018 -0500

    Merge pull request #48 from gammison/TrishaBranch

    sample fail tests

commit 3bd5a5b98cc49e95baee8c31e9119ae25b0ca5dc
Author: Yesha Maniar <yeshamaniar@Yeshas-MacBook-Air.local>
Date:   Fri Dec 14 14:24:01 2018 -0500

    sample fail tests

commit 840e46471758c2438823938815d24c93a415dac3
Merge: 30f547f e427b07
```

```
Author: trishamaniar1 <trm2144@barnard.edu>
Date:   Thu Dec 13 22:59:52 2018 -0500

    Merge pull request #47 from gammison/TrishaBranch

    test files edited

commit e427b0757b3c4da2995aa9b17e92d25e5d6780de
Author: Yesha Maniar <yeshamaniar@Yeshas-Air.lan1>
Date:   Thu Dec 13 22:52:46 2018 -0500

    test changes

commit 8221d3565ca65d49d533c117b7c0280f0629673d
Merge: 9e33797 30f547f
Author: Yesha Maniar <yeshamaniar@Yeshas-Air.lan1>
Date:   Thu Dec 13 21:36:38 2018 -0500

    pulling from master

    Merge branch 'master' of https://github.com/gammison/SOSL

commit 30f547fa1bc3b66399871f63c8ac6c88d3855e19
Merge: 242fb4a 1302016
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Thu Dec 13 19:47:07 2018 -0500

    Merge pull request #46 from gammison/RKbranch2

    R kbranch2

commit 130201600bb249c51b31ee8ec4187c1008580461
Merge: d63d3cc 242fb4a
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Thu Dec 13 19:46:20 2018 -0500

    resolved conflicts

commit d63d3cc6ab7a914fabe063178cc31a7977723807
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Thu Dec 13 19:41:57 2018 -0500

    ast and parser Set and SetLit are done.
    ast and parser Set and SetLit are done.
    ast and parser Set and SetLit are done.

commit 9e337979ed87fd4a688d5bee42d795f9ccabac64
Merge: 853fa50 242fb4a
```

```
Author: Yesha Maniar <yeshamaniar@dyn-209-2-228-159.dyn.columbia.edu>
Date:   Thu Dec 13 18:20:07 2018 -0500

    pull from master

    Merge branch 'master' of https://github.com/gammison/SOSL

commit 853fa5006e04ed8d3d6e7114ddc16dcdb256c58a
Merge: 70f5af3 15b204c
Author: Yesha Maniar <yeshamaniar@dyn-209-2-228-159.dyn.columbia.edu>
Date:   Thu Dec 13 18:12:54 2018 -0500

    Merge branch 'TrishaBranch'

commit 242fb4a6703bca1b2f9e8c80c3946bc710dd7dea
Merge: 0657e71 6ce1501
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Thu Dec 13 18:07:43 2018 -0500

    Merge pull request #45 from gammison/RKbranch2

    Made changes to fix shift/reduce error in parser and fix lit_list for…

commit 6ce15014850f4351ba01d6e2dd666d5b6c0a3c81
Merge: 3e8f500 0657e71
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Thu Dec 13 18:07:34 2018 -0500

    Merge branch 'master' into RKbranch2

commit 3e8f500f68d34271caf1e24dc41f69ba958fbc34
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Thu Dec 13 18:05:24 2018 -0500

    Made changes to fix shift/reduce error in parser and fix lit_list for set
Literals.

commit 0657e7146b31de395780c25a6cc0debb3a0d56cc
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Thu Dec 13 18:05:01 2018 -0500

    files moving and scanner

commit fc82b76b928dad50ab6046bc52f947e556a30e08
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Thu Dec 13 16:54:24 2018 -0500

    parser and scanner fixed for sets, need to fix ast now
```

```
commit 06f14805757e660ebc9393f534894d7e25ef4271
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Thu Dec 13 16:26:59 2018 -0500

    changed parser for set and scanner

commit 01cdee11f276ac73be2e13974b9c07361a606bd9
Merge: cf1597d 750ba26
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 12 18:40:42 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit 750ba26b866af4a5df4fab373670179d868379ba
Merge: 4ebecc5 15b204c
Author: trishamaniar1 <trm2144@barnard.edu>
Date:   Wed Dec 12 17:43:28 2018 -0500

    Merge pull request #44 from gammison/TrishaBranch

    tests for basic ops

commit 15b204c3176c8af76d365baa1c7479cb3cf17030
Author: Yesha Maniar <yeshamaniar@Yeshas-MacBook-Air.local>
Date:   Wed Dec 12 17:42:33 2018 -0500

    tests for basic ops

commit 4ebecc56dbbdce701c6a87e368d72374171dbc4c
Merge: d2e2091 82438e0
Author: trishamaniar1 <trm2144@barnard.edu>
Date:   Wed Dec 12 17:33:22 2018 -0500

    Merge pull request #43 from gammison/TrishaBranch

    bug fix for for loop

commit 82438e041d26d06a38fd2e54d05ea31688c838bb
Author: Yesha Maniar <yeshamaniar@Yeshas-MacBook-Air.local>
Date:   Wed Dec 12 17:32:16 2018 -0500

    bug fix for for loop

commit d2e20912df6b4ab13af490d538096790251ff3d3
Merge: 70f5af3 b310371
Author: trishamaniar1 <trm2144@barnard.edu>
Date:   Wed Dec 12 16:48:14 2018 -0500
```

```
    Merge pull request #42 from gammison/TrishaBranch

    new if tests

commit b310371595b435c3fcbebce39d529d740eac288c
Author: Yesha Maniar <yeshamaniar@Yeshas-MacBook-Air.local>
Date:   Wed Dec 12 16:35:22 2018 -0500

    new if tests

commit cf1597d933164f04a9face70b3bb59a9ce55369a
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 12 16:19:07 2018 -0500

    change to printlib

commit 70f5af39fdeff40570510d2d54b95ce4679e20cb
Merge: ee07689 c58ad46
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 12 16:10:02 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit ee076899f06b72641ec29c1b19406613a3201ec3
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 12 16:09:49 2018 -0500

    complement added to set.c

commit c58ad4635320972ba0f44f2420004916466c5c5a
Merge: 020c975 b2fcb22
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Wed Dec 12 15:23:18 2018 -0500

    Merge pull request #41 from gammison/RKbranch2

    adding test files for set

commit b2fcb2208bff7bc8831606ef3689ce46033d9abe
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Wed Dec 12 15:22:05 2018 -0500

    adding test files for set

commit 020c975102af701b4f906264f4f46ec8d7670250
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 12 15:20:28 2018 -0500
```

```
    name fix

commit 0b31c97156251b727ae7c28ea270b6bfef80960f
Merge: 9d69d61 b7f1338
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Dec 12 15:15:18 2018 -0500

    Merge pull request #40 from gammison/ryan-test

    Ryan test

commit b7f133809c568e60399ec4bc8238a5f1e253cb80
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 12 15:14:31 2018 -0500

    more tests

commit 9d69d613b70928cf6f8164a9a07ab2d222b86fae
Merge: 221366b 8bd7bba
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Wed Dec 12 15:13:37 2018 -0500

    Merge pull request #39 from gammison/RKbranch2

    Fixed set and setLit implementation for decl and ast.

commit 8bd7bba19fbfdce1e97f12780ff205b68039bb33
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Wed Dec 12 15:12:41 2018 -0500

    Fixed set and setLit implementation for decl and ast.

commit 0764a0dd85ab7430cbb49e7c8e937858405a02ed
Merge: 49afb19 221366b
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 12 14:57:44 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL into ryan-test

commit 221366b2574a1c1cd8ea29bcabae5ea98f15d926
Merge: e4e9435 3cfa1fd
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 12 14:24:11 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit e4e943531220cc9b5884f4d2206038e8904ea7cc
```

```
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 12 14:23:53 2018 -0500

    parse S_LIT, type needed in parser

commit 3cfa1fda6155d39840f596dc3f76ea3f8977903d
Merge: dd8bda8 769c46d
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Dec 12 14:06:17 2018 -0500

    Merge pull request #38 from gammison/ryan-test

    typo fix

commit 49afb199a6c45ea7071cfd195ec1869aec9494e8
Merge: 769c46d dd8bda8
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 12 14:05:46 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL into ryan-test

commit 769c46db5df1471fd161f1a99612aba0b889b269
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 12 14:05:05 2018 -0500

    typo fix

commit dd8bda875b6e5aac2027ab933f11bf7cc8518d3b
Merge: c356b31 e829312
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Dec 12 14:04:48 2018 -0500

    Merge pull request #37 from gammison/ryan-test

    Ryan test

commit e8293121ba30973df8ced340dcf05974857b140a
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 12 14:03:37 2018 -0500

    edited destroy function

commit 3638c7e386e5c4bb287a26f5b8b00c0400f8b0c8
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Dec 12 14:02:31 2018 -0500

    new destory function
```

```
commit c356b317773ddcf8b27b966e9802fe9706f8103a
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 12 13:55:02 2018 -0500

    syntax error on parsing setliteral

commit 9ef6049751ab6b008c452f943fe83528b9ef1fbd
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Dec 12 13:28:18 2018 -0500

    moving fns around

commit 8e57135a1fac91fb75116dcfe19195a27a4fa732
Merge: 1807306 324cba8
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Wed Dec 12 11:58:01 2018 -0500

    Merge pull request #36 from gammison/RKBranch

    Rk branch

commit 324cba83e476e521e40a83417e83a0c111dde65c (origin/RKBranch)
Merge: 2c6c085 1807306
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Wed Dec 12 11:57:48 2018 -0500

    Merge branch 'master' into RKBranch

commit 2c6c085978cf9693353d5b698b17df4d56aeb197
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Wed Dec 12 11:53:05 2018 -0500

    made changes for readability

commit 90e4b3afac462e1e2697e81b25a6567edef4c823
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Wed Dec 12 11:50:38 2018 -0500

    Added SSet and corresponding pretty printing function.

commit 2f688980a95a146a22ed089f801642d53d524f6a
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Wed Dec 12 11:50:07 2018 -0500

    uncommented remaining sections so we can actually print our syntax trees.

commit ba33fc1641cfbe89ec1e35bc932f9b46374777ed
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
```

```
Date:   Wed Dec 12 11:49:14 2018 -0500

    added set declaration and refactored the rest of the files to accept our
definition.

commit 3d1433c55906e9b6df57fbeebbf93c75a20eea59
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Wed Dec 12 11:48:16 2018 -0500

    Added set type, and corresponding pretty printing function for it.

commit 1807306883442642aad3e4674b75b3caaefecdf4
Merge: ac7b010 d149cf0
Author: Ryan <kryanchun@gmail.com>
Date:   Tue Dec 11 16:42:38 2018 -0500

    Merge pull request #35 from gammison/ryan-test

    set operations except complement

commit d149cf0dee9883c5b81e9efed21244a05d7d2542
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Tue Dec 11 16:31:27 2018 -0500

    set operations except complement

commit ac7b010093beb4464d4c31085ce83d02b195d109
Merge: e8cfeb3 a08d175
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Thu Dec 6 20:04:05 2018 -0500

    Merge pull request #34 from gammison/RKBranch

    Rk branch

commit a08d175a6630990c08ec4679fa297e7c448b786b
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Thu Dec 6 20:03:08 2018 -0500

    Added ArrLit to necessary files, and fixed compilation errors.

commit 20f21cf5cfc08c7f541c65d63d39884399bb6916
Merge: dc9847e e8cfeb3
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Thu Dec 6 19:40:06 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL into RKBranch
```

```
commit dc9847e97032a58e321c43455a40c7bdf9f34f7e
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Thu Dec 6 19:38:31 2018 -0500

    added arr in parser

commit e8cfeb3ab91686570ec66566df5a6e671efdcb75
Merge: 26aee0c c230c52
Author: Ryan <kryanchun@gmail.com>
Date:   Thu Dec 6 19:37:01 2018 -0500

    Merge pull request #33 from gammison/ryan-test

    Ryan test

commit c230c5258d1481f57452247b367ecbc271961f88
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Thu Dec 6 19:36:12 2018 -0500

    set.c functions

commit 026579b4ffed57d1fa8e79497fc13524649b8a2e
Merge: 08b154d 26aee0c
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Thu Dec 6 18:34:18 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL into ryan-test

commit 26aee0c114c2f9907e00e68f9f84b958188cc850
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Thu Dec 6 17:39:59 2018 -0500

    things progressing

commit 4d030a449cef20e9b673fb9db1823172ac6f5572
Merge: 5f3573a 0ee1ecd
Author: Ryan <34100806+CodeKoning@users.noreply.github.com>
Date:   Thu Dec 6 16:44:27 2018 -0500

    Merge pull request #32 from gammison/RKBranch

    adding c files for linked list

commit 0ee1ecdd06e6e3aec37667020fafec9d6421c726
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Thu Dec 6 16:42:41 2018 -0500

    adding c files for linked list
```

```
commit 08b154daaae04bf6215edbcd93d7163acf05a73b
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Mon Dec 3 20:52:04 2018 -0500

    if test + pattern-matching removal from ast.ml

commit 5f3573a03ff3ea9aa2b815406c2e024309d6ac15
Merge: b860942 a75b0f9
Author: CodeKoning <34100806+CodeKoning@users.noreply.github.com>
Date:   Sat Dec 1 23:45:31 2018 -0500

    Merge pull request #31 from gammison/RKBranch

    Rk branch

commit a75b0f9b45e6c36466cb75752664cd6b9e3a11bd
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Sat Dec 1 23:41:09 2018 -0500

    needed to change test file on master to make sure test-var2 passed.

commit 83309b5e9896eecdb83e909dc947c9c2c1a91145
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Sat Dec 1 23:40:29 2018 -0500

    fixing error from last merge

commit b860942e17b0586d09dba3c9ff2266c40a2c6791
Merge: e4ec690 2b97ef6
Author: CodeKoning <34100806+CodeKoning@users.noreply.github.com>
Date:   Sat Dec 1 23:28:27 2018 -0500

    Merge pull request #30 from gammison/RKBranch

    made necessary changes to various files to finally fix all the print
functions so all our tests pass.

commit 2b97ef632818f2848ebee65401516aac28698060
Merge: 35e69e1 e4ec690
Author: CodeKoning <34100806+CodeKoning@users.noreply.github.com>
Date:   Sat Dec 1 23:28:14 2018 -0500

    Merge branch 'master' into RKBranch

commit 35e69e1d4559f124a7721cf19531e2eb9044cfdd
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Sat Dec 1 23:22:03 2018 -0500
```

```
    make necessary changes to various files to finally fix all the print
functions so all our tests pass.

commit e4ec690cc02dc34386756e796902bb19997b9a4f
Merge: 4e90230 0d879cf
Author: CodeKoning <34100806+CodeKoning@users.noreply.github.com>
Date:   Sat Dec 1 17:17:24 2018 -0500

    Merge pull request #29 from gammison/RKBranch

    Rk branch

commit 0d879cfed11bc8b8013752088ca90ffaa02aaad5
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Sat Dec 1 17:15:12 2018 -0500

    Also made changes for boolit in scanner.

commit 87d1d93cc1856ddd19d4634ff19c5ea1741430a2
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Sat Dec 1 17:14:53 2018 -0500

    Boolit was coded incorrectly. I made necessary changes.

commit 98878546182dcdaf55933340df27e99c619004d5
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Sat Dec 1 17:14:13 2018 -0500

    changed boolean test to print result so it works with testall.

commit bfc09e19451ebc3b678744479dc31e3110a5b030
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Sat Dec 1 17:13:28 2018 -0500

    changed out file to 1 so test will pass, we will use 1 and 0 instead of
true and false

commit d0584cacf3e6ed0dab8ec95f9dcc3511a788fbd4
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Sat Dec 1 17:12:19 2018 -0500

    test file for boolean false

commit d8a0dfb893b244a767ae05d7045fd649beed1646
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Sat Dec 1 17:11:46 2018 -0500
```

```
    test file for boolean false

commit 4e90230ec23d1cc4298332c9ea59bd2e4024a568
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sat Dec 1 16:48:47 2018 -0500

    silly strings work now

commit 65de167ab4428c9cecf4a96a3190fe08355c1ad9
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sat Dec 1 16:09:19 2018 -0500

    forgot c

commit 304c8fb87e500db8dff997291f895d0eabde73fd
Merge: 2a81256 1c90b4c
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sat Dec 1 16:08:25 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit 2a81256c6b729f61410f51ca7cf26f11c4e643c0
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sat Dec 1 16:08:13 2018 -0500

    links working, not printing right

commit 1c90b4cbdc7fcb25428df056a6a815f870ec9da6
Merge: e49187d 3643471
Author: trishamaniar1 <trm2144@barnard.edu>
Date:   Fri Nov 30 21:59:58 2018 -0500

    Merge pull request #28 from gammison/TrishaBranch

    assignment order for addition to work

commit 3643471eba26f15ab1c90d2451455e6964596473
Author: Yesha Maniar <yeshamaniar@dyn-209-2-228-173.dyn.columbia.edu>
Date:   Fri Nov 30 21:58:04 2018 -0500

    assignment order for addition to work

commit e49187d1b19517dee9ba57ea67c6b0af84e7c217
Merge: 3c7a145 8f89156
Author: trishamaniar1 <trm2144@barnard.edu>
Date:   Fri Nov 30 19:38:37 2018 -0500

    Merge pull request #27 from gammison/TrishaBranch
```

```
    updates for pattern matching warnings

commit 8f891564194406af7082c6c058dd33f704bd314f
Author: Yesha Maniar <yeshamaniar@dyn-209-2-228-173.dyn.columbia.edu>
Date:   Fri Nov 30 19:36:25 2018 -0500

    updates for pattern matching warnings

commit 3c7a1457eb036bcdb70f80897f515c7bcc72308b
Merge: 1d33556 3a2dfe8
Author: CodeKoning <34100806+CodeKoning@users.noreply.github.com>
Date:   Fri Nov 30 17:34:22 2018 -0500

    Merge pull request #26 from gammison/RKBranch

    Rk branch

commit 3a2dfe8c139d9ae2887a8ed9fe030493cff861b3
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Fri Nov 30 17:30:50 2018 -0500

    Updated sast to reflect changes in codegen--had to add SWhile which had
been removed. I also Uncommented and integrated the pretty printing functions
for the sast, which had not been done yet. There are some pattern matchings yet
to be completed, but it compiles successfully.

commit 107373b524833dec43b8106c68e959d8c9014862
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Fri Nov 30 17:29:00 2018 -0500

    Made fixes and integrated conditional logic for SIf, SWhile, SFor, SForEach
etc. These all compile successfully.

commit 1d3355680cc336de546b79737685acdc9a61765c
Merge: 5c33ffb 77424ec
Author: trishamaniar1 <trm2144@barnard.edu>
Date:   Fri Nov 30 00:38:18 2018 -0500

    Merge pull request #25 from gammison/TrishaBranch

    Trisha branch

commit 77424ec8bd70664f16afb0d1bdfca5510deab940
Author: Yesha Maniar <yeshamaniar@Yeshas-Air.lan1>
Date:   Fri Nov 30 00:35:46 2018 -0500

    ast pattern matching
```

```
commit 2d475ce08cee987b09befda7b9bd00da62dd92a0
Merge: 5c33ffb a0552c9
Author: Yesha Maniar <yeshamaniar@Yeshas-Air.lan1>
Date:   Fri Nov 30 00:33:06 2018 -0500

    Merge branch 'TrishaBranch'

    updates

commit a0552c95b9c4367fd316a8e1c63807b8c74239b5
Merge: 7d39f88 5c33ffb
Author: Yesha Maniar <yeshamaniar@Yeshas-Air.lan1>
Date:   Fri Nov 30 00:27:26 2018 -0500

    updates

commit 7d39f88ec621b7ed6a3d5401a26fb281ab43b817
Author: Yesha Maniar <yeshamaniar@Yeshas-Air.lan1>
Date:   Thu Nov 29 23:55:07 2018 -0500

    pattern matching in ast

commit 5c33ffb933a85c4154ebe40d28a7dbfb0eb21aca
Merge: 9c4b86d 72af023
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Thu Nov 29 18:03:22 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit 9c4b86d23134360b887b190fd7f1cb1c04f03af5
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Thu Nov 29 18:03:11 2018 -0500

    last clean

commit 72af0235612d263de6a5912a49607d346baef145
Merge: 6537c56 c7d1168
Author: Ryan <kryanchun@gmail.com>
Date:   Thu Nov 29 17:01:53 2018 -0500

    Merge pull request #24 from gammison/ryan-test

    uncommented string_of_stmt

commit c7d116860f65b0a5a1ad3fdd6403abd9cbd4dcf4
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Thu Nov 29 17:01:19 2018 -0500
```

```
    uncommented string_of_stmt

commit 7674cde7e25b6e8c3cafe5340c9d8d8653de44d9
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Thu Nov 29 16:16:15 2018 -0500

    better clean

commit 6537c567203c51c1615d7074c27c7224f4bc51f9
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Thu Nov 29 16:08:19 2018 -0500

    some updates

commit fcb37360c5869c6b6052840ff2c987f7364b2c33
Merge: 115206a 6594f32
Author: CodeKoning <34100806+CodeKoning@users.noreply.github.com>
Date:   Wed Nov 28 22:58:56 2018 -0500

    Merge pull request #23 from gammison/RKBranch

    ALL TESTS COMPILEgit add tests/test-var2.sl git add tests/test-var2.s…

commit 6594f3212477f5e958b9e15e12992c8ea41d91b6
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Wed Nov 28 22:48:18 2018 -0500

    ALL TESTS COMPILEgit add tests/test-var2.sl git add tests/test-var2.sl
Just need to figure out what the problem is with testall.sh and why the
generated file aren't working properly. The problem with our char and string
had to do with the scanner actually. That's where we were supposed to tell the
compiler to look for quotes, not in the parser. Then I figured out how to add
strlit properly to our pretty printing function section of the ast so that
match error(69, 25) stops showing up.

commit 115206a27aa70de51812662d9d3cd82725cfe208
Merge: ac4507f 6e9ad54
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Nov 28 20:05:33 2018 -0500

    Merge pull request #22 from gammison/ryan-test

    added tests

commit 6e9ad5413ffdab99eadeeffd0952c3186956a8a9
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Nov 28 20:04:21 2018 -0500
```

```
    added tests

commit ac4507fa72d361b194b13f6ff55a26a5cb0daab0
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Wed Nov 28 20:01:35 2018 -0500

    Changed printbig to accept chars in semant and codegen.

commit 34b487130de1a3de181d50af13f45d7fa2587669
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Tue Nov 27 19:03:10 2018 -0500

    test-var1 compiles with sosl.native but for some reason test still differs.
GARRISON - I uncommented the section that was incorrectly generating exe files
in testall.sh. Your problem should be solved.

commit b0db567602623b0bb748dd9cfaf8595ca964704a
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Tue Nov 27 17:14:47 2018 -0500

    Successful linking of semant.ml and creation of sast through codegen and
sosl.ml. Everything compiles without errors, still lots of warnings though. We
need to complete the pattern matching.

commit 0d1548b6cfa8d1464aca2d1ab8545bf46431cb30
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Nov 25 12:34:52 2018 -0500

    assignment works test var1 outs a .s file, script for testing is wrong
though

commit bf9da4eee5ad23c98a732efcc97031fa5a64423f
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Nov 25 11:38:03 2018 -0500

    ridiculous file format fix, forgot to commit before

commit d8449c7c7709c2e45357ba7c3d2039238fa7ba6c
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Nov 25 11:31:36 2018 -0500

    cleanup file and gitignore

commit 81d4a4c5e3a8bd6cf0ba69f2511f9f22f6404298
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Thu Nov 22 01:07:58 2018 -0500
```

```
    fixed parser error, now fixing scanner for strings, makefile needs a fix
(it doesn't always remake when some files are edited)

commit 78e4cce549cad10d36d64d6f73de3c5d705fe775
Merge: d336bd7 55d338b
Author: Ryan <kryanchun@gmail.com>
Date:   Mon Nov 19 16:05:25 2018 -0500

    Merge pull request #21 from gammison/ryan-test

    set, void, some fixes

commit 55d338bbd61f4668f3d4f6ccc6e44954d66c6c31
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Mon Nov 19 16:00:14 2018 -0500

    set, void, some fixes

commit d336bd769c3b0512fe7434560945b83c17286c7b
Merge: be8207b bdf6585
Author: Ryan <kryanchun@gmail.com>
Date:   Sun Nov 18 18:51:10 2018 -0500

    Merge pull request #20 from gammison/ryan-test

    comments on future implementation | fix on test

commit bdf65854f5e5a0d1dce4363eb27ffab7cc802277
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Sun Nov 18 18:49:59 2018 -0500

    comments on future implementation | fix on test

commit be8207b7f69b426885d7b50677efb977e28f40fc
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Sat Nov 17 18:17:11 2018 -0500

    Added data types in codegen for String and Char as well as StrLit and
CharLit, made appropriate changes in all other files and fixed a couple other
errors I discovered. Language compiles successfully, testall still fails due to
a problem with the parser.

commit e98193ecded199c7492f547a5d1595486ca25ad2
Merge: 8e63495 74ba171
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Nov 14 23:51:00 2018 -0500

    Merge pull request #19 from gammison/ryan-test
```

```
    kinda works

commit 74ba171210d31b4d8f273772134634c2f5fe8857
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Nov 14 23:50:19 2018 -0500

    kinda works

commit 8e63495efa2e802c67756e8740a6cfb0188b8871
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Wed Nov 14 23:07:31 2018 -0500

    test stuff

commit 9f5fc2f28a744b09b19e4e5b229c9742c425c773
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Wed Nov 14 22:29:58 2018 -0500

    testall.sh update

commit 17d6818be6f0fd56913f4d3bb69c75f85f489d29
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Wed Nov 14 22:16:23 2018 -0500

    Hallelujah

commit ce030e4b8e8b0dec150ab5f9950064b9ed421f72
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Wed Nov 14 22:02:01 2018 -0500

    Finally!

commit 3feec5ad8ac542b7556f541d4055c9fde6c4c8a2
Merge: d6b1b2c b112b20
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Nov 14 21:55:00 2018 -0500

    Merge pull request #18 from gammison/ryan-test

    vdecl

commit b112b206ffed061b8a8fc6132f469792904e5c50
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Nov 14 21:54:11 2018 -0500

    vdecl
```

```
commit d6b1b2cd7f68e7b1785759a83f340238e1ecf070
Merge: f5f6736 810b98a
Author: Ryan <kryanchun@gmail.com>
Date:   Wed Nov 14 21:15:39 2018 -0500

    Merge pull request #17 from gammison/ryan-test

    type to dtype

commit 810b98af1eb66a4aebf8ab670555fd45353536b5
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Wed Nov 14 21:14:58 2018 -0500

    type to dtype

commit f5f6736446881673a1ab93a9dc6973a787db4ffe
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Nov 14 20:51:27 2018 -0500

    fix

commit 3444fe73e98ec41d62c37c398b9f6caeb8eac4ee
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Wed Nov 14 20:49:13 2018 -0500

    silly comment fixes and taking out sast

commit 9fa32e049d5d409e082653fba26a3e4170e12318
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Wed Nov 14 20:36:24 2018 -0500

    committing latest changes

commit f62bb69a2ef9be9008a2c55c9676b24f563ee297
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 20:58:43 2018 -0500

    closer

commit d53650a50c32c49f08ecfa20a9ddf89556fff782
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 20:17:34 2018 -0500

    so close

commit bbe45dd415aa28bbb1921609a6ac1f3bb3e6400a
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Tue Nov 13 20:06:11 2018 -0500
```

```
    removed erroneous 'in' statement

commit 70ceb0eaf9ae8efe1df5a06e6dabbae5a69e5df4
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 19:59:03 2018 -0500

    bad types

commit 3aac0ba039c818049d56be0f601107cac33b1768
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 19:55:20 2018 -0500

    pretty print get rid of again

commit f736646751a1eca3835c8df627ae4195e502b505
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 19:52:10 2018 -0500

    in fix from comments

commit cc4430611b50fbe974ba1253bf47cd19ac365edc
Merge: 691dc46 c6421d9
Author: Ryan <kryanchun@gmail.com>
Date:   Tue Nov 13 19:49:48 2018 -0500

    Merge pull request #16 from gammison/ryan-test

    no in

commit c6421d96b3e6bd4927ddb14046f12bdf3ee11e36
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Tue Nov 13 19:49:18 2018 -0500

    no in

commit 691dc46cb8642d9c4ebd3842c22a1edc8094b30e
Merge: 5c930be a81165a
Author: Ryan <kryanchun@gmail.com>
Date:   Tue Nov 13 19:48:04 2018 -0500

    Merge pull request #15 from gammison/ryan-test

    Ryan test

commit a81165a228c55b0dec42dd721a4b4f98a6db5d59
Merge: d58ad6f 8004c28
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
```

```
Date:   Tue Nov 13 19:47:04 2018 -0500

    codegen first draft

commit 5c930bedf68ac167b43f38ae34edcc4ece8f6fb3
Merge: 407bbfd 8004c28
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 19:47:02 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit 407bbfdd0b0f61026a80ba946c4d6ebe207d8e93
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 19:46:50 2018 -0500

    string variable add

commit d58ad6fbca1737dd78a80a12c9d13cc1716430cb
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Tue Nov 13 19:29:31 2018 -0500

    codegen types

commit 8004c281918ee274f98636257a199deea9ce9d06
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Tue Nov 13 19:28:36 2018 -0500

    commented version of codegen.ml

commit d7b39256632eff44d78092e03bfd22ee98c59b52
Merge: 5500711 1f68340
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Tue Nov 13 19:17:39 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit 1f683405571b1145b7b18595bf2be7f018499ba7
Merge: c0dc1f4 a056c15
Author: Ryan <kryanchun@gmail.com>
Date:   Tue Nov 13 19:14:41 2018 -0500

    Merge pull request #14 from gammison/ryan-test

    added codegen.ml

commit a056c157c20658ea2872a6866d7bddf586179c03
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Tue Nov 13 19:10:52 2018 -0500
```

```
    added codegen.ml

commit 5500711d82abee096ab6df07581976e2d69f5527
Merge: e24885f c0dc1f4
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Tue Nov 13 18:55:12 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit c0dc1f42fa77a734f73f3978961912a9b974357a
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 18:54:45 2018 -0500

    ocaml naming schemes are bad

commit d9840a36c0526d794cd40c9655e0049bf369c713
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 18:53:04 2018 -0500

    file name fixes, now everything builds except codegen

commit e24885f28ce03d45e846908fef19c55b183786c4
Merge: 2d77fb5 2d8afc1
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Tue Nov 13 18:45:00 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit 2d8afc12e23d39d50faadbd26b4fdeb77fc71d0f
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 18:42:19 2018 -0500

    parser and scanner build sucessfully, sast can stay blank for now

commit fa95cd8547c67329d1d6e155c932a26d84299d27
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 18:12:51 2018 -0500

    ocamlbuild

commit f38bcef90500f634a871cac23e12b7a681d6c7ce
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 18:12:38 2018 -0500

    we changed ast to have local vdecls, so need to change that in the parser,
also added ocamlbuild tags file
```

```
commit 6a77f40d85d2beaf7fdfe29d3cc71ae682a8b262
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 17:53:49 2018 -0500

    all non printing errors resolved

commit ea2179f0b1c29232db657989a92fa76a0bf8302b
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 17:45:55 2018 -0500

    all pretty print shelved, more meaningful erors present

commit df6e72534be28e26e14d7d18835a52b4d8dfce23
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 17:39:30 2018 -0500

    more syntax error fixes

commit 2860af43487fd868a4cfcc640e422324c54a5b6a
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 17:31:34 2018 -0500

    break type fixed, more ast errors

commit 2d77fb5a9d375b5bbcc76e3ccd12cb1c4c5598a9
Merge: 7b81855 1debf70
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Tue Nov 13 17:13:26 2018 -0500

    Merge branch 'master' of https://github.com/gammison/SOSL

commit 1debf700acbc636b44f1c4ce8de309bb398b4774
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 17:12:17 2018 -0500

    syntax error in ast fixed, missed capitalization, spell fix on sosl.ml

commit dc379ca6351a31c6282b6a1078ffad2325b768af
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 17:04:56 2018 -0500

    bind fixes, naming change of sosl_parser.mly

commit 1d15d948c171fed9c955d64b4443c969b27d7ffc
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Tue Nov 13 16:55:01 2018 -0500

    rename, wierd syntax
```

```
commit 7b8185517b80d5ae675ca65303b83c6fc77fd6a1
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Tue Nov 13 16:53:45 2018 -0500

    exchanged microc's testall file with ours

commit 724d36f567d3d36daf8d8d36103a177208ba1c67
Author: CodeKoning <rjk2153@P50s-1.allurehome.com>
Date:   Thu Nov 8 19:47:19 2018 -0500

    makefile header fix and sosl.ml comments

commit 6f13acceffff1dbe9cd2d07c43a3e6fadc55eeb1
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Thu Nov 8 19:06:46 2018 -0500

    naming update

commit e9a7e165335ba7c775a23b625af1afe933fabcb8
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Thu Nov 8 18:53:39 2018 -0500

    skeleton compiler created

commit e387b20a6ff89ee943e9e7f54ac51792c3a61da7
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Thu Nov 8 18:39:33 2018 -0500

    printbig old test somehow was in the directory

commit aa8dcd5b2be12f229377b1bf41ad0ece5bc60ee3
Merge: 2d66506 fe96ac2
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Thu Nov 8 18:36:15 2018 -0500

    resolved Makefile merge conflict

commit 2d66506f771708e8e72b86d7eeb19a006aeac3d5
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Thu Nov 8 18:34:45 2018 -0500

    skeleton makefile for needed tests and ocaml compile

commit fe96ac2eff10bc6a62be86a79a18beedfe416697
Merge: a4cefdf e0cb59c
Author: Ryan <kryanchun@gmail.com>
Date:   Thu Nov 1 19:01:11 2018 -0400
```

```
    Merge pull request #13 from gammison/ryan-test

    Ryan test

commit e0cb59c24792c3b1dc961b04c8af042c00c036b0
Merge: 28f0e54 96d311b
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Thu Nov 1 18:53:55 2018 -0400

    Merge branch 'ryan-test' of https://github.com/gammison/SOSL into ryan-test

commit 28f0e545f9b3e1a87938429217e2c36c9d408d2e
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Thu Nov 1 18:50:17 2018 -0400

    SET/NOT error on parser fixed; A test case is added; printbig.c semant.ml
testall.sh Makefile

commit a4cefdf7c77fedf806539fa20b2b32ea14d7d2e6
Merge: 9d09ee4 96d311b
Author: Garrison Grogan <gammison@users.noreply.github.com>
Date:   Mon Oct 22 22:21:27 2018 -0400

    Merge pull request #12 from gammison/ryan-test

    Ryan test good

commit 96d311b62cf1fbf9615c8bcb234482b8c97ae956
Merge: 0d9a9c8 9d09ee4
Author: Garrison Grogan <gammison@users.noreply.github.com>
Date:   Mon Oct 22 22:21:14 2018 -0400

    Merge branch 'master' into ryan-test

commit 0d9a9c8788fcb9cde44c695411b42079b7883a30
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Mon Oct 22 17:27:25 2018 -0400

    Parser Edit / AST Edit / More formatting

commit 0d2044558cb64fc082bfaee646f9ca22aeafb8ae
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Mon Oct 22 16:59:17 2018 -0400

    Typo fix on parser.mly

commit af9becf25364bed08fb265c6425a0dade53e7dbc
```

```
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Mon Oct 22 16:57:06 2018 -0400

    AST Change: Formatting; Added some comments for future discussion

commit 9d09ee4455cf6d387eaca8e67ac8220627d3ccd2
Author: Garrison Grogan <gammison@users.noreply.github.com>
Date:   Thu Oct 18 19:17:23 2018 -0400

    Update README.md

commit 4598d215d0a8c3bf7bd0f726d09ef5e0cf1f5577
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Oct 14 23:46:37 2018 -0400

    README

commit 4b8872bcbd32727cb4ededc6c12a08d58a1c437c
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Oct 14 23:45:30 2018 -0400

    README

commit f13ad37455874f95a480032f8b8e1b0513198ea2
Merge: 5318ecc 6dac8b6
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Oct 14 23:44:19 2018 -0400

    Merge branch 'master' of https://github.com/gammison/SOSL

commit 5318ecc810a4e0c279adb020a57546d9c567396c
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Oct 14 23:44:03 2018 -0400

    more stuff, readme

commit 6dac8b60158467a6381198af76a4239db1852aeb
Merge: 12f34d0 df698e2
Author: Ryan <kryanchun@gmail.com>
Date:   Sun Oct 14 23:41:46 2018 -0400

    Merge pull request #10 from gammison/ryan-test

    First Draft

commit df698e281586f89226a243c7b0adab255057bdd6
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Sun Oct 14 23:41:14 2018 -0400
```

```
    First Draft

commit 12f34d02c8b631fca8c67d844dd97cb27fd40d65
Merge: 436d07c 46900ca
Author: Ryan <kryanchun@gmail.com>
Date:   Sun Oct 14 23:07:08 2018 -0400

    Merge pull request #9 from gammison/ryan-test

    fparams

commit 46900caf33c2f9d14b6314728a3055c3ece431b1
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Sun Oct 14 23:06:36 2018 -0400

    fparams

commit 436d07c5229ebe6b696289210257d70131e50bf0
Merge: ef4d7b0 c31c9d0
Author: Ryan <kryanchun@gmail.com>
Date:   Sun Oct 14 22:57:56 2018 -0400

    Merge pull request #8 from gammison/ryan-test

    Ryan test

commit c31c9d00e360f9bb6cd8966d5beb8a56e6ac6f8d
Merge: 37d3b84 ef4d7b0
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Sun Oct 14 22:56:58 2018 -0400

    almost done

commit 37d3b840da3113b869604875d4a3524b8e99b295
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Sun Oct 14 22:52:53 2018 -0400

    Temp commit

commit ef4d7b09d3416ee144d9b4489759feea6d0a91f0
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Oct 14 22:52:21 2018 -0400

    got rid of extraneous for def with block explicit, shouldn't matter

commit fbfcb4d53908ac5c929263ead604b50472e60d87
Author: Garrison Grogan <garrison.grogan@columbia.edu>
```

```
Date:   Sun Oct 14 22:50:59 2018 -0400

    typo fix

commit ffbdbda3c44c2b0bff6a1f829a153c22b4ef008b
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Oct 14 22:44:47 2018 -0400

    uop to unop for consistency

commit 36e351e1952640a95a24b72c78e927241d5918d3
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Oct 14 22:38:44 2018 -0400

    ast done maybe

commit 95c9ce8ea7ec8928dade73e886e5ef209fa7449e
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Oct 14 22:21:18 2018 -0400

    more pretty printing

commit 8865983f51e6ee1956b1b20f1786ff90999b88b4
Merge: 4d1b7c5 c934709
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Oct 14 22:19:58 2018 -0400

    Merge branch 'master' of https://github.com/gammison/SOSL

commit 4d1b7c5bea7226c05b13a7521ab3e695075b1e11
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Oct 14 22:19:42 2018 -0400

    pretty printing

commit c934709f12b50b90b1cf7c852b74f634bd33d8e2
Merge: 5207db1 4d753ff
Author: Ryan <kryanchun@gmail.com>
Date:   Sun Oct 14 22:10:49 2018 -0400

    Merge pull request #7 from gammison/ryan-test

    Some change on parser.mly

commit 4d753ff316c54ae18645632912e209950cbc0fe5
commit 4d753ff316c54ae18645632912e209950cbc0fe5
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Sun Oct 14 22:10:14 2018 -0400
```

```
    Some change on parser.mly

commit 5207db17cc2d33983b614996170cce5f2a26907b
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Oct 14 21:49:03 2018 -0400

    fdecl done, doing pretty printing

commit fc5b097f4fa9a8dd4878f1a576387f5b0f487266
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Oct 14 21:04:06 2018 -0400

    more confusion

commit 0478835e5ac4bfe6df6d958b8ad9bf71f42b230f
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Sun Oct 14 20:58:03 2018 -0400

    expr types, stmt types

commit 21686fc216ab40291ca7968dae8c7334b5e2d457
Merge: f0be3d8 444e908
Author: Ryan <kryanchun@gmail.com>
Date:   Sun Oct 14 20:10:43 2018 -0400

    Merge pull request #6 from gammison/ryan-test

    scanner first draft

commit 444e908cdd9f58883cdd591853778f9a866453dc
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Sun Oct 14 20:06:34 2018 -0400

    scanner first draft

commit f0be3d84ab45bd0030f19c2d75295596430c09ab
Merge: 8e8016b 14ac7bd
Author: CodeKoning <34100806+CodeKoning@users.noreply.github.com>
Date:   Sun Oct 14 20:01:13 2018 -0400

    Merge pull request #5 from gammison/RKBranch

    Added the basic operators etc. to the ast file

commit 14ac7bddabd0b78db794857682b568221ffc8063
Author: U-P50S-1\Ryan.Koning <P50S-1+Ryan.Koning@P50s-1.allurehome.com>
Date:   Sun Oct 14 19:58:22 2018 -0400
```

```
    Added the basic operators etc. to the ast file

commit 8e8016b7eb9c4aa7788c8b700665344953139254
Merge: 050b653 191a874
Author: Ryan <kryanchun@gmail.com>
Date:   Sun Oct 14 19:54:04 2018 -0400

    Merge pull request #4 from gammison/ryan-test

    scanner update

commit 191a8742ad6941041154a2097ba91ab698cdb471
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Sun Oct 14 19:53:21 2018 -0400

    scanner update

commit 050b653c25eb7e1c03e40b59fe4b0854acad029d
Merge: 6fe85c0 109bf40
Author: Ryan <kryanchun@gmail.com>
Date:   Thu Oct 11 19:24:12 2018 -0400

    Merge pull request #3 from gammison/ryan-test

    replaced parser.mli with parser.mly

commit 109bf4090e9e7f500f335d031151b52a1326cf41
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Thu Oct 11 19:23:14 2018 -0400

    replaced parser.mli with parser.mly

commit 6fe85c090de6f50395883e2252956951a5a0ad21
Merge: 8dcf831 a16353f
Author: Ryan <kryanchun@gmail.com>
Date:   Thu Oct 11 19:15:43 2018 -0400

    Merge pull request #2 from gammison/ryan-test

    Added ast.mli; added set types in ast file

commit a16353fb49878b68d5aab164df3dd5dc72d10ace
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Thu Oct 11 19:14:55 2018 -0400

    Added ast.mli; added set types in ast file
```

```
commit 8dcf831091e7afa6e61f1e9a013e3e2f3a3b77cd
Merge: 7eb606e 97e8162
Author: Ryan <kryanchun@gmail.com>
Date:   Mon Oct 8 21:34:08 2018 -0400

    Merge pull request #1 from gammison/ryan-test

    Added parser.mli and scanner.mll

commit 97e816254a71e6f2bf2d1fba5ec2e7a7def2f97a
Author: Ryan Chun <ryan.chun@columbiaspectator.com>
Date:   Mon Oct 8 21:32:45 2018 -0400

    Added parser.mli and scanner.mll

commit 7eb606eb8703c9296a3b675cc335be2115384e9f
Author: Garrison Grogan <garrison.grogan@columbia.edu>
Date:   Thu Sep 27 18:45:15 2018 -0400

    first commit, added proposal tex document
(END)
```

Scanner.mll

```
{open Parser }
(* nice string parser from
https://v1.realworldocaml.org/v1/en/html/parsing-with-ocamllex-and-menhir.html *)
```

```
(*let read_string(buf) =
      parse
        | '"'       { STR_LIT (Buffer.contents buf) }
        | '\\' '/'  { Buffer.add_char buf '/'; read_string buf lexbuf }
        | '\\' '\\' { Buffer.add_char buf '\\'; read_string buf lexbuf }
        | '\\' 'b'  { Buffer.add_char buf '\b'; read_string buf lexbuf }
        | '\\' 'f'  { Buffer.add_char buf '\012'; read_string buf lexbuf }
        | '\\' 'n'  { Buffer.add_char buf '\n'; read_string buf lexbuf }
        | '\\' 'r'  { Buffer.add_char buf '\r'; read_string buf lexbuf }
        | '\\' 't'  { Buffer.add_char buf '\t'; read_string buf lexbuf }
        | [^ '"' '\\']+
        { Buffer.add_string buf (Lexing.lexeme lexbuf);
               read_string buf lexbuf
        }
        | _ { raise (Failure ("Illegal string character: " ^ Lexing.lexeme lexbuf)) }
        | eof { raise (Failure ("String is not terminated")) }
        | eof { EOF }
*)

(* parse_set, possibly typecheck here, or that should be runtime error *)
rule token = parse
(* Whitespace*)
[' ' '\t' '\r' '\n'] { token lexbuf }

(* Delimiters *)
| '(' { LPAREN }
| ')' { RPAREN }
| '[' { LBRACKET }
| ']' { RBRACKET }
| '{' { LBRACE }
| '}' { RBRACE }
| '|' { CARD }
| ';' { SEMI }
| ',' { COMMA }
| ':' { COLON }

(* Arithmetic Operators *)
| '+' { PLUS }
| '-' { MINUS }
| '*' { TIMES }
| '/' { DIVIDE }
| '=' { ASSIGN }
| '%' { MOD }

(* Data Type *)
| "int"     { INT }
| "char"    { CHAR }
| "boolean" { BOOL }
| "void"    { VOID }
| "set"     { SET } (* handle typing need solution *)
| "string"  { STRING }

(* Boolean Type *)
| "true"     { BLIT(true) }
| "false"    { BLIT(false) }

(* Set Operators *)
| ":u" { UNION }
| ":n" { INTSEC }
| ":i" { ELEM }
| ":c" { COMP }
```

```
(* Relational Operators *)
| '<'   { LT }
| "<="  { LEQ }
| '>'   { GT }
| ">="  { GEQ }
| "=="  { EQ }
| "!="  { NEQ }
| "AND" { AND }
| "OR"  { OR }
| "!"   { NOT }

(* Control Flow *)
| "if"  { IF }
| "else" { ELSE }
| "for"  { FOR }
| "forEach" { FOREACH }
| "in" { IN }
| "return" { RETURN }
| "break" { BREAK }

(* Literals and EOF *)
| ['0'-'9']+ as lxm { NUM_LIT(int_of_string lxm)}
| ['a'-'z' 'A'-'Z']['a'-'z' 'A'-'Z' '0'-'9' '_']*+ as lxm { VARIABLE(lxm) }

(*| '"' { read_string (Buffer.create 17) lexbuf } *)
| '"' (([^ '"'] | "\\\"")* as strlit) '"' { STR_LIT(strlit) }
| '''([' '-'!' '#'-'[' ']'-'~' ]|['0'-'9'])''' as lxm {CHAR_LIT( String.get lxm 1)}
| eof { EOF }
| _ { raise (Failure ("Unexpected char: " ^ Lexing.lexeme lexbuf)) }
```

Parser.mly

```
%{ open Ast %}

/* Delimiters */
%token LPAREN RPAREN LBRACKET RBRACKET LBRACE RBRACE
%token CARD SEMI COMMA COLON

/* Arithmetic Operators */
%token PLUS MINUS TIMES DIVIDE ASSIGN MOD

/* Data Types */
%token INT CHAR BOOL VOID STRING SET ARRAY

/* Boolean Values */
%token <bool> BLIT

/* Boolean Values */
%token UNION INTSEC ELEM COMP

/* Relational Operators */
%token LT LEQ GT GEQ EQ NEQ AND OR NOT

/* Control Flow */
%token IF ELSE FOR FOREACH IN RETURN BREAK

/* Literals, Identifiers, EOF */
%token <int> NUM_LIT /* we are only doing ints rn, if add floats will need to make an AST.num
type that handles both */
%token <char> CHAR_LIT
```

```
%token <string> VARIABLE
%token <string> STR_LIT
%token EOF

/* Order and Associativity */
%nonassoc NOELSE
%nonassoc ELSE
%right ASSIGN
%left PLUS MINUS TIMES DIVIDE MOD
%left UNION INTSEC ELEM COMP
%left LT LEQ GT GEQ EQ NEQ NSEQ
%left OR AND
%right NOT

%start program
%type <Ast.program> program
%%

program: decls EOF { $1 }

decls: /* nothing */   { [], []                 }
     | decls vdecl    { ($2 :: fst $1), snd $1 }
     | decls fdecls    { fst $1, ($2 :: snd $1) }

fdecls: dtype VARIABLE LPAREN params_opt RPAREN LBRACE vdecls stmts RBRACE {
          {
            ftype = $1;
            fname = $2;
            parameters = $4;
            locals = List.rev $7;
            body = List.rev $8
          }
        }

params_opt: /* nothing */              { []             }
          | params                     { List.rev $1 }


params:
        dtype VARIABLE               { [($1,$2)]      }
      | params COMMA dtype VARIABLE  { ($3, $4) :: $1 }


dtype: INT                                              { Int }
     | BOOL                                             { Boolean }
     | CHAR                                             { Char }
     | SET COLON LBRACE stypes RBRACE COLON     { Set($4)}
     | STRING                                           { String }
     | VOID                                             { Void }

stypes: INT                                            { Int }
     | BOOL                                             { Boolean }
     | CHAR                                             { Char }
     | STRING                                           { String }
     | SET COLON LBRACE stypes RBRACE COLON     { Set($4) }


vdecls: /* nothing */   { [] }
      | vdecls vdecl    { $2 :: $1 }

vdecl: dtype VARIABLE SEMI { ($1, $2) }
```

```
S_LIT: COLON LBRACE lit_list_opt RBRACE COLON { SetLit($3) }

lit_list_opt:
        {[]}
       | lit_list { List.rev $1 }

lit_list:
         expr                 { [$1]  }
       | lit_list COMMA expr     { $3 :: $1 }

stmts: /* nothing */    { [] }
     | stmts stmt       { $2 :: $1 }

stmt:
    expr SEMI                                                       { Expr $1 }
  | BREAK SEMI                                                      { Break }
  | RETURN expr SEMI                                                { Return $2 } /* Consider
optional expressions */
  | LBRACE stmts RBRACE                                             { 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 SEMI expr SEMI expr RPAREN stmt                 { For($3, $5, $7, $9) } /*
Consider optional expressions */
  | FOREACH LPAREN expr IN expr RPAREN stmt                         { ForEach($3, $5, $7) }

literals:
    NUM_LIT                                                         { IntLit($1) }
  | CHAR_LIT                                                      { CharLit($1) }
  | STR_LIT                                                     { StrLit($1) }
  | BLIT                                                          { BoolLit($1) }
  | S_LIT                                                       { $1 }

expr:
    literals                                                        { $1 }
  | VARIABLE                                                        { Variable($1) }
  | expr PLUS expr                                                  { Binop($1, Add, $3) }
  | expr MINUS expr                                                 { Binop($1, Sub, $3) }
  | expr TIMES expr                                                 { Binop($1, Mul, $3) }
  | expr DIVIDE expr                                                { Binop($1, Div, $3) }
  | expr MOD expr                                                   { Binop($1, Mod, $3) }
  | expr EQ expr                                                    { Binop($1, Eq, $3) }
  | expr NEQ expr                                                   { Binop($1, Neq, $3) }
  | expr LT expr                                                    { Binop($1, Less, $3) }
  | expr LEQ expr                                                   { Binop($1, LessEq, $3) }
  | expr GT expr                                                    { Binop($1, More, $3) }
  | expr GEQ expr                                                   { Binop($1, MoreEq, $3) }
  | expr AND expr                                                   { Binop($1, And, $3) }
  | expr OR expr                                                    { Binop($1, Or, $3) }
  | expr UNION expr                                                 { Binop($1, Union, $3) }
  | expr INTSEC expr                                                { Binop($1, Isec, $3) }
  | expr COMP expr                                                  { Binop($1, Comp, $3) }
  | expr ELEM expr                                                  { Binop($1, Elof, $3) }
  /*| NOT expr                                                        { Unop(Not, $2) }*/
  | VARIABLE LPAREN fparams_opt RPAREN                               { Call($1, $3) } /* consider
using optional args */
  | LPAREN expr RPAREN                                              { $2 }
  | VARIABLE ASSIGN expr                                            { Assign($1, $3) }

fparams_opt:
```

```
     /* nothing */{ [] }
    |fparams { List.rev $1 }

fparams:
    expr                        { [$1] }
  | fparams COMMA expr          { $3 :: $1 }
```

Ast.ml

```
type op = Add | Sub | Mul | Div | Mod | Eq
          | Union | Isec | Elof | Comp | Neq
          | And | Or | LessEq | MoreEq
          | More | Less
type unop = Not (* cardinality is a delim like () *)
type elmTypes = Int | Boolean | Char | String | Void | Set of elmTypes
type bind = elmTypes * string
type expr =
          | IntLit              of int
          | CharLit             of char
          | BoolLit             of bool
          | StrLit              of string
           | SetLit           of expr list
          | Variable            of string
          (*| SetAccess           of string * expr
          | ArrLit           of expr list
          | ArrayAccess         of string * expr *)
          | Call                of string * expr list
          | Binop               of expr * op * expr
          | Unop                of unop * expr (* if we do string types or array slicing,
their syntactic sugar needs to go here *)
          | Noexpr
          | Assign               of string * expr

(* and arr = ArrLit of expr *)

and stmt = Block                of stmt list
         | Expr                 of expr
         | If                   of expr * stmt * stmt
         | For                  of expr * expr * expr * stmt
         | ForEach              of expr * expr * stmt
         | Return               of expr
         | Break
          | While                   of expr * stmt
         (* | SetElementAssign     of string * expr * expr
         | ArrayElementAssign   of string * expr * expr *)


type fdecl = {  (* function declaration *)
                ftype : elmTypes;
                fname : string;
                parameters: bind list;
                locals: bind list;
                body : stmt list;
            }



type program = bind list * fdecl list (* a valid program is some globals and function
declarations *)

(* add pretty printing for the AST ie Add -> "+" *)
```

```
let string_of_op = function
    Add         -> "+"
    | Sub       -> "-"
    | Mul       -> "*"
    | Div       -> "/"
    | Mod       -> "%"
    | Eq        -> "=="
    | Neq       -> "!="
    | Less      -> "<"
    | More      -> ">"
    | LessEq    -> "<="
    | MoreEq    -> ">="
    | Union     -> ":u"
    | Isec      -> ":n"
    | Comp      -> ":c"
    | Elof      -> ":i"
    | And       -> "AND"
    | Or        -> "OR"

let string_of_unop = function
    Not -> "!"


let rec string_of_expr = function
      IntLit(l)             -> string_of_int l
    | CharLit(c)                -> Char.escaped c
    | StrLit(strlit)        -> strlit
    | BoolLit(true)         -> "true"
    | BoolLit(false)        -> "false"
    | Assign(s, e)          -> s ^ " = " ^ string_of_expr e ^ ";\n"
    | SetLit(s)             -> "Set:{" ^ String.concat "" (List.map string_of_expr s) ^ "}:\n"
    | Variable(s)           -> s
    | Binop(e1, o, e2)      -> string_of_expr e1 ^ " " ^ string_of_op o ^ " " ^ string_of_expr
e2
    | Unop(o, e)            -> string_of_unop o ^ string_of_expr e
    | Call(f, e1)           -> f ^ "(" ^ String.concat", "(List.map string_of_expr e1)^ ")"
    | Noexpr                -> "noexpr"
    (* | SetAccess (s,e)       -> s ^ "{" ^ string_of_expr e ^ "}"
    | ArrayAccess (s,e)     -> s ^ "[" ^ string_of_expr e ^ "]" *)

let rec string_of_stmt = function
    Block(stmts)                -> "Block{\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,s1,s2)               -> "if(" ^ string_of_expr e ^ ")\n" ^ string_of_stmt s1 ^
"else\n" ^ string_of_stmt s2 (*if else*)
    | For(e1,e2,e3,s)           -> "for(" ^ string_of_expr e1 ^ "; " ^ string_of_expr e2 ^ ";
" ^ string_of_expr e3 ^ ")\n" ^ string_of_stmt s
    | ForEach(e1,e2,s)          -> "foreach(" ^ string_of_expr e1 ^ " in " ^ string_of_expr e2
^ ")\n" ^ string_of_stmt s
    | While(e, s)               -> "while (" ^ string_of_expr e ^ ") " ^ string_of_stmt s
    (*| SetElementAssign(s,e1,e2)     -> s ^ "{" ^ string_of_expr e1 ^"} = " ^ string_of_expr
e2 ^";\n"
    | ArrayElementAssign(a,e1,e2)   -> a ^"[" ^ string_of_expr e1 ^"] = " ^ string_of_expr e2
^";\n"*)
    | Break                     -> "break;\n"

let rec string_of_typ = function
        Int                 -> "int"
      | String              -> "string"
```

```
      | Char                -> "char"
      | Boolean             -> "bool"
      | Void                -> "void"
      | Set(l)                      -> "set:{" ^ string_of_typ l ^ "}:"


let string_of_set (e) = "Set{" ^ String.concat "" (List.map string_of_expr e) ^ "}\n"

let string_of_bind (t, id) = string_of_typ t ^ " " ^ id ^ ";\n"

let string_of_vinit (s, e) = s ^ " = " ^ string_of_expr e ^ ";\n"

let string_of_fdecl fdecl =
    string_of_typ fdecl.ftype ^ " " ^ fdecl.fname ^ "(" ^ String.concat "," (List.map snd
fdecl.parameters) ^
    ")\n{\n" ^ String.concat "" (List.map string_of_stmt fdecl.body) ^ "}\n"

let string_of_program (vars, funcs) =
    String.concat "" (List.map string_of_bind vars) ^ "\n" ^ String.concat "\n" (List.map
string_of_fdecl funcs) ^
    String.concat ";\n" (List.map string_of_fdecl funcs)
```

Semant.ml

```
(* Semantic checking for the MicroC compiler *)

open Ast
open Sast

module StringMap = Map.Make(String)

(* Semantic checking of the AST. Returns an SAST if successful,
   throws an exception if something is wrong.

   Check each global variable, then check each function *)

let check (globals, functions) =

  (* Verify a list of bindings has no void types or duplicate names *)
  let check_binds (kind : string) (binds : bind list) =
    List.iter (function
        (Void, b) -> raise (Failure ("illegal void " ^ kind ^ " " ^ b))
      | _ -> ()) binds;
    let rec dups = function
        [] -> ()
      | ((_,n1) :: (_,n2) :: _) when n1 = n2 ->
          raise (Failure ("duplicate " ^ kind ^ " " ^ n1))
      | _ :: t -> dups t
    in dups (List.sort (fun (_,a) (_,b) -> compare a b) binds)
  in

  (**** Check global variables ****)

  check_binds "global" globals;

  (**** Check functions ****)

  (* Collect function declarations for built-in functions: no bodies *)
```

```
let built_in_decls =
  let add_bind map (name, ty) = StringMap.add name {
    ftype = ty;
    fname = name;
    parameters = [(ty, "x")];
    locals = []; body = [] } map
  in List.fold_left add_bind StringMap.empty [ ("print", Int);
                                               ("printb", Boolean);
                                               ("printf", String);
                                              ("prints", String);
                                                 ("printc", Char);
                                                 ("print_set", Set(Int));]

in

(* Add function name to symbol table *)
let add_func map fd =
  let built_in_err = "function " ^ fd.fname ^ " may not be defined"
  and dup_err = "duplicate function " ^ fd.fname
  and make_err er = raise (Failure er)
  and n = fd.fname (* Name of the function *)
  in match fd with (* No duplicate functions or redefinitions of built-ins *)
       _ when StringMap.mem n built_in_decls -> make_err built_in_err
     | _ when StringMap.mem n map -> make_err dup_err
     | _ ->  StringMap.add n fd map
in

(* Collect all function names into one symbol table *)
let function_decls = List.fold_left add_func built_in_decls functions
in

(* Return a function from our symbol table *)
let find_func s =
  try StringMap.find s function_decls
  with Not_found -> raise (Failure ("unrecognized function " ^ s))
in

let _ = find_func "main" in (* Ensure "main" is defined *)

let check_function func =
  (* Make sure no formals or locals are void or duplicates *)
  check_binds "parameter" func.parameters;
  check_binds "local" func.locals;

  (* Raise an exception if 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 (Failure err)
  in

  (* Build local symbol table of variables for this function *)
  let symbols = List.fold_left (fun m (ty, name) -> StringMap.add name ty m)
                    StringMap.empty (globals @ func.parameters @ func.locals )
  in

  (* Return a variable from our local symbol table *)
  let type_of_identifier s =
    try StringMap.find s symbols
    with Not_found -> raise (Failure ("undeclared identifier " ^ s))
  in
```

```
(* Return a semantically-checked expression, i.e., with a type *)
let rec expr = function
    IntLit  l  -> (Int, SIntLit l)
  | CharLit l  -> (Char, SCharLit l)
  | BoolLit l  -> (Boolean, SBoolLit l)
  | StrLit l   -> (String, SStrLit l)
  | SetLit sl   ->
    ( match sl with
    | [] -> (Set(Void), SSetLit [])
    | hd :: _ ->
      let (t', _) = expr hd in
        let expr_to_sexpr ex =
          let (t1, e1) = expr ex in
            match t1 with
            | Int        -> (t1, e1)
            | Char       -> (t1, e1)
            | Boolean    -> (t1, e1)
            | String     -> (t1, e1)
            | Void       -> (t1, e1)
            | Set(_)     -> (t1, e1)
            in
          let ssl = List.map expr_to_sexpr sl in (Set(t'), SSetLit ssl))
  | Noexpr     -> (Void, SNoexpr)
  | Variable s -> (type_of_identifier s, SVariable s)
  | Assign(var, e) as ex ->
      let lt = type_of_identifier var
        and (rt, e') = expr e in
      let err = "illegal assignment " ^ string_of_typ lt ^ " = " ^
        string_of_typ rt ^ " in " ^ string_of_expr ex
      in (check_assign lt rt err,
            SAssign(var, (rt, e')))
  | Unop(op, e) as ex ->
      let (t, e') = expr e in
      let ty = match op with
        Not when t = Boolean -> Boolean
        | _ -> raise (Failure ("illegal unary operator " ^
                               string_of_unop op ^ string_of_typ t ^
                              " in " ^ string_of_expr ex))
      in (ty, SUnop(op, (t, e')))
  | Binop(e1, op, e2) as e ->
      let (t1, e1') = expr e1 and (t2, e2') = expr e2 in
      (* All binary operators require operands of the same type *)
      let same = t1 = t2 in
      (* Determine expression type based on operator and operand types *)
      let ty = match op with
        Add
             | Sub
            | Mul
            | Div      when same && t1 = Int   -> Int
        | Eq
            | Neq      when same               -> Boolean
        | Less
            | LessEq
            | More
        | MoreEq   when same && t1 = Int   -> Boolean
        | Union
        | Isec
        | Comp     when same               -> t1
        | Elof     when t1 = Set(t2)       -> Int
        | And
        | Or       when same && t1 = Boolean -> Boolean
```

```
            | _ -> raise (Failure ("illegal binary operator " ^
                      string_of_typ t1 ^ " " ^ string_of_op op ^ " " ^
                      string_of_typ t2 ^ " in " ^ string_of_expr e))in
                      (ty, SBinop((t1, e1'), op, (t2, e2')))
      | Call(fname, args) as call ->
          let fd = find_func fname in
          let param_length = List.length fd.parameters in
          if List.length args != param_length then
            raise (Failure ("expecting " ^ string_of_int param_length ^ " arguments in " ^
string_of_expr call))
          else let check_call (ft, _) e =
            let (et, e') = expr e in
            let err = "illegal argument found " ^ string_of_typ et ^ " expected " ^
string_of_typ ft ^ " in " ^ string_of_expr e
            in (check_assign ft et err, e')
          in
            let args' = List.map2 check_call fd.parameters args
              in (fd.ftype, SCall(fname, args'))
    in

    let check_bool_expr e =
      let (t', e') = expr e
      and err = "expected Boolean expression in " ^ string_of_expr e
      in if t' != Boolean then raise (Failure err) else (t', e')
    in

    (* Return a semantically-checked statement i.e. containing sexprs *)
    let rec check_stmt = function
        Expr e -> SExpr (expr e)
      | If(p, b1, b2) -> SIf(check_bool_expr p, check_stmt b1, check_stmt b2)
      | For(e1, e2, e3, st) -> SFor(expr e1, check_bool_expr e2, expr e3, check_stmt st)
      | ForEach(e1, e2, st) -> SForEach(expr e1, expr e2, check_stmt st)
      | While(p, s) -> SWhile(check_bool_expr p, check_stmt s)
      | Break -> SBreak
      | Return e -> let (t, e') = expr e in
        if t = func.ftype then SReturn (t, e')
        else raise(
          Failure ("return gives " ^ string_of_typ t ^ " expected " ^ string_of_typ func.ftype
^ " in " ^ string_of_expr e))

           (* A block is correct if each statement is correct and nothing
              follows any Return statement.  Nested blocks are flattened. *)
      | Block sl ->
          let rec check_stmt_list = function
              [Return _ as s] -> [check_stmt s]
            | Return _ :: _   -> raise (Failure "nothing may follow a return")
            | Block sl :: ss  -> check_stmt_list (sl @ ss) (* Flatten blocks *)
            | s :: ss         -> check_stmt s :: check_stmt_list ss
            | []              -> []
          in SBlock(check_stmt_list sl)

    in (* body of check_function *)
    { sftype = func.ftype;
      sfname = func.fname;
      sparameters = func.parameters;
      slocals  = func.locals;
      sbody = match check_stmt (Block func.body) with
       SBlock(sl) -> sl
      | _ -> raise (Failure ("internal error: block didn't become a block?"))
    }
  in (globals, List.map check_function functions)
```

Sast.ml

```
open Ast

type sexpr = elmTypes * sx
and sx =
          | SIntLit              of int
          | SCharLit             of char
          | SBoolLit             of bool
          | SStrLit              of string
          | SSetLit              of sexpr list
          | SVariable            of string
          (*| SSetAccess           of string * sexpr *)
               (*| SArrLit           of sexpr list*)
          (*| SArrayAccess         of string * sexpr *)
          | SCall                of string * sexpr list
          | SBinop               of sexpr * op * sexpr
          | SUnop                of unop * sexpr (* if we do string or array slicing, their
syntactic sugar here *)
          | SNoexpr
          | SAssign              of string * sexpr

(*and arr = SArrLit of sexpr list*)
and sstmt =
          SBlock                of sstmt list
         | SExpr                 of sexpr
         | SIf                   of sexpr * sstmt * sstmt
              | SWhile                of sexpr * sstmt
         | SFor                  of sexpr * sexpr * sexpr * sstmt
         | SForEach              of sexpr * sexpr * sstmt
         | SReturn               of sexpr
         | SBreak
         (*| SSetElementAssign     of string * sexpr * sexpr not done and we probably will
just do remove and ins ops*)
         (*| SArrayElementAssign   of string * sexpr * sexpr we dont have arrays rn*)

type sfdecl = { (* function declaration *)
                sftype : elmTypes;
                sfname : string;
                sparameters: bind list;
                slocals: bind list;
                sbody : sstmt list;
            }

type sprogram = bind list * sfdecl list (* a valid program is some globals and function
declarations *)

let string_of_sunop = function
    Not -> "!"

let rec string_of_sexpr (t , e) =
      "(" ^ string_of_typ t ^ " : " ^ (match e with
      SIntLit(l)             -> string_of_int l
    | SCharLit(c)          -> Char.escaped c
    | SStrLit(strlit)        -> strlit
    | SBoolLit(true)         -> "true"
    | SBoolLit(false)        -> "false"
    | SSetLit(setlit)        -> ":{" ^ List.fold_left (^) "" (List.map string_of_sexpr setlit)
^ "}:"
    | SAssign(s, e)          -> s ^ " = " ^ string_of_sexpr e ^ ";\n"
```

```
    | SVariable(s)           -> s
    | SBinop(e1, o, e2)      -> string_of_sexpr e1 ^ " " ^ string_of_op o ^ " " ^
string_of_sexpr e2
    | SUnop(o, e)            -> string_of_sunop o ^ string_of_sexpr e
    | SCall(f, e1)           -> f ^ "(" ^ String.concat", "(List.map string_of_sexpr e1)^ ")"
    | SNoexpr                -> "noexpr"
    (*| SSetAccess (s,e)       -> s ^ "{" ^ string_of_sexpr e ^ "}"
    | SArrayAccess (s,e)     -> s ^ "[" ^ string_of_sexpr e ^ "]"*)
              ) ^ ")"

let rec string_of_sstmt = function
      SBlock(stmts)                -> "Block{\n" ^ String.concat "" (List.map string_of_sstmt
stmts) ^ "}\n"
    | SExpr(expr)               -> string_of_sexpr expr ^ ";\n"
    | SReturn(expr)             -> "return " ^ string_of_sexpr expr ^ ";\n"
    | SIf(e,s1,s2)              -> "if(" ^ string_of_sexpr e ^ ")\n" ^ string_of_sstmt s1 ^
"else\n" ^ string_of_sstmt s2 (*if else*)
    | SWhile(e, s) -> "while (" ^ string_of_sexpr e ^ ") " ^ string_of_sstmt s
    | SFor(e1,e2,e3,s)          -> "for(" ^ string_of_sexpr e1 ^ "; " ^ string_of_sexpr e2 ^
"; " ^ string_of_sexpr e3 ^ ")\n" ^ string_of_sstmt s
    | SForEach(e1,e2,s)         -> "foreach(" ^ string_of_sexpr e1 ^ " in " ^ string_of_sexpr
e2 ^ ")\n" ^ string_of_sstmt s
    (*| SSetElementAssign(s,e1,e2)     -> s ^ "{" ^ string_of_sexpr e1 ^"} = " ^
string_of_sexpr e2 ^";\n"
    | SArrayElementAssign(a,e1,e2)   -> a ^"[" ^ string_of_sexpr e1 ^"] = " ^ string_of_sexpr
e2 ^";\n"*)
    | SBreak                    -> "break;\n"

let string_of_vinit (s, e) = s ^ " = " ^ string_of_sexpr e ^ ";\n"


let string_of_sfdecl fdecl =
  string_of_typ fdecl.sftype ^ " " ^
  fdecl.sfname ^ "(" ^ String.concat ", " (List.map snd fdecl.sparameters) ^
  ")\n{\n" ^
  String.concat "" (List.map string_of_bind fdecl.slocals) ^
  String.concat "" (List.map string_of_sstmt fdecl.sbody) ^
  "}\n"


let string_of_sprogram (vars, funcs) =
    String.concat "" (List.map string_of_bind vars) ^ "\n" ^ String.concat "\n" (List.map
string_of_sfdecl funcs)
```

Codegen.ml

```
(* COPY from MicroC codegen.ml*)

(* 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 L = Llvm
module A = Ast
open Sast

module StringMap = Map.Make(String)

(* translate : Sast.program -> Llvm.module *)
let translate (globals, functions) =
  let context    = L.global_context () in

  (* Create the LLVM compilation module into which
     we will generate code *)
  let the_module = L.create_module context "SOSL" in

  (* Get types from the context *)
  let i32_t          = L.i32_type    context
  and i1_t           = L.i1_type     context
  and i8_t           = L.i8_type     context
  and void_t         = L.void_type   context
  and str_t          = L.pointer_type (L.i8_type context)
  and void_ptr_t     = L.pointer_type (L.i8_type context) in

  (*
  and set_t             = L.named_struct_type context "set"

  in
  let void_ptr_t  = L.pointer_type set_t
  (* and array_t    = L.array_type*)

  in
  *)

  let br_block    = ref (L.block_of_value (L.const_int i32_t 0)) in

  (* Return the LLVM type for a SOSL type *)
  let ltype_of_typ = function
      A.Int              -> i32_t
    | A.Boolean          -> i1_t
    | A.Char             -> i8_t
    | A.String           -> str_t
    | A.Void               -> void_t
    | A.Set(ltype_of_typ)  -> void_ptr_t
    | _ -> raise (Failure "not a supported data type")
  in

  (* Create a map of global variables after creating each *)
  let global_vars : L.llvalue StringMap.t =
    let global_var m (t, n) =
      let init = match t with
          _ -> L.const_int (ltype_of_typ t) 0
      in StringMap.add n (L.define_global n init the_module) m in
    List.fold_left global_var StringMap.empty globals in
  (*Create set.c functions*)


  (*Build printf function from C*)

  let printf_t : L.lltype =
      L.var_arg_function_type i32_t [| L.pointer_type i8_t |] in
  let printf_func : L.llvalue =
      L.declare_function "printf" printf_t the_module in
```

```
let print_set_t : L.lltype =
    L.var_arg_function_type i32_t [| void_ptr_t |] in
let print_set_func : L.llvalue =
    L.declare_function "print_set" print_set_t the_module in
let create_set : L.lltype =
    L.var_arg_function_type void_ptr_t [| i32_t |] in
let create_set_func : L.llvalue =
    L.declare_function "create_set" create_set the_module in
(*let get_head : L.lltype =
    L.var_arg_function_type void_ptr_t [| void_ptr_t |] in
let get_head_func : L.llvalue =
    L.declare_function "get_head" get_head the_module in
let get_data_from_node : L.lltype =
    L.var_arg_function_type void_ptr_t [| void_ptr_t |] in
let get_data_from_node_func : L.llvalue =
    L.declare_function "get_data_from_node" get_data_from_node the_module in
let get_next_node : L.lltype =
    L.var_arg_function_type void_ptr_t [| void_ptr_t |] in
let get_next_node_func : L.llvalue =
    L.declare_function "get_next_node" get_next_node the_module in *)

(*let compare_int_bool_char : L.lltype =
    L.var_arg_function_type i32_t [| void_ptr_t ; void_ptr_t |] in
let compare_int_bool_char_func : L.llvalue =
    L.declare_function "compare_int_bool_char" compare_int_bool_char the_module in
let compare_string : L.lltype =
    L.var_arg_function_type i32_t [| void_ptr_t ; void_ptr_t |] in
let compare_string_func : L.llvalue =
    L.declare_function "compare_string" compare_string the_module in
let compare_set : L.lltype =
    L.var_arg_function_type void_ptr_t [| void_ptr_t ; void_ptr_t |] in
let compare_set_func : L.llvalue =
    L.declare_function "comare_set" compare_set the_module in *)

let add_set : L.lltype =
    L.var_arg_function_type void_ptr_t [| void_ptr_t ; i32_t |] in
let add_set_func : L.llvalue =
    L.declare_function "adds" add_set the_module in
let destroy_set : L.lltype =
    L.var_arg_function_type void_t [| void_ptr_t |] in
let destroy_set_func : L.llvalue =
    L.declare_function "destroy" destroy_set the_module in
(*let rem_set : L.lltype =
    L.var_arg_function_type void_t [| void_ptr_t ; void_ptr_t |] in
let rem_set_func : L.llvalue =
    L.declare_function "remove_elm" rem_set the_module in  *)
let has_elmt : L.lltype =
    L.var_arg_function_type i32_t [| void_ptr_t ; void_ptr_t |] in
let has_elmt_func : L.llvalue =
    L.declare_function "has" has_elmt the_module in
let has_elmt_const : L.lltype =
    L.var_arg_function_type i32_t [| void_ptr_t ; i32_t |] in
let has_elmt_func_const : L.llvalue =
    L.declare_function "has_const" has_elmt_const the_module in
let complement_set : L.lltype =
    L.var_arg_function_type  void_ptr_t [| void_ptr_t ; void_ptr_t |] in
let complement_set_func : L.llvalue =
    L.declare_function "complement" complement_set the_module in
(*let copy_set : L.lltype =
    L.var_arg_function_type void_ptr_t [|void_ptr_t |] in
let copy_set_func : L.llvalue =
```

```
    L.declare_function "copy" copy_set the_module in *)
let union_set : L.lltype =
    L.var_arg_function_type void_ptr_t [|void_ptr_t ; void_ptr_t |] in
let union_set_func : L.llvalue =
    L.declare_function "set_union" union_set the_module in
let intsect_set : L.lltype =
    L.var_arg_function_type void_ptr_t [| void_ptr_t ; void_ptr_t |] in
let intsect_set_func : L.llvalue =
    L.declare_function "intersect" intsect_set the_module in
(* let get_card : L.lltype =
    L.var_arg_function_type i32_t [| void_ptr_t |] in
let get_card_func : L.llvalue =
    L.declare_function "get_card" intsect_set the_module in *)


 let function_decls : (L.llvalue * sfdecl) StringMap.t =
  let function_decl m fdecl =
    let name = fdecl.sfname
    and parameter_types =
     Array.of_list (List.map (fun (t,_) -> ltype_of_typ t) fdecl.sparameters)
    in let ftype = L.function_type (ltype_of_typ fdecl.sftype) parameter_types in
    StringMap.add name (L.define_function name ftype the_module, fdecl) 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.sfname function_decls in
  let builder = L.builder_at_end context (L.entry_block the_function) in

  (*declare variables and formatting for each C printf type to be called below in SCall*)
  let int_format_str = L.build_global_stringptr "%d\n" "fmt" builder
  and float_format_str = L.build_global_stringptr "%g\n" "fmt" builder
  and str_format_str = L.build_global_stringptr "%s\n" "fmt" builder
  and char_format_str = L.build_global_stringptr "%c\n" "fmt" builder in

  (* Construct the function's "locals": formal arguments and locally
     declared variables.  Allocate each on the stack, initialize their
     value, if appropriate, and remember their values in the "locals" map *)
  let local_vars =
    let add_formal m (t, n) p =
     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);
     StringMap.add n local m

    (* Allocate space for any locally declared variables and add the
     * resulting registers to our map *)
    and add_local m (t, n) =
     let local_var = L.build_alloca (ltype_of_typ t) n builder
     in StringMap.add n local_var m
    in

    let formals = List.fold_left2 add_formal StringMap.empty fdecl.sparameters
        (Array.to_list (L.params the_function)) in
    List.fold_left add_local formals fdecl.slocals
  in

  (* Return the value for a variable or formal argument.
     Check local names first, then global names *)
  let lookup n = try StringMap.find n local_vars
                 with Not_found -> StringMap.find n global_vars
```

```
    in

   (* Construct code for an expression; return its value *)
    let rec expr builder = function
        SIntLit i     -> L.const_int i32_t i
      | SBoolLit b    -> L.const_int i1_t (if b then 1 else 0)
      | SCharLit c    -> L.const_int i8_t (Char.code c)
      | SStrLit str   -> L.build_global_stringptr str "string" builder
      | SSetLit sl    ->
        (match sl with
        | [] -> L.build_call create_set_func [| L.const_int i32_t 5 |] "tmp" builder
        | hd :: _ ->
            let (ty1, _) = hd in
            let s =
                (match ty1 with
                Int         -> L.build_call create_set_func [| L.const_int i32_t 0 |] "tmp"
builder
                | Char      -> L.build_call create_set_func [| L.const_int i32_t 1 |] "tmp"
builder
                | Boolean   -> L.build_call create_set_func [| L.const_int i32_t 2 |] "tmp"
builder
                | String    -> L.build_call create_set_func [| L.const_int i32_t 3 |] "tmp"
builder
                | Set(_)    -> L.build_call create_set_func [| L.const_int i32_t 4 |] "tmp"
builder ) in
            let addNodes ex =
                let (ty, e2) = ex in
                    L.build_call add_set_func [| s; expr builder e2 |] "s" builder in
                    List.map addNodes sl;  s)
      | SNoexpr       -> L.const_int i32_t 0
      | SVariable s   -> L.build_load (lookup s) s builder
      | SAssign (s,ex) -> let (_ , e) = ex in
                       let e' = expr builder e in
                         ignore(L.build_store e' (lookup s) builder); e'
      | SBinop ((_,e1), op, (tyy,e2)) ->
        let e1' = expr builder e1 and e2' = expr builder e2 in
        (match op with
          A.Add     -> L.build_add e1' e2' "tmp" builder
        | A.Sub     -> L.build_sub e1' e2' "tmp" builder
        | A.Mul     -> L.build_mul e1' e2' "tmp" builder
        | A.Div     -> L.build_sdiv e1' e2' "tmp" builder
        | A.And     -> L.build_and e1' e2' "tmp" builder
        | A.Or      -> L.build_or e1' e2' "tmp" builder
        | A.Eq      -> L.build_icmp L.Icmp.Eq e1' e2' "tmp" builder
        | A.Neq     -> L.build_icmp L.Icmp.Ne e1' e2' "tmp" builder
        | A.Less    -> L.build_icmp L.Icmp.Slt e1' e2' "tmp" builder
        | A.LessEq  -> L.build_icmp L.Icmp.Sle e1' e2' "tmp" builder
        | A.More    -> L.build_icmp L.Icmp.Sgt e1' e2' "tmp" builder
        | A.MoreEq  -> L.build_icmp L.Icmp.Sge e1' e2' "tmp" builder
        | A.Mod     -> L.build_frem e1' e2' "tmp" builder
        | A.Elof    -> (match tyy with
                       Int  ->     L.build_call has_elmt_func_const [| e1'; e2' |] "has_const"
builder
                       | Char  ->     L.build_call has_elmt_func_const [| e1'; e2' |]
"has_const" builder
                       | Boolean  ->     L.build_call has_elmt_func_const [| e1'; e2' |]
"has_const" builder
                       | String ->     L.build_call has_elmt_func [| e1'; e2' |] "has" builder
                       | Set(_)->     L.build_call has_elmt_func[| e1'; e2' |] "has" builder )

        | A.Comp    -> L.build_call complement_set_func [| e1' ; e2' |] "complement" builder
        | A.Isec    -> L.build_call intsect_set_func [| e1' ; e2' |] "intersect" builder
```

```
        | A.Union   -> L.build_call union_set_func [| e1' ; e2' |] "set_union" builder)
      | SUnop(op, (_, e)) ->
          let e' = expr builder e in
                (match op with
           A.Not          -> L.build_not) e' "tmp" builder

      (*Map various print functions back to C's printf function*)
      | SCall ("print", [(_, e)]) | SCall ("printb", [(_, e)]) ->
           L.build_call printf_func [| int_format_str ; (expr builder e) |] "print" builder
      | SCall ("prints", [(_, e)]) ->
           L.build_call printf_func [| str_format_str ; (expr builder e) |] "prints" builder
      | SCall ("printc", [(_, e)]) ->
           L.build_call printf_func [| char_format_str;  (expr builder e) |] "print_char"
builder
      | SCall ("printf", [(_, e)]) ->
        L.build_call printf_func [| float_format_str ; (expr builder e) |] "printf" builder
      | SCall ("print_set_int", [(_,e)]) ->
        L.build_call print_set_func [| expr builder e |] "print_set" builder
      (*| SCall ("print_set_bool", [(_,e)]) ->
        L.build_call print_set_func [| expr builder e |] "print_set" builder
      | SCall ("print_set_char", [(_,e)]) ->
        L.build_call print_set_func [| expr builder e |] "print_set" builder
      | SCall ("print_set_string", [(_,e)]) ->
        L.build_call print_set_func [| expr builder e |] "print_set" builder*)
      (*| SCall ("print_set_set", [(_,e)]) ->
        L.build_call print_set_func [| expr builder e |] "print_set" builder*)

      (*| SCall ("adds_int", [(_,e1); (_,e2)]) ->
       L.build_call add_set_func [| (expr builder e1) ; (expr builder e2) |] "adds" builder
      | SCall ("adds_int", [(_,e1); (_,e2)]) ->
       L.build_call add_set_func [| (expr builder e1) ; (expr builder e2) |] "adds" builder
      | SCall ("adds_int", [(_,e1); (_,e2)]) ->
       L.build_call add_set_func [| (expr builder e1) ; (expr builder e2) |] "adds" builder
      | SCall ("adds_int", [(_,e1); (_,e2)]) ->
       L.build_call add_set_func [| (expr builder e1) ; (expr builder e2) |] "adds" builder
      | SCall ("adds_int", [(_,e1); (_,e2)]) ->
       L.build_call add_set_func [| (expr builder e1) ; (expr builder e2) |] "adds" builder
      | SCall ("rems_int", [(_,e1); (_,e2)]) ->
       L.build_call add_set_func [| (expr builder e1) ; (expr builder e2) |] "adds" builder
      | SCall ("rems_int", [(_,e1); (_,e2)]) ->
       L.build_call rem_set_func [| (expr builder e1) ; (expr builder e2) |] "remove_elm"
builder
      | SCall ("rems_int", [(_,e1); (_,e2)]) ->
       L.build_call rem_set_func [| (expr builder e1) ; (expr builder e2) |] "remove_elm"
builder
      | SCall ("rems_int", [(_,e1); (_,e2)]) ->
       L.build_call rem_set_func [| (expr builder e1) ; (expr builder e2) |] "remove_elm"
builder
      | SCall ("rems_int", [(_,e1); (_,e2)]) ->
       L.build_call rem_set_func [| (expr builder e1) ; (expr builder e2) |] "remove_elm"
builder
      (*| SCall ("rems_set", [(_,e1); (_,e2)]) ->
       L.build_call rem_set_func [| (expr builder e1) ; (expr builder e2) |] "remove_elm"
builder*)*)


      | SCall (f, args) ->
            let (fdef, fdecl) = StringMap.find f function_decls in
                let llargs = List.rev (List.map (expr builder) (List.rev_map snd args)) in
            let result =
                (match fdecl.sftype with
                    _ -> f ^ "_result") in
```

```
          L.build_call fdef (Array.of_list llargs) result builder
    in

    (* LLVM insists each basic block end with exactly one "terminator"
       instruction that transfers control.  This function runs "instr builder"
       if the current block does not already have a terminator.  Used,
       e.g., to handle the "fall off the end of the function" case. *)
    let add_terminal builder instr =
      match L.block_terminator (L.insertion_block builder) with
        Some _ -> ()
      | None -> ignore (instr builder) in

    (* Build the code for the given statement; return the builder for
       the statement's successor (i.e., the next instruction will be built
       after the one generated by this call) *)

    let rec stmt builder = function
            SBlock sl       -> List.fold_left stmt builder sl
      | SExpr (_, e)    -> ignore(expr builder e); builder
      | SReturn (_, e)       -> ignore(match fdecl.sftype with
                    Void          -> L.build_ret_void builder (* Special "return nothing" instr
*)
                  | _             -> L.build_ret (expr builder e) builder ); builder

      | SIf ((_, predicate), then_stmt, else_stmt) ->
        let bool_val = expr builder predicate in
                let merge_bb = L.append_block context "merge" the_function in
            let build_br_merge = L.build_br merge_bb in (* partial function *)

                  let then_bb = L.append_block context "then" the_function in
                    add_terminal (stmt (L.builder_at_end context then_bb) then_stmt)
                      build_br_merge;

              let else_bb = L.append_block context "else" the_function in
                add_terminal (stmt (L.builder_at_end context else_bb) else_stmt)
                  build_br_merge;

              ignore(L.build_cond_br bool_val then_bb else_bb builder);
              L.builder_at_end context merge_bb

      | SBreak -> ignore (L.build_br !br_block builder);  builder

      | SWhile ((_, 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 = 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

        (*
        | SForEach (e1, e2, s) ->
            let set_ptr = expr builder e2 in
            let counter = L.build_alloca i32_t "counter" builder in
            ignore(L.build_store (L.const_int i32_t 0) counter builder);
```

```
            let size = L.build_call get_card_func [| set_ptr |] "size" builder in
            let node_var = L.build_alloca void_ptr_t n builder in
            let vars = StringMap.add n set_var vars in

            let current_node_ptr = L.build_alloca void_ptr_t "current" builder in
            let head_node = L.build_call get_head_func [| set_ptr |] "head" builder in
            ignore(L.build_store head_node current_node_ptr builder);

            let body_bb = L.append_block context "while_body" the_function in
            let body_builder = L.builder_at_end context body_bb in

            let current_node = L.build_load current_node_ptr "current_tmp" body_builder in

            let data_ptr = L.build_call get_data_from_vertex_func [| current_node |] (n ^
"_tmp") body_builder in
            ignore(L.build_store data_ptr node_var body_builder);

            let next_node = L.build_call get_next_vertex_func [| current_node |] "next"
body_builder in
            ignore(L.build_store next_node current_node_ptr body_builder);

            let counter_val = L.build_load counter "counter_tmp" body_builder in
            let counter_incr = L.build_add (L.const_int i32_t 1) counter_val "counter_incr"
body_builder in
            ignore(L.build_store counter_incr counter body_builder);

            add_terminal (stmt vars body_builder body) (L.build_br pred_bb);

            let pred_builder = L.builder_at_end context pred_bb in
            let counter_val = L.build_load counter "counter_tmp" pred_builder in
            let done_bool_val = L.build_icmp L.Icmp.Slt counter_val size "done" pred_builder
in

            let merge_bb = L.append_block context "merge" the_function in
            ignore (L.build_cond_br done_bool_val body_bb merge_bb pred_builder);
            L.builder_at_end context merge_bb *)

      (* Implement for loops as while loops *)
      | SFor (e1, e2, e3, body) -> stmt builder
            ( SBlock [SExpr e1 ; SWhile (e2, SBlock [body ; SExpr e3]) ] ) in

      (* Build the code for each statement in the function *)
        let builder = stmt builder (SBlock fdecl.sbody) in

        (* Add a return if the last block falls off the end *)
        add_terminal builder (match fdecl.sftype with
          (* A.Void -> L.build_ret_void*)
          | t -> L.build_ret (L.const_int (ltype_of_typ t) 0))
 in

 List.iter build_function_body functions;
 the_module
```

Linkedlist.h

```
#ifndef _MYLIST_H_
#define _MYLIST_H_

/*
 * A node in a linked list.
```

```
 */
struct Node {
    void *data;
    struct Node *next;
};

/*
 * A linked list.
 * 'head' points to the first node in the list.
 */
struct List {
    struct Node *head;
};

/*
 * Initialize an empty list.
 */

static inline void initList(struct List *list)
{
    list->head = 0;
}

/*
 * In all functions below, the 'list' parameter is assumed to point to
 * a valid List structure.
 */

/*
 * Create a node that holds the given data pointer,
 * and add the node to the front of the list.
 *
 * Note that this function does not manage the lifetime of the object
 * pointed to by 'data'.
 *
 * It returns the newly created node on success and NULL on failure.
 */
struct Node *addFront(struct List *list, void *data);

/*
 * Traverse the list, calling f() with each data item.
 */
void traverseList(struct List *list, void (*f)(void *));

/*
 * Traverse the list, comparing each data item with 'dataSought' using
 * 'compar' function.  ('compar' returns 0 if the data pointed to by
 * the two parameters are equal, non-zero value otherwise.)
 *
 * Returns the first node containing the matching data,
 * NULL if not found.
 */
struct Node *findNode(struct List *list, const void *dataSought,
       int (*compar)(const void *, const void *));


/*
 * Returns 1 if the list is empty, 0 otherwise.
 */
static inline int isEmptyList(struct List *list)
{
```

```
    return (list->head == 0);
}

/*
 * Remove the first node from the list, deallocate the memory for the
 * ndoe, and return the 'data' pointer that was stored in the node.
 * Returns NULL is the list is empty.
 */
void *popFront(struct List *list);

/*
 * Remove all nodes from the list, deallocating the memory for the
 * nodes.  You can implement this function using popFront().
 */
void removeAllNodes(struct List *list);

/*
 * Create a node that holds the given data pointer,
 * and add the node right after the node passed in as the 'prevNode'
 * parameter.  If 'prevNode' is NULL, this function is equivalent to
 * addFront().
 *
 * Note that prevNode, if not NULL, is assumed to be one of the nodes
 * in the given list.  The behavior of this function is undefined if
 * prevNode does not belong in the given list.
 *
 * Note that this function does not manage the lifetime of the object
 * pointed to by 'data'.
 *
 * It returns the newly created node on success and NULL on failure.
 */
struct Node *addAfter(struct List *list,
       struct Node *prevNode, void *data);

/*
 * Reverse the list.
 *
 * Note that this function reverses the list purely by manipulating
 * pointers.  It does NOT call malloc directly or indirectly (which
 * means that it does not call addFront() or addAfter()).
 *
 * Implementation hint: keep track of 3 consecutive nodes (previous,
 * current, next) and move them along in a while loop.  Your function
 * should start like this:


      struct Node *prv = NULL;
      struct Node *cur = list->head;
      struct Node *nxt;

      while (cur) {
          ......


 * And at the end, prv will end up pointing to the first element of
 * the reversed list.  Don't forget to assign it to list->head.
 */
void reverseList(struct List *list);

#endif /* #ifndef _MYLIST_H_ */
```

Linkedlist.c

```
#include <stdio.h>
#include <stdlib.h>
#include "linkedlist.h"

struct Node *addFront(struct List *list, void *data){

        struct Node *node = malloc(sizeof(struct Node));
        node->data = data;
        node->next = list->head;
        list->head = node;
        return list->head;

}
void traverseList(struct List *list, void(*f)(void *)){

    struct Node *tmp = list->head;
    while(tmp != NULL){
        (*f)(tmp->data);
        tmp = tmp->next;
    }
}
struct Node *findNode(struct List *list, const void *dataSought, int (*compar)(const void *,
const void *)){

    struct Node *tmp = list->head;

    while(tmp){
        if((*compar)(dataSought, tmp->data) == 0){
            return tmp;
        }
        tmp = tmp->next;
    }
    return NULL;

}
void *popFront(struct List *list){


        if(list->head == 0){
            return NULL;
        }
        void *data;
        struct Node *n = list->head;
        list->head = list->head->next;
        data = n->data;
        free(n);
        return data;

}
void removeAllNodes(struct List *list){

    while(list->head != NULL){
        popFront(list);
    }
}
struct Node *addAfter(struct List *list, struct Node *prevNode, void *data){

    struct Node *node = malloc(sizeof(struct Node));
```

```
    if(prevNode == NULL){
        list->head= node;
        return list->head;
    }
    prevNode->next = node;
    return prevNode->next;

}
void reverseList(struct List *list){

    struct Node *prv;
    struct Node *cur = list->head;
    struct Node *nxt;
    prv = NULL;
    nxt = NULL;

    while(cur){
        nxt = cur->next;
        cur->next = prv;
        prv = cur;
        cur = nxt;
    }
    cur = prv;
    list->head = cur;
}
```

Setlib.h

```
#include "linkedlist.h"

/*type corrosponds int,boolean,char,string,set
 * 0 -> int
 * 1 -> boolean
 * 2 -> char
 * 3 -> string
 * 4 -> set */
struct set{
    struct List list;
    int card;
    int type;
};
void *create_set(int dType);

void destroy(void *s_ptr);

int has(void *s_ptr, void *value);
int has_const(void *s_ptr, int value);
int compare_int_bool_char(const void *data_sought, const void *against);
int compare_string(const void *data_sought, const void *against);
int compare_char(const void *data_sought, const void *against);
int compare_set(const void *data_sought, const void *against);
int get_card(void *A_ptr);
void *get_head(void *set_ptr);
void *get_data_from_node(void *node_ptr);
void *get_next_node(void *data);

void* adds(void *s_ptr, void *value);
void* remove_elm(void *s_ptr, void *value);

void print_set(void *A_ptr);
```

```
void *complement(void *A_ptr, void *U);
void *set_union(void *A_ptr, void *B);
void *intersect(void *A_ptr, void *B);
void *copy(void *A_ptr);
void *cartesian(void *A_ptr, void *B);
```

Setlib.c

```
#include<string.h>
#include<stdlib.h>
#include<stdio.h>
#include "setlib.h"

/* type = 0 -> int
 *      = 1 -> boolean
 *      = 2 -> char
 *      = 3 -> string
 *      = 4 -> set
 *      = 5 -> void

struct set* create(struct List *list, int dType){
    struct set *result;
    struct List *curr = result->head;

    result->card = 0;
    result->type = dType;

    while (list != 0) {
        curr = list;
        curr = curr->next;
        list = list->next;
        result->card++;
    }

    return result;

}*/

void *create_set(int dType){  //not necessairly from llvm
    struct set *newset = malloc(sizeof(struct set));
    initList(&(newset->list));
    newset->card = 0;
    newset->type = 1;
    return (void *) newset;
}

void *get_head(void *set_ptr){
    struct set *s = (struct set *) set_ptr;

    return (void *) (s->list).head;
}

void *get_data_from_node(void *node_ptr){
    struct Node *node = (struct Node *) node_ptr;
    return node->data;
}

void *get_next_node(void *node_ptr){
    struct Node *node = (struct Node *) node_ptr;
```

```
    return (void *) node->next;
}

void destroy(void *set_ptr){
    struct set *s = (struct set *) set_ptr;

    struct List nodes = s->list;
    struct Node *next = nodes.head;

    if ((s->type)==4){
        for (int i=0; i<(s->card); i++){
            struct set *temp = next->data;
            next = next->next;
            destroy(temp);
        }
    }
    else {
        removeAllNodes(&nodes);
    }
}

int compare_int_bool_char(const void *data_sought, const void *against){
//they're all the same so we just cast to int and do equivalence
    if(*(int *)data_sought != *(int *)against)
        return 1;
    return 0;
}
int compare_string(const void *data_sought, const void *against){
    return strcmp((char *)data_sought,(char *)against);
}

int compare_set(const void *data_sought, const void *against){
    int type_sought = ((struct set *)data_sought)->type;
    int type_against =((struct set *)against)->type;//will throw error if screw up types
    if(type_sought != type_against)
        return 1;//sets of different types are clearly not the same thing
    else{
        //loop over all the elements and run the right compare method
        struct Node *tmpCurr = (((struct set *)data_sought)->list).head;

        for(int i=0; i<(((struct set *)data_sought)->card); i++){

            if(has((struct set *)against,tmpCurr->data)==0){
                return 1;
            }
            else{
                tmpCurr= tmpCurr->next;
            }
        }
    }
    return 0;

}

int has(void *set_ptr, void *value){
    struct set *s = (struct set *) set_ptr;

    struct List *nodes = &(s->list);
    int (*compar)(const void *, const void *);
    if(s->type == 0 || s->type==1 || s->type == 2)
        compar = compare_int_bool_char;
```

```
    else if(s->type == 3)
        compar = compare_string;
    else if(s->type == 4)
        compar = compare_set;
    if (findNode(nodes, value, compar) != 0){
        return 1;
    }

    return 0;
}

int has_const(void *set_ptr, int value){
    return has(set_ptr, &value);
}


void *adds(void *set_ptr, void *value){
    struct set *s = (struct set *) set_ptr;
    struct List nodes = s->list;

    if (!has(s,value)){
        addFront(&nodes, value);
    }

    return (void *) s;
}

void *remove_elm(void *set_ptr, void *value){
    struct set *s = (struct set *) set_ptr;
    struct Node *tmpNode = (s->list).head;
    struct Node *prev;

    while(tmpNode != 0 && tmpNode->data != value){
        prev = tmpNode;
        tmpNode = tmpNode->next;
    }

    if (tmpNode == 0) return (void *) s;

    prev->next = tmpNode->next;
    free(tmpNode);
    return (void *) s;
}



void *complement(void *A_ptr, void* U_ptr){
    struct set *A = (struct set *) A_ptr;
    struct set *U = (struct set *) U_ptr;

    struct set *tmp = create_set(A->type);
    struct List tmpNodes = tmp->list;
    struct Node *tmpCurr = tmpNodes.head;

    struct List uNodes = U->list;
    struct Node *uCurr = uNodes.head;


    struct set *AiU = intersect(A,U);
    for(int i=0; i<(U->card); i++){
```

```
        if(!has(AiU, uCurr->data)){
            adds(tmp, uCurr->data);
        }
        uCurr = uCurr->next;
    }
    destroy(AiU);
    return (void *) tmp;
}
void* copy(void *A){                //maybe put in a set_lib.c?
    return 0;
}

void* set_union(void *A_ptr, void *B_ptr){
    struct set *A = (struct set *) A_ptr;
    struct set *B = (struct set *) B_ptr;

    struct List aNodes = A->list;
    struct Node *aCurr = aNodes.head;

    struct List bNodes = B->list;
    struct Node *bCurr = bNodes.head;

    struct set *AuB=create_set(A->type);

    int bigger_card = (A->card > B->card) ? A->card : B->card;
    for (int i=0; i<bigger_card; i++){
        if(i<A->card){
            adds(AuB,aCurr);
            aCurr = aCurr->next;
        }
        if(i< B->card){
            adds(AuB,bCurr);
            bCurr = bCurr->next;
        }
    }

    return (void *) AuB;
}
int compare(void *Adata, void *Bdata, int type){
    if(type == 0 || type == 1 || type == 2)
        return compare_int_bool_char(Adata,Bdata);
    else if(type == 3)
        return compare_string(Adata,Bdata);
    else if(type == 4)
        return compare_set(Adata,Bdata);

    return -1;
}
void *intersect(void *A_ptr, void *B_ptr){
    struct set *A = (struct set *) A_ptr;
    struct set *B = (struct set *) B_ptr;

    struct set *tmp = create_set(A->type);

    struct List aNodes = A->list;
    struct Node *aCurr = aNodes.head;

    struct List bNodes = B->list;
    struct Node *bCurr = bNodes.head;

    int larger_card;
```

```
    A->card > B->card ? (larger_card = A->card):(larger_card = B-> card);
    int smaller_card;
    A->card <= B->card ? (smaller_card = A->card) : (smaller_card = B->card);
    for (int i=0; i<smaller_card; i++){
        for (int j=0; j<larger_card; j++){
            if(compare(aCurr->data,bCurr->data, A->type) == 0){
                adds(tmp, A->card > B->card ? aCurr->data: bCurr->data);
            }
            A->card > A->card ? (bCurr = bCurr->next):(aCurr = aCurr->next);
        }
        A->card <= B->card ? (aCurr = aCurr->next):(bCurr = bCurr->next);
    }

    return (void *) tmp;
}

int get_card(void *A_ptr){
    struct set *A = (struct set *) A_ptr;

    return A->card;
}

void *cartesian(void *A_ptr, void *B_ptr){                // not done -Ryan C.
    struct set *A = (struct set *) A_ptr;
    struct set *B = (struct set *) B_ptr;


    struct set *tmp;
    struct List tmpNodes = tmp->list;
    struct Node *tmpCurr = tmpNodes.head;

    struct List aNodes = A->list;
    struct Node *aCurr = aNodes.head;

    struct List bNodes = B->list;
    struct Node *bCurr = bNodes.head;

    return tmp;
}

void print_set(void *A_ptr){
    struct set *A = (struct set *) A_ptr;
    printf(":{");
    struct Node *Acurr = A->list.head;
    for(int i = 0; i<get_card(A); i++){
        int typ = A->type;
        if(i < get_card(A) - 1){
            if(typ == 0)
                printf("%d,",*((int *)Acurr->data));
            else if(typ == 1)
                printf(*((int *)Acurr->data) == 0 ? "false," : "true,");
            else if(typ == 2)
                printf("%c,",*((char *)Acurr->data));
            else if(typ == 3)
                printf("%s,",(char *)(Acurr->data));
            else if(typ == 4)
                print_set(Acurr->data);
        }
        else{
            if(typ == 0)
                printf("%d",*((int *)Acurr->data));
```

```
            else if(typ == 1)
                printf((*(int *)Acurr->data) == 0 ? "false" : "true");
            else if(typ == 2)
                printf("%c",*((char *)Acurr->data));
            else if(typ == 3)
                printf("%s",(char *)(Acurr->data));
            else if(typ == 4)
                print_set(Acurr->data);
        }
        Acurr = Acurr->next;
    }
}
```