# Contents

1 Introduction ............................................. 3
   1.1 Goals .............................................. 3
      1.1.1 Syntax Simplification ......................... 3
      1.1.2 String and Array Improvements .............. 4

2 Language Tutorial .................................. 4
   2.1 Docker Configuration ............................ 4
   2.2 Compilation and Execution ...................... 5

3 Language Reference Manual ....................... 5
   3.1 Lexical Structure ................................ 5
      3.1.1 Comments ................................. 5
      3.1.2 Identifiers ............................... 5
      3.1.3 Keywords ................................. 6
      3.1.4 Separators ............................... 6
      3.1.5 Operators ............................... 6
      3.1.6 Literals ................................. 6
   3.2 Types and Variables ............................ 7
      3.2.1 Primitive Types ......................... 8
      3.2.2 Objects ................................. 9
      3.2.3 Variables ............................... 9
      3.2.4 Strings ................................. 10
      3.2.5 Arrays ................................. 10
   3.3 Statements and Expressions ..................... 11
      3.3.1 Statements .............................. 11
      3.3.2 Expressions .............................. 12
      3.3.3 Operator Precedence ...................... 13
      3.3.4 Functions .............................. 13
   3.4 Classes ........................................... 14
      3.4.1 Dot Operator ............................. 14
   3.5 Built-In Functions ................................. 14
      3.5.1 Arrays .................................. 15
      3.5.2 Strings .................................. 15
      3.5.3 Print ..................................... 16
   3.6 Syntax/Style Guide .............................. 16
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Project Plan</td>
<td>17</td>
</tr>
<tr>
<td>4.1</td>
<td>Planning Process</td>
<td>17</td>
</tr>
<tr>
<td>4.2</td>
<td>Specification Process</td>
<td>17</td>
</tr>
<tr>
<td>4.3</td>
<td>Development Process</td>
<td>17</td>
</tr>
<tr>
<td>4.4</td>
<td>Testing Process</td>
<td>18</td>
</tr>
<tr>
<td>4.5</td>
<td>Roles and Responsibilities</td>
<td>18</td>
</tr>
<tr>
<td>4.6</td>
<td>Project Timeline</td>
<td>19</td>
</tr>
<tr>
<td>4.7</td>
<td>Project Log</td>
<td>19</td>
</tr>
<tr>
<td>4.8</td>
<td>Software Development Environment</td>
<td>19</td>
</tr>
<tr>
<td>4.9</td>
<td>Programming Style Guide</td>
<td>19</td>
</tr>
<tr>
<td>5</td>
<td>Architectural Design</td>
<td>20</td>
</tr>
<tr>
<td>5.1</td>
<td>Block Diagram</td>
<td>20</td>
</tr>
<tr>
<td>5.2</td>
<td>Scanner</td>
<td>20</td>
</tr>
<tr>
<td>5.3</td>
<td>Parser and AST</td>
<td>20</td>
</tr>
<tr>
<td>5.4</td>
<td>Semantic Checking</td>
<td>21</td>
</tr>
<tr>
<td>5.5</td>
<td>Code Generation</td>
<td>21</td>
</tr>
<tr>
<td>5.6</td>
<td>C Libraries</td>
<td>21</td>
</tr>
<tr>
<td>6</td>
<td>Test Plan</td>
<td>22</td>
</tr>
<tr>
<td>6.1</td>
<td>Representative Programs</td>
<td>22</td>
</tr>
<tr>
<td>6.2</td>
<td>Testing Tools</td>
<td>23</td>
</tr>
<tr>
<td>6.3</td>
<td>Automated CD/CI</td>
<td>24</td>
</tr>
<tr>
<td>6.4</td>
<td>Performance Benchmarking</td>
<td>24</td>
</tr>
<tr>
<td>6.5</td>
<td>Target Language Program (LLVM)</td>
<td>25</td>
</tr>
<tr>
<td>6.5.1</td>
<td>GCD</td>
<td>25</td>
</tr>
<tr>
<td>6.5.2</td>
<td>Bubble Sort</td>
<td>26</td>
</tr>
<tr>
<td>6.5.3</td>
<td>Class</td>
<td>30</td>
</tr>
<tr>
<td>7</td>
<td>Lessons Learned</td>
<td>33</td>
</tr>
<tr>
<td>7.1</td>
<td>Frances Cao</td>
<td>33</td>
</tr>
<tr>
<td>7.2</td>
<td>Mateo Maturana</td>
<td>33</td>
</tr>
<tr>
<td>7.3</td>
<td>Hongfei Chen</td>
<td>33</td>
</tr>
<tr>
<td>7.4</td>
<td>Ian Chen</td>
<td>34</td>
</tr>
<tr>
<td>8</td>
<td>Appendix</td>
<td>34</td>
</tr>
<tr>
<td>8.1</td>
<td>scanner.ml</td>
<td>34</td>
</tr>
<tr>
<td>8.2</td>
<td>parser.mly</td>
<td>36</td>
</tr>
<tr>
<td>8.3</td>
<td>ast.ml</td>
<td>38</td>
</tr>
<tr>
<td>8.4</td>
<td>semant.ml</td>
<td>41</td>
</tr>
<tr>
<td>8.5</td>
<td>sast.ml</td>
<td>49</td>
</tr>
<tr>
<td>8.6</td>
<td>codegen.ml</td>
<td>52</td>
</tr>
<tr>
<td>8.7</td>
<td>javalite.ml</td>
<td>61</td>
</tr>
<tr>
<td>8.8</td>
<td>stringfunsc.c</td>
<td>62</td>
</tr>
<tr>
<td>8.9</td>
<td>Makefile</td>
<td>64</td>
</tr>
<tr>
<td>8.10</td>
<td>testall.sh</td>
<td>65</td>
</tr>
</tbody>
</table>
1 Introduction

JavaLite is a multi-paradigm language that allows the user to freely use both functional and object-oriented paradigms. While Java is one of the most popular programming languages for beginners, it can be challenging for new programmers to get familiar with its excessive syntax and strict object-oriented programming rules. In contrast, JavaLite has simplified syntax compared to Java. Further, JavaLite supports more built-in functionality for non-primitive data types, including String and Array. These built-in methods are inspired by Python and the goal is to provide a more intuitive way of String/Array manipulation.

1.1 Goals

1.1.1 Syntax Simplification

In Java, each program has to have at least the same basic outline.

```java
public class Program {
    public static void main(String[] args) {
        // main method begins here
    }
}
```

For new programmers, this is often confusing and can lead to many questions that are usually not answered until much later in an introductory class such as public vs private, static methods, method arguments, and arrays. This is mainly because Java is completely object-oriented, so every method must be part of a class, including the main method. JavaLite removes the need for this complicated syntax by allowing functional program. Therefore, there is no main method and no class is needed to execute any code, similar to JavaScript.

Additionally, consider the default print method in Java: `system.out.print()`. The only relevant part to a new programmer is print since they won’t initially understand the purpose behind system.out. In JavaLite, printing is done with `print()`.
1.1.2 String and Array Improvements

In Java, Strings and Arrays present many unique issues that may be confusing to new programmers. With strings, we cannot directly compare strings with <, >, <=, or >=. Instead in Java we have to use `compareTo()`, however this may be confusing to new programmers so these binary operations are implemented in JavaLite. Also, in Java `==` is not accurate in comparing the values of two strings so `equals()` must be used instead. This problem is resolved in JavaLite so string comparisons remain consistent with primitive types such as int.

One simple change that we make is array literals. Although arrays are declared with `type []`, arrays literals are declared with a list of values enclosed by `{}`. This doesn’t make a lot of sense, so instead JavaLite array literals are list of values enclosed by `[]`.

2 Language Tutorial

Some familiarity with git and bash/shell is assumed. First the repository must be cloned into your local machine using `git clone`.

```
$ git clone git@github.com:FranCao/javalite.git
```

2.1 Docker Configuration

To run JavaLite, Docker must be installed on your local machine. To install Docker, refer to Docker’s [documentation](#).

1. Navigate to the root of the JavaLite repository. If you haven’t changed directory since you cloned, you can run the following command.

```
$ cd javalite
```

2. Make sure docker is running correctly. To test, you can run

```
> docker run hello-world
```

If this fails, you will need to correct your installation.

3. Next, invoke the docker image.

```
> docker run --rm -it -v 'pwd':/home/javalite -w=/home/javalite columbiasowards/plt
```

4. Finally, if you’re not in the correct directory, run `cd javalite` to switch to the correct directory.
2.2 Compilation and Execution

To compile the JavaLite compiler and run JavaLite’s test suite, you can use the following command.

```
make
```

Otherwise, you can also just write, compile, and execute a single program. First, the JavaLite compiler must be compiled. To do this, run the following command.

```
make all
```

The following sample code is a hello world program in JavaLite.

```
print("Hello, World!");
```

Save this code into a `helloworld.jl` file. To compile and run this code, run the following commands in your terminal.

```
$ ./BuildAndRun helloworld.jl
Hello, World!
```

3 Language Reference Manual

3.1 Lexical Structure

3.1.1 Comments

JavaLite supports both single line and multi-line comments.

```
// This is a single line comment
```

A single line comment starts with “//” and ends at the end of the line. Any ASCII characters before the end of the line are considered comments.

```
/* This is a
  multi-line
  comment */
```

A multi-line comment is wrapped by “/*” and “*/”. Namely, any ASCII characters after “/*” are considered comments until there is a “*/”.

JavaLite does not allow nested comments. The comments will be ignored by the compiler and will not be executed.

3.1.2 Identifiers

An identifier in JavaLite is an unlimited-length character sequence consisting of letters and digits ”a-zA-Z1-9”, except for those reserved for keywords [3.1.3]. Note that identifiers must begin with letters.
3.1.3 Keywords

There are a total number of 12 keywords reserved in JavaLite. Each keyword is a character sequence consisting of ASCII letters. The following are all the keywords.

```
bool? else int
string for void
class if while
double return null
```

3.1.4 Separators

There are 9 separators in JavaLite. They are tokens formed from ASCII characters. The following are all the separators.

```
( ) ]
) ]
[ [ ]
```

3.1.5 Operators

There are 15 operators in JavaLite. They are tokens formed from ASCII characters. The following are all the operators.

```
= = - /
== < &&
+ <= ||
> * !=
>= % !
```

3.1.6 Literals

Literals in JavaLite represent any constant value of a primitive type (int, double, boolean) or the string type.

**Int Literals**

An integer literal is either the single ASCII digit 0, representing the integer zero, or any ASCII digit between 1 and 9, optionally followed by any sequence of ASCII digits between 0 and 9.
The int data type is a 32-bit signed primitive data type. The valid range of the integer literal is between \(-2^{31}, 147, 483, 648(-2^{31})\) and \(2^{31}, 147, 483, 647(2^{31}-1)\).

If the integer literal is larger than \(2^{31}, 147, 483, 647\) or less than the unary minus operator \(2^{31}, 147, 483, 648\), it will result in a compile-time error.

Examples of int literals:

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-5640</td>
<td>42014</td>
<td></td>
</tr>
</tbody>
</table>

**Double Literals**

A floating point literal has a whole number, a decimal, followed by another whole number.

In decimal float literals, at least one digit, and either a decimal, an exponent, or a float type suffix is required.

Floating point literals are always of type double.

Examples of floating point literals of double type:

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>.23</td>
<td>3.145</td>
<td>0.0</td>
<td>-1.3</td>
</tr>
</tbody>
</table>

**Boolean Literals**

The Boolean type is represented by the literals true and false.

**String Literals**

A string literal is any sequence of zero or more characters enclosed in double quotes. Characters such as newlines may be represented by escape sequences.

Examples of string literals:

```
"" // the empty string
"\n"
"Hello World!"
"This is a string literal\n that spans two lines" // this forms a string with \n two lines of text
```

### 3.2 Types and Variables

JavaLite is a statically typed language. All variables and expressions will have a known type at compile time. JavaLite is also a strongly typed language, and limits the values that a variable can hold. Variables must be assigned to a value when declared.
3.2.1 Primitive Types

JavaLite supports three primitive data types:

- int (number, 4 bytes)
- double (floating point number, 8 bytes)
- boolean (true or false, 1 byte)

Int Operations
JavaLite provides a number of operations for ints.

The first type of operators are the comparison operators, which result in a value of type int representing the boolean values. 0 indicates false while 1 indicates true. These operators are `<`, `<=`, `>`, `>=`, `==`, `!=`.

```java
2 < 4; // 1
5 <= 4; // 0
0 > -10; // 1
3 >= 3; // 1
2 == 0; // 0
2 != 0; // 1
```

The second type of operators are the numerical operators, which result in a value of type int. These operators are `+`, `−`, `∗`, `/`. Note that `/` returns values of type int, not double, so the result will be the floor of the division.

```java
10 + 2; // 12
-8 - 10; // -18
2 * 3; // 6
4 / 5; // 0
```

Double Operations
JavaLite provides a similar number of operations for doubles.

The first type of operators are the comparison operators, which result in a value of type int. These operators are `<`, `<=`, `>`, `>=`, `==`, `!=`.

```java
2.3 < 2.1; // 0
5.0 <= 4.3; // 0
0.0 > -4; // 1
3.2 >= 3.2; // 1
0.1 == 0.2; // 0
2.10 != 2.100; // 0
```

The second type of operators are the numerical operators, which result in a value of type double. These operators are `+`, `−`, `∗`, `/`.

```java
10.3 + 4.7; // 16.0
-8.0 - 10.0; // -18.0
2.4 * 3.0; // 7.2
4.0 / 5.0; // 0.8
```
### Boolean Operations

JavaLite provides two relational operators for booleans, which result in a value of type int. These are != and ==.

```java
false != false; // 0
true != false; // 1
false == false; // 1
true == false; // 0
```

JavaLite also provides three conditional operators for booleans, which result in a value of type int. These are !, ||, and &&.

```java
!false; // 1
!true; // 0
true || true; // 1
true || false; // 1
false || false; // 0
true && true; // 1
true && false; // 0
false && false; // 0
```

### 3.2.2 Objects

Aside from primitive types, JavaLite contains three kinds of reference types (class types, type variables, and array types). Class types have a type name. If arguments appear in a class type, then it is a parameterized type. Objects are a class instance or an array. Reference values are pointers to these objects, along with a special null reference that refers to no object.

### 3.2.3 Variables

Variables in JavaLite provide a storage location that is named. A variable in JavaLite must have a specific type to determine the size and layout of its memory. Once a type is declared, the value held in the variable must be of that exact type. A variable is declared using the operator ‘=’ and must be declared with a name of a valid string type.

```java
// valid variable declarations
int x = 1000;
string s = "s";

// invalid variable declarations
int = 3;
4;
int x = 4.5;
int y;
char c = "string";
bool b = 2;
```

Each variable be initialized with a value.
3.2.4 Strings

JavaLite uses the String class to create and manipulate strings. A string object has a constant value and represents sequences of Unicode characters. String literals are references to instances of a string object. Strings in JavaLite are mutable. Strings can be declared like follows.

```
string str = "Hello, World!";
```

String Operations

JavaLite provides six comparison operators for strings, which result in a value of type int. These are !=, ==, <, >, <=, >=.

```
"str" != "stra"; // 1
"hi" == "Hi"; // 0
"a" < "c"; // 1
"z" > "y"; // 1
"ab" <= "aab" // 0
"aaaab" >= "aaaab" // 1
```

JavaLite also provides the ability to use the '+' operator to concatenate two strings.

```
"str" + "ing"; // "string"
```

3.2.5 Arrays

An array in JavaLite is an object, and is assigned to variables of an object type. An array contains zero or more elements, in which the array is said to be empty in the case of zero elements. The elements in an array have no names and are referenced by array access expressions that use positive integer index values. In an array with n elements, with n being the length of the array, the elements of the array are referenced using integers from 0 to n-1.

An array type in JavaLite is written as the name of the elements type followed by one or more empty pairs of square brackets [], determining the depth of array nesting. All elements in an array are of the same type. If the components in an array are of type T, then the type of the array is written T[]. Elements in an array may be of any primitive or reference type. Arrays can be declared like follows.

```
string[] arr = ["hello", "world"];
```

Additionally, we can declare two-dimensional arrays as follows.

```
int[] arr1 = [1, 2, 3];
int[] arr2 = [4, 5, 6];
int[][] arr = [arr1, arr2];
```

Array Operations

JavaLite also provides the ability to use the "[]" operator to access elements of an array.
3.3 Statements and Expressions

3.3.1 Statements

Statements in JavaLite are of the following forms:

- If-Else Statements
- While Statements
- For Statements
- Expressions
- Return Statements

If-Else Statements

If statements in JavaLite allow for the conditional execution of a statement.

```java
if (expression) {
    statement1;
} else {
    statement2;
}
```

The expression must be of type boolean, or a compile-time error occurs.

In this case, the first statement will execute if expression evaluates to true and otherwise, the second statement will be executed. The conditional must be wrapped in ( ) and the statement to be executed must be wrapped in { }. However, a single line statement does not need to be wrapped in { }.

It is also possible to have standalone if statements or if-else if blocks without the trailing else.

```java
if (expression) {
    statement;
}
```

Here, statement would be executed if expression evaluates to true. Nothing is executed if the expression evaluates to false.

While Statements

While statements in JavaLite execute an expression and a statement repeatedly until the value of the expression evaluates to false.
while (expression) {
  statement;
}

The expression must be of type boolean, or a compile-time error occurs. Similar to If-Else Statements, the expressions must be wrapped in () and the statements must be wrapped in {} (unless one line).

For Statements
For statements in JavaLite execute an initial statement and then executes an expression, statement, and update statement until the expression evaluate to false.

for (initial; expression; update) {
  statement;
}

The expression must be of type boolean, or a compile-time error occurs.

Return Statements
In JavaLite, return statements are used in functions to return control (and data in some cases) to the invoker of a method.

T fun() {
  return expression;
}

Expression must be of type T, or a compile-time error occurs. In return statements, the expression is first evaluated then returned. For example,

int fun() {
  int x = 2;
  return x + 1;
}

In this case, \( x + 1 \) will be evaluated first, so the function will return 3.

3.3.2 Expressions
Expressions in JavaLite are a type of Statement that are of the following three syntactic forms:

- Primary Expressions
- Unary Operator Expressions
- Binary Operator Expressions

Primary Expressions
Primary Expressions in JavaLite are of the form of literals.

2; // evaluates to 2
true; // evaluates to true
Operation Expressions

Operator Expressions can be of unary or binary form. Operators specific to each of the types provided by JavaLite are discussed in 3.2.1, 3.2.1, 3.2.1, 3.2.4, and 3.2.5.

Assignment Operator

In JavaLite, the assignment operator ‘=' stores values into variables. This expression is evaluated right-to-left.

```
T var = expression;
```

The expression is first evaluated then stored into the variable of type T with name var. If the variable and evaluated value of the expression are not of the same type, a compile-time error occurs.

3.3.3 Operator Precedence

JavaLite executes operators with different precedence. The following table details the different levels of precedence with the highest levels of precedence at the top.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
<th>Associativity</th>
</tr>
</thead>
<tbody>
<tr>
<td>[]</td>
<td>Array Access</td>
<td>Left</td>
</tr>
<tr>
<td>.</td>
<td>Object Member Access</td>
<td></td>
</tr>
<tr>
<td>()</td>
<td>Parentheses</td>
<td>Left</td>
</tr>
<tr>
<td>!</td>
<td>Not</td>
<td>Right</td>
</tr>
<tr>
<td>*</td>
<td>Multiplication</td>
<td>Left</td>
</tr>
<tr>
<td>/</td>
<td>Division</td>
<td>Left</td>
</tr>
<tr>
<td>+</td>
<td>Addition</td>
<td>Left</td>
</tr>
<tr>
<td>-</td>
<td>Subtraction</td>
<td>Left</td>
</tr>
<tr>
<td>+</td>
<td>Array/String Concatenation</td>
<td></td>
</tr>
<tr>
<td>&lt;, &lt;=, &gt;, &gt;=</td>
<td>Comparison</td>
<td>Left</td>
</tr>
<tr>
<td>==, !=</td>
<td>Equality</td>
<td>Left</td>
</tr>
<tr>
<td>&amp;&amp;</td>
<td>Logical AND</td>
<td>Left</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>=</td>
<td>Assignment</td>
<td>Right</td>
</tr>
</tbody>
</table>

3.3.4 Functions

Functions in JavaLite are standalone. Functions take in a list of arguments and return one value.

```
T fun(R arg1, V arg2, ...) {
    /* do something */
    return expression;
}
```

Expression must be of type T. If not, a run-time error will occur. For example, we can write a function that increments values by 5 based on a boolean condition.
```java
int fun(int x, bool b) {
    if (b) {
        x = x + 5;
    }
    return x;
}
```

Functions are called by calling its name and providing the required number of parameters. A mismatch in the evaluated type of parameters or number of parameters will result in a compile-time error.

```java
int fun(int x) {
    return x + 5;
}
```

fun(); // compile-time error
fun("hi"); // compile-time error
fun(2.3); // compile-time error
fun(5); // evaluates to 10

### 3.4 Classes
JavaLite supports classes with variables. Classes can be declared using the following syntax.

```java
class Test {
    int x;
}
```

The constructor is called when a object is initialized. For example,

```java
x = Test(3);
```

An object of type Test and name x would be initialized by passing one argument of value 3 into the constructor function. The constructor is assumed to be of return type void.

#### 3.4.1 Dot Operator
In JavaLite, the dot operator, ".", is used to access values of an object (an instance of a class). These values are in the form of variables. The dot operator is used in the following format.

```java
x.var;
```

### 3.5 Built-In Functions
JavaLite supports built-in array structure with dynamic length similar to Python.
3.5.1 Arrays

`arr[n]`
`arr[n]` returns the nth element of the array.

```
int[] arr = [1, 2, 3];
arr[0]; // 1
arr[1]; // 2
```

`int length(T[] arr)`
`length(T[] arr)` returns the number of elements in arr.

```
int[] arr = [1, 2, 3];
length(arr); // 3
```

3.5.2 Strings

`string reverse(string str)`
`reverse(str)` returns a string in the reverse order of str.

```
string str = "hello";
str = reverse(str);
// str = "olleh"
```

`string upper(string str)`
`upper(str)` converts all characters of str to upper-case.

```
string str = "hello";
str = upper(str);
// str = "HELLO"
```

`string lower(string str)`
`lower(str)` converts all characters of str to lower-case.

```
string str = "Hello";
str = lower(str);
// str = "hello"
```

`string substring(string str, int a, int b)`
`substring(str, a, b)` returns the substring of str between index a (included) and b (excluded).

```
string str = "hello";
string str_sub = substring(str, 0, 2);
// str_sub = "he"
```

- If a < 0, a compile-time error will occur
- If b < a, a compile-time error will occur
- If b > len(str), a run-time error will occur
int indexOf(string str, string c)
indexOf(str, c) returns the index of the first occurrence of c in str. If there is no occurrence of c, -1 is returned.

```java
string str = "hello";
int a = str.indexOf(str, "h");
int b = str.indexOf(str, "l");
int c = str.indexOf(str, "k");
// a = 0, b = 2, c = -1
```

int len(string str)
len(str) returns the number of characters in str.

```java
string str = "hello";
int x = length(str);
// x = 5
```

string concat(string str1, string str2)
concat(str1, str2) returns the concatenated string of str1 to str2.

```java
string str1 = "hello";
string str2 = " world";
string str = concat(str1, str2);
// str = "hello world"
```

### 3.5.3 Print

JavaLite uses print to print strings to the terminal/console. Print can accept literals and variables of any type.

```java
print("hi"); // "hi"
print(10); // "10"
int x = 2;
string s = "hello";
print(x); // 2
print(s); // "hello"
```

- If print is called with more than one argument, a compile-time error will occur
- If print is called with any type other than int, double, boolean, or string, a compile-time error will occur

### 3.6 Syntax/Style Guide

- Any line that is not the beginning or end of a bracket block must end with a semi-colon. If not, a compile-time error will occur.
- All opening symbols must be matched with a closing symbol. More specifically, (), {}, and [] must be matched. If not, a compile-time error will occur.
- Each line should be a maximum of 80 characters wide.
• Proper indentation of 4 spaces should be used, however indentation is ignored by the compiler.

4 Project Plan

4.1 Planning Process

The main goals and milestones are set during the first few project meetings, which were decided in consultation with Professor Edwards and T.A. Jianan Yao. Throughout the evolution of the project, we adjusted the timeline of individual tasks based on the progress. We have regular weekly group meetings to share progress and discuss next steps, as well as additional meetings within the group and with T.A. when encountering roadblocks during development.

4.2 Specification Process

The motivation of our language is to create a Java-like language but with simplified syntax and semantics, so that it is more user-friendly for beginner programmers. We want our language to incorporate the basic and key functionality of the Java language, as well as to accommodate the programming and learning needs of beginners. Therefore, all the features we designed for the language are based on such a purpose. We first determined the data types, such as integer, strings, and arrays, that are commonly seen in Java and other programming languages. Next, as a key part of any object-oriented language, we defined the syntax for class. To better emulate Java, a class defined in our language also has a constructor. Finally, in order for the language to be more beginner-friendly, we borrowed the idea from Python, so that the programmer does not need to define a main function to compile their program. We wrote the language reference manual based on these ideas and also added some helpful built-in functions for string and array operations. After turning in our LRM, a few adjustments were made to the specifications as the situation called for it, but no notable changes that diverged from our main purposes.

4.3 Development Process

We divided each individual task feature wise. For the development of each feature, we followed the stages of the compiler architecture. In other words, we defined the syntax part of the feature first, including scanner.mll, ast.ml and parser.mly, which is followed by the semantic checking part, semant.ml and sast.ml. We would use ocamllyacc and menhir commands to check for any parsing error. For the semantic part, we wrote test cases that covered both success and failure situations and used the pretty print functions to print out the typed version of the test programs. After thoroughly testing the parts stated above, we moved on to the code generation part of the feature (codegen.ml). Often, we would use the clang command to generate a LLVM file for a C program that
realizes the feature we were implementing and use that as a reference for writing the corresponding OCaml code.

4.4 Testing Process

The test cases are written throughout the project development process along the implementation of each individual feature. The tests for failure cases are mostly done while defining the syntax and semantics, which check whether the expected error message occur when using syntax that is not compatible with our language. The tests for successful cases were written after the code generation part was completed for each feature. Every aspect and possible usage of a feature are examined through these tests. More details about testing are covered in Section 6 Test Plan.

4.5 Roles and Responsibilities

<table>
<thead>
<tr>
<th>Team Member</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frances Cao</td>
<td>System Architect</td>
</tr>
<tr>
<td>Mateo Maturana</td>
<td>Language Guru</td>
</tr>
<tr>
<td>Hongfei Chen</td>
<td>Manager</td>
</tr>
<tr>
<td>Ian Chen</td>
<td>Tester</td>
</tr>
</tbody>
</table>

Below are the features and the group members who are responsible for each feature:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Members</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrays</td>
<td>Hongfei, Frances, Mateo</td>
</tr>
<tr>
<td>Built-In Functions</td>
<td>Frances</td>
</tr>
<tr>
<td>Classes</td>
<td>Hongfei</td>
</tr>
<tr>
<td>Pretty-printer</td>
<td>Hongfei, Frances</td>
</tr>
<tr>
<td>Main Dependency Removal</td>
<td>Ian, Mateo, Hongfei</td>
</tr>
<tr>
<td>Print (All types)</td>
<td>Frances, Hongfei</td>
</tr>
<tr>
<td>Variable Assignment</td>
<td>Mateo, Hongfei</td>
</tr>
<tr>
<td>String Binops</td>
<td>Hongfei, Frances</td>
</tr>
</tbody>
</table>

Testing and the final report were completed by every group member. Mateo was in charge of typesetting and documentation for the final report and LRM.
4.6 Project Timeline

<table>
<thead>
<tr>
<th>Date</th>
<th>Milestone</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/23</td>
<td>Initial Meeting</td>
</tr>
<tr>
<td>1/24</td>
<td>JavaLite Proposal Idea</td>
</tr>
<tr>
<td>1/25</td>
<td>Proposal Finalized</td>
</tr>
<tr>
<td>2/13</td>
<td>Initial Scanner + Parser Work</td>
</tr>
<tr>
<td>2/19</td>
<td>LRM Draft</td>
</tr>
<tr>
<td>2/20</td>
<td>LRM Finalized + Scanner/Parser Work</td>
</tr>
<tr>
<td>2/23</td>
<td>TA Meeting</td>
</tr>
<tr>
<td>2/24</td>
<td>Finalized Parser + Scanner</td>
</tr>
<tr>
<td>3/15</td>
<td>TA Meeting</td>
</tr>
<tr>
<td>3/21</td>
<td>TA Meeting</td>
</tr>
<tr>
<td>4/18</td>
<td>Meeting to Establish Deliverables for Final Week</td>
</tr>
<tr>
<td>4/21</td>
<td>Final Week Check in</td>
</tr>
<tr>
<td>4/24</td>
<td>Meeting for Presentation Layout</td>
</tr>
<tr>
<td>4/26</td>
<td>Final Meeting; Presentation and Report</td>
</tr>
</tbody>
</table>

4.7 Project Log

Our project log is included in Section 9.

4.8 Software Development Environment

The main programming language that we used was OCaml with the LLVM library. OCamlyacc and OCamllex extensions were used for compiling the scanner and parser front end. We used C language to build some of the built-in functions and GCC was used to compile and link the C files. Each member used different development environments, but the most commonly used one was Visual Studio Code with its OCaml Platform extension.

4.9 Programming Style Guide

The general guidelines we followed while programming are as followings:

- Formatting: We followed the OCaml formatting and editing style. We made sure the code is indented properly so that the scopes would be understandable to the readers. Blocks of codes that are implemented for different features are separated by empty lines. The naming convention we used for variables is variable_name style (as opposed to camelCase).

- Commenting: Comments were included for each block of code explaining its general implementation idea and pointing out the corresponding feature for which it is used.
5 Architectural Design

5.1 Block Diagram

5.2 Scanner

File: scanner.mll

The scanner takes in a Javalite source program and translates it into a stream of tokens for identifiers, keywords, operators, and literals. Comments and whitespaces are ignored at this stage, and there is support for both single line and multiline comments. If any characters are found to be illegal, the scanner will throw an error. The tokens from the scanner are read in by the parser next to create the Abstract Syntax Tree.

Frances and Hongfei implemented the scanner.

5.3 Parser and AST

Files: parser.mly, ast.ml

The parser, written with OCamlYacc, takes in the tokens generated by the lexer and creates an Abstract Syntax Tree with the grammar defined in the parser and the datatypes defined in ast. The context-free grammar for the JavaLite syntax is defined in the parser through a set of production rules and precedence levels. If the source program is successfully parsed, it means the grammar is syntactically correct, and it will return the AST which is printed through pretty printing functions found in ast.ml. If there are violations, such as missing semicolons or missing arguments, the parser will throw errors.

Hongfei, Frances, and Mateo implemented the parser and AST.
5.4 Semantic Checking

Files: semant.ml, sast.ml

During semantic checking, the compiler traverses through the Abstract Syntax Tree created by the parser for type checking, and converts it to a semantically checked Abstract Syntax Tree. This step checks that types are consistent throughout each expression, variables are declared only once and in the proper scope, functions and classes are not redefined, among other things. This step is also important for built-in functions since LLVM requires types and argument types to be specified at compile time. If it passes semantic checking, it will return the SAST that can be printed through pretty printing functions found in sast.ml.

Hongfei and Frances implemented most of semantic checking, with a few additions from Mateo.

5.5 Code Generation

File: codegen.ml

The code generator takes in the semantically checked AST and constructs the LLVM code. The mappings between expressions and other features are accomplished with the OCaml LLVM module. On top of the int and double binary operations found in MicroC, we added string binary operations with the same modules with the help of LLVM build pointercast that creates a pointer to int instruction. The global print function is implemented in codegen through several helper functions that determine the type of the argument passed in. Further, it is at this step where modules are imported for the various built-in functions and converted to functions for Javalite in the call wrapper. In addition to functions, the code generation step also includes memory allocation for array and class type. More specifically, arrays are stored in the memory as pointers and each element is stored sequentially. For class, a named struct type would be created when a class is first declared and the values of all its fields are stored when the constructor gets called.

Hongfei and Frances implemented code generation.

5.6 C Libraries

Files: stringfuncs.c

Javalite provides several built-in functions for string and array manipulation that are reviewed in greater detail in section 3.5. Some of those functions were built with support from linked external C libraries.
Frances and Hongfei implemented the C libraries.

6 Test Plan

6.1 Representative Programs

GCD

```c
int gcd(int a, int b) {
    while (a != b)
        if (a > b) a = a - b;
        else b = b - a;
    return a;
}
```

```c
print(gcd(14,21));
print(gcd(8,36));
print(gcd(99,121));
```

Bubble Sort

```c
int[] sortingArr = [52,14,72,5,66,7,12,31,9,3,54,41,53,12,61];
int i = 0;
int j = 0;
int tmpForSwap = 0;

int length = length(sortingArr);

for (i = 0 ; i < length-1 ; i = i + 1) {
    for (j =0; j< length-i-1;j = j + 1){
        if (sortingArr[j] > sortingArr[j+1]){
            tmpForSwap = sortingArr[j];
            sortingArr[j] = sortingArr[j+1];
            sortingArr[j+1] = tmpForSwap;
        }
    }
}

for (i = 0 ; i < length; i = i + 1) {
    print(sortingArr[i]);
}
```

Counting Sort

```c
int[] countingSortArray = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0];
int countingSortArrayLength = length(countingSortArray);
int[] sortingArr = [32,17,31,5,26,13,47,31,9,3,26,41,43,12,11];
int length = length(sortingArr);
int i = 0;
int j = 0;
```
for (i = 0 ; i < length ; i = i + 1) {
    countingSortArray[sortingArr[i]] = countingSortArray[sortingArr[i]] + 1;
}

for (i = 0 ; i < countingSortArrayLength ; i = i + 1) {
    if(countingSortArray[i] > 0){
        for(j=0;j<countingSortArray[i];j = j+1){
            print(i);
        }
    }
}

Class Declaration and Usage

class Person {
    string name;
    int age;
    string phrase;
}

void sayhi(class Person p) {
    string n = concat(p.name, " say:");
    print(n);
    string s = p.phrase;
    for (int i = 0; i < p.age; i = i + 1) {
        s = concat(s, p.phrase);
    }
    print(s);
}

class Person alice = Person("Alice", 3, "hey");
sayhi(alice);

The LLVM output for these sections will be in Section 6.5.

6.2 Testing Tools

The syntax and execution results can be tested by running testall.sh file, this will run all the test cases and check the execution result. Test cases are stored under the /test folder. Success test cases follow the format of test-*.jl, and Fail cases follow the format of fail-*.jl. Success cases have its corresponding *.out file as its expected output, while fail cases have its corresponding *.err file as its expected error.

To test the execution result of a single program, we can use the BuildAndRun.sh script. This will simply build the target source code file and show the execution result. E.g: If we run "./BuildAndRun.sh SampleCode1.jl" , it will build and execute SampleCode1.jl, and print out the execution result in the console.

To benchmark the program performance, we can run the PerformanceBench.sh
tool, this will benchmark all the performance tests under /test/performance folder, and print out the test result to the console.

6.3 Automated CD/CI

For automated testing, we integrated CircleCI. Everytime a new commit is pushed, or a new pull request is created, CircleCI will spin up an container environment using columbiasedwards/plt docker image, and run the entire test suite to make sure everything passes. This automation process usually takes 20-60 seconds to complete.

The config.yml file for the automated testing process is located under the ./circleci directory in the source code repository.

6.4 Performance Benchmarking

The following is the performance benchmark result compared to Java. As the result shows, even though Javalite is easier to learn with a more Python-like syntax, the performance it has can be pretty close to Java. This may be improved even more by further optimizing the compiler.
To run Javalite performance benchmark, use `PerformanceBench.sh` in the root directory. All of the test cases used are under `/test/performance` folder.

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Javalite</th>
<th>Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int-Arithmetic Test</td>
<td>2873 ms</td>
<td>1121 ms</td>
</tr>
<tr>
<td>Double-Arithmetic Test</td>
<td>4692 ms</td>
<td>3094 ms</td>
</tr>
<tr>
<td>If-Condition Test</td>
<td>2949 ms</td>
<td>1991 ms</td>
</tr>
<tr>
<td>Array-Access Test</td>
<td>4835 ms</td>
<td>3278 ms</td>
</tr>
<tr>
<td>Class-Field-Access Test</td>
<td>5268 ms</td>
<td>2289 ms</td>
</tr>
</tbody>
</table>

## 6.5 Target Language Program (LLVM)

These are the llvm outputs to some of our representative programs.

### 6.5.1 GCD

```
; ModuleID = 'JavaLite'
source_filename = "JavaLite"

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

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

declare i8* @reverse(i8*)

declare i8* @upper(i8*)

declare i8* @lower(i8*)

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

declare i32 @indexOf(i8*, i8*)

declare i32 @len(i8*)

declare i8* @concat(i8*, i8*)

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

define i32 @length(i32*)

define i32 @main() {
entry:

%gcd_result = call i32 @gcd(i32 14, i32 21)
%printf = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4 i8* x i8]* @fmt, i32 0, i32 0), i32 %gcd_result)

%gcd_result1 = call i32 @gcd(i32 8, i32 36)
%printf2 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4 i8* x i8]* @fmt, i32 0, i32 0), i32 %gcd_result1)
```

6.5.2 Bubble Sort
; ModuleID = 'JavaLite'
source_filename = "JavaLite"

@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"

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

declare i8* @reverse(i8*)

declare i8* @upper(i8*)

declare i8* @lower(i8*)

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

declare i32 @indexOf(i8*, i8*)

declare i32 @len(i8*)

declare i8* @concat(i8*, i8*)

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

define i32 @length(i32*)

define i32 @main() {
  entry:
    %malloccall = tail call i8* @malloc(i32 mul (i32 ptrtoint (i1** getelementptr (i1*, i1** null, i32 1) to i32), i32 15))
    %arr = bitcast i8* %malloccall to i32**
    %arrptr = bitcast i32** %arr to i32*
    %arrelt = getelementptr i32, i32* %arrptr, i32 0
    store i32 52, i32* %arrelt
    %arrelt1 = getelementptr i32, i32* %arrptr, i32 1
    store i32 14, i32* %arrelt1
    %arrelt2 = getelementptr i32, i32* %arrptr, i32 2
    store i32 72, i32* %arrelt2
    %arrelt3 = getelementptr i32, i32* %arrptr, i32 3
    store i32 5, i32* %arrelt3
    %arrelt4 = getelementptr i32, i32* %arrptr, i32 4
    store i32 66, i32* %arrelt4
    %arrelt5 = getelementptr i32, i32* %arrptr, i32 5
    store i32 7, i32* %arrelt5
    %arrelt6 = getelementptr i32, i32* %arrptr, i32 6
    store i32 12, i32* %arrelt6
    %arrelt7 = getelementptr i32, i32* %arrptr, i32 7
    store i32 31, i32* %arrelt7
    %arrelt8 = getelementptr i32, i32* %arrptr, i32 8
    store i32 9, i32* %arrelt8
    %arrelt9 = getelementptr i32, i32* %arrptr, i32 9
    store i32 3, i32* %arrelt9
    %arrelt10 = getelementptr i32, i32* %arrptr, i32 10
    store i32 64, i32* %arrelt10
    %arrelt11 = getelementptr i32, i32* %arrptr, i32 11
    store i32 41, i32* %arrelt11
}

}
%arrelt12 = getelementptr i32, i32* %arrptr, i32 12
store i32 53, i32* %arrelt12
%arrelt13 = getelementptr i32, i32* %arrptr, i32 13
store i32 12, i32* %arrelt13
%arrelt14 = getelementptr i32, i32* %arrptr, i32 14
store i32 61, i32* %arrelt14
%malloccall15 = tail call i8* @malloc(i32 ptrtoint (i1** getelementptr (i1*, i1* null, i32 1) to i32))
%sortingArr = bitcast i8* %malloccall15 to i32**
store i32* %arrptr, i32** %sortingArr
%malloccall16 = tail call i8* @malloc(i32 ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to i32))
%i = bitcast i8* %malloccall16 to i32*
store i32 0, i32* %i
%malloccall17 = tail call i8* @malloc(i32 ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to i32))
%j = bitcast i8* %malloccall17 to i32*
store i32 0, i32* %j
%malloccall18 = tail call i8* @malloc(i32 ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to i32))
%tmpForSwap = bitcast i8* %malloccall18 to i32*
store i32 0, i32* %tmpForSwap
%malloccall19 = tail call i8* @malloc(i32 ptrtoint (i32* getelementptr (i32, i32* null, i32 1) to i32))
%length = bitcast i8* %malloccall19 to i32*
store i32 15, i32* %length
store i32 0, i32* %i
br label %while
while:
  ; preds = %merge59, %entry
  %i62 = load i32, i32* %i
  %length63 = load i32, i32* %length
  %tmp64 = sub i32 %length63, 1
  %tmp65 = icmp slt i32 %i62, %tmp64
  br i1 %tmp65, label %while_body, label %merge66
while_body:
  ; preds = %while
  store i32 0, i32* %j
  br label %while20
while20:
  ; preds = %merge, %while_body
  %j83 = load i32, i32* %j
  %length54 = load i32, i32* %length
  %i55 = load i32, i32* %i
  %tmp56 = sub i32 %length54, %i55
  %tmp57 = sub i32 %tmp56, 1
  %tmp58 = icmp slt i32 %j83, %tmp57
  br i1 %tmp58, label %while_body21, label %merge59
while_body21:
  ; preds = %while20
  %j22 = load i32, i32* %j
  %accpos = add i32 %j22, 0
  %sortingArr23 = load i32*, i32** %sortingArr
  %acceltptr = getelementptr i32, i32* %sortingArr23, i32 %accpos
  %accelt = load i32, i32* %acceltptr
  %j24 = load i32, i32* %j
  %tmp = add i32 %j24, 1
%accpos25 = add i32 %tmp, 0
%sortingArr26 = load i32*, i32** %sortingArr
%acceltptr27 = getelementptr i32, i32* %sortingArr26, i32 %accpos25
%accelt28 = load i32, i32* %acceltptr27
%tmp29 = icmp sgt i32 %accelt, %accelt28
br i1 %tmp29, label %then, label %else

merge: ; preds = %else, %then
%j51 = load i32, i32* %j
%tmp52 = add i32 %j51, 1
store i32 %tmp52, 132* %j
br label %while20

then: ; preds = %while_body21
%j30 = load i32, 132* %j
%accpos31 = add i32 %j30, 0
%sortingArr32 = load i32*, i32** %sortingArr
%acceltptr33 = getelementptr i32, i32* %sortingArr32, i32 %accpos31
%accelt34 = load i32, i32* %acceltptr33
store i32 %accelt34, i32* %tmpForSwap
%j35 = load i32, 132* %j
%accpos36 = add i32 %j35, 0
%sortingArr37 = load i32*, i32** %sortingArr
%j38 = load i32, 132* %j
%tmp39 = add i32 %j38, 1
%accpos40 = add i32 %tmp39, 0
%sortingArr41 = load i32*, i32** %sortingArr
%acceltptr42 = getelementptr i32, i32* %sortingArr41, i32 %accpos40
%accelt43 = load i32, i32* %acceltptr42
%arrelt44 = getelementptr i32, i32* %sortingArr37, i32 %accpos36
store i32 %accelt43, i32* %arrelt44
%j45 = load i32, 132* %j
%tmp46 = add i32 %j45, 1
%accpos47 = add i32 %tmp46, 0
%sortingArr48 = load i32*, i32** %sortingArr
%tmpForSwap49 = load i32, i32* %tmpForSwap
%arrelt50 = getelementptr i32, i32* %sortingArr48, i32 %accpos47
store i32 %tmpForSwap49, i32* %arrelt50
br label %merge

else: ; preds = %while_body21
br label %merge
merge59: ; preds = %while20
%i60 = load i32, 132* %i
%tmp61 = add i32 %i60, 1
store i32 %tmp61, 132* %i
br label %while
merge66: ; preds = %while
store i32 0, 132* %i
br label %while67
while67: ; preds = %while_body68, %
%length77 = load i32, i32* %length
while_body68: ; preds = %while67
%169 = load i32, i32* %i
%170 = add i32 %169, 0
%171 = load i32*, i32** %sortingArr
%172 = getelementptr i32, i32* %171, i32 %170
%173 = load i32, i32* %172
%174 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4 x i8]* @fmt, i32 0, i32 0), i32 %173)
%175 = load i32, i32* %i
%176 = add i32 %175, 1
store i32 %176, i32* %i
br label %while67
merge79: ; preds = %while67
ret i32 0
}
declare noalias i8* @malloc(i32)

6.5.3 Class

; ModuleID = 'JavaLite'
source_filename = "JavaLite"

%Person = type { i8*, i32, i8* }
@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"%f\0A\00"
@fmt.3 = private unnamed_addr constant [4 x i8] c"%d\0A\00"
@fmt.4 = private unnamed_addr constant [4 x i8] c"%g\0A\00"
@fmt.5 = private unnamed_addr constant [4 x i8] c"%s\0A\00"
@str.6 = private unnamed_addr constant [6 x i8] c"hey\00"
@fmt.7 = private unnamed_addr constant [4 x i8] c"%d\0A\00"
@fmt.8 = private unnamed_addr constant [4 x i8] c"%g\0A\00"
@fmt.9 = private unnamed_addr constant [4 x i8] c"%s\0A\00"
@str.10 = private unnamed_addr constant [6 x i8] c"say:\00"
declare i32 @printf(i8*, ...)
declare i8* @reverse(i8*)
declare i8* @upper(i8*)
declare i8* @lower(i8*)
declare i8* @substring(i8*, i32, i32)
declare i32 @indexOf(i8*, i8*)
declare i32 @len(i8*)
declare i8* @concat(i8*, i8*)
declare i8* @to_string(i8*, ...)
declare i32 @length(i32*)

define %Person* @Person(i8* %name, i32 %age, i8* %phrase) {
  entry:
  %malloccall = tail call i8* @malloc(i32 ptrtoint (i1** getelementptr (i1*, i1**
  → null, i32 1) to i32))
  %name1 = bitcast i8* %malloccall to i8**
  store i8* %name, i8** %name1
  %malloccall2 = tail call i8* @malloc(i32 ptrtoint (i32* getelementptr (i32, i32
  → * null, i32 1) to i32))
  %age3 = bitcast i8* %malloccall2 to i32*
  store i32 %age, i32* %age3
  %malloccall4 = tail call i8* @malloc(i32 ptrtoint (i1** getelementptr (i1*, i1
  → ** null, i32 1) to i32))
  %phrase5 = bitcast i8* %malloccall4 to i8**
  store i8* %phrase, i8** %phrase5
  %malloccall6 = tail call i8* @malloc(i32 ptrtoint (%Person* getelementptr (%
  → Person, %Person* null, i32 1) to i32))
  %constrObj = bitcast i8* %malloccall6 to %Person*
  %name7 = load i8*, i8** %name1
  %name8 = getelementptr inbounds %Person, %Person* %constrObj, i32 0, i32 0
  store i8* %name7, i8** %name8
  %age9 = load i32, i32* %age3
  %age10 = getelementptr inbounds %Person, %Person* %constrObj, i32 0, i32 1
  store i32 %age9, i32* %age10
  %phrase11 = load i8*, i8** %phrase5
  %phrase12 = getelementptr inbounds %Person, %Person* %constrObj, i32 0, i32 2
  store i8* %phrase11, i8** %phrase12
  %malloccall13 = tail call i8* @malloc(i32 ptrtoint (i1** getelementptr (i1*, i1
  → ** null, i32 1) to i32))
  %obj = bitcast i8* %malloccall13 to %Person**
  store %Person* %constrObj, %Person** %obj
  %obj14 = load %Person*, %Person** %obj
  ret %Person* %obj14
}
define i32 @main() {
  entry:
  %Person_result = call %Person* @Person(i8* getelementptr inbounds ([6 x i8], [6
  → x i8] @str.6, i32 0, i32 0), i32 3, i8* getelementptr inbounds ([4 x i8
  → ], [4 x i8] @str, i32 0, i32 0))
  %malloccall = tail call i8* @malloc(i32 ptrtoint (i1** getelementptr (i1*, i1**
  → null, i32 1) to i32))
  %alice = bitcast i8* %malloccall to %Person*
  store %Person* %Person_result, %Person** %alice
  %alice1 = load %Person*, %Person** %alice
  call void @sayhi(%Person* %alice1)
  ret i32 0
}
define void @sayhi(%Person* %p) {
  entry:
%malloccall = tail call i8* @malloc(i32 ptrtoint (i1** getelementptr (i1*, i1**
  ← null, i32 1) to i32))
%p1 = bitcast i8* %malloccall to %Person**
store %Person* %p, %Person** %p1
%p2 = load %Person*, %Person** %p1
%objld1 = load %Person, %Person* %p2
%name = extractvalue %Person %objld1, 0
%concat = call i8* @concat(i8* %name, i8* getelementptr inbounds ([6 x i8], [6
  ← x i8]* @str.10, i32 0, i32 0))
%malloccall3 = tail call i8* @malloc(i32 ptrtoint (i1** getelementptr (i1*, i1
  ← ** null, i32 1) to i32))
%n = bitcast i8* %malloccall3 to i8**
store i8* %concat, i8** %n
%n4 = load i8*, i8** %n
%printf = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4
  ← x i8]* @fmt.9, i32 0, i32 0), i8* %n4)
%n5 = load %Person*, %Person** %p1
%objld6 = load %Person, %Person* %p5
%phrase = extractvalue %Person %objld6, 2
%malloccall7 = tail call i8* @malloc(i32 ptrtoint (i1** getelementptr (i1*, i1
  ← ** null, i32 1) to i32))
%n6 = bitcast i8* %malloccall7 to i8**
%n7 = bitcast i8* %malloccall8 to i8**
store i32 0, i32* %i
while: ; preds = %while_body, %entry
%i15 = load i32, i32* %i
%p16 = load %Person*, %Person** %p1
%objld17 = load %Person, %Person* %p16
%age = extractvalue %Person %objld17, 1
%tmp18 = icmp slt i32 %i15, %age
br i1 %tmp18, label %while_body, label %merge
while_body: ; preds = %while_body
%phrase11 = extractvalue %Person %objld10, 2
%phrase11 = extractvalue %Person %phrase11, 1
%phrase12 = load i8*, i8** %s
%concat13 = call i8* @concat(i8* %phrase12, i8* %phrase11)
store i8* %concat13, i8** %s
%phrase13 = load i8*, i8** %s
%phrase14 = load i32, i32* %i
%phrase15 = add i32 %phrase14, 1
store i32 %phrase15, i32* %i
br label %while
merge: ; preds = %while
%phrase16 = load i8*, i8** %s
%printf20 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4
  ← x i8]* @fmt.9, i32 0, i32 0), i8* %phrase16)
ret void
}
declare noalias i8* @malloc(i32)
7 Lessons Learned

7.1 Frances Cao

This project requires full commitment and a ton of planning, which may be difficult for a few. OCaml is challenging at first, but after a bit of practice, it starts to click. One thing teams need to decide early on is whether they will be building their language with the MicroC architecture, which already compiles, or start fresh. Our team originally wanted to build our language from scratch, but realized it would be much better to work off of something that already works and build on top of it. Thus, another challenge is being able to understand the MicroC code, which is sizable, but once that happens, things start to get a little easier. Documentation is tough to understand at times, and there aren’t many resources online, so reach out to the professor and T.As whenever you’re stuck. Lastly, communication and collaboration is key, so make sure to voice issues early on. It’s important that all team members hold their weight more or less, as building a compiler is not trivial.

7.2 Mateo Maturana

This project, while challenging, as also proven to be very rewarding when looking at the final project. At the beginning it does seem daunting, so my advice to future groups is to first start early and plan aggressively. The more you plan early on, the less burden you place on yourself and the rest of your team later one when it comes to making decisions. Additionally, you want to have a clear idea about how you want to build your language so work done during some of the earlier weeks is actually effective and makes it into the final project rather than making sideways or backwards progress and delaying the progress of your group. OCaml is not a simple language to learn as it is not very similar to languages many of us have used before such as Java, C, Python. Therefore, it is important that you spend time in the earlier weeks getting to know OCaml, reading documentation, and feeling comfortable programming in it. My final piece of advice is to study the MicroC code intensively, regardless of whether you plan to build your compiler on top of it or independent to it. By doing this, you are able to more easily construct your own compiler code since you have a much better idea of how the Codegen and Semant work and won’t send as much time trying to debug these programs or just trying to figure out the syntax/logic.

7.3 Hongfei Chen

OCaml is an interesting programming language, but it could be challenging at first as it was new to almost everyone. However, when getting more familiar with OCaml as well as the LLVM library, programming becomes much more intuitive. One thing that I have learned, and I believe could be very helpful when implementing code generation part of the language, is starting with considering
memory allocation and how it would be done in C programming. Another aspect of the project was team collaboration - communication is crucial, especially when dealing with a project that has so many different components. If any issue was found, address it as early as possible and never have wishful thinking. Other than that, make good comments and notes, which would make it easier when trying to add in new features based on the old ones and this would also be great help when putting together the final report.

7.4 Ian Chen

Through this project, I learned how the compiler works and how a programming language is implemented. I now have a better understanding of how scanner, parser, ast, semant, sast, and codegen work, and how they can be modified to implement desired features for a new programming language. I also learned how new features can conflict with original codes and the importance of understanding the existing code before extending it. Before this project, I also didn’t have much experience with MakeFile, shell script, and docker. But after this project, I now have a better idea of how to utilize them to construct an automatic workflow which could save a lot of time. This is also my first time setting up an automatic CD/CI using the docker image. Initially, I tried GitHub action, it took me a while to understand the docker command, yml configuration, and how to spin up a testing environment using the above combined. Unfortunately, even though I was pretty sure that the docker was already running, we were still not able to run through the tests using GitHub action, which is pretty frustrating. I then try to switch to CircleCI, this time is much faster as many CD/CI concepts I learned can already be applied. The testing environment spins-up correctly and we are able to run through all the tests with an automated workflow. Overall, through this project, I learned many valuable lessons and plenty of useful skills that can be used for future projects.

8 Appendix

8.1 scanner.mll

```mll
(* Template taken from MicroC *)

{ open Parser }

let digit = ['0' - '9']
let digits = digit+

rule token = parse

[' ' '	' '' '
'] { token lexbuf } (* Whitespace *)
| "/*" { comment lexbuf } (* Multi-line Comments *)
| "//" { singcomment lexbuf } (* Single-line Comment *)
| '(' { LPAREN }
| ')' { RPAREN }
| '{' { LBRACE }
```
8.2 parser.mly

```ml
%{
open Ast
%

%token LPAREN RPAREN LBRACE RBRACE LBRACK RBRACK
%token SEMI COMMA ASSIGN
%token NOT EQ NEQ LT LEQ GT GEQ AND OR
%token PLUS MINUS TIMES DIVIDE
%token RETURN IF ELSE FOR WHILE
%token INT BOOL DOUBLE VOID STRING
%token ARRAY
%token CLASS DOT
%token NULL
%token <int> INT_LIT
%token <bool> BOOL_LIT
%token <string> VARIABLE DOUBLE_LIT STRING_LIT
%token EOF
%
%start program
%type <Ast.program> program

%nonassoc NOELSE
%nonassoc ELSE
%right ASSIGN
%left OR
%left AND
%left EQ NEQ
%left LT GT LEQ GEQ
%left PLUS MINUS
%right NOT
%
%}

program:
  decls EOF { $1 }

/*decls:
  ( [], [] )
| decls cdecl { (($2 :: fst $1), snd $1) }
| decls fdecl { (fst $1, ($2 :: snd $1)) }*/

decls:
  /* nothing */ { ([], [], []) }
| decls cdecl { let (stmt, cdecl, fdecl) = $1 in ($2::stmt, cdecl, fdecl) }
| decls cdecl { let (stmt, cdecl, fdecl) = $1 in ($2::cdecl, fdecl) }
| decls fdecl { let (stmt, cdecl, fdecl) = $1 in ($2::cdecl, fdecl) }

cdecl:
  CLASS VARIABLE LBRACE vdecl_list RBRACE
  { { cname = $2;
     fields = List.rev $4 } }
```
fdecl:
   typ VARIABLE formals_opt RPAREN LBRACE stmt_list RBRACE
   { typ = $1;
     fname = $2;
     formals = List.rev $4;
     locals = [];
     body = List.rev $7 }

/*
sdecl:
   stmt_list { $1 } */

formals_opt:
   /* nothing */ { [] }
   | formal_list { $1 }

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

typ:
   INT { Int }
   | BOOL { Bool }
   | DOUBLE { Double }
   | VOID { Void }
   | STRING { String }
   | typ ARRAY { Arr($1, 0) }
   | CLASS VARIABLE { Object($2) }
vdecl_list:
   /* nothing */ { [] }
   | vdecl_list vdecl { $2 :: $1 }
vdecl:
   typ VARIABLE SEMI { ($1, $2) }

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

stmt:
   expr SEMI { Expr $1 }
   | RETURN expr_opt SEMI { Return $2 }
   | LBRACE stmt_list 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_opt SEMI expr SEMI expr_opt RPAREN stmt { For($3, $5, $7, $9) }
   | WHILE LPAREN expr RPAREN stmt { While($3, $5) }

expr_opt:
   /* nothing */ { Noexpr }
   | expr { $1 }

expr:
   INT_LIT { IntLit($1) }
   | DOUBLE_LIT { DoubleLit($1) }
8.3  ast.ml

(* Abstract Syntax Tree and functions for printing it *)

type op = Add | Sub | Mult | Div | Equal | Neq | Less | Leq | Greater | Geq | And | Or

type uop = Neg | Not

type typ = Int | Bool | Double | Void | String | Arr of (typ * int) | Object of string | Any | Null

type bind = typ * string

type expr =

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

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

/* Arrays */
| VARIABLE LBRACK expr RBRACK { ArrayAccess($1, $3) } |
| LBRACK args_list RBRACK { ArrayLit($2) } |
| VARIABLE LBRACK expr RBRACK ASSIGN expr { ArrAssign($1, $3, $6) } |

/* CLASSES */
| VARIABLE DOT VARIABLE { ObjAccess($1, $3) } |
| VARIABLE DOT VARIABLE ASSIGN expr { ObjAssign($1, $3, $5) } |

/* VARIABLE DECLEARATION */
| typ VARIABLE ASSIGN expr { DecAssn($1, $2, $4) } |
| NULL { NullPtr(Null) } |

args_opt:
   /* nothing */ { [] } |
   | args_list { List.rev $1 } |

arg_list:
   expr { [$1] } |
   arg_list COMMA expr { $3 :: $1 } |
```
| IntLit  of int         |
| DoubleLit of string   |
| BoolLit of bool       |
| StrLit  of string     |
| Var of string         |
| Binop   of expr * op * expr |
| Unop    of uop * expr  |
| Assign  of string * expr |
| Call    of string * expr list |
| ArrayAccess of string * expr |
| ArrayLit of expr list |
| ArrAssign of string * expr * expr |
| ObjAccess of string * string |
| ObjAssign of string * string * expr |
| Construct of string * (string * expr) list |
| DecAssn of typ * string * expr |
| NullPtr of typ        |
| Noexpr               |

**type stmt =**
- Block of stmt list
- Expr of expr
- Return of expr
- If of expr * stmt * stmt
- For of expr * expr * expr * stmt
- While of expr * stmt

**type func_decl =**
- typ : typ;
- fname : string;
- formals : bind list;
- locals : bind list;
- body : stmt list;

**type class_decl =**
- cname: string;
- fields: bind list;

**type program =** stmt list * class_decl list * func_decl list

(* Pretty-printing functions *)

**let string_of_op =** function
  - Add -> "+
  - Sub -> "-"
  - Mult -> "*
  - Div -> "/"
  - Equal -> "=="
  - Neq -> "!="
  - Less -> "<"
  - Leq -> "<="
  - Greater -> ">
  - Geq -> ">="
  - And -> "&&"
  - Or -> "||"
```
let string_of_uop = function
| Neg -> "-" |
| Not -> "!"

let rec string_of_typ = function
| Int -> "int" |
| Bool -> "bool" |
| Double -> "double" |
| Void -> "void" |
| String -> "string" |
| Any -> "any" |
| Null -> "null"
| Arr(t, _) -> string_of_typ t ^ "[]"
| Object(s) -> "class " ^ s

let rec string_of_expr = function
| IntLit(l) -> string_of_int l |
| DoubleLit(l) -> l |
| BoolLit(true) -> "true" |
| BoolLit(false) -> "false" |
| StrLit(s) -> "" ^ s ^ ""
| Var(s) -> s |
| Binop(e1, o, e2) -> string_of_expr e1 ^ " " ^ string_of_op o ^ " " ^ string_of_expr e2 |
| Unop(o, e) -> string_of_uop o ^ string_of_expr e |
| Assign(v, e) -> v ^ " = " ^ string_of_expr e |
| Call(f, el) -> f ^ "(" ^ String.concat ", " (List.map string_of_expr el) ^ ")" |
| ArrayAccess (s, e) -> s ^ "\[" ^ string_of_expr e ^ "\]" |
| ArrayLit(e) -> "]" ^ String.concat ", " (List.map string_of_expr (List.rev e)) ^ "]" |
| ArrayAssign(s, e1, e2) -> s ^ "[" ^ string_of_expr e1 ^ " = " ^ string_of_expr e2 |
| ObjAccess(s1, s2) -> s1 ^ "." ^ s2 |
| ObjAssign(s1, s2, e) -> s1 ^ "." ^ s2 ^ " = " ^ string_of_expr e |
| Construct(s, _) -> "Constructor " ^ s |
| DecAsn(t, s, e) -> string_of_typ t ^ " = " ^ string_of_expr e |
| NullPtr(t) -> string_of_typ t ^ " Null" |
| Noexpr -> ""

let rec string_of_stmt = function
| Block(stmts) -> "{
" ^ String.concat "" (List.map string_of_stmt stmts) ^ "}\n"
| Expr(expr) -> string_of_expr expr ^ ";\n"
| Return(expr) -> "return " ^ string_of_expr expr ^ ";\n"
| If(e, s, Block([])) -> "if (" ^ string_of_expr e ^ ")\n" ^ string_of_stmt s |
| If(e, s1, s2) -> "if (" ^ string_of_expr e ^ ")\n" ^ string_of_stmt s1 ^ "else\n" ^ string_of_stmt s2 |
| For(e1, e2, e3, s) -> "for (" ^ string_of_expr e1 ^ "; " ^ string_of_expr e2 ^ "; " ^ string_of_expr e3 ^ ")\n" ^ string_of_stmt s |
| While(e, s) -> "while (" ^ string_of_expr e ^ ")\n" ^ string_of_stmt s |

let string_of_vdecl (t, id) = string_of_typ t ^ " = " ^ id ^ ";\n"
let string_of_fdecl fdecl = 
  string_of_typ fdecl.typ ^ " " ^
  fdecl.fname ^ "(String.concat ", " (List.map snd fdecl.formals) ~
)\n(\n" ^
  String.concat " (List.map string_of_vdecl fdecl.locals) ~
  String.concat " (List.map string_of_stmt fdecl.body) ~
)\n"

let string_of_cdecl cdecl = 
  "class " ^ cdecl.cname ^ " {\n" ^
  String.concat " (List.map string_of_vdecl cdecl.fields) ~
  "}\n"

let string_of_program (statements, classes, funcs) = 
  (* String.concat " (List.map string_of_vdecl vars) ~ "\n" ~ *)
  String.concat " (List.map string_of_cdecl classes) ~ "\n" ~
  String.concat " (List.map string_of_fdecl funcs) ~ "\n" ~
  String.concat " (List.map string_of_stmt statements)

8.4 semant.ml

(* Semantic checking for the MicroC compiler *)
open Ast
open Sast

module StringMap = Map.Make(String);

module HashtblString =
  struct
    type t = string
    let equal = ( = )
    let hash = Hashtbl.hash
  end;;

module StringHash = Hashtbl.Make(HashtblString);

(* 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 (statements, classes, 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)
let check_bind tbl (t, n) =  
  if t = Void then raise (Failure ("illegal void " ^ n)) else  
  if (StringHash.mem tbl n) then raise (Failure ("duplicate " ^ n)) else  
    StringHash.add tbl n t; tbl  
  in

(** Check classes ****)

(* Add class name to symbol table *)
let add_class map cd =  
  let dup_err = "duplicate class " ^ cd.cname in  
  let n = cd.cname in  
  match cd with  
    _ when StringMap.mem n map -> raise (Failure dup_err)  
  | _ -> StringMap.add n cd map  
  in

(* Collect all class names into one symbol table *)
let class_decls = List.fold_left add_class StringMap.empty classes  
  in

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

let check_class cls =  
  (* Make sure no fields are void or duplicates *)
  ignore (check_binds "field" cls.fields);  
  { scname = cls.cname;  
    sfields = cls.fields }  
  in

(* find the field bind in class *)
let find_field (cname, field) =  
  let cd = find_class cname in  
  let add_field m (t,n) = StringMap.add n (t,n) m in  
  let f_names = List.fold_left add_field StringMap.empty cd.fields in  
  try StringMap.find field f_names  
  with Not_found ->  
    raise (Failure ("unrecognized field " ^ field ^ " in class " ^ cname))  
  in

(** Check functions ****)

(* Collect function declarations for built-in functions: no bodies *)
let built_in_print_decls =  
  let add_bind map name = StringMap.add name {  
    typ = Void;  
    fname = name;  
    formals = [(Any, "x")];  
    locals = []; body = [] } map  
  in List.fold_left add_bind StringMap.empty [ "print" ]
(* Array print support *)
(* let built_in_print_decls =
  StringMap.add "to_string" {
    typ = String;
    fname = "to_string";
    formals = [Any, "x"];
    locals = [];
    body = [] } built_in_print_decls
in *)

let built_in_decls =
  let add_str_func map (name, ty) = StringMap.add name {
    typ = String;
    fname = name;
    formals = [(ty, "x")];
    locals = [];
    body = [] } map
  in List.fold_left add_str_func built_in_print_decls [ ("reverse", String);
    ("upper", String);
    ("lower", String) ]
in

let built_in_decls =
  StringMap.add "substring" {
    typ = String;
    fname = "substring";
    formals = [(String, "str"); (Int, "from"); (Int, "to")];
    locals = [];
    body = [] } built_in_decls
in

let built_in_decls =
  StringMap.add "concat" {
    typ = String;
    fname = "concat";
    formals = [(String, "str1"); (String, "str2")];
    locals = [];
    body = [] } built_in_decls
in

let built_in_decls =
  StringMap.add "indexOf" {
    typ = Int;
    fname = "indexOf";
    formals = [(String, "str"); (String, "find")];
    locals = [];
    body = [] } built_in_decls
in

let built_in_decls =
  StringMap.add "len" {
    typ = Int;
    fname = "len";
    formals = [(String, "str")];
    locals = [];
    body = [] } built_in_decls
in
let built_in_decls = StringMap.add "length" {
  typ = Int;
  fname = "length";
  formals = [(Any, "arr")];
  locals = [];
  body = [] } built_in_decls

let main_decl = {
  typ = Int;
  fname = "main";
  formals = [];
  locals = [];
  body = List.rev statements }

let functions = main_decl :: functions in

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

let form_constructor_body (cname, field_lst) = let field_args = List.map (fun (_,n) -> (n,Var(n))) field_lst in
let return_body = [ Return (Var("obj")) ] in
let constructor = Expr (DecAssn(Object(cname), "obj", Construct(cname, field_args))) in
let body_rev = constructor :: return_body in

let form_constructor cdecl = let cn = cdecl.cname in
{ typ = Object(cn);
  fname = cn;
  formals = cdecl.fields;
  locals = [];
  body = form_constructor_body (cn, cdecl.fields) }
in

let all_constructors = List.map (fun cd -> form_constructor cd) classes in

let all_functions = List.fold_left (fun l c -> c::l) functions all_constructors in

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

let find_func s =
  try StringMap.find s function_decls
  with Not_found -> raise (Failure ("unrecognized function " ^ s))

let rec check_arr (arrtyp, lv) =
  (match arrtyp with
   | Arr(ty,_) -> check_arr (ty, lv+1)
   | _ -> (arrtyp, lv))

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 "formal" func.formals;
  check_binds "local" func.locals;
  let tbl = StringHash.create 10 in
  let formal_tbl = StringHash.create 5 in

  let check_assign lvaluet rvaluet err =
    if lvaluet = Any then rvaluet else
      if lvaluet = rvaluet then lvaluet else
        (match lvaluet with
         | Object(_) -> if rvaluet = Null then lvaluet else raise (Failure err)
         | Arr _ -> (match rvaluet with
                     | Arr _ -> let r_arr = check_arr (rvaluet, 0) in
                       let l_arr = check_arr (lvaluet, 0) in
                       if r_arr = l_arr then rvaluet else raise (Failure err)
         )
        )

let type_of_identifier s =
  try StringHash.find tbl s
  with Not_found ->
    try StringHash.find formal_tbl s
    with Not_found -> raise (Failure ("undeclared identifier " ^ s))

let is_arr_ty (v, ty) = match ty with
  | Arr(t,_) ->
    if t = Void then raise (Failure ("void type array " ^ v ^ " is not allowed"))
  else t
_ -> raise (Failure ("cannot access an element in variable " ^ v ^ " of type " ^ string_of_typ ty))

(* Return a semantically-checked expression, i.e., with a type *)
let rec expr e =
  let check_assign_null e lt err =
    let (rt, e') = expr e in
    let ty = check_assign lt rt err in
    if e' = (SNullPtr Null) then (ty, (SNullPtr ty)) else (ty, e')
  in

  (match e with
  | IntLit l -> (Int, SIntLit l)
  | DoubleLit l -> (Double, SDoubleLit l)
  | BoolLit b -> (Bool, SBoolLit b)
  | StrLit s -> (String, SStrLit s)
  | Noexpr -> (Void, SNoexpr)
  | NullPtr t -> (t, SNullPtr t)
  | Var s -> (type_of_identifier s, SVar s)
  | Assign(var, e) as ex ->
    let lt = type_of_identifier var in
    let (rt, _) = expr e in
    let err = "illegal assignment " ^ string_of_typ lt ^ " = " ^
              string_of_typ rt ^ " in " ^ string_of_expr ex in
    let (ty, e') = check_assign_null e lt err in
    (* update array size *)
    in let _ = (match ty with
      Arr _ -> StringHash.replace tbl var ty
      | _ -> ignore 1)
    in (ty, SAssign(var, (ty, e')))
  | DecAssn(ty, var, e) as decassgn ->
    ignore (check_bind tbl (ty, var));
    let (rt, _) = expr e in
    let err = "illegal assignment " ^ string_of_typ ty ^ " = " ^
              string_of_typ rt ^ " in " ^ string_of_expr decassgn in
    let (ty, e') = check_assign_null e ty err in
    (* update array size *)
    in let _ = (match ty with
      Arr _ -> StringHash.replace tbl var ty
      | _ -> ignore 1)
    in (ty, SDecAssign(ty, var, (ty, e')))
  | Unop(op, e) as ex ->
    let (t, e') = expr e in
    let ty = match op with
      Neg when t = Int || t = Double -> t
      | Not when t = Bool -> Bool
      | _ -> raise (Failure ("illegal unary operator " ^
                               string_of_uop op ^ " string_of_typ t ^
                               " in " ^ string_of_expr ex))
    in (ty, SUlop(op, (t, e')))
  | Binop(e1, op, e2) as e ->
    let (t1, e1') = expr e1 in
    let (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 | Mult | Div when same && t1 = Int -> Int
| Add | Sub | Mult | Div when same && t1 = Double -> Double
| Equal | Neq when same -> Bool

(* special case for null *)
| Equal | Neq when (t2 = Null) -> Bool
| Less | Leq | Greater | Geq

when same && (t1 = Int || t1 = Double || t1 = String) ->
  Bool
| And | Or when same && t1 = Bool -> Bool
| Add when same && t1 = String -> String
| _ -> raise (Failure ("illegal binary operator " ^
    string_of_typ t1 ^ " " ^ string_of_op op ^ " " ^
    string_of_typ t2 ^ " in " ^ string_of_expr e))

let fd = find_func fname in
let param_length = List.length fd.formals 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, n) e =
    let (et, _) = expr e in
    let err = "illegal argument found " ^
      string_of_typ et ^ " expected " ^
      string_of_typ ft ^ " in " ^ string_of_expr e
    in (ty, SBinop((t1, e1'), op, (t2, e2')))
  in
  let args' = List.map2 check_call fd.formals args
  in (fd.typ, SCall(fname, args'))

| ObjAccess(obj, f) as objaccess ->
  let cname =
    let obj_ty = type_of_identifier obj in
    (match obj_ty with
     Object(cn) -> cn
    | _ -> raise (Failure (obj ^ " is not a class object in " ^
        string_of_expr objaccess)))
  in
  let (field_ty, _) = find_field (cname, f) in
  (match ty with
   Arr _ -> StringHash.replace formal_tbl n ty
   | _ -> ignore 1) in (ty, e')

in
let args' = List.map2 check_call fd.formals args
in (fd.typ, SCall(fname, args'))

| ObjAssign(obj, f, e) as objassign ->
  let cname =
    let obj_ty = type_of_identifier obj in
    (match obj_ty with
     Object(cn) -> cn
    | _ -> raise (Failure (obj ^ " is not a class object in " ^
        string_of_expr objassign))
     string_of_expr objassign))
  in
  let (field_ty, _) = find_field (cname, f) in
  let t = ObjAccess(obj, f) in
  let (rt, _) = expr e in
let err = "illegal assignment " ^ string_of_typ field_ty ^ " = " ^
  string_of_typ rt ^ " in " ^ string_of_expr objassign in
let (ty, e') = check_assign_null e field_ty err
in (ty, SObjAssign(obj, cname, f, (ty, e')))
| ArrayLit(el) as arraylit ->
  (* check if types of expr are consistent *)
  let ty_inconsistent_err = "inconsistent types in array " ^
  string_of_expr arraylit in
  let fst_e = List.hd el in
  let (fst_ty, _) = expr fst_e in
  let (arr_ty_len, arr_ty_e) = List.fold_left (fun (t, l) e ->
    let (et, e') = expr e in
    let is_arr = (
      match et with
      | Arr _ -> true
      | _ -> false) in
    if ((et = fst_ty) || is_arr) then (t+1, (et, e')::l) else (t, (et, e')::l)) (0,[]) el
  in if arr_ty_len != List.length el
  then raise (Failure ty_inconsistent_err)
  (* determine arr type *)
  else let arr_ty = Arr(fst_ty, arr_ty_len)
in (arr_ty, SArrayLit(arr_ty_e))
| Construct(cname, args) ->
  let args' = List.map (fun (s,e) -> (s, (expr e))) args
  in (Object(cname), SConstruct(cname, args'))
| ArrayAccess(v, e) as arrayacess ->
  (* check if type of e is an int *)
  let (t, e1') = expr e in
  if t != Int then raise (Failure (string_of_expr e ^ " is not of int type in " ^ string_of_expr arrayacess))
  (* check if variable is array type *)
  let v_ty = type_of_identifier v in
  let e_ty = is_arr_ty (v, v_ty) in
  (e_ty, SArrayAccess(v, (t, e1')))
| ArrAssign(v, e1, e2) as arrassign ->
  (* check if type of e is an int *)
  let (t, e1') = expr e1 in
  if t != Int then raise (Failure (string_of_expr e1 ^ " is not of int type in " ^ string_of_expr arrassign))
  (* check if variable is array type *)
  let v_ty = type_of_identifier v in
  let e_ty = is_arr_ty (v, v_ty) in
  (e_ty, SArrayAssign(v, (t, e1'), (ty, e2')))
in (ty, SArrayAssign(v, (t, e1'), (ty, e2')))

let check_bool_expr e =
  let (t', e') = expr e in
  if t' != Bool then raise (Failure err) else (t', e')
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) ->
    let e1' = expr e1 in
    SFor(e1', check_bool_expr e2, expr e3, check_stmt st)
  | While(p, s) -> SWhile(check_bool_expr p, check_stmt s)
  | Return e -> let (t, e') = expr e in
    if "main" = func.fname then
      raise (Failure("expected no return statement in outer body")) else
    let err = "return gives " ^ string_of_typ t ^ " expected " ^
      string_of_typ func.typ ^ " in " ^ string_of_expr e in
    if t = func.typ then SReturn (t, e') else
      (match t with
        Arr _ -> (match func.typ with
          Arr _ -> let r_arr = check_arr (func.typ, 0) in
            let l_arr = check_arr (t, 0) in
            if r_arr = l_arr then SReturn (func.typ, e')
            else raise (Failure err)
        | _ -> raise (Failure err))
    | _ -> raise (Failure err))
  | 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 -> let c = check_stmt s in
        c :: check_stmt_list ss
      | [] -> []
in SBlock(check_stmt_list sl)

in (* body of check_function *)
{ styp = func.typ;
  sfname = func.fname;
  sformals = func.formals;
  slocals = func.locals;
  sbody = match check_stmt (Block func.body) with
  | _ -> raise (Failure ("internal error: block didn't become a block?"))
}
in (List.map check_class classes, List.map check_function all_functions)

(* 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) ->
    let e1' = expr e1 in
    SFor(e1', check_bool_expr e2, expr e3, check_stmt st)
  | While(p, s) -> SWhile(check_bool_expr p, check_stmt s)
  | Return e -> let (t, e') = expr e in
    if "main" = func.fname then
      raise (Failure("expected no return statement in outer body")) else
    let err = "return gives " ^ string_of_typ t ^ " expected " ^
      string_of_typ func.typ ^ " in " ^ string_of_expr e in
    if t = func.typ then SReturn (t, e') else
      (match t with
        Arr _ -> (match func.typ with
          Arr _ -> let r_arr = check_arr (func.typ, 0) in
            let l_arr = check_arr (t, 0) in
            if r_arr = l_arr then SReturn (func.typ, e')
            else raise (Failure err)
        | _ -> raise (Failure err))
    | _ -> raise (Failure err))
  | 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 -> let c = check_stmt s in
        c :: check_stmt_list ss
      | [] -> []
in SBlock(check_stmt_list sl)

in (* body of check_function *)
{ styp = func.typ;
  sfname = func.fname;
  sformals = func.formals;
  slocals = func.locals;
  sbody = match check_stmt (Block func.body) with
  | _ -> raise (Failure ("internal error: block didn't become a block?"))
}
in (List.map check_class classes, List.map check_function all_functions)

8.5 sast.ml

(* Semantically-checked Abstract Syntax Tree and functions for printing it *)
open Ast

(type sexpr = typ * sx

and sx = 
SIntLit of int
SDoubleLit of string
SBoolLit of bool
SStrLit of string
SVar of string
SBinop of sexpr * op * sexpr
SUnop of uop * sexpr
SAssign of string * sexpr
SCall of string * sexpr list
SArrayAccess of string * sexpr
SArrayLit of sexpr list
SArrayAssign of string * sexpr * sexpr
SObjAccess of string * string * string
SObjAssign of string * string * string * sexpr
(* cname * (field_name * field_value) list *)
SConstruct of string * (string * sexpr) list
SDecAssn of typ * string * sexpr
SNullPtr of typ
SNoexpr

type sstmt =
SBlock of sstmt list
SExpr of sexpr
SReturn of sexpr
SIf of sexpr * sstmt * sstmt
SFor of sexpr * sexpr * sexpr * sstmt
SWhile of sexpr * sstmt

type sfunc_decl = { 
styp : typ;
sfname : string;
sformals : bind list;
slocals : bind list;
sbody : sstmt list;
}

type sclass_decl = { 
scname : string;
sfields : bind list;
}

type sprogram = sclass_decl list * sfunc_decl list

(* Pretty-printing functions *)

let rec string_of_sexpr (t, e) =
"(" ^ string_of_typ t ^ " : " ^ (match e with
SIntLit(l) -> string_of_int l
| SBoolLit(true) -> "true"
| SBoolLit(false) -> "false"
| SDoubleLit(l) -> l
| SStrLit(s) -> "\"" ^ s ^ "\""
| SVar(s) -> s
| SBinop(e1, o, e2) ->
string_of_sexpr e1 ^ " " ^ string_of_op o ^ " " ^ string_of_sexpr e2
| SUnop(o, e) -> string_of_uop o ^ string_of_sexpr e

50
| SAssign(v, e) -> v ^ " = " ^ string_of_sexpr e |
| SCall(f, el) -> f ^ "(" ^ String.concat ", " (List.map string_of_sexpr el) ^ ")"
| SArrayAccess(s, e) -> s ^ "[" ^ string_of_sexpr e ^ "]"
| SArrayLit(e) -> "[" ^ String.concat ", " (List.map string_of_sexpr (List.rev e)) ^ "]"
| SArrAssign(s, e1, e2) -> s ^ "[" ^ string_of_sexpr e1 ^ "] = " ^ string_of_sexpr e2 |
| SObjAccess(s1, c, s2) -> s1 ^ "(class " ^ c ^ ")." ^ s2 |
| SObjAssign(s1, c, s2, e) -> s1 ^ "(class " ^ c ^ ")." ^ s2 ^ " = " ^ string_of_sexpr e |
| SConstruct(s, _) -> "Constructor " ^ s |
| SDecAccess(t, s, e) -> string_of_typ t ^ " s" |
| SDecAssign(t, s, e) -> string_of_typ t ^ " = " ^ string_of_sexpr e |
| SNullPtr(t) -> string_of_typ t ^ " Null" |
| SNoexpr -> ""
|)

let print_sstring (_, e) = match e with
| SStrLit(s) -> s
| _ -> raise (Failure "error: only string type allowed")

let compare_sstring ((_, e1),(_,e2)) = if SStrLit(e1) = SStrLit(e2) then true else false

let rec string_of_sstmt = function
| SBlock(stmts) -> "{" ^ String.concat "" (List.map string_of_sstmt stmts) ^ "}\n"
| SExpr(expr) -> string_of_expr expr ^ " ;\n"
| SReturn(expr) -> "return " ^ string_of_expr expr ^ " ;\n"
| SIf(e, s, SBlock([])) -> "if (" ^ string_of_expr e ^ ")\n" ^ string_of_sstmt s
| SIf(e, s1, s2) -> "if (" ^ string_of_expr e ^ ")\n" ^ string_of_sstmt s1 ^ " else\n" ^ string_of_sstmt s2
| SFor(e1, e2, e3, s) -> "for (" ^ string_of_expr e1 ^ ") ; " ^ string_of_expr e2 ^ ") ; " ^ string_of_expr e3 ^ ")\n" ^ string_of_sstmt s
| SWhile(e, s) -> "while (" ^ string_of_expr e ^ ")\n" ^ string_of_sstmt s

let string_of_sfdecl fdecl = string_of_typ fdecl.styp ^ ";" ^ string_of_sdecl fdecl.sfdecl

let string_of_scdecl cdecl = "class " ^ cdecl.scname ^ "}\n"

let string_of_sprogram (classes, funcs) = (* String.concat "\n" (List.map string_of_vdecl vars) ^ "\n" * ) String.concat "\n" (List.map string_of_scdecl classes) ^ "\n" ^ String.concat "\n" (List.map string_of_sfdecl funcs)
8.6 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);

module HashtblString =
struct
  type t = string
  let equal = (=)
  let hash = Hashtbl.hash
end;;

module StringHash = Hashtbl.Make(HashtblString);

(* translate : Sast.program -> Llvm.module *)

let translate (classes, functions) =
  let context = L.global_context () in
  let the_module = L.create_module context "JavaLite" in

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

  (* string type *)
  let string_t = L.pointer_type i8_t in

  let add_class_typ map cd =
    let cname = cd.scname in
    let typ_lst = List.map (fun (t,_) -> t) cd.sfields in
    StringMap.add cname typ_lst map in
  (* store all class name and corresponding field type list *)
  let class_types = List.fold_left add_class_typ StringMap.empty classes in
(* get the list of field types for a class *)

let find_field_typs cname =
  try StringMap.find cname class_types
  with Not_found -> raise (Failure ("find_field_typs not found " ^ cname)) in

let add_field_name map cd =
  let cname = cd.scname in
  let field_lst = List.map (fun (_,n) -> n) cd.sfields in
  StringMap.add cname field_lst map in

let class_fields = List.fold_left add_field_name StringMap.empty classes in

let add_cd m cd = StringMap.add cd.scname cd.sfields m in

let class_ty_fields = List.fold_left add_cd StringMap.empty classes in

let get_field_ind (cname, fname) =
  let f_lst =
    try StringMap.find cname class_fields
    with Not_found -> raise (Failure ("get_field_ind not found " ^ cname)) in
  let f_lst_ind = List.mapi (fun i n -> (n, i)) f_lst in
  snd (List.hd (List.filter (fun (n, _) -> n = fname) f_lst_ind)) in

let class_tbl = StringHash.create 10 in

let find_struct cls =
  try StringHash.find class_tbl cls
  with Not_found ->
    let cls_struct = L.named_struct_type context cls in
    StringHash.add class_tbl cls cls_struct; cls_struct in

let rec ltype_of_typ = function
  A.Int -> i32_t
  | A.Bool -> i1_t
  | A.Double -> double_t
  | A.Void -> void_t
  | A.String -> string_t
  | A.Arr(ty,_) -> L.pointer_type (ltype_of_typ ty)
  | A.Object(cls) -> L.pointer_type (find_struct cls)
  | _ -> raise (Failure "Unmatched LLVM type in ltype_of_typ") in

(* Import modules for our built-in functions and print *)

let printf_t : L.lltype = L.var_arg_function_type i32_t [| string_t |] in

let printf_func : L.llvalue = L.declare_function "printf" printf_t the_module in

let reversestring_t : L.lltype = L.function_type string_t [| string_t |] in

let reversestring_func : L.llvalue = L.declare_function "reverse" reversestring_t the_module in
let stringupper_t : L.lltype = L.function_type string_t [ | string_t | ] in
let stringupper_func : L.llvalue = L.define_function "upper" stringupper_t the_module in

let stringlower_t : L.lltype = L.function_type string_t [ | string_t | ] in
let stringlower_func : L.llvalue = L.define_function "lower" stringlower_t the_module in

let stringsubstring_t : L.lltype = L.function_type string_t [ | string_t ; i32_t ; i32_t | ] in
let stringsubstring_func : L.llvalue = L.define_function "substring" stringsubstring_t the_module in

let stringindexof_t : L.lltype = L.function_type i32_t [ | string_t ; string_t | ] in
let stringindexof_func : L.llvalue = L.define_function "indexOf" stringindexof_t the_module in

let stringlen_t : L.lltype = L.function_type i32_t [ | string_t | ] in
let stringlen_func : L.llvalue = L.define_function "len" stringlen_t the_module in

let stringconcat_t : L.lltype = L.function_type string_t [ | string_t ; string_t | ] in
let stringconcat_func : L.llvalue = L.define_function "concat" stringconcat_t the_module in

(* Array functions *)
let to_string_t : L.lltype = L.var_arg_function_type string_t [ | string_t | ] in
let to_string : L.llvalue = L.define_function "to_string" to_string_t the_module in

(* let arraylen_t : L.lltype = L.function_type i32_t [ | L.pointer_type i32_t | ] in
let arraylen_func : L.llvalue = L.define_function "length" arraylen_t the_module in *)

(* Define each function (arguments and return type) so we can
call it even before we’ve created its body *)
let function_decls : (L.llvalue * sfunc_decl) StringMap.t =
  let function_decl m fdecl =
    let name = fdecl.sfname
    and formal_types = Array.of_list (List.map (fun (t,_) -> ltype_of_typ t) fdecl.sformals)
    in let ftype = L.function_type (ltype_of_typ fdecl.styp) formal_types in
    StringMap.add name (L.define_function name ftype the_module, fdecl) m in
  List.fold_left function_decl StringMap.empty functions

(* Fill in the body of the given function *)
let build_function_body fdecl =
  try StringMap.find fdecl.sfname function_decls
  with _ ->
    let (the_function, _) = StringMap.find fdecl.sfname function_decls
    in try
try

with Not_found -> raise (Failure ("build_function_body not found " ^ fdecl. ^ 
  ^ sname))

let builder = L.builder_at_end context (L.entry_block the_function) in

let int_format_str = L.build_global_stringptr "%d\n" "fmt" builder
and float_format_str = L.build_global_stringptr "%g\n" "fmt" builder
and str_format_str = L.build_global_stringptr "%s\n" "fmt" builder in

(* Find the type of the input *)
let find_str_typ = function
  A.Int -> int_format_str
| A.Bool -> int_format_str
| A.Double -> float_format_str
| A.String -> str_format_str
| _ -> raise (Failure "Invalid type") in

(* Construct a hash table for function formals and locals
  add all the formals first *)
let tbl = StringHash.create 10 in
let formal_tbl = StringHash.create 5 in
let add_formal tbl (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);
  StringHash.add tbl n local; tbl in
let _ = List.fold_left2 add_formal formal_tbl fdecl.sformals
  (Array.to_list (L.params the_function)) in

(* Return the value for a variable or formal argument.
  Check local names first, then global names *)
let lookup n = try StringHash.find tbl n
  with Not_found ->
    try StringHash.find formal_tbl n
    with Not_found -> raise (Failure ("variable " ^ n ^ " not
    " ^ sname ^ found in lookup"))

let i32_t_pt = L.string_of_lltype i32_t in
let i1_t_pt = L.string_of_lltype i1_t in
let double_t_pt = L.string_of_lltype double_t in
let string_t_pt = L.string_of_lltype string_t in

let match_typ t =
  let t_pt = L.string_of_lltype t in
  if t_pt = string_t_pt then A.String
  else
    if t_pt = double_t_pt then A.Double
    else
      if t_pt = i32_t_pt then A.Int
      else
        if t_pt = i1_t_pt then A.Int
        else
          raise (Failure ("cannot match type: " ^ t_pt))

let is_arith = function
  A.Add -> true
| A.Sub -> true
let rec find_type = function
| SIntLit _ -> A.Int
| SBoolLit _ -> A.Int
| SDoubleLit _ -> A.Double
| SStrLit _ -> A.String
| SBinop((_, e_x), op, _) -> if (is_arith op) = true then find_type e_x
else A.Int
| SUnop(_, (_, e_x)) -> find_type e_x
| SNoexpr -> raise (Failure "Unmatched NoExpr")
| SNullPtr _ -> A.String
| SCall(f, _) -> let (_, fdecl) = StringMap.find f function_decls in
(match fdecl.styp with
A.Void -> raise (Failure "Cannot print void")
| _ -> fdecl.styp)
| SVar v -> match_typ (L.element_type (L.type_of (lookup v)))
| SAssign(v, _) -> find_type (SVar(v))
| SArrayAccess(s, _) -> match_typ (L.element_type (L.element_type (L.
-> type_of (lookup s))))
| SObjAccess(_, c, f) ->
let f_lst = StringMap.find c class_ty_fields in
fst (List.hd (List.filter (fun (_,n) -> n = f) f_lst))
(* todo *)
| SArrayLit (el) -> let (_, e_fst) = (List.nth el 0) in
A.Arr (find_type e_fst, List.length el)
| SConstruct _ -> raise (Failure "SConstruct cannot be printed")
| SObjAssign _ -> raise (Failure "SObjAssign cannot be printed")
| SDecAssn _ -> raise (Failure "SDecAssn not implemented")
in
(* Construct code for an expression; return its value *)
let rec expr builder ((_, e) : sexpr) = match e with
| SIntLit i -> L.const_int i32_t i
| SBoolLit b -> L.const_int i1_t (if b then 1 else 0)
| SDoubleLit l -> L.const_float_of_string double_t l
| SStrLit s -> L.build_global_stringptr s "str" builder
| SArrayLit arr ->
(* arr: sexpr list = typ * sx list *)
(* let len = L.const_int i32_t ((List.length arr) + 1) in *)
let len = L.const_int i32_t (List.length arr) in
let size = L.const_int i32_t ((List.length arr) + 1) in
let (fast_t, _) = List.hd arr in
let ty = ltype_of_typ (A.Arr(fast_t, List.length arr)) in
(* allocate memory for array *)
let arr_alloca = L.build_array_alloca ty size "arr" builder
(* bitcast *)
let arr_ptr = L.build_pointercast arr_alloca ty "arrptr" builder
(* store all elements *)
let elts = List.map (expr builder) arr
let store_elt ind elt =
let pos = L.const_int i32_t (ind) in
let elt_ptr = L.build_gep arr_ptr [| pos |] "arrelt" builder in
ignore(L.build_store elt elt_ptr builder)
in List.iteri store_elt elts;
let elt_ptr = L.build_gep arr_ptr [| len |] "arrlast" builder in
let null_elt = L.const_null (L.element_type ty) in
ignore(L.build_store null_elt elt_ptr builder);

| SConstruct (cname, args) ->
| (* let obj_ty = ltype_of_typ (A.Object(cname)) in *)
| let obj_ty = find_struct cname in
| let fields' = List.map (fun ty -> ltype_of_typ ty) (find_field_types cname
| → ) in
| let _ = L.struct_set_body obj_ty (Array.of_list fields') false in
| let obj_alloc = L.build_malloc obj_ty "constrObj" builder in
| (* bitcast *)
| let obj_ptr = L.build_pointercast obj_alloc (L.pointer_type obj_ty) "
| → constrPtr" builder in
| (* store all fields *)
| let store_field (fname, e) =
| let f_val = expr builder e in
| let ind = get_field_ind (cname, fname) in
| let f_ptr = L.build_struct_gep obj_ptr ind fname builder in
| ignore(L.build_store f_val f_ptr builder)
in List.iteri store_field args; obj_ptr

| SArrayAccess (s, e) ->
| let ind = expr builder e in
| let (ty, _) = e in
| (* increment index by one to get actual ptr position *)
| let pos = L.build_add ind (L.const_int i32_t 0) "accpos" builder in
| let arr = expr builder (ty, (SVar s)) in
| let elt = L.build_gep arr [| pos |] "acceltptr" builder in
| L.build_load elt "accelt" builder

| SObjAccess (s, cname, f) ->
| let obj = expr builder (A.String, (SVar s)) in
| let obj_ptr = L.build_load obj "objld" builder in
| let ind = get_field_ind (cname, f) in
| L.build_extractvalue obj_ptr ind f builder

| SASign Assign (s, e1, e2) ->
| let ind = expr builder e1 in
| let (ty, _) = e in
| (* increment index by one to get actual ptr position *)
| let pos = L.build_add ind (L.const_int i32_t 0) "accpos" builder in
| let arr = expr builder (ty, (SVar s)) in
| let new_val = expr builder e2 in
| let elt_ptr = L.build_gep arr [| pos |] "arrrelt" builder in
| L.build_store new_val elt_ptr builder

| SObjAssign Assign (s, cname, f, e) ->
| let obj = expr builder (A.String, (SVar s)) in
| let f_val = expr builder e in
| let ind = get_field_ind (cname, f) in
| let f_ptr = L.build_struct_gep obj_ptr ind f builder in
| L.build_store f_val f_ptr builder

| SNoexpr -> L.const_int i32_t 0
| SNullPtr t -> L.const_pointer_null (ltype_of_typ t)
| SVar s -> L.build_load (lookup s) s builder
| SAssign Assign (s, e) -> let e' = expr builder e in
| ignore(L.build_store e' (lookup s) builder); e'
| SDecAssn DecAssign (t, s, e) ->
| let e' = expr builder e in

57
let var = L.build_alloca (ltype_of_typ t) s builder in
StringHash.add tbl s var; e'
let e1' = expr builder e1
and e2' = expr builder e2 in
(match op with
  | A.Add       -> L.build_fadd
  | A.Sub       -> L.build_fsub
  | A.Mult      -> L.build_fmul
  | A.Div       -> L.build_fdiv
  | A.Equal     -> L.build_fcmp L.Fcmp.Oeq
  | A.Neq       -> L.build_fcmp L.Fcmp.One
  | A.Less      -> L.build_fcmp L.Fcmp.Olt
  | A.Leq       -> L.build_fcmp L.Fcmp.Ole
  | A.Greater   -> L.build_fcmp L.Fcmp.Ogt
  | A.Geq       -> L.build_fcmp L.Fcmp.Oge
  | A.And | A.Or       ->
      raise (Failure "internal error: semant should have rejected and/or on float") e1' e2' "tmp" builder

(* String operations *)
| SBinop ((A.String,_) as e1, op, e2) ->
  (match op with
   | A.Add       -> expr builder (A.String, SStrLit((print_sstring e1) ^ (print_sstring e2)))
   | _ ->
     let load_str e =
     let e' = expr builder e in
     let e_ptr = L.build_pointercast e' (L.pointer_type string_t) "strPtr"
     L.build_load e_ptr "strptrld" builder in
    L.build_load e_ptr "strptrld" builder in
     let s1 = load_str e1
     and e2 = load_str e2 in
     (match op with
     | A.Equal     -> L.build_icmp L.Icmp.Eq
     | A.Neq       -> L.build_icmp L.Icmp.Ne
     | A.Less      -> L.build_icmp L.Icmp.Slt
     | A.Leq       -> L.build_icmp L.Icmp.Sle
     | A.Greater   -> L.build_icmp L.Icmp.Sgt
     | A.Geq       -> L.build_icmp L.Icmp.Sge
     | _ -> raise (Failure "internal error: cannot perform this operation on string")
      s1 s2 "strcmp" builder
      )

  (* object Null equality *)
  | SBinop ((A.Object(_),_) as e1, op, _) ->
    let obj = expr builder e1 in
    (match op with
     | A.Equal     -> L.build_is_null
     | A.Neq       -> L.build_is_not_null
     | _ -> raise (Failure "internal error: cannot perform this operation on object")
    obj "objcmp" builder
let el' = expr builder e1
and e2' = expr builder e2 in
(match op with
| A.Add -> L.build_add
| A.Sub -> L.build_sub
| A.Mult -> L.build_mul
| A.Div -> L.build_sdiv
| A.And -> L.build_and
| A.Or -> L.build_or
| A.Equal -> L.build_icmp L.Icmp.Eq
| A.Neq -> L.build_icmp L.Icmp.Ne
| A.Less -> L.build_icmp L.Icmp.Slt
| A.Leq -> L.build_icmp L.Icmp.Sle
| A.Greater -> L.build_icmp L.Icmp.Sgt
| A.Geq -> L.build_icmp L.Icmp.Sge
) e1' e2' "tmp" builder
| SUnop(op, ((t, _) as e)) ->
let e' = expr builder e in
(match op with
| A.Neg when t = A.Double -> L.build_fneg
| A.Neg -> L.build_neg
| A.Not -> L.build_not) e' "tmp" builder

| SCall ("print", [e]) ->
let (_, e_x) = e in
let e_type = find_type e_x in
L.build_call printf_func [] (find_str_typ e_type) (expr builder e) [] "printf" builder

| SCall ("reverse", [e]) ->
L.build_call reversestring_func [] (expr builder e) [] "reverse" builder
| SCall ("upper", [e]) ->
L.build_call stringupper_func [] (expr builder e) [] "upper" builder
| SCall ("lower", [e]) ->
L.build_call stringlower_func [] (expr builder e) [] "lower" builder
| SCall ("substring", [e_s1;e_s2]) ->
L.build_call stringsubstring_func [] (expr builder e) (expr builder e_s1) (expr builder e_s2) [] "substring" builder
| SCall ("indexOf", [e1;e2]) ->
L.build_call stringindexof_func [] (expr builder e1) (expr builder e2) [] "indexOf" builder
| SCall ("len", [e]) ->
L.build_call stringlen_func [] (expr builder e) [] "len" builder
| SCall ("concat", [e1;e2]) ->
L.build_call stringconcat_func [] (expr builder e1) (expr builder e2) [] "concat" builder
| SCall ("to_string", [e]) ->
let e' = expr builder e in
let p_e = L.build_alloca (L.type_of e') "to_string" builder in
ignore(L.build_store e' p_e builder);
let v_e = L.build_bitcast p_e (string_t) "cast" builder in
L.build_call to_string [] (v_e) [] "to_string" builder

(* array length function *)
| SCall ("length", [e]) ->
let (ty,_) = e in
(match ty with
  | A.Arr(_,l) -> L.const_int i32_t l
  | _ -> raise (Failure "function length cannot be called on non array type "))

| SCall (f, args) ->
  let (fdef, fdecl) =
    try StringMap.find f function_decls
    with Not_found -> raise (Failure ("SCALL not found " ^ f))
  in
  let llargs = List.rev (List.map (expr builder) (List.rev args)) in
  let result = (match fdecl.styp with
    | A.Void -> ""
    | _ -> f ^ ".result") in
  L.build_call fdef (Array.of_list llargs) result builder

let add_terminal builder instr =
  match L.block_terminator (L.insertion_block builder) with
  | Some _ -> ()
  | None -> ignore (instr builder) in

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.styp with
    | A.Void -> L.build_ret_void builder
    | _ -> L.build_ret (expr builder e) 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);
  | SWhile (predicate, body) ->
    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
  | 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);

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

(* 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.styp with
  | A.Void -> L.build_ret_void
  | A.Double -> L.build_ret (L.const_float double_t 0.0)
  | t -> L.build_ret (L.const_int (ltype_of_typ t) 0))
in

List.iter build_function_body functions;
the_module

8.7 javalite.ml

(* Top-level of the JavaLite compiler: scan & parse the input,
  check the resulting AST and generate an SAST from it, generate LLVM IR,
  and dump the module *)
type action = Ast | Sast | LLVM_IR | Compile

let () =
  let action = ref Compile in
  let set_action a () = action := a in
  let speclist = [
    ("-a", Arg.Unit (set_action Ast), "Print the AST");
    ("-s", Arg.Unit (set_action Sast), "Print the SAST");
    ("-l", Arg.Unit (set_action LLVM_IR), "Print the generated LLVM IR");
    ("-c", Arg.Unit (set_action Compile),
     "Check and print the generated LLVM IR (default)");
  ] in
  let usage_msg = "usage: ./javalite.native [-a|-s|-l|-c] [file.jl]" in
  let channel = ref stdin in
  Arg.parse speclist (fun filename -> channel := open_in filename) usage_msg;
let lexbuf = Lexing.from_channel !channel in
let ast = Parser.program Scanner.token lexbuf in
match !action with
  | Ast -> print_string (Ast.string_of_program ast)
  | _ -> let sast = Semant.check ast in
  | Sast -> print_string (Sast.string_of_sprogram sast)
match !action with
  | Ast -> ()
  | LLVM_IR -> print_string (Llvm.string_of_llmodule (Codegen.translate sast))
  | Compile -> (let m = Codegen.translate sast in
      Llvm_analysis.assert_valid_module m;
      print_string (Llvm.string_of_llmodule m))

8.8  stringfuncs.c

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>

cchar* reverse(const char *str)
{
    size_t len = strlen(str);
    char* revStr = (char*)malloc(len + 1);

    size_t dst, src;
    for (dst = len - 1, src = 0; src < len; src++, dst--) {
        revStr[dst] = str[src];
    }

    revStr[len] = '\0';
    return revStr;
}

cchar* lower(const char *str)
{
    size_t len = strlen(str);
    char* lowStr = (char*)malloc(len + 1);

    size_t src;
    for (src = len - 1, src = 0; src < len; src++) {
        lowStr[src] = tolower(str[src]);
    }

    lowStr[len] = '\0';
    return lowStr;
}

cchar* substring(const char *str, size_t start, size_t end)
{
    size_t len = strlen(str);
    char* sub = (char*)malloc(len + 1);
    size_t src = 0;
while (start < end && start < len) {
    sub[src] = str[start];
    src++;
    start++;
}
sub[src] = '\0';
return sub;
}
char* upper(const char *str)
{
    size_t len = strlen(str);
    char* upStr = (char*)malloc(len + 1);
    size_t src;
    for (src = len - 1, src = 0; src < len; src++) {
        upStr[src] = toupper(str[src]);
    }
    upStr[len] = '\0';
    return upStr;
}
int indexOf(const char *str, const char *find)
{
    int i;
    size_t len = strlen(str);
    for (i = 0; i < len; i++)
    {
        if (str[i] == *find)
            return i;
    }
    return -1;
}
int len(const char *str)
{
    int l;
    size_t len = strlen(str);
    // l = (int*)(len);
    l = (int)(len);
}
char* concat(const char *str1, const char *str2)
{
    size_t len1 = strlen(str1);
    size_t len2 = strlen(str2);
    char* both = (char*)malloc(len1 + len2 + 1);
    strcpy(both, str1);
    strcat(both, str2);
    return both;
8.9 Makefile

```makefile
# "make test" Compiles everything and runs the regression tests

.PHONY : test
test : all testall.sh
./testall.sh

# "make all" builds the executable as well as the built-in libraries designed
# to test linking external code

.PHONY : all
all : javalite.native stringfuncs.o

# "make javalite.native" compiles the compiler
#
# The _tags file controls the operation of ocamlbuild, e.g., by including
# packages, enabling warnings
#
# See https://github.com/ocaml/ocamlbuild/blob/master/manual/manual.adoc

javalite.native :
opam config exec -- \n  ocamlbuild -use-ocamlfind javalite.native

# "make clean" removes all generated files

.PHONY : clean
clean :
ocamlbuild -clean

# Testing the "string" example

stringfuncs : stringfuncs.c
  cc -o string -DBUILD_TEST stringfuncs.c

# Building the tarball

TESTS = \
  add1 arith1 arith2 arith3 fib float1 float2 float3 for1 for2 func1 \n  func2 func3 func4 func5 func6 func7 func8 func9 gcd2 gcd global1 \n  global2 global3 hello if1 if2 if3 if4 if5 if6 local1 local2 ops1 \n  ops2 var1 var2 while1 while2

FAILS = \
  assign1 assign2 assign3 dead1 dead2 expr1 expr2 expr3 float1 float2 \n  for1 for2 for3 for4 for5 func1 func2 func3 func4 func5 func6 func7 \n  func8 func9 global1 global2 if1 if2 if3 if4 if5 global printb print \n  return1 return2 while1 while2

TESTFILES = $(TESTS:%=test-%.jl) $(TESTS:%=test-%.out) \

```

8.10  testall.sh

```bash
#!/bin/sh

# Regression testing script for JavaLite
# Step through a list of files
# Compile, run, and check the output of each expected-to-work test
# Compile and check the error of each expected-to-fail test

# Path to the LLVM interpreter
LLI="lli"
#LLI="/usr/local/opt/llvm/bin/lli"

# Path to the LLVM compiler
LLC="llc"

# Path to the javalite compiler. Usually ".javalite.native"
# Try ".build/javalite.native" if ocamlbuild was unable to create a symbolic link
JAVALITE="./javalite.native"
#JAVALITE="_build/javalite.native"

# Set time limit for all operations
ulimit -t 30

globallog=testall.log
rm -f $globallog
tmp=
error=0
globalerror=0
keep=0

Usage() {  
echo "Usage: testall.sh [options] [.jl files]"
echo "-k  Keep intermediate files"  
echo "-h  Print this help"
exit 1
}

SignalError() {
```
if [ $error -eq 0 ] ; then
error=1
fi
  echo " $1"

# Compare <outfile> <reffile> <difffile>
# Compares the outfile with reffile. Differences, if any, written to difffile
Compare() {
  generatedfiles="" $generatedfiles $3"
  echo diff -b "$1 $2 "" $3 1>&2
  diff -b "$1" "$2" > "$3" 2>&1 || {
    SignalError '"1 differs"
    echo "FAILED $1 differs from $2" 1>&2
  }
}

# Run <args>
# Report the command, run it, and report any errors
Run() {
  echo $* 1>&2
  eval $* || {
    SignalError "$1 failed on $*
    return 1
  }
}

# RunFail <args>
# Report the command, run it, and expect an error
RunFail() {
  echo $* 1>&2
  eval $* && {
    SignalError "failed: $* did not report an error"
    return 1
  }
  return 0
}

Check() {
  error=0
  basename='echo $1 | sed 's/.*/\///
  s/.j1//';
  reffile='echo $1 | sed 's/.jl$//';
  basedir='echo $1 | sed 's/\[/\]/\*/\*/\*/.*$//'."
  echo -n "$basename..."
  echo $>1>&2
  echo "#### Testing $basename" $>1>&2
  generatedfiles=""
  generatedfiles="" $generatedfiles "$basename).ll "$basename.s "$basename).exe
  -> "$basename).out" &&
  Run "$JAVA LITE" "$1" "$basename).ll "$ &&
  Run "$LLC" "-relocation-model=pic" "$basename).ll" "$basename.s" "$
Run "$CC" "-o" "$\{basename\}.exe" "$\{basename\}.s" "stringfuncs.o" && Run "./\{basename\}.exe" > "$\{basename\}.out" && Compare $\{basename\}.out $\{reffile\}.out $\{basename\}.diff

# Report the status and clean up the generated files

if [ $\{error\} -eq 0 ]; then
    if [ $\{keep\} -eq 0 ]; then
        rm -f $\{generatedfiles\}
    fi
    echo "OK"
echo "##### SUCCESS" 1>&2
else
    echo "##### FAILED" 1>&2
globalerror=$\{error\}
fi

CheckFail() {
    error=0
    basename='echo $1 | sed 's/.*\///'
    s/.jl//'
    reffile='echo $1 | sed 's/\s*\//\//'
    basedir='echo $1 | sed 's/\s*\//\//'
    echo -n "$basename..."
    echo 1>&2
echo "##### Testing $basename" 1>&2
generatedfiles=""
generatedfiles="$\{generatedfiles\} $\{basename\}.err $\{basename\}.diff" && RunFail "$\{JAVALITE\}" "$1" "2>" "$\{basename\}.err" ">>" $\{globallog\} && Compare $\{basename\}.err $\{reffile\}.err $\{basename\}.diff

# Report the status and clean up the generated files

if [ $\{error\} -eq 0 ]; then
    if [ $\{keep\} -eq 0 ]; then
        rm -f $\{generatedfiles\}
    fi
    echo "OK"
echo "##### SUCCESS" 1>&2
else
    echo "##### FAILED" 1>&2
globalerror=$\{error\}
fi
}

while getopts kdpsh c; do
    case $c in
        k) # Keep intermediate files
            keep=1
            ;;
        h) # Help
            Usage
            ;;
    esac
    echo "$1 "$2" "$3" "$4" "$5" "$6" "$7" "$8" "$9"
    esac
    esac
    esac
    esac
    esac
    esac
    esac
    esac
    esac
    fi
}
shift 'expr $OPTIND - 1'

LLIFail() {  
  echo "Could not find the LLVM interpreter "$LLI"."  
  echo "Check your LLVM installation and/or modify the LLI variable in testall.sh → "  
  exit 1
}

which "$LLI" >> $globallog || LLIFail

if [ ! -f stringfuncs.o ]
then
  echo "Could not find stringfuncs.o"
  echo "Try "make stringfuncs.o"
  exit 1
fi

if [ ! -f arrayfuncs.o ]
then
  echo "Could not find arrayfuncs.o"
  echo "Try "make arrayfuncs.o"
  exit 1
fi

if [ $# -ge 1 ]
then
  files=$@
else
  files="tests/test-*.jl tests/fail-*.jl"
fi

for file in $files
do  
case $file in
  *test*)
    Check $file 2>> $globallog
    ;;
  *fail*)
    CheckFail $file 2>> $globallog
    ;;
  *)
    echo "unknown file type $file"
    globalerror=1
    ;;
esac
done

exit $globalerror
8.11 Testing Files

8.11.1 Test Cases

test-add1.jl

```c
int add(int x, int y)
{
    return x + y;
}
```

```c
print( add(17, 25) );
```

test-arith1.jl

```c
print(39 + 3);
```

test-arith2.jl

```c
print(1 + 2 * 3 + 4);
```

test-arith3.jl

```c
int foo(int a)
{
    return a;
}
```

```c
int a = 42;
a = a + 5;
print(a);
```

test-arith4.jl

```c
print(3/2);
```

test-array1.jl

```c
int[] intArr = [1, 2, 3];
int i = intArr[2];
print(i);
```

```c
string[] strArr = ["h", "e", "y"]; for (i = 0; i < length(strArr); i = i + 1) {
    print(strArr[i]);
}
```

test-array2.jl

```c
int[] intArr = [1, 3, 5, 7];
int i = 0;
int j = 0;
while (i < length(intArr)) {
    print(j);
    j = j + intArr[i];
i = i + 1;
}
```

```c
print(j);
```
test-array3.jl

```julia
string[] strArr1 = ["te", "st"];
string[] strArr2 = ["ar", "ry", "3"];
print(strArr2[1]);
string[][] str2dArr = [strArr1, strArr2];
strArr2 = str2dArr[0];
print(strArr2[1]);
```

```
int[][] int2dArr = [[1,2,3],[1,2]];
int[] intArr = int2dArr[1];
print(intArr[0]);
```

```
void addOne(int[] arr, int len) {
    for (int i = 0; i < len; i = i + 1) {
        arr[i] = arr[i] + 1;
    }
}

int[] intArr = [1,2,3];
print(intArr[0]);
print(intArr[1]);
print(intArr[2]);
addOne(intArr, 3);
print(intArr[0]);
print(intArr[1]);
print(intArr[2]);
```

```
void repeatStr(string[] arr, int len) {
    for (int i = 0; i < len; i = i + 1) {
        arr[i] = concat(arr[i], arr[i]);
    }
}

string[] strArr = ["hi", "hello", "hey"]; 
print(strArr[0]);
print(strArr[1]);
print(strArr[2]);
repeatStr(strArr, 3);
print(strArr[0]);
print(strArr[1]);
print(strArr[2]);
```

```
int[] createArr() {
    int[] nums = [1,2,3];
    return nums;
}

int[] ns = [3,4];
print(ns[1]);
ns = createArr();
print(ns[1]);
```
test-array-length1.jl

```julia
int[] nums = [1,2,42,4,43];
int i = length(nums);
print(i);
```

test-array-length2.jl

```julia
string[] strArr1 = ['h', 'e', 'y'];
int l1 = length(strArr1);
print(l1);
string[] strArr2 = ['hello', 'world'];
int l2 = length(strArr2);
print(l2);
string[][] str2dArr = [strArr1, strArr2];
int l2d = length(str2dArr);
print(l2d);
```

test-bubble-sort.jl

```julia
int[] sortingArr = [52,14,72,5,66,7,12,31,9,3,54,41,53,12,61];
int i = 0;
int j = 0;
int tmpForSwap = 0;
int length = length(sortingArr);

for (i = 0 ; i < length-1 ; i = i + 1) {
    for (j =0; j< length-i-1;j = j + 1){
        if (sortingArr[j] > sortingArr[j+1]){
            tmpForSwap = sortingArr[j];
            sortingArr[j] = sortingArr[j+1];
            sortingArr[j+1] = tmpForSwap;
        }
    }
}

for (i = 0 ; i < length; i = i + 1) {
    print(sortingArr[i]);
}
```

test-class1.jl

```julia
class square {
    string name;
    int side;
}

class square sq = square("fst_sq", 42);
print(sq.side);
print(sq.name);
sq.side = 2;
sq.name = "snd_sq";
print(sq.side);
print(sq.name);
```
### test-class2.jl

```java
class Person {
    string name;
    int age;
    string phrase;
}

void sayhi(Person p) {
    string n = concat(p.name, " say:");
    print(n);
    string s = p.phrase;
    for (int i = 0; i < p.age; i = i + 1) {
        s = concat(s, p.phrase);
    }
    print(s);
}

class Person alice = Person("Alice", 3, "hey");
sayhi(alice);
```

### test-class3.jl

```java
class Item {
    string data;
    int priority;
}

class Item getHighestPriority(Item[] arr, int len) {
    Item hi = arr[0];
    for (int i = 1; i < len; i = i + 1) {
        Item ci = arr[i];
        if (ci.priority > hi.priority) {
            hi = ci;
        }
    }
    return hi;
}

// may want to add "insert(data, priority)" , "deleteHighestPriority()"
// after implemented array functions

class Item i1 = Item("medium", 1);
class Item i2 = Item("high", 2);
class Item i3 = Item("extra high", 3);
class Item i4 = Item("low", 0);
class Item[] itemArr = [i1, i2, i3, i4];
class Item hi = getHighestPriority(itemArr, 4);
print(hi.priority);
print(hi.data);
```

### test-class4.jl

```java
class Record {
    int index;
}```
int data;
string comment;
}

void updateComment(class Record r, string cnt) {
    string msg = concat("old comment: ", r.comment);
    print(msg);
    r.comment = cnt;
}

class Record rec = Record(0, 42, "added on Apr 10, 2021");
updateComment(rec, "updated on Apr 18, 2021");
string msg = concat("new comment: ", rec.comment);
print(msg);

class Course {
    string name;
    string department;
}

class Person {
    string name;
    int id;
    class Course[] take_courses;
}

void printCourses(class Person p, int len) {
    class Course[] courses = p.take_courses;
    for (int i = 0; i < len; i = i + 1) {
        class Course curr_course = courses[i];
        string fullname = concat(curr_course.department, curr_course.name);
        print(fullname);
    }
}

void updateDepartment(class Person p, int len) {
    class Course[] courses = p.take_courses;
    for (int i = 0; i < len; i = i + 1) {
        class Course curr_course = courses[i];
        if (curr_course.department != "COMS") {
            curr_course.department = "NO";
        }
    }
}

class Course cs4115 = Course("PLT", "COMS");
class Course m4042 = Course("Algebra", "MATH");
class Course cs6111 = Course("Database", "COMS");
class Person jess = Person("Jesse", 3222, [cs4115, m4042, cs6111]);
print(jess.id);
printCourses(jess, 3);
updateDepartment(jess, 3);
printCourses(jess, 3);
class Node {
  int data;
  class Node next;
}

void push(class Node head, int data) {
  class Node n = head;
  while (n.next != null) {
    n = n.next;
  }
  n.next = Node(data, null);
}

int pop(class Node head) {
  class Node n = head.next;
  int data = head.data;
  if (n == null) {
    head = null;
  } else {
    class Node prev = head;
    while (n.next != null) {
      n = n.next;
      prev = prev.next;
    }
    data = n.data;
    prev.next = null;
  }
  return data;
}

void printNodes(class Node head) {
  class Node n = head;
  while (n.next != null) {
    print(n.data);
    n = n.next;
  }
  print(n.data);
}

class Node head = Node(42, null);
printNodes(head);
int[] dataArr = [3,4,5];
push(head, dataArr[0]);
push(head, dataArr[1]);
push(head, dataArr[2]);
printNodes(head);
pop(head);
printNodes(head);
```c
int countingSortArrayLength = length(countingSortArray);
int[] sortingArr = [32,17,31,5,26,13,47,31,9,3,24,41,43,12,11];
int length = length(sortingArr);
int i = 0;
int j = 0;

for (i = 0 ; i < length ; i = i + 1) {
    countingSortArray[sortingArr[i]] = countingSortArray[sortingArr[i]] + 1;
}

for (i = 0 ; i < countingSortArrayLength ; i = i + 1) {
    if(countingSortArray[i] > 0){
        for(j=0;j<countingSortArray[i];j = j+1){
            print(i);
        }
    }
}
```

```plaintext
test-double1.jl

double a = 3.14159267;
print(a);

test-double2.jl

double a = 3.14159267;
double b = -2.71828;
double c = a + b;
print(c);

test-double3.jl

void testdouble(double a, double b)
{
    print(a + b);
    print(a - b);
    print(a * b);
    print(a / b);
    print(a == b);
    print(a == a);
    print(a != b);
    print(a != a);
    print(a > b);
    print(a >= b);
    print(a < b);
    print(a <= b);
}

double c = 42.0;
double d = 3.14159;
testdouble(c, d);
testdouble(d, d);

test-fib.jl

int fib(int x)
```
```plaintext
if (x < 2) return 1;
return fib(x-1) + fib(x-2);
}

print(fib(0));
print(fib(1));
print(fib(2));
print(fib(3));
print(fib(4));
print(fib(5));
```

test-for1.jl
```
for (int i = 0 ; i < 5 ; i = i + 1) {
    print(i);
}
print(42);
```

test-for2.jl
```
int i = 0;
for ( ; i < 5; ) {
    print(i);
    i = i + 1;
}
print(42);
```

test-func-string1.jl
```
string s = "Hello World";
string r = reverse(s);
print(s);
print(r);
r = reverse(r);
print(r);
s = "Foo Foo Doo Bar";
string u = upper(s);
print(s);
print(u);
```

test-func-string2.jl
```
string x = "This is a very long sentence that I am writing";
string l = lower(x);
int len_x = len(x);
print(x);
print(l);
print(len_x);
```

test-func-string3.jl
```
string s = "Hello World!!";
string t = "FooFooDooDooBar";
```
```json
print(s);
string r = substring(s,1,5);
print(r);

r = substring(s,4,7);
print(r);

r = substring(t,0,10);
print(r);

r = substring(t,4,6);
print(r);
```

```javascript
test-func-string4.jl

string s = "Hello World!!";
string t = "FooFooDooDooBar";

int si = indexOf(s,"e");
print(si);

int ti = indexOf(t,"D");
print(ti);

int sn = indexOf(s,"D");
print(sn);

string c = concat(s,t);
print(c);

s = substring(s,0,6);
c = concat(s,t);
c = concat(c,"!!!!!");
print(c);
```

```javascript
test-func1.jl

int add(int a, int b)
{
    return a + b;
}

int a = add(39, 3);
print(a);
```

```javascript
test-func2.jl

/*@ Bug noticed by Pin-Chin Huang */

int fun(int x, int y)
{
    return 0;
}

int i = 1;
fun(i = 2, i = i+1);
```
```plaintext
print(i);

**test-func3.jl**

```plaintext
def printem(a, b, c, d):
    print(a);
    print(b);
    print(c);
    print(d);

printem(42, 17, 192, 8);
```

**test-func4.jl**

```plaintext
def add(a, b):
    c = a + b;
    return c;

d = add(52, 10);
print(d);
```

**test-func5.jl**

```plaintext
def foo(a):
    return a;

foo(40);
```

**test-func6.jl**

```plaintext
def foo() {}
def bar(a, b, c):
    return a + c;

print(bar(17, False, 25));
```

**test-func7.jl**

```plaintext
def foo(a):
    print(a + 3);

foo(40);
```

**test-func8.jl**

```plaintext
def foo(a):
    ;
```
test-functional1.jl

```java
int x = 2;
print(x);
```

test-functional2.jl

```java
int x = 0;
print(x);

int update(int x)
{
    x = x + 1;
    return x;
}

x = update(x);
print(x);
```

test-functional3.jl

```java
void print1()
{
    print("hello");
}

print1();

void print2()
{
    print("hi");
}

print2();

void doNothing() {} 
doNothing();
print1();
print2();
```

test-gcd1.jl

```java
int gcd(int a, int b) {
    while (a != b) {
        if (a > b) a = a - b;
        else b = b - a;
    }
    return a;
}

print(gcd(2,14));
```
test-gcd2.jl

```c
int gcd(int a, int b) {
    while (a != b)
        if (a > b) a = a - b;
        else b = b - a;
    return a;
}
```

```
print(gcd(3, 15));
print(gcd(99, 121));
print(gcd(14, 21));
print(gcd(8, 36));
print(gcd(99, 121));
```

---

test-if1.jl

```c
if (true) print(42);
print(17);
```

---

test-if2.jl

```c
if (true) print(42); else print(8);
print(17);
```

---

test-if3.jl

```c
if (false) print(42);
print(17);
```

---

test-if4.jl

```c
if (false) print(42); else print(8);
print(17);
```

---

test-if5.jl

```c
int cond(bool b) {
    int x = 0;
    if (b)
        x = 42;
    else
        x = 17;
    return x;
}
```

```
print(cond(true));
print(cond(false));
```

---

test-if6.jl

```c
int cond(bool b) {
    int x = 10;
    if (b)
        x = 42;
    else
        x = 17;
    return x;
}
```

```
print(cond(true));
print(cond(false));
```
```plaintext
if (x == 10)
    x = 42;
else
    x = 17;
return x;
}

print(cond(true));
print(cond(false));

test-local1.jl

void foo(bool i)
{
    int i = 42; /* Should hide the formal i */
    print(i + i);
}
foo(true);

test-local2.jl

int foo(int a, bool b)
{
    int c = a;
    bool d = false;
    return c + 10;
}
print(foo(37, false));

test-ops1.jl

print(1 + 2);
print(1 - 2);
print(1 * 2);
print(100 / 2);
print(99);
print(1 == 2);
print(1 == 1);
print(99);
print(1 != 2);
print(1 != 1);
print(99);
print(1 < 2);
print(2 < 1);
print(99);
print(1 <= 2);
print(1 <= 1);
print(99);
print(1 > 2);
print(2 > 1);
print(99);
print(1 >= 2);
print(1 >= 1);
print(99);
```
24  \texttt{print}(2 \geq 1);

\begin{verbatim}
test-ops2.jl
1 print(true);
2 print(false);
3 print(true && true);
4 print(true && false);
5 print(false && true);
6 print(false && false);
7 print(true || true);
8 print(true || false);
9 print(false || true);
10 print(false || false);
11 print(!false);
12 print(!true);
13 print(-10);
14 print(-42);
\end{verbatim}

\begin{verbatim}
test-print.jl
1 int i = 3;
2 int j = 9;
3 print(i + j);
4 print(i - j);
5 print(i / j);
6 string s = "Foo";
7 print(s);
8 double a = 3.14159267;
9 double b = -2.71828;
10 print(a + b);
11 print(a == b);
12 print(true && false);
13 print(!false);
14 print("hello" + " world");
15 print((i + 2 * 3 + 4);
16 string[] strArr = ["hello", "world"]; 
17 print(strArr[0]);
18 int fib(int x)
19 { 
20  if (x < 2) return 1;
21  return fib(x-1) + fib(x-2);
22 } 
23 print(fib(0));
24 print(fib(3));
\end{verbatim}

\begin{verbatim}
test-string-binop1.jl
1 string c = "Hello " + "FooFooDooDooBar";
2 print(c);
3 c = "Foo " + "Bar";
4 print(c);
\end{verbatim}
```julia
test-string-binop2.jl
string s1 = "hello";
string s2 = "hey";
string s3 = "hello";
print(s1 == s2); // 0
print(s1 == s3); // 1
print(s1 != s2); // 1
print(s1 > s2); // 1
print(s1 < s2); // 0
print(s1 >= s2); // 1
print(s1 <= s3); // 1
```

test-var1.jl
```julia
int a = 42;
print(a);
```

test-var2.jl
```julia
string s = "hello";
int a = 1;
print(s);
print(a);
```

test-while1.jl
```julia
int i = 5;
while (i > 0) {
    print(i);
    i = i - 1;
}
print(42);
```

test-while2.jl
```julia
int foo(int a)
{
    int j = 0;
    while (a > 0) {
        j = j + 2;
        a = a - 1;
    }
    return j;
}
print(foo(7));
```

8.11.2 Fail Cases

fail-array1.jl
```julia
int[] nums = ["h", "e", "l", "l", "o"];
```

fail-array2.jl
string[] arr = ["h", "e", 1, "l", "l", "o"];

int nums = 1;
string[] arr = ["h", "e", "l", "l", "o"];
string se = arr[2];
int ie = nums[1];

string[] strArr2 = ["l", "l", "o"];
strArr2[2] = 1;

int[] int2dArr = [[1,2,3],[4,5]];

string s = "hello world";
int i = length(s);
print(i);

int i = 42;
bool b = true;
b = false;
i = false; // Fail: assigning a bool to an integer */

int i = 0;
bool b = 48; // Fail: assigning an integer to a bool */

void myvoid()
{
    return;
}

int i = myvoid(); // Fail: assigning a void to an integer */

class square {
    string name;
    int side;
}
fail-class2.jl

```java
class square {
    string name;
    int side;
}

class square sq = square("fst_sq", 42);
int s = sq.side;
int n = sq.name;
```

fail-dead1.jl

```java
int i = 15;
return i;
i = 32; /* Error: code after a return */
```

fail-dead2.jl

```java
int i = 0;
{
i = 15;
    return i;
}
i = 32; /* Error: code after a return */
```

fail-double1.jl

```java
-3.5 && 1; /* Double with AND? */
```

fail-double2.jl

```java
-3.5 && 2.5; /* Double with AND? */
```

fail-expr1.jl

```java
void foo(int c, bool d)
{
    int dd = 0;
    bool e = true;
    e + dd; /* Error: bool + int */
}
```

fail-expr2.jl

```java
void foo(int c, bool d)
{
    int d = 0;
    bool e = false;
    e + d; /* Error: bool + int */
}
```

fail-expr3.jl

```java
void foo(int c, double d)
{
    int d = 0;
```
```cpp
double e = 0.0;
e + d; /* Error: double + int */
}

fail-for1.jl
int i = 0;
int k = 0;
for ( ; true ; ) {} /* OK: Forever */
for (i = 0 ; i < 10 ; i = i + 1) {
  if (i == 3) {
    j = j + 1;
  }
}
for (j = 0; i < 10 ; i = i + 1) {} /* j undefined */

fail-for2.jl
int i = 0;
for (i = 0; j < 10 ; i = i + 1) {} /* j undefined */

fail-for3.jl
int i = 0;
for (i = 0; i ; i = i + 1) {} /* i is an integer, not Boolean */

fail-for4.jl
int i = 0;
for (i = 0; i < 10 ; i = i + 1) {
  foo(); /* Error: no function foo */
}

fail-func1.jl
int foo() {}
int bar() {}
int baz() {}
void bar() {} /* Error: duplicate function bar */

fail-func2.jl
int foo(int a, bool b, int c) {}
void bar(int a, bool b, int a) {} /* Error: duplicate formal a in bar */
```
fail-func3.jl

```c
int foo(int a, bool b, int c) {}  
void bar(int a, void b, int c) {/* Error: illegal void formal b */}
```

fail-func4.jl

```c
int foo() {}  
void bar() {}  
int print() {/* Should not be able to define print */}
void baz() {}
```

fail-func5.jl

```c
void foo(int a, bool b)  
{  
}

foo(42, true);
foo(42); /* Wrong number of arguments */
```

fail-func6.jl

```c
void foo(int a, bool b)  
{  
}

foo(42, true);
foo(42, 42); /* Fail: int, not bool */
```

fail-func7.jl

```c
void foo(int a, bool b)  
{  
}

foo(42, true);
foo(42, true, false); /* Wrong number of arguments */
```

fail-func8.jl

```c
void foo(int a, bool b)  
{  
}

void bar()  
{  
}

foo(42, true);
foo(42, bar()); /* int and void, not int and bool */
```
fail-functional1.jl

```java
int x = 0;
void update()
{
    x = x + 1;
}
update();
print(x);
```

fail-if1.jl

```java
if (true) {}
if (false) {} else {}
if (42) {} /* Error: non-bool predicate */
```

fail-if2.jl

```java
if (true) {
    foo; /* Error: undeclared variable */
}
```

fail-if3.jl

```java
if (true) {
    42;
} else {
    bar; /* Error: undeclared variable */
}
```

fail-print1.jl

```java
/* Should be illegal to redefine */
void print() {}
```

fail-print2.jl

```java
string[] strArr = ["hello", "world"];
print(strArr);
```

fail-return1.jl

```java
int foo(){
    return 12;
    int a=2;
}
```

fail-return2.jl

```java
void foo()
{
    if (true) return 42; /* Should return void */
    else return;
}
```
fail-return3.jl

```julia
int x = 0;
return x;
```

fail-string-binop1.jl

```julia
string s1 = "hello";
string s2 = "hey";
string s3 = "hello";
print(s1 - s2);
```

fail-while1.jl

```julia
int i = 0;

while (true) {
    i = i + 1;
}

while (42) {  /* Should be boolean */
    i = i + 1;
}
```

fail-while2.jl

```julia
int i = 0;

while (true) {
    i = i + 1;
}

while (true) {
    foo();  /* foo undefined */
}
```

8.11.3 Performance Tests
test-Array-Access.jl

```julia
int[] Arr = [52,14,72,5,66,7,12,31,9,3,54,41,53,12,61];
int length = 15;
int count = 100000000;
int i=0;

int v = 0;
while(count > 0){
    for(i=0;i<length;i = i+1){
        v = Arr[i];
        Arr[i] = 3;
    }
    count = count-1;
}
print("Array-Access Test Completed");
```
test-Class-Field-Access.jl

```plaintext
class Person {
    string name;
    int age;
    string phrase;
}
int count = 200000000;

class Person alice = Person("Alice", 3, "hey");
while(count > 0){
    string s = alice.phrase;
    s = alice.phrase;
    count = count-1;
}
print("Class-Field-Access Test Completed");
```

test-Double-Arithmetic.jl

```plaintext
double AddTest = 0.0;
double MinusTest = 2100000005.0;
double MultiplyTest = 2.0;
double DivideTest = 2.0;
int count = 2100000005;
while(count > 0){
    AddTest = AddTest + 1.0;
    MinusTest = MinusTest - 1.0;
    MultiplyTest = 3.0 * 3.0;
    DivideTest = 6.0/2.0;
    count = count-1;
}
print("Double-Arithmetic Test Completed");
```

test-If-Condition.jl

```plaintext
int count = 2100000005;
int n=0;
while(count > 0){
    if(1 < 3){
        n = 2;
    }
    count = count-1;
}
print("If-Condition Test Completed");
```

test-Int-Arithmetic.jl

```plaintext
int AddTest = 0;
int MinusTest = 2100000005;
```
```cpp
int MultiplyTest = 2;
int DivideTest = 2;
int count = 2100000005;

while(count > 0){
    AddTest = AddTest + 1;
    MinusTest = MinusTest - 1;
    MultiplyTest = 3*3;
    DivideTest = 6/2;
    count = count-1;
}

print("Int-Arithmetic Test Completed");
```

### 8.11.4 Demo Files

#### demo-arrays.jl

```jl
int[] a = [[2, 5], [1, 2]];
int[] b = [[4, 5], [2, 3]];
deterMult(a, b);
```

```jl
// assuming size of 2x2
void deterMult(int[][] a, int[][] b) {
    int[] row1A = a[0];
    int[] row2A = a[1];
    int[] row1B = b[0];
    int[] row2B = b[1];

    int f0 = (row1A[0] * row1B[0]) + (row1A[1] * row2B[0]);
    int f1 = (row1A[0] * row1B[1]) + (row1A[1] * row2B[1]);
    int f2 = (row2A[0] * row1B[0]) + (row2A[1] * row2B[0]);
    print((f0 * f3) - (f1 * f2));
}
```

/*
[[ 2 5 ] [ 4 5 ]] * [[ 2 5 ]]
[[[ 1 2 ] [ 2 3 ]]] = [[ 8 11 ]] = (198 - 200) = -2 */

#### demo-class.jl

```jl
class Course {
    string name;
    string department;
}

class Student {
    string name;
    int id;
    string class_of;
    class Course[] courses_taken;
}
```
void printCourses(class Student s, int len) {
    class Course[] courses = s.courses_taken;
    for (int i = 0; i < len; i = i + 1) {
        class Course curr_course = courses[i];
        string fullname = concat(curr_course.department, curr_course.name);
        print(fullname);
    }
}

void updateDepartment(class Student s, int len) {
    class Course[] courses = s.courses_taken;
    for (int i = 0; i < len; i = i + 1) {
        class Course curr_course = courses[i];
        if (curr_course.department != "COMS") {
            curr_course.department = "NOTCS! --- ";
        }
    }
}

void prettyPrint(class Student s) {
    string j = upper(s.name);
    print(j);
    print(s.id);
    string c = concat("Class Of: ", s.class_of);
    print(c);
    print("----------");
    printCourses(s, 3);
    print("----------");
    updateDepartment(s, 3);
    printCourses(s, 3);
}

class Course coms4115 = Course("PLT", "COMS");
class Course math4042 = Course("Algebra", "MATH");
class Course coms6111 = Course("Database", "COMS");

class Student jess = Student("Jesse Blah", 5555, "Spring 2022", [coms4115, math4042, coms6111]);
prettyPrint(jess);

demo-linkedlist.jl

class Node {
    int data;
    class Node next;
}

void push(class Node head, int data) {
    class Node n = head;
    while (n.next != null) {
        n = n.next;
    }
    n.next = Node(data, null);
}

int pop(class Node head) {
class Node n = head.next;
int data = head.data;
if (n == null) {
    head = null;
} else {
    class Node prev = head;
    while (n.next != null) {
        n = n.next;
        prev = prev.next;
    }
data = n.data;
    prev.next = null;
}
return data;

// initialize head
class Node head = Node(42, null);
printNodes(head);

// push new nodes
int[] dataArr = [3,4,5];
for (int i = 0; i < length(dataArr); i = i + 1) {
push(head, dataArr[i]);
}
printNodes(head);

// pop the last node
print("popped:");
pop(head);
print("*");
printNodes(head);

void printNodes(class Node head) {
    print("list:");
    class Node n = head;
    while (n.next != null) {
        print(n.data);
        n = n.next;
    }
    print(n.data);
    print("*");
}

9 Project Log
commit 354e3a465987be2a29d3a44d8b91dc5cb2056e0f
Author: HongfeiChenCU <hc3222@columbia.edu>
Date:  Mon Apr 26 14:31:09 2021 -0700
memory allocation updated
codegen cleanup

codegen.ml | 47 ---------------------------------------------------------------
1 file changed, 47 deletions(-)

commit 3117364856185177531cd3722cb181f35edccab6
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Sun Apr 25 15:22:18 2021 -0700

array length, 2d array fix

Makefile | 6 ------
ast.ml | 2 +--
codegen.ml | 45 ++++++++++++++++++----------------------
dev_tests/ast-test-arrays.jl | 8 +++++--
semant.ml | 15 ++++++-
tests/fail-array1.err | 2 +--
tests/fail-array5.err | 1 +
tests/fail-array5.ml | 1 +
tests/test-array3.ml | 3 +++
tests/test-array3.out | 3 ++
10 files changed, 49 insertions(+), 37 deletions(-)

commit 4f7914b39a0ed64b07164955ce275013ad3e89
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Sun Apr 25 15:42:34 2021 -0400

trying printArr

codegen.ml | 34 +++++++++++++++++++++++++++--
dev_tests/ast-test-arrays.jl | 6 +++++--
2 files changed, 36 insertions(+), 4 deletions(-)

commit 3f1aed640fc6fee3bf0994b069e662a528a5d4c8
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Sun Apr 25 11:49:58 2021 -0700

half working array length

codegen.ml | 10 +++++--
dev_tests/ast-test-arrays.jl | 2 +--
semant.ml | 43 ++++++----------------------
stringfuncs.o | Bin 3272 -> 0 bytes
tests/fail-array-length.err | 1 +
tests/fail-array-length.ml | 3 ++++
tests/test-array-length1.ml | 3 +++
tests/test-array-length1.out | 1 +
tests/test-array-length2.ml | 9 +++++++
tests/test-array-length2.out | 3 +++
tests/test-array1.ml | 2 +--
tests/test-array2.ml | 2 +--
tests/test-bubble-sort.ml | 2 +--
tests/test-counting-sort.ml | 4 +--
14 files changed, 65 insertions(+), 20 deletions(-)

commit 9710da0c129c5e42a96817fe8801a9be29b81254
update class demo, add print pass and fail test, add buildandrun.log to
→ makefile

Makefile | 4 ++--
demo/demo-class.jl | 48 +++++++++++++++++++++++++++++++++++++++
demo/demo-class.out | 9 ++
dev_tests/demo/demo-class.jl | 48 +------------------------------------------
dev_tests/demo/demo-class.out | 8 -----stringfuncs.o |  Bin 0 -> 3272 bytes
tests/{fail-print.err => fail-print1.err} | 0
tests/{fail-print.jl => fail-print1.jl} | 0
tests/fail-print2.err | 1 +
tests/fail-print2.jl | 2 ++
tests/test-print.jl | 29 +++++++++++++++++
tests/test-print.out | 13 ++++++
13 files changed, 107 insertions(+), 63 deletions(-)

add print tests to dev

dev_tests/demo/demo-class.jl | 48 +++++++++++++++++++++++++++++++++++++++
dev_tests/demo/demo-class.out | 8 +++
dev_tests/test-print.jl | 29 +++++++++++++++++
tests/test-class5.jl | 1 -
4 files changed, 85 insertions(+), 1 deletion(-)

cleanup external array funcs

Makefile | 6 ----
_tags | 5 ----
arrayfuncs.c | 61 ------------------------------------------
arrayfuncs.o |  Bin 1240 -> 0 bytes
codegen.ml | 8 +++-
testall.sh | 14 +++++-----
tests/test-array-func1.jl | 3 ----
tests/test-array-func1.out | 1 -
8 files changed, 13 insertions(+), 85 deletions(-)

commit 314bb3b5be3e6c3c1007141bf5c57ca1e4ac51b6
Merge: 8ceea35 ba5c9ad
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Sat Apr 24 14:34:06 2021 -0700

Merge branch 'main' of https://github.com/FranCao/javalite into main

commit 8ceea3513afdac2f06290f5b533ceaf6e14680c4c
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Sat Apr 24 14:33:29 2021 -0700

codegen warning resolved
arrayfuncs.o | Bin 1240 -> 0 bytes
codegen.ml | 2 +-
stringfuncs.o | Bin 3272 -> 0 bytes
3 files changed, 1 insertion(+), 1 deletion(-)

commit ba5c9ad1f42ff259b9562a3c7d9335e01f108be
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Sat Apr 24 17:05:21 2021 -0400
test reorganization
tests/fail-func5.err | 1 +
tests/!(fail-func9.jl => fail-func5.jl) | 2 +-
tests/fail-func6.err | 2 +-
tests/fail-func5.jl | 2 +-
tests/fail-func9.err | 1 -
5 files changed, 4 insertions(+), 4 deletions(-)

commit 50532e1daf6277db32c053f8fe88c4f5e35d1685
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Sat Apr 24 16:21:40 2021 -0400
update boolean back to bool, cleanup unnecessary files
arrayfuncs.o | Bin 0 -> 1240 bytes
ast.ml | 2 +-
gitlog.txt | 510 ----------------------------------------------------------
scanner.ml | 2 +-
stringfuncs.o | Bin 0 -> 3272 bytes
5 files changed, 2 insertions(+), 512 deletions(-)

commit cb0876f6f728d36c6479abf4e637301b3ac81a
Author: 314pies <ian121363@gmail.com>
Date: Sat Apr 24 12:11:53 2021 -0700
Add Java performance benchmarking code for compare
tests/performance/Java/Person.java | 11 ++++++++++
tests/performance/Java/TestArrayAccess.java | 20 ++++++++++++++++++++++
tests/performance/Java/TestClassFieldAccess.java | 11 ++++++++++
tests/performance/Java/TestDoubleArithmetic.java | 20 ++++++++++++++++++++++
tests/performance/Java/TestIntArithmetic.java | 16 +++++++++++++++
tests/performance/Java/TestIfCondition.java | 20 ++++++++++++++++++++++
6 files changed, 98 insertions(*)
commit 63e9e6d880a142e11eac8476d47ab9c6446cb0a6
Merge: 2b2107c cadf4dc
Author: 314pies <ian121363@gmail.com>
Date: Sat Apr 24 14:34:26 2021 -0400

Merge branch 'main' into main

commit 2b2107c242996709c93bd90de37c95b57c875827
Author: 314pies <ian121363@gmail.com>
Date: Sat Apr 24 11:28:27 2021 -0700

Add Performance Benchmark

PerformanceBench.sh | 19 +++++++++++++++++++
arrayfuncs.o | Bin 1240 -> 0 bytes
stringfuncs.o | Bin 3272 -> 0 bytes
tests/performance/test-Array-Access.jl | 16 +++++++++++++++++
tests/performance/test-Class-Field-Access.jl | 15 +++++++++++++++++
tests/performance/test-Double-Arithmetic.jl | 18 +++++++++++++++++
tests/performance/test-If-Condition.jl | 12 +++++++
tests/performance/test-Int-Arithmetic.jl | 17 +++++++++++++++++
8 files changed, 97 insertions(+)

commit 60c8a136b9243a68d03e88bdfc94c2e826e2d18f
Author: 314pies <ian121363@gmail.com>
Date: Sat Apr 24 13:08:18 2021 -0400

Add config.yml for CD/CI

.circleci/config.yml | 14 +++++++++++++++++
1 file changed, 14 insertions(+)

commit cadf4dcb64209e72d6d480db39044f185fb3c1d4
Merge: dbfc3fa 2573484
Author: 314pies <ian121363@gmail.com>
Date: Sat Apr 24 12:52:27 2021 -0400

Merge pull request #1 from 314pies/circleci-project-setup

Add .circleci/config.yml

.commitlog.txt | 6 +++++++
1 file changed, 6 insertions(+)

commit 87a730f264bf03a968237282f3d6367535e2843
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Sat Apr 24 12:20:58 2021 -0400

  changes to test file names, added the git log

gitlog.txt | 504 ++++++++++++++++++++++++++++++++
  tests/{test-func9.jl => test-func7.jl} | 2 +- test-func9.out => test-func7.out | 0
  tests/test-func8.jl | 2 +- tests/{test-gcd.jl => test-gcd1.jl} | 0
tests/test-gcd.out => test-gcd1.out | 0
tests/{test-var3.jl => test-var2.jl} | 0
tests/test-var3.out => test-var2.out | 0
8 files changed, 506 insertions(+), 2 deletions(-)

commit a7ff0c4284d99c82c901d0240714fbd7d4ce646d
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Sat Apr 24 11:25:40 2021 -0400

  update bool -> boolean
  ast.ml | 2 +- scanner.mll | 2 +-
2 files changed, 2 insertions(+), 2 deletions(-)

commit c95703154f3b2315f3ccb670bfbb28193464ea451
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Sat Apr 24 00:24:35 2021 -0400

  added some print array stuff, change arr layout, update null
  Makefile | 2 +- arrayfuncs.c | 21 ++++++++++++++++++++ ast.ml | 4 +--
  codegen.ml | 27 +++++++++++++++++++++--- dev_tests/ast-test-arrays.jl | 41 ++++++++++++++++++++++++++++++
  parser.mly | 2 +- scanner.mll | 2 +-
  semant.ml | 11 ++++++++ tests/fail-array1.err | 2 +-
tests/test-class6.jl | 16 ++++++++ 10 files changed, 85 insertions(+), 43 deletions(-)

commit 5eea65c90d2a3f1c5914c1c6fc54a6b6e2e89760
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Thu Apr 22 17:39:09 2021 -0400

  pieces of length for array

  Makefile | 6 +-- _tags | 5 +-- arrayfuncs.c | 42 ++++++++++++++++++++++++++++++++ arrayfuncs.o | Bin 0 -> 1240 bytes
codegen.ml | 8 ++++++
<table>
<thead>
<tr>
<th>File Name</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>semant.ml</td>
<td>13 +++++++++++++</td>
</tr>
<tr>
<td>testall.sh</td>
<td>9 +++++++++--</td>
</tr>
<tr>
<td>tests/test-array-func1.jl</td>
<td>3 +++</td>
</tr>
<tr>
<td>tests/test-array-func1.out</td>
<td>1 +</td>
</tr>
<tr>
<td>tests/test-array2.jl</td>
<td>1 -</td>
</tr>
<tr>
<td>tests/test-performance1.jl</td>
<td>8 --------</td>
</tr>
<tr>
<td>tests/test-performance1.out</td>
<td>1 -</td>
</tr>
</tbody>
</table>

12 files changed, 84 insertions(+), 13 deletions(-)

Commit: cd8f383f357f7a0d4c2c638534385d2b3069a494
Author: FranCao <frances.te.cao@gmail.com>
Date: Thu Apr 22 15:33:46 2021 -0400

cleanup

SampleCode1.jl | 3 ---
1 file changed, 3 deletions(-)

Commit: 29e741924b0d487bd509f9ad244a9fe36ec6306c
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Thu Apr 22 08:21:08 2021 +0000

linkedlist test added

codegen.ml | 17 +++++++++---
dev_tests/aastest-class.jl | 26 +++++++++++++++++++++
dev_tests/classinc.c | 5 +++--
semant.ml | 27 +++++++++++++++++-----
tests/test-class6.jl | 48 ++++++++++++++++++++++++++++++++--
tests/test-class6.out | 8 +++++
6 files changed, 99 insertions(+), 42 deletions(-)

Commit: 777a2e23114ddfd698db61e2a8b60ae6f6a86fb4
Merge: cd8b094 f1e952e
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Wed Apr 21 23:22:02 2021 -0700

Merge branch 'main' into hc-inheritance

Commit: cd6b094715abaa03ecef7dffa32145932620af3
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Wed Apr 21 23:17:57 2021 -0700

recursive class and null pointer done

ast.ml | 5 +++
codegen.ml | 26 +++++++++++++++++
dev_tests/aastest-class.jl | 23 +++++++++++++++++
parser.mly | 6 +++
sast.ml | 2 ++
sast.ml | 2 ++
scanner.mll | 2 ++
semant.ml | 66 ++++++++++++++++++++++++--
8 files changed, 90 insertions(+), 42 deletions(-)

Commit: f1e952e4b34cf0c8a8a48a7a224c436a3fa96700
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Wed Apr 21 18:02:20 2021 -0400

added return check to main in semant

semant.ml | 6 ++++
tests/fail-for1.jl | 5 ++++
tests/fail-return3.err | 1 +
tests/fail-return3.jl | 2 ++
tests/test-add1.jl | 1 -
tests/test-while2.jl | 1 -
6 files changed, 11 insertions(+), 5 deletions(-)

commit 43d0873a9072de93ba3e36153e01ac3cf0c83
Author: 314pies <ian121363@gmail.com>
Date: Wed Apr 21 14:19:50 2021 -0700

remove test-array-func1, add fail-return1 test

tests/fail-return1.err | 1 +
tests/fail-return1.jl | 4 ++++
tests/test-array-func1.jl | 6 ------
tests/test-array-func1.out | 1 -
4 files changed, 5 insertions(+), 7 deletions(-)

commit b013dd6747aca6fe4205b6f90fde3fc0000d6a46
Merge: c150b0e a8d00d8
Author: 314pies <ian121363@gmail.com>
Date: Wed Apr 21 16:54:17 2021 -0400

Merge pull request #2 from FranCao/functional

Functional

commit a8d00d8c5d14bff07a2c6a8c181ea4f52f3cd6e
Author: 314pies <ian121363@gmail.com>
Date: Wed Apr 21 13:52:49 2021 -0700

Fixes duplicate functional declaration in test cases

SampleCode1.jl | 10 +++
tests/fail-array1.jl | 5 ----
tests/fail-array2.jl | 5 ----
tests/fail-array3.jl | 13 ++++
tests/fail-array4.jl | 7 ++
tests/fail-assign1.jl | 13 ++++
tests/fail-assign2.jl | 8 ++
tests/fail-assign3.jl | 7 ++
tests/fail-class1.jl | 9 ++
tests/fail-class2.jl | 9 ++
tests/fail-dead1.jl | 9 ++
tests/fail-dead2.jl | 14 ++++
tests/fail-double1.jl | 6 ----
tests/fail-double2.jl | 6 ----
tests/fail-expr1.jl | 5 ----
tests/fail-expr2.jl | 5 ----
tests/fail-expr3.jl | 5 ----
tests/fail-for1.jl | 17 ++++
<table>
<thead>
<tr>
<th>Path</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>tests/test-func8.jl</td>
<td>8 ++-----</td>
</tr>
<tr>
<td>tests/test-func9.jl</td>
<td>7 -------</td>
</tr>
<tr>
<td>tests/test-gcd.jl</td>
<td>11 +++++---</td>
</tr>
<tr>
<td>tests/test-gcd2.jl</td>
<td>10 +++++---</td>
</tr>
<tr>
<td>tests/test-if1.jl</td>
<td>8 -------</td>
</tr>
<tr>
<td>tests/test-if2.jl</td>
<td>8 -------</td>
</tr>
<tr>
<td>tests/test-if3.jl</td>
<td>8 -------</td>
</tr>
<tr>
<td>tests/test-if4.jl</td>
<td>8 -------</td>
</tr>
<tr>
<td>tests/test-if5.jl</td>
<td>11 +++++---</td>
</tr>
<tr>
<td>tests/test-if6.jl</td>
<td>10 +++++---</td>
</tr>
<tr>
<td>tests/test-local1.jl</td>
<td>7 -------</td>
</tr>
<tr>
<td>tests/test-local2.jl</td>
<td>5 -------</td>
</tr>
<tr>
<td>tests/test-ops1.jl</td>
<td>53 ++++++++++++++++++------------------------</td>
</tr>
<tr>
<td>tests/test-ops2.jl</td>
<td>32 ++++++++++++++++++--</td>
</tr>
<tr>
<td>tests/test-performance1.jl</td>
<td>11 +++++---</td>
</tr>
<tr>
<td>tests/test-string-binop1.jl</td>
<td>11 +++++---</td>
</tr>
<tr>
<td>tests/test-string-binop2.jl</td>
<td>22 ++++++++++++---</td>
</tr>
<tr>
<td>tests/test-var1.jl</td>
<td>8 -------</td>
</tr>
<tr>
<td>tests/test-var3.jl</td>
<td>11 +++++---</td>
</tr>
<tr>
<td>tests/test-while1.jl</td>
<td>14 +++++---</td>
</tr>
<tr>
<td>tests/test-while2.jl</td>
<td>9 +++++---</td>
</tr>
<tr>
<td>96 files changed, 431 insertions(+), 701 deletions(-)</td>
<td></td>
</tr>
</tbody>
</table>

Commit: 86a733aaa2ff1c4146c23b161b146f10ee5ddc392
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Wed Apr 21 15:56:49 2021 -0400

main fix done -- test cases need to be changed

<table>
<thead>
<tr>
<th>Path</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ast.ml</td>
<td>7 +++++--</td>
</tr>
<tr>
<td>parser.mly</td>
<td>20 +++++++++++++++++++</td>
</tr>
<tr>
<td>semant.ml</td>
<td>12 ++++++++--</td>
</tr>
<tr>
<td>tests/fail-functional1.err</td>
<td>1 +</td>
</tr>
<tr>
<td>tests/fail-functional1.jl</td>
<td>9 +++++++</td>
</tr>
<tr>
<td>tests/test-functional1.jl</td>
<td>2 ++</td>
</tr>
<tr>
<td>tests/test-functional1.out</td>
<td>1 +</td>
</tr>
<tr>
<td>tests/test-functional2.jl</td>
<td>11 +++++++</td>
</tr>
<tr>
<td>tests/test-functional2.out</td>
<td>2 ++</td>
</tr>
<tr>
<td>tests/test-functional3.jl</td>
<td>20 ++++++++--</td>
</tr>
<tr>
<td>tests/test-functional3.out</td>
<td>4 +++++</td>
</tr>
<tr>
<td>11 files changed, 77 insertions(+), 12 deletions(-)</td>
<td></td>
</tr>
</tbody>
</table>

Commit: c150b0e5ecf8c66883c64899ed4890ab4e1ef7fb
Merge: di3682f118a270
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Wed Apr 21 13:08:40 2021 -0400

Merge branch 'main' of github.com:FranCao/javalite into main

Commit: di3682f1b5fbc96b5a21b0bc1a07eb236fb6a3368
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Wed Apr 21 13:08:34 2021 -0400

minor change to test-for1

<table>
<thead>
<tr>
<th>Path</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>tests/test-for1.jl</td>
<td>3 +++</td>
</tr>
<tr>
<td>1 file changed, 1 insertion(+), 2 deletions(-)</td>
<td></td>
</tr>
</tbody>
</table>
commit 118a27009484af3c3ab53a8f1a887a4bfa59c98
Author: Hongfei ChenCU <hc3222@columbia.edu>
Date: Tue Apr 20 22:10:58 2021 -0700

    local hide formal updated

codegen.ml | 7 ++++++--
seman.ml | 13 +++++----------
tests/test-local2.jl | 2 +--
3 files changed, 14 insertions(+), 8 deletions(-)

commit 6b8db289efa7f3f8c7b1dd4d64f8a73057c6e359b783
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Wed Apr 21 01:02:05 2021 -0400

    fix tests, remove some, rename

tests/fail-assign1.jl | 3 +--
tests/fail-assign2.err | 2 +--
tests/fail-assign3.err | 2 +--
tests/fail-dead1.jl | 1 +
tests/test-arith5.jl | 5 ------
tests/test-arith6.out | 1 -
tests/test-arith6.out | 5 ------
tests/test-counting-sort.jl | 4 +---
tests/{test-performence1.jl => test-performance1.jl} | 0
tests/{test-performence1.out => test-performance1.out} | 0
11 files changed, 6 insertions(+), 18 deletions(-)

commit c45a9c3bec03fae95f77c5fc6f96b62e1a022198
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Tue Apr 20 23:26:07 2021 -0400

    refactoring/fixing of tests + new arith tests

Makefile | 2 +--
stringfuncs.o | Bin 0 -> 3272 bytes
{testold => tests}/fail-assign1.err | 0
{testold => tests}/fail-assign1.jl | 7 +-----
{testold => tests}/fail-assign2.err | 0
tests/fail-assign2.jl | 5 +-----
{testold => tests}/fail-assign3.err | 0
tests/fail-assign3.jl | 9 ++++++++}
{testold => tests}/fail-dead1.err | 0
{testold => tests}/fail-dead1.jl | 5 ------
{testold => tests}/fail-dead2.err | 0
{testold => tests}/fail-dead2.jl | 2 +--
{testold => tests}/fail-double1.err | 0
{testold => tests}/fail-double1.jl | 0
{testold => tests}/fail-double2.err | 0
{testold => tests}/fail-double2.jl | 0
{testold => tests}/fail-expr1.err | 2 +--
tests/fail-expr1.jl | 11 ++++++++}
{testold => tests}/fail-expr2.err | 2 +--
tests/fail-expr2.jl | 11 ++++++++
632  {testsold => tests}/fail-expr3.err |  2 ++
633  tests/fail-expr3.jl |  11 ++++++++  
634  {testsold => tests}/fail-for1.err |  0
635  {testsold => tests}/fail-for1.jl |  2 ++
636  {testsold => tests}/fail-for2.err |  0
637  {testsold => tests}/fail-for2.jl |  2 ++
638  {testsold => tests}/fail-for3.err |  0
639  {testsold => tests}/fail-for3.jl |  2 ++
640  {testsold => tests}/fail-for4.err |  0
641  {testsold => tests}/fail-for4.jl |  2 ++
642  {testsold => tests}/fail-for5.err |  0
643  {testsold => tests}/fail-for5.jl |  2 ++
644  {testsold => tests}/fail-func1.err |  0
645  {testsold => tests}/fail-func1.jl |  0
646  {testsold => tests}/fail-func2.err |  0
647  {testsold => tests}/fail-func2.jl |  0
648  {testsold => tests}/fail-func3.err |  0
649  {testsold => tests}/fail-func3.jl |  0
650  {testsold => tests}/fail-func4.err |  0
651  {testsold => tests}/fail-func4.jl |  0
652  {testsold => tests}/fail-func6.err |  0
653  {testsold => tests}/fail-func6.jl |  0
654  {testsold => tests}/fail-func7.err |  0
655  {testsold => tests}/fail-func7.jl |  0
656  {testsold => tests}/fail-func8.err |  0
657  {testsold => tests}/fail-func8.jl |  0
658  {testsold => tests}/fail-func9.err |  0
659  {testsold => tests}/fail-func9.jl |  0
660  {testsold => tests}/fail-if1.err |  0
661  {testsold => tests}/fail-if1.jl |  0
662  {testsold => tests}/fail-if2.err |  0
663  {testsold => tests}/fail-if2.jl |  0
664  {testsold => tests}/fail-if3.err |  0
665  {testsold => tests}/fail-if3.jl |  0
666  {testsold => tests}/fail-nomain.err |  0
667  {testsold => tests}/fail-nomain.jl |  0
668  {testsold => tests}/fail-print.err |  0
669  {testsold => tests}/fail-print.jl |  0
670  {testsold => tests}/fail-return1.err |  0
671  {testsold => tests}/fail-return1.jl |  0
672  {testsold => tests}/fail-return2.err |  0
673  {testsold => tests}/fail-return2.jl |  0
674  {testsold => tests}/fail-while1.err |  0
675  {testsold => tests}/fail-while1.jl |  2 ++
676  {testsold => tests}/fail-while2.err |  0
677  {testsold => tests}/fail-while2.jl |  2 ++
678  {testsold => tests}/test-add1.err |  0
679  {testsold => tests}/test-add1.out |  0
680  {testsold => tests}/test-arith1.err |  0
681  {testsold => tests}/test-arith1.out |  0
682  {testsold => tests}/test-arith2.err |  0
683  {testsold => tests}/test-arith2.out |  0
684  {testsold => tests}/test-arith3.err |  3 ++-
685  {testsold => tests}/test-arith3.out |  0
686  tests/test-arith4.jl |  5 ++++
687  tests/test-arith4.out |  1 +
688  tests/test-arith5.jl |  5 ++++
<table>
<thead>
<tr>
<th>Path</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>{testsold =&gt; tests}/test-ops2.jl</td>
<td>0</td>
</tr>
<tr>
<td>{testsold =&gt; tests}/test-ops2.out</td>
<td>0</td>
</tr>
<tr>
<td>{testsold =&gt; tests}/test-performance1.jl</td>
<td>6 ++-----</td>
</tr>
<tr>
<td>{testsold =&gt; tests}/test-performance1.out</td>
<td>0</td>
</tr>
<tr>
<td>{testsold =&gt; tests}/test-var1.jl</td>
<td>3 --</td>
</tr>
<tr>
<td>{testsold =&gt; tests}/test-var1.out</td>
<td>0</td>
</tr>
<tr>
<td>tests/test-var3.jl</td>
<td>8 +++++++</td>
</tr>
<tr>
<td>{testsold =&gt; tests}/test-var3.out</td>
<td>0</td>
</tr>
<tr>
<td>{testsold =&gt; tests}/test-while1.jl</td>
<td>3 --</td>
</tr>
<tr>
<td>{testsold =&gt; tests}/test-while1.out</td>
<td>0</td>
</tr>
<tr>
<td>{testsold =&gt; tests}/test-while2.jl</td>
<td>3 --</td>
</tr>
<tr>
<td>{testsold =&gt; tests}/test-while2.out</td>
<td>0</td>
</tr>
<tr>
<td>testsold/fail-assign2.jl</td>
<td>7 -------</td>
</tr>
<tr>
<td>testsold/fail-assign3.jl</td>
<td>11 --------</td>
</tr>
<tr>
<td>testsold/fail-exp1.jl</td>
<td>18 ------------</td>
</tr>
<tr>
<td>testsold/fail-exp2.jl</td>
<td>14 ------------</td>
</tr>
<tr>
<td>testsold/fail-exp3.jl</td>
<td>14 ------------</td>
</tr>
<tr>
<td>testsold/fail-func5.err</td>
<td>1 -</td>
</tr>
<tr>
<td>testsold/fail-func5.jl</td>
<td>14 ------------------</td>
</tr>
<tr>
<td>testsold/fail-global1.err</td>
<td>1 -</td>
</tr>
<tr>
<td>testsold/fail-global1.jl</td>
<td>9 ----------</td>
</tr>
<tr>
<td>testsold/fail-global2.err</td>
<td>1 -</td>
</tr>
<tr>
<td>testsold/fail-global2.jl</td>
<td>9 ----------</td>
</tr>
<tr>
<td>testsold/test-aglobal4.jl</td>
<td>7 -------</td>
</tr>
<tr>
<td>testsold/test-aglobal4.out</td>
<td>2 --</td>
</tr>
<tr>
<td>testsold/test-bubble-sort.jl</td>
<td>27 -------------------------------</td>
</tr>
<tr>
<td>testsold/test-double2.jl</td>
<td>11 --</td>
</tr>
<tr>
<td>testsold/test-func7.jl</td>
<td>13 ------------</td>
</tr>
<tr>
<td>testsold/test-func7.out</td>
<td>1 -</td>
</tr>
<tr>
<td>testsold/test-global1.jl</td>
<td>30 ---------------------------------</td>
</tr>
<tr>
<td>testsold/test-global1.out</td>
<td>4 --</td>
</tr>
<tr>
<td>testsold/test-global12.jl</td>
<td>10 -----------</td>
</tr>
<tr>
<td>testsold/test-global12.out</td>
<td>1 -</td>
</tr>
<tr>
<td>testsold/test-global13.jl</td>
<td>11 ----------</td>
</tr>
<tr>
<td>testsold/test-global13.out</td>
<td>1 -</td>
</tr>
<tr>
<td>testsold/test-var2.jl</td>
<td>13 ----------</td>
</tr>
<tr>
<td>testsold/test-var2.out</td>
<td>1 -</td>
</tr>
<tr>
<td>testsold/test-var3.jl</td>
<td>10 ----------</td>
</tr>
<tr>
<td>172 files changed, 148 insertions(+), 315 deletions(-)</td>
<td></td>
</tr>
</tbody>
</table>

commit 09308ae1f1aa31e422bc3eb5ca506c7f80c06b3c
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Tue Apr 20 22:07:30 2021 -0400

fixed Makefile, removed stringfuncs.o

<table>
<thead>
<tr>
<th>Path</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Makefile</td>
<td>4 +++--</td>
</tr>
<tr>
<td>stringfuncs.o</td>
<td>Bin 3272 -&gt; 0 bytes</td>
</tr>
<tr>
<td>2 files changed, 2 insertions(+), 2 deletions(-)</td>
<td></td>
</tr>
</tbody>
</table>

commit bc4d028dc5d333f8bbc5d43b0e85031851d5f91d3
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Tue Apr 20 22:06:26 2021 -0400

added object files to clean target Makefile

<table>
<thead>
<tr>
<th>Path</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Makefile</td>
<td>2 --</td>
</tr>
</tbody>
</table>
update tests, cleanup old files

funcs/reversestring.c | 27 -------------------
funcs/stringlower.c  | 28 -------------------
funcs/stringsubstring.c | 31 -------------------
funcs/stringupper.c  | 28 -------------------
tests/fail-string-binop1.err | 1 +
tests/fail-string-binop1.jl | 6 ++++
{testsold => tests}/test-array-func1.jl | 0
{testsold => tests}/test-array-func1.out | 0
tests/test-func-string1.jl | 14 ++++++++++
tests/test-func-string1.out | 5 ++++
tests/test-func-string2.jl | 10 +++++++
.../test-func-string2.out | 0
.../test-func-string3.jl | 11 ++++++
.../test-func-string3.out | 0
tests/test-func-string4.jl | 22 +++++++++++
.../test-func-string4.out | 0
tests/test-string-binop1.jl | 8 ++++
tests/test-string-binop1.out | 2 ++
testsold/test-afunc-string1.jl | 11 --------
testsold/test-afunc-string1.out | 3 ---
testsold/test-afunc-string2.jl | 9 -------
testsold/test-afunc-string2.out | 2 --
testsold/test-afunc-string3.jl | 14 --------
testsold/test-afunc-string5.jl | 29 -----------------
testsold/test-astring-binop.jl | 6 ----
testsold/test-astring-binop.out | 1 -
testsold/test-hello.jl | 7 ----
testsold/test-hello.out | 3 ---
testsold/test-helloworld.jl | 4 ---
testsold/test-helloworld.out | 1 -
testsold/test-printstring.jl | 6 ----
testsold/test-printstring.out | 1 -
32 files changed, 73 insertions(+), 217 deletions(-)

variable decl assign done; tests partially updated

codegen.ml | 56 +++++-----------------------------
semant.ml | 14 +++++---
tests/fail-array1.err | 2 +-
tests/fail-array1.jl | 3 +-
tests/fail-array2.jl | 3 +-
tests/fail-array3.jl | 11 ++++
tests/fail-array4.jl | 3 +-
tests/fail-class1.jl | 9 ++++
tests/fail-class2.err | 2 +-
<table>
<thead>
<tr>
<th>Number</th>
<th>File Path</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>{tests =&gt; testsold}/test-performence1.jl</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>{tests =&gt; testsold}/test-performence1.out</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>{tests =&gt; testsold}/test-printstring.jl</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>{tests =&gt; testsold}/test-printstring.out</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>{tests =&gt; testsold}/test-var1.jl</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>{tests =&gt; testsold}/test-var1.out</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>{tests =&gt; testsold}/test-var2.jl</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>{tests =&gt; testsold}/test-var2.out</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>{tests =&gt; testsold}/test-var3.jl</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>{tests =&gt; testsold}/test-var3.out</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>{tests =&gt; testsold}/test-while1.jl</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>{tests =&gt; testsold}/test-while1.out</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>{tests =&gt; testsold}/test-while2.jl</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>{tests =&gt; testsold}/test-while2.out</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>194 files changed, 59 insertions(+), 146 deletions(-)</td>
</tr>
</tbody>
</table>

commit 60f2abf52f0f2a26ff62777e2bd63a7cd97fbff
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Tue Apr 20 12:23:15 2021 -0700

still in progress

codegen.ml | 42 ++++++++++++++++++++++++++++++++
dev_tests/stest-var.jl | 3 ++-
parser.mly   | 2 +- tests/test-class1.jl | 3 ++-
tests/test-class2.jl | 12 +++++-
tests/test-class3.jl | 27 ++++++++----
7 files changed, 61 insertions(+), 45 deletions(-)

commit 9d4f97f0d3a612e1d1a231d724d490d3fa27906ec
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Tue Apr 20 10:39:15 2021 -0700

  up to semant (remove extra)
_build/javalite.o | Bin 7960 -> 0 bytes
_build/ocamlc.where | 1 -
_build/parser.cmi | Bin 1373 -> 0 bytes
_build/parser.cmx | Bin 4700 -> 0 bytes
_build/parser.ml | 885 ------------------------
_build/parser.ml.depends | 1 -
_build/parser.ml | 45 --
_build/parser.ml.depends | 1 -
_build/parser.mly | 145 -----
_build/parser.o | Bin 44024 -> 0 bytes
_build/sast.cmi | Bin 2888 -> 0 bytes
_build/sast.cmo | Bin 5667 -> 0 bytes
_build/sast.cmx | Bin 620 -> 0 bytes
_build/sast.ml | 114 ----
_build/sast.ml.depends | 1 -
_build/sast.o | Bin 25896 -> 0 bytes
_build/scanner.cmi | Bin 1270 -> 0 bytes
_build/scanner.cmo | Bin 21774 -> 0 bytes
_build/scanner.cmx | Bin 20233 -> 0 bytes
_build/scanner.ml | 1528 --------------------------------------------
_build/scanner.ml.depends | 1 -
_build/scanner.mll | 66 --
_build/scanner.o | Bin 30608 -> 0 bytes
_build/semant.cmi | Bin 7135 -> 0 bytes
_build/semant.cmo | Bin 13720 -> 0 bytes
_build/semant.cmx | Bin 2431 -> 0 bytes
_build/semant.ml | 397 -----------
_build/semant.ml.depends | 1 -
_build/semant.o | Bin 68168 -> 0 bytes
javalite.native | 1 -
50 files changed, 3972 deletions(-)

commit 131fe05d109210cad21617eb0425e047542d782e
Author: HongfeiChenCU <hc3222@columbia.edu>
Date:  Tue Apr 20 10:38:46 2021 -0700
up to semant

_build/_digests | 32 +
_build/_log | 80 +++
_build/ast.cmi | Bin 0 -> 3328 bytes
_build/ast.cmo | Bin 0 -> 5857 bytes
_build/ast.cmx | Bin 0 -> 687 bytes
_build/ast.ml | 139 ++++
_build/ast.ml.depends | 1 +
_build/ast.o | Bin 0 -> 25760 bytes
_build/codegen.cmi | Bin 0 -> 5059 bytes
_build/codegen.cmo | Bin 0 -> 17658 bytes
_build/codegen.cmx | Bin 0 -> 1616 bytes
_build/codegen.ml | 499 ++++++++++++++
_build/codegen.ml.depends | 1 +
_build/codegen.o | Bin 0 -> 87592 bytes
_build/javalite.cmi | Bin 0 -> 795 bytes
_build/javalite.cmo | Bin 0 -> 1956 bytes
_build/javalite.cmx | Bin 0 -> 903 bytes
_build/javalite.ml | 32 +
_build/javalite.ml.depends | 1 +
<table>
<thead>
<tr>
<th>File</th>
<th>Size Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>_build/javalite.native</td>
<td>Bin 0 -&gt; 1423656 bytes</td>
</tr>
<tr>
<td>_build/javalite.o</td>
<td>Bin 0 -&gt; 7960 bytes</td>
</tr>
<tr>
<td>_build/ocamlc.where</td>
<td>1 +</td>
</tr>
<tr>
<td>_build/parser.cmi</td>
<td>Bin 0 -&gt; 1373 bytes</td>
</tr>
<tr>
<td>_build/parser.cmx</td>
<td>Bin 0 -&gt; 4700 bytes</td>
</tr>
<tr>
<td>_build/parser.ml</td>
<td>885 +++++++++++++++++++++++</td>
</tr>
<tr>
<td>_build/parser.ml.depends</td>
<td>1 +</td>
</tr>
<tr>
<td>_build/parser.mli</td>
<td>45 ++</td>
</tr>
<tr>
<td>_build/parser.mli.depends</td>
<td>1 +</td>
</tr>
<tr>
<td>_build/parser.mly</td>
<td>145 +++++</td>
</tr>
<tr>
<td>_build/parser.o</td>
<td>Bin 0 -&gt; 44024 bytes</td>
</tr>
<tr>
<td>_build/sast.cmi</td>
<td>Bin 0 -&gt; 2888 bytes</td>
</tr>
<tr>
<td>_build/sast.cmo</td>
<td>Bin 0 -&gt; 5667 bytes</td>
</tr>
<tr>
<td>_build/sast.cmx</td>
<td>Bin 0 -&gt; 620 bytes</td>
</tr>
<tr>
<td>_build/sast.ml</td>
<td>114 +++++</td>
</tr>
<tr>
<td>_build/sast.ml.depends</td>
<td>1 +</td>
</tr>
<tr>
<td>_build/sast.o</td>
<td>Bin 0 -&gt; 25896 bytes</td>
</tr>
<tr>
<td>_build/scanner.cmi</td>
<td>Bin 0 -&gt; 1270 bytes</td>
</tr>
<tr>
<td>_build/scanner.cmo</td>
<td>Bin 0 -&gt; 21774 bytes</td>
</tr>
<tr>
<td>_build/scanner.cmx</td>
<td>Bin 0 -&gt; 20233 bytes</td>
</tr>
<tr>
<td>_build/scanner.ml</td>
<td>1528 +++++++++++++++++++++++</td>
</tr>
<tr>
<td>_build/scanner.ml.depends</td>
<td>1 +</td>
</tr>
<tr>
<td>_build/scanner.ml1</td>
<td>66 ++</td>
</tr>
<tr>
<td>_build/scanner.o</td>
<td>Bin 0 -&gt; 30608 bytes</td>
</tr>
<tr>
<td>_build/semant.cmi</td>
<td>Bin 0 -&gt; 7135 bytes</td>
</tr>
<tr>
<td>_build/semant.cmo</td>
<td>Bin 0 -&gt; 13720 bytes</td>
</tr>
<tr>
<td>_build/semant.cmx</td>
<td>Bin 0 -&gt; 2431 bytes</td>
</tr>
<tr>
<td>_build/semant.ml</td>
<td>397 +++++++++++++</td>
</tr>
<tr>
<td>_build/semant.ml.depends</td>
<td>1 +</td>
</tr>
<tr>
<td>_build/semant.o</td>
<td>Bin 0 -&gt; 68168 bytes</td>
</tr>
<tr>
<td>ast.ml</td>
<td>28 +</td>
</tr>
<tr>
<td>codegen.ml</td>
<td>9 +</td>
</tr>
<tr>
<td>dev_tests/stest-var.jl</td>
<td>4 +</td>
</tr>
<tr>
<td>javalite.native</td>
<td>1 +</td>
</tr>
<tr>
<td>parser.mly</td>
<td>24 +</td>
</tr>
<tr>
<td>sast.ml</td>
<td>8 +</td>
</tr>
<tr>
<td>semant.ml</td>
<td>41 +</td>
</tr>
<tr>
<td>tests/test-class4.jl</td>
<td>9 +</td>
</tr>
<tr>
<td></td>
<td>57 files changed, 4051 insertions(+), 44 deletions(-)</td>
</tr>
</tbody>
</table>

commit bb14d66b5acfc2578f39f0a0abb8ef494d5b6c670
Merge: d58a198b510aa8
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Tue Apr 20 09:02:35 2021 -0700

Merge branch 'main' of https://github.com/FranCao/javalite into main

commit d58a198ab64b6c4351e69f901d303bd7e1ce580
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Tue Apr 20 09:02:18 2021 -0700

string binop done

codegen.ml | 30 +++++++++++++++++++++
dev_tests/eqaulinc.c | 8 +++++
semant.ml | 2 +
tests/test-class5.jl | 62 +++++++++++++++++++++++
1202 tests/test-class5.out    |  7 +++++
1203 tests/test-string-binop2.jl | 15 ++++++++ 
1204 tests/test-string-binop2.out |  7 +++++
1205 7 files changed, 120 insertions(+), 11 deletions(-)
1206 commit b510aa844210ccda421d3193b6e672f93abf3eb
1207 Merge: eebeb3f b70a0fd
1208 Author: Frances Cao <frances.te.cao@gmail.com>
1209 Date: Mon Apr 19 21:41:57 2021 -0400
1210
1211 resolve conflict
1212
1213 commit eebeb3f9818d3e9013b7b1cefe87210e106499715
1214 Author: Frances Cao <frances.te.cao@gmail.com>
1215 Date: Mon Apr 19 21:40:49 2021 -0400
1216
1217 start print array in codegen and semant, add test
1218
codegen.ml |  23 +++++++++++++++++----
1219 semant.ml | 12 ++++++++---
1220 tests/test-array-func1.jl |  8 +++++
1221 tests/test-array-func1.out |  1 +
1222 4 files changed, 33 insertions(+), 11 deletions(-)
1223 commit b70a0fd96c3399d068524ddf459d0d521ab716f
1224 Author: 314pies <ian121363@gmail.com>
1225 Date: Mon Apr 19 17:41:30 2021 -0700
1226
1227 Rename buildAndRun.sh and change its log file name
1228
1229 buildAndRun.sh => BuildAndRun.sh |  4 +---
1230 1 file changed, 2 insertions(+), 2 deletions(-)
1231 commit 1db655421a19e1829a27d8298a375f35ea0fccc9
1232 Author: 314pies <ian121363@gmail.com>
1233 Date: Mon Apr 19 17:36:36 2021 -0700
1234
1235 Add BuildAndRun.sh for simply build, execute and show the execution result of the source code file. For example: You can run './BuildAndRun.sh SampleCode1.jl' and it will show the execution result of SampleCode1.jl
1236 SampleCode1.jl |  7 +++
1237 buildAndRun.sh | 145 ++++++++++++++++++++++++++++++++++++++++
1238 2 files changed, 152 insertions(+)
1239 commit badd72f7b972e1310cda7e2859b5e2237d0f38c
1240 Author: 314pies <ian121363@gmail.com>
1241 Date: Mon Apr 19 16:53:45 2021 -0700
1242
1243 Add (optional) representative source language program - counting sort
1244 tests/test-counting-sort.jl | 32 ++++++++++++++++++++++++
1245 tests/test-counting-sort.out | 15 ++++++++ 
1246 2 files changed, 47 insertions(+)
1247 commit badd72f7b972e1310cda7e2859b5e2237d0f38c
1248 Author: 314pies <ian121363@gmail.com>
Date: Mon Apr 19 16:14:35 2021 -0700

Add performance measurement testing code

```
tests/test-performance1.jl | 14 ++++++++++++++
tests/test-performance1.out | 1 +
2 files changed, 15 insertions(+)
```

commit 8477bd4f62099db447646b6220e31abac8bef726
Author: 314pies <ian121363@gmail.com>
Date: Mon Apr 19 15:54:27 2021 -0700

Add representative source language program - bubble sort

```
tests/test-bubble-sort.jl | 27 ++++++++++++++++++++++++++
tests/test-bubble-sort.out | 15 +++++++++++++++
2 files changed, 42 insertions(+)
```

commit 70abb26e2460bddd3c512ac1f035f357a4276442a
Merge: 4e1867f a74452b
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Sun Apr 18 23:16:34 2021 -0700

Merge branch 'hc-class' into main

```
commit a74452bd6b44c9e745ee5ab3e4b735e109ed6711
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Sun Apr 18 23:15:50 2021 -0700

class done
```

```
ast.ml | 4 +
codegen.ml | 94 ++++++++++++++++++++----
dev_tests/aatst-class.jl | 26 +++++
arrayinc.c => dev_tests/arrayinc.c | 0
ast-test-arrays.jl => dev_tests/ast-test-arrays.jl | 0
dev_tests/classinc.c | 24 +++++
parser.mly | 2 +-s
ast.ml | 9 +-s
semant.ml | 55 +++++++++--
aatst-class.jl => tests/test-class1.jl | 10 +--
tests/test-class1.out | 4 +
tests/test-class2.jl | 24 +++++
tests/test-class2.out | 2 +
tests/test-class3.jl | 40 +++++++++
tests/test-class3.out | 2 +
tests/test-class4.jl | 21 +++++
tests/test-class4.out | 2 +
17 files changed, 275 insertions(+), 44 deletions(-)
```

commit 2925e93ddfaff0d20879bfff5b9e230019980ea
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Sun Apr 18 12:45:42 2021 -0700

class semant done

```
Makefile | 2 +-
| _build/scanner.cmi | Bin 1270 -> 0 bytes |
| _build/scanner.cmo | Bin 20198 -> 0 bytes |
| _build/scanner.cmx | Bin 18697 -> 0 bytes |
| _build/scanner.ml | 1421 | 1 - |
| _build/scanner.ml1 | 66 | -- |
| _build/scanner.o | Bin 28968 -> 0 bytes |
| _build/semant.cmi | Bin 4962 -> 0 bytes |
| _build/semant.cmo | Bin 9976 -> 0 bytes |
| _build/semant.cmx | Bin 1453 -> 0 bytes |
| _build/semant.ml | 288 | ------- |
| _build/semant.ml1 | 1 - |
| _build/semant.o | Bin 50280 -> 0 bytes |
| javalite.native | 1 - |
| testall.log | 628 | ----------------- |
| 51 files changed, 4157 deletions(-) |

commit 6876d88fd96477255fd712ee8c2e7b562178c1c
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Sun Apr 18 10:42:35 2021 -0700

array mutation test case added for string[]
commit fb6832d8fa3d91cd82078327d089b1281ed5c0d42
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Sun Apr 18 10:31:46 2021 -0700

  string mutation done

  aatest-class.jl | 0     
  ast-test-arrays.jl | 3 +++
  ast.ml | 2 ++
  codegen.ml | 12 **********
  parser.mly | 18 **************
  sast.ml | 2 ++
  scanner.mll | 2 ++
  semant.ml | 14 **************
  tests/fail-array4.err | 1 +
  tests/fail-array4.jl | 6 ++++
  tests/test-array4.jl | 19 ***************
  tests/test-array4.out | 6 ++++
  12 files changed, 83 insertions(+), 2 deletions(-)

commit de71fe27cd9e336f2a63be684b37f75c1cd6bc07f
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Fri Apr 16 16:01:04 2021 -0700

  string indexof fixed

  stringfuncs.c | 6 ++++
  stringfuncs.o | Bin 3272 -> 3272 bytes
  tests/test-afunc-string5.jl | 6 ++++
  tests/test-afunc-string5.out | 3 +++
  4 files changed, 10 insertions(+), 5 deletions(-)

commit ce589b39600c4b80727cd15c6d6f199aca600a0c
Merge: bfe0ddf 784a98f
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Thu Apr 15 20:39:49 2021 -0400
Merge branch 'main' of github.com:FranCao/javalite into main

commit 5fe0ddfecaebad72d85a8d25bc722490d9ef656
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Thu Apr 15 20:39:42 2021 -0400

combine funcs, add concat, len, and indexof funcs for string

codegen.ml | 21 +++++++++++++
semant.ml | 28 +++++++++++++++++
stringfuncs.c | 35 ++++++++++++++++++++ stringfuncs.o | Bin 2528 -> 3272 bytes
tests/test-afunc-string3.jl | 5 ++
tests/test-afunc-string3.out | 3 --
tests/test-afunc-string5.jl | 25 +++++++++++++
tests/test-afunc-string5.out | 4 ++
...est-afunc-string5.out => test-afunc-string5.out) | 0
tests/test-afunc-string5.out => test-afunc-string1.out) | 0
tests/test-afunc-string1.out => test-afunc-string5.out) | 8 ----
tests/test-afunc-string1.out => test-afunc-string5.out) | 1 -
12 files changed, 120 insertions(+), 10 deletions(-)

commit 784a98f4e4ebf3e935c130f55bb2164d4e25a10
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Thu Apr 15 16:49:17 2021 -0700

print for arrayaccess added
codegen.ml | 2 --
tests/test-array1.jl | 4 ----
tests/test-array3.jl | 11 ++++++++
tests/test-array3.out | 2 ++
4 files changed, 15 insertions(+), 4 deletions(-)

commit 1922ed0e773974bc34abdfef7eb9d830b50ab4c
Merge: 955689f bc6d10f
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Thu Apr 15 17:27:55 2021 -0400

Merge branch 'main' of github.com:FranCao/javalite into main

commit 955689f14a95db3ad3c0ac7460e1d9e8de445bb3
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Thu Apr 15 17:27:51 2021 -0400

concatenate built in functions to one file

Makefile | 20 +--------
_tags | 13 +-----
codegen.ml | 5 +++
reversestring.c => funcs/reversestring.c | 0
stringlower.c => funcs/stringlower.c | 0
stringsubstring.c => funcs/stringsubstring.c | 0
stringupper.c => funcs/stringupper.c | 0
reversestring.o | Bin 1592 -> 0 bytes
sast.ml | 4 ++
commit bc6d10f7d6a1c88601a3a6d34e6653f0a7da0d0
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Thu Apr 15 12:24:09 2021 -0700

small array fix

codegen.ml | 4 +++-
1 file changed, 1 insertion(+), 3 deletions(-)

commit 034481221c28b92937df7fe3fbc7cc310ef3d8
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Wed Apr 14 16:41:42 2021 -0700

array lit and access done

codegen.ml | 33 +++++++++++++++++++-----------------
tests/test-array2.jl | 13 +++++++++++
tests/test-array2.out | 5 +++
3 files changed, 37 insertions(+), 14 deletions(-)

commit cf08e0c0c31f15ec1d5172830bc781a27556b120
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Wed Apr 14 15:48:02 2021 -0700

array basic done

arrayinc.c | 4 +++
ast.ml | 2 +-
codegen.ml | 40 +++++++++++++++++++++++++++++++++------
parser.mly | 2 +-
sast.ml | 2 +-
tests/test-aglobal4.jl | 7 ++++
tests/test-aglobal4.out | 2 ++
tests/test-array1.jl | 15 ++++++++
tests/test-array1.out | 4 ++++
9 files changed, 69 insertions(+), 9 deletions(-)

commit 528a48b65ca698caecabc1e7db9d46d67f9d4dd9
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Sun Apr 11 19:30:00 2021 -0400

add string binop equals

tests/test-afunc-string5.jl | 6 -----
tests/test-afunc-string5.out | 1 -
tests/test-astring-binop.jl | 14 ~~~~~~~~~~
tests/test-astring-binop.out | 3 ---
tests/test-astring-binop1.jl | 6 +++++
tests/test-string-binop1.out | 1 +
tests/test-string-binop2.jl | 8 ++++++++
tests/test-string-binop2.out | 1 +
8 files changed, 16 insertions(+), 24 deletions(-)

commit 09fad9a4c8942809ff82b2b6d3a81e88aab0f66b82
Merge: 6dbb993 456f5f0e
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Sun Apr 11 16:12:06 2021 -0700
merge string binop

commit 6dbb99309c0c8c24380aadb570a1e7a1cd1b6d22
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Sun Apr 11 16:10:29 2021 -0700

string addition added

codegen.ml | 26 ++++++++++++++++++++------
sast.ml | 6 +++++
semant.ml | 10 +++++++++
tests/test-afunc-string5.jl | 6 +++++
tests/test-afunc-string5.out | 1 +
5 files changed, 44 insertions(+), 5 deletions(-)

commit 456f0e4d3048a86a2b3283c850f0b3688f
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Sun Apr 11 00:59:32 2021 -0400

minor print changes, add test for string binop

codegen.ml | 16 ++++++++++++++++++++------
tests/test-string-binop.jl | 14 +++++++++++++
tests/test-string-binop.out | 3 +++
3 files changed, 29 insertions(+), 4 deletions(-)

commit f2910da50b50b94dd1ed7f3ba7dd20614f50f98cf
Merge: 61edc04 28dc68c
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Fri Apr 9 23:25:38 2021 -0700

merged array changes

commit 61edc04af63e3a6eb75da5e904991bb4293b6258
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Sat Apr 10 06:18:32 2021 +0000

print done

Makefile | 2 ++
ast.ml | 4 ++
codegen.ml | 72 +--------------------------------------------------
semant.ml | 4 ++
tests/test-afunc-string2.jl | 2 ++
tests/test-afunc-string4.jl | 2 ++
tests/test-helloworld.jl | 2 ++
7 files changed, 33 insertions(+), 55 deletions(-)
commit ce81a3s827bca3107bab07ad36ae1df2bd7a6e6c
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Fri Apr 9 21:58:41 2021 -0700

    print working except string and double

    Makefile  |  7 ++--
    ast.ml    |  3 +-  
    codegen.ml| 103 +++++++++++++++++++++++++++++++++++++++-
    semant.ml | 12 +++++-
    tests/fail-printb.err  |  1 -
    tests/fail-printb.jl    |  2 -
    tests/test-afunc-string1.jl |  6 +++-
    tests/test-afunc-string2.jl |  2 +-
    tests/test-afunc-string3.jl |  4 +-
    tests/test-afunc-string4.jl |  8 +++-
    tests/test-double1.jl    |  2 +-  
    tests/test-double2.jl    |  2 +-  
    tests/test-double3.jl    | 24 +++++------
    tests/test-ops1.jl       | 28 +++++------
    tests/test-ops2.jl       | 24 +++++------
    tests/test-printstring.jl|  2 +-
    tests/test-var3.jl       |  2 +-  

17 files changed, 138 insertions(+), 94 deletions(-)

commit aca4bb28c4e8b27999fc30df25886d7fc0439ecd6
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Fri Apr 9 00:34:44 2021 -0400

    complete substring built-in function

    codegen.ml            | 34 ++++++++++++++++++++++++++++++++++++++
    semant.ml             | 12 +++++++++--
    stringssubstring.c    |  6 +++--
    stringssubstring.o    | Bin 1600 -> 1608 bytes
    test-afunc-string4.ll |  0 
    tests/test-afunc-string4.jl |  4 +++
    tests/test-afunc-string4.out |  3 ++-

7 files changed, 48 insertions(+), 11 deletions(-)

commit 01a20e573971a33abf28cb3cd07532415351c6e
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Thu Apr 8 20:54:38 2021 -0400

    substring edits

    codegen.ml            |  6 +++++
    test-afunc-string4.ll |  0 
    tests/test-afunc-string4.jl |  4 +++
    tests/test-afunc-string4.out |  2 ++

4 files changed, 6 insertions(+), 6 deletions(-)

commit b578d1e866cab755b6e5e9c2dab2b2e475f769
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Thu Apr 8 20:13:05 2021 -0400
add substring func

| Makefile       | 6 +++++- |
| _tags          | 5 +++++- |
| codegen.ml     | 7 +++++- |
| semant.ml      | 3 +--   |
| stringsubstring.c | 31 +++++++++++++++++++++++++++++++++ |
| stringsubstring.o | Bin 0 -> 1600 bytes |
| testall.sh     | 9 +++++-- |
| tests/test-afunc-string4.jl | 19 +++++++++++++++++++ |
| tests/test-afunc-string4.out | 6 +++++ |

9 files changed, 82 insertions(+), 4 deletions(-)

commit 28dc68c56f7b47b126728903c526a48a912b94bd
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Thu Apr 8 01:47:08 2021 -0700

allow 2d arrays

| ast-test-arrays.jl | 17 +++++++++-- |
| ast.ml            | 10 +------- |
| parser.mly        | 8 +------- |
| scanner.ml        | 5 +--     |
| semant.ml         | 18 +++++-- |

5 files changed, 22 insertions(+), 36 deletions(-)

commit f7c322d464d16a33ea9219e854f2b56e7ebf6274
Merge: b7255e5 b8cdd73
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Wed Apr 7 21:34:48 2021 -0400

Merge branch 'main' of github.com:FranCao/javalite into main

commit b7255e5d76082167e53e04552d4c02f1e15c2280
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Wed Apr 7 21:34:36 2021 -0400

cleanup printbig

| Makefile       | 15 +------ |
| _tags          | 3 --     |
| codegen.ml     | 7 ----   |
| printbig.c     | 75 +++++++++++++++++++++++++++++++++ |
| printbig.o     | Bin 2080 -> 0 bytes |
| semant.ml      | 3 +      |
| testall.sh     | 9 +----- |
| tests/fail-printbig.err | 1 - |
| tests/fail-printbig.jl | 2 -- |
| tests/test-printbig.jl | 25 +------ |
| tests/test-printbig.out | 88 +++++++++++++++++++++++++++++++++ |

11 files changed, 7 insertions(+), 221 deletions(-)

commit b8cdd73554829c81562b578abdd7d018d26e0792
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Wed Apr 7 18:26:32 2021 -0700

class part deleted
commit 4f81c56468f7778f89a585c5dd58e85e8bddd8921
Merge: e0789fc cfc69f
commits: Frances Cao <frances.te.cao@gmail.com>
date: Wed Apr 7 21:00:03 2021 -0400

merge

commits: Frances Cao <frances.te.cao@gmail.com>
date: Wed Apr 7 20:56:45 2021 -0400

etit tests for print functions

delta 125
add upper and lower functions for string

commit 9a656fb885987dd5e45d238039f1c3a243090bd4
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Sun Apr 4 18:30:39 2021 -0400

12 files changed, 81 insertions(+), 19 deletions(-)
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Sun Apr 4 13:24:32 2021 -0400

  minor changes
  
  ast.ml  |  2 +-  
  semant.ml  |  2 ++  
  2 files changed, 3 insertions(+), 1 deletion(-)

commit ab788cc135fdbee3dd5e5e9fc045aa50f964265d
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Sat Mar 27 17:55:21 2021 -0400

  finished sast
  
  sast.ml  |  5 ++++  
  1 file changed, 5 insertions(+)

commit 622ab5357df0a5c7c7c3b7c6a11879c97afaf5d7d4
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Sat Mar 27 13:41:35 2021 -0400

  improvements to pretty printing, mandating int for arr access, not expr
  
  ast-test-arrays.jl  |  2 ++  
  ast.ml  |  6 ++++  
  parser.mly  |  2 ++  
  3 files changed, 5 insertions(+), 5 deletions(-)

commit c8d647564132f314fe93f7eb995668b81591d8df
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Sat Mar 27 13:24:45 2021 -0400

  arrays complete in ast, parser, scanner
  
  ast-test-arrays.jl  |  4 ++++  
  ast.ml  |  10 ++++++++  
  parser.mly  |  12 ++++++++++++  
  scanner.mll  |  4 +++  
  4 files changed, 20 insertions(+), 10 deletions(-)

commit 093fe067fe67f64f2f3141fe93f7eb995668b81591d8df
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Fri Mar 26 06:29:23 2021 -0400

  arrays partially implemented
  
  ast-test-arrays.jl  |  6 ++++  
  ast.ml  |  10 ++++++++  
  parser.mly  |  3 ++  
  3 files changed, 13 insertions(+), 6 deletions(-)

commit f223498628777f72baec1f920b47a3df37128a5
Author: Mateo Maturana <mateomaturana@yahoo.com>
Date: Fri Mar 26 06:03:05 2021 -0400

  ast, parser, scanner for array prelim
CLASS parser, sanner, ast finished

ast-test-class.jl | 26 ++++++++++++++++++++++++++
ast.ml | 38 +++++++++++++++++++++++++++++++++++---
codegen.ml | 11 ++++++++++++++++-----------
parser.mly | 40 +++++++++++++++++++++++++++++++++++--
scanner.mll | 14 ++++++++------
semant.ml | 2 +
tests/test-var3.jl | 10 +++++++
tests/test-var3.out | 2 ++
8 files changed, 126 insertions(+), 17 deletions(-)

commit e82ab35a20f1ea9f2f53bfcfb89fc09fda30e8c2
Author: Frances Cao <frances.te.cao@gmail.com>
Date: Mon Mar 22 21:43:40 2021 -0400

add single line comments, edit test cases, replace float with double

ast.ml | 16 +++++------
codegen.ml | 16 +++++------
parser.mly | 41 ++++++++++++++++--------------
sast.ml | 12 +++++------
scanner.mll | 24 +++++--------
semant.ml | 14 +++++------
tests/fail-double1.err | 1 +
tests/fail-double1.jl | 5 +++
tests/fail-double2.err | 1 +
tests/fail-double2.jl | 5 ++++
tests/fail-exp3.err | 2 ++
tests/fail-float1.err | 1 -
tests/fail-float1.jl | 5 ----
tests/fail-float2.err | 1 -
tests/fail-float2.jl | 5 ----
tests/{test-float3.jl => test-double3.jl} | 10 +++++----
tests/{test-float3.out => test-double3.out} | 0

24 files changed, 97 insertions(+), 85 deletions(-)

commit fde4100ebc3b2cd081c2bd415ebabc681915744d
Author: HongfeiChenCU <hc3222@columbia.edu>
Date: Sun Mar 21 20:47:57 2021 -0700
renamed all files for javalite

<table>
<thead>
<tr>
<th>File</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Makefile</td>
<td>20 ++++++-</td>
</tr>
<tr>
<td>codegen.ml</td>
<td>2 +--</td>
</tr>
<tr>
<td>microc.ml =&gt; javalite.ml</td>
<td>6 ++++-</td>
</tr>
<tr>
<td>microparse.mly =&gt; parser.mly</td>
<td>4 ++-</td>
</tr>
<tr>
<td>scanner.ml</td>
<td>2 +--</td>
</tr>
<tr>
<td>testall.sh</td>
<td>26 ++++++-</td>
</tr>
<tr>
<td>tests/{fail-assign1.mc =&gt; fail-assign1.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-assign2.mc =&gt; fail-assign2.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-assign3.mc =&gt; fail-assign3.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-dead1.mc =&gt; fail-dead1.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-dead2.mc =&gt; fail-dead2.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-expr1.mc =&gt; fail-expr1.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-expr2.mc =&gt; fail-expr2.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-expr3.mc =&gt; fail-expr3.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-float1.mc =&gt; fail-float1.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-float2.mc =&gt; fail-float2.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-for1.mc =&gt; fail-for1.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-for2.mc =&gt; fail-for2.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-for3.mc =&gt; fail-for3.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-for4.mc =&gt; fail-for4.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-for5.mc =&gt; fail-for5.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-func1.mc =&gt; fail-func1.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-func2.mc =&gt; fail-func2.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-func3.mc =&gt; fail-func3.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-func4.mc =&gt; fail-func4.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-func5.mc =&gt; fail-func5.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-func6.mc =&gt; fail-func6.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-func7.mc =&gt; fail-func7.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-func8.mc =&gt; fail-func8.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-func9.mc =&gt; fail-func9.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-global1.mc =&gt; fail-global1.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-global2.mc =&gt; fail-global2.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-if1.mc =&gt; fail-if1.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-if2.mc =&gt; fail-if2.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-if3.mc =&gt; fail-if3.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-nomain.mc =&gt; fail-nomain.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-print.mc =&gt; fail-print.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-printb.mc =&gt; fail-printb.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-printbig.mc =&gt; fail-printbig.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-return1.mc =&gt; fail-return1.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-return2.mc =&gt; fail-return2.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-while1.mc =&gt; fail-while1.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{fail-while2.mc =&gt; fail-while2.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{test-add1.mc =&gt; test-add1.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{test-arithmetic1.mc =&gt; test-arithmetic1.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{test-arithmetic2.mc =&gt; test-arithmetic2.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{test-arithmetic3.mc =&gt; test-arithmetic3.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{test-fib.mc =&gt; test-fib.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{test-float1.mc =&gt; test-float1.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{test-float2.mc =&gt; test-float2.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{test-float3.mc =&gt; test-float3.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{test-for1.mc =&gt; test-for1.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{test-for2.mc =&gt; test-for2.jl}</td>
<td>0</td>
</tr>
<tr>
<td>tests/{test-func1.mc =&gt; test-func1.jl}</td>
<td>0</td>
</tr>
</tbody>
</table>
tests/{test-func2.mc => test-func2.jl} | 0
2170 tests/{test-func3.mc => test-func3.jl} | 0
2171 tests/{test-func4.mc => test-func4.jl} | 0
2172 tests/{test-func5.mc => test-func5.jl} | 0
2173 tests/{test-func6.mc => test-func6.jl} | 0
2174 tests/{test-func7.mc => test-func7.jl} | 0
2175 tests/{test-func8.mc => test-func8.jl} | 0
2176 tests/{test-func9.mc => test-func9.jl} | 0
2177 tests/{test-gcd.mc => test-gcd.jl} | 0
2178 tests/{test-gcd2.mc => test-gcd2.jl} | 0
2179 tests/{test-global1.mc => test-global1.jl} | 0
2180 tests/{test-global2.mc => test-global2.jl} | 0
2181 tests/{test-global3.mc => test-global3.jl} | 0
2182 tests/{test-hello.mc => test-hello.jl} | 0
2183 tests/{test-helloworld.mc => test-helloworld.jl} | 0
2184 tests/{test-if1.mc => test-if1.jl} | 0
2185 tests/{test-if2.mc => test-if2.jl} | 0
2186 tests/{test-if3.mc => test-if3.jl} | 0
2187 tests/{test-if4.mc => test-if4.jl} | 0
2188 tests/{test-if5.mc => test-if5.jl} | 0
2189 tests/{test-if6.mc => test-if6.jl} | 0
2190 tests/{test-local1.mc => test-local1.jl} | 0
2191 tests/{test-local2.mc => test-local2.jl} | 0
2192 tests/{test-ops1.mc => test-ops1.jl} | 0
2193 tests/{test-ops2.mc => test-ops2.jl} | 0
2194 tests/{test-printbig.mc => test-printbig.jl} | 0
2195 tests/{test-var1.mc => test-var1.jl} | 0
2196 tests/{test-var2.mc => test-var2.jl} | 0
2197 tests/{test-while1.mc => test-while1.jl} | 0
2198 tests/{test-while2.mc => test-while2.jl} | 0
2199 84 files changed, 30 insertions(+), 30 deletions(-)
2200 commit cd9a779b7d1be4d0d82f0d2a36319ac9a073aa86
2201 Author: Frances Cao <frances.te.cao@gmail.com>
2202 Date: Sun Mar 21 14:08:34 2021 -0400
2203 add microc template, hello-world deliverable
2204 Dockerfile | 32 ++++++
2205 Makefile | 61 +++++++++
2206 README | 3 +
2207 _tags | 9 ++
2208 arcade-font.pbm | Bin 0 -> 344 bytes
2209 ast.ml | 109 +++++++++++++++++++++++++++++++
2210 codegen.ml | 251 ++++++++++++++++++++++++++++++++++++++
2211 font2c | 9 ++
2212 microc.ml | 32 +++++
2213 microcparse.mly | 117 +++++++++++++++++++++
2214 printbig.c | 75 +++++++++++
2215 printbig.o | Bin 0 -> 2080 bytes
2216 sast.ml | 79 +++++++++++++
2217 scanner.mll | 59 +++++++++
2218 semant.ml | 190 ++++++++++++++++++++++++++++++++++++++
2219 testall.sh | 198 ++++++++++++++++++++++++++++++++
<table>
<thead>
<tr>
<th>Line</th>
<th>File</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>2226</td>
<td>tests/fail-assign2.mc</td>
<td>7 ++</td>
</tr>
<tr>
<td>2227</td>
<td>tests/fail-assign3.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2228</td>
<td>tests/fail-assign3.mc</td>
<td>11 ++</td>
</tr>
<tr>
<td>2229</td>
<td>tests/fail-dead1.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2230</td>
<td>tests/fail-dead1.mc</td>
<td>8 ++</td>
</tr>
<tr>
<td>2231</td>
<td>tests/fail-dead2.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2232</td>
<td>tests/fail-dead2.mc</td>
<td>10 ++</td>
</tr>
<tr>
<td>2233</td>
<td>tests/fail-expr1.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2234</td>
<td>tests/fail-expr1.mc</td>
<td>18 ++++</td>
</tr>
<tr>
<td>2235</td>
<td>tests/fail-expr2.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2236</td>
<td>tests/fail-expr2.mc</td>
<td>14 ++++</td>
</tr>
<tr>
<td>2237</td>
<td>tests/fail-expr3.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2238</td>
<td>tests/fail-expr3.mc</td>
<td>14 ++++</td>
</tr>
<tr>
<td>2239</td>
<td>tests/fail-float1.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2240</td>
<td>tests/fail-float1.mc</td>
<td>5 +</td>
</tr>
<tr>
<td>2241</td>
<td>tests/fail-float2.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2242</td>
<td>tests/fail-float2.mc</td>
<td>5 +</td>
</tr>
<tr>
<td>2243</td>
<td>tests/fail-for1.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2244</td>
<td>tests/fail-for1.mc</td>
<td>13 ++++</td>
</tr>
<tr>
<td>2245</td>
<td>tests/fail-for2.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2246</td>
<td>tests/fail-for2.mc</td>
<td>8 ++</td>
</tr>
<tr>
<td>2247</td>
<td>tests/fail-for3.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2248</td>
<td>tests/fail-for3.mc</td>
<td>8 ++</td>
</tr>
<tr>
<td>2249</td>
<td>tests/fail-for4.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2250</td>
<td>tests/fail-for4.mc</td>
<td>8 ++</td>
</tr>
<tr>
<td>2251</td>
<td>tests/fail-for5.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2252</td>
<td>tests/fail-for5.mc</td>
<td>10 ++</td>
</tr>
<tr>
<td>2253</td>
<td>tests/fail-func1.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2254</td>
<td>tests/fail-func1.mc</td>
<td>12 ++++</td>
</tr>
<tr>
<td>2255</td>
<td>tests/fail-func2.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2256</td>
<td>tests/fail-func2.mc</td>
<td>8 ++</td>
</tr>
<tr>
<td>2257</td>
<td>tests/fail-func3.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2258</td>
<td>tests/fail-func3.mc</td>
<td>8 ++</td>
</tr>
<tr>
<td>2259</td>
<td>tests/fail-func4.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2260</td>
<td>tests/fail-func4.mc</td>
<td>12 ++++</td>
</tr>
<tr>
<td>2261</td>
<td>tests/fail-func5.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2262</td>
<td>tests/fail-func5.mc</td>
<td>14 ++++</td>
</tr>
<tr>
<td>2263</td>
<td>tests/fail-func6.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2264</td>
<td>tests/fail-func6.mc</td>
<td>9 ++</td>
</tr>
<tr>
<td>2265</td>
<td>tests/fail-func7.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2266</td>
<td>tests/fail-func7.mc</td>
<td>9 ++</td>
</tr>
<tr>
<td>2267</td>
<td>tests/fail-func8.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2268</td>
<td>tests/fail-func8.mc</td>
<td>13 ++++</td>
</tr>
<tr>
<td>2269</td>
<td>tests/fail-func9.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2270</td>
<td>tests/fail-func9.mc</td>
<td>9 ++</td>
</tr>
<tr>
<td>2271</td>
<td>tests/fail-global1.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2272</td>
<td>tests/fail-global1.mc</td>
<td>9 ++</td>
</tr>
<tr>
<td>2273</td>
<td>tests/fail-global2.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2274</td>
<td>tests/fail-global2.mc</td>
<td>9 ++</td>
</tr>
<tr>
<td>2275</td>
<td>tests/fail-if1.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2276</td>
<td>tests/fail-if1.mc</td>
<td>6 ++</td>
</tr>
<tr>
<td>2277</td>
<td>tests/fail-if2.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2278</td>
<td>tests/fail-if2.mc</td>
<td>6 ++</td>
</tr>
<tr>
<td>2279</td>
<td>tests/fail-if3.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2280</td>
<td>tests/fail-if3.mc</td>
<td>8 ++</td>
</tr>
<tr>
<td>2281</td>
<td>tests/fail-nomain.err</td>
<td>1 +</td>
</tr>
<tr>
<td>2282</td>
<td>tests/fail-nomain.mc</td>
<td>0</td>
</tr>
<tr>
<td>File</td>
<td>Status</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>tests/fail-print.err</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/fail-print.mc</td>
<td>2 +</td>
<td></td>
</tr>
<tr>
<td>tests/fail-printb.err</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/fail-printb.mc</td>
<td>2 +</td>
<td></td>
</tr>
<tr>
<td>tests/fail-printbig.err</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/fail-printbig.mc</td>
<td>2 +</td>
<td></td>
</tr>
<tr>
<td>tests/fail-return1.err</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/fail-return1.mc</td>
<td>4 +</td>
<td></td>
</tr>
<tr>
<td>tests/fail-return2.err</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/fail-return2.mc</td>
<td>10 ++</td>
<td></td>
</tr>
<tr>
<td>tests/fail-while1.err</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/fail-while1.mc</td>
<td>13 +++</td>
<td></td>
</tr>
<tr>
<td>tests/fail-while2.err</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/fail-while2.mc</td>
<td>13 +++</td>
<td></td>
</tr>
<tr>
<td>tests/test-add1.mc</td>
<td>10 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-add1.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-arith1.mc</td>
<td>5 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-arith1.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-arith2.mc</td>
<td>5 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-arith2.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-arith3.mc</td>
<td>13 +++</td>
<td></td>
</tr>
<tr>
<td>tests/test-arith3.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-fib.mc</td>
<td>16 +++</td>
<td></td>
</tr>
<tr>
<td>tests/test-fib.out</td>
<td>6 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-float1.mc</td>
<td>7 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-float1.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-float2.mc</td>
<td>11 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-float2.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-float3.mc</td>
<td>30 +++++</td>
<td></td>
</tr>
<tr>
<td>tests/test-float3.out</td>
<td>24 +++++</td>
<td></td>
</tr>
<tr>
<td>tests/test-for1.mc</td>
<td>9 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-for1.out</td>
<td>6 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-for2.mc</td>
<td>11 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-for2.out</td>
<td>6 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-func1.mc</td>
<td>12 +++</td>
<td></td>
</tr>
<tr>
<td>tests/test-func1.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-func2.mc</td>
<td>18 +++++</td>
<td></td>
</tr>
<tr>
<td>tests/test-func2.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-func3.mc</td>
<td>13 +++</td>
<td></td>
</tr>
<tr>
<td>tests/test-func3.out</td>
<td>4 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-func4.mc</td>
<td>14 +++</td>
<td></td>
</tr>
<tr>
<td>tests/test-func4.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-func5.mc</td>
<td>9 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-func5.out</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>tests/test-func6.mc</td>
<td>9 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-func6.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-func7.mc</td>
<td>13 +++</td>
<td></td>
</tr>
<tr>
<td>tests/test-func7.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-func8.mc</td>
<td>10 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-func8.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-func9.mc</td>
<td>11 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-func9.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-gcd.mc</td>
<td>15 +++</td>
<td></td>
</tr>
<tr>
<td>tests/test-gcd.out</td>
<td>3 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-gcd2.mc</td>
<td>14 +++</td>
<td></td>
</tr>
<tr>
<td>tests/test-gcd2.out</td>
<td>3 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-global1.mc</td>
<td>30 +++++</td>
<td></td>
</tr>
<tr>
<td>File</td>
<td>Changes</td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>tests/test-global1.out</td>
<td>4 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-global2.mc</td>
<td>10 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-global2.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-global3.mc</td>
<td>11 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-global3.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-hello.mc</td>
<td>7 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-hello.out</td>
<td>3 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-helloworld.mc</td>
<td>4 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-helloworld.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-if1.mc</td>
<td>6 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-if1.out</td>
<td>2 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-if2.mc</td>
<td>6 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-if2.out</td>
<td>2 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-if3.mc</td>
<td>6 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-if3.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-if4.mc</td>
<td>6 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-if4.out</td>
<td>2 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-if5.mc</td>
<td>16 ++++</td>
<td></td>
</tr>
<tr>
<td>tests/test-if5.out</td>
<td>2 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-if6.mc</td>
<td>18 ++++</td>
<td></td>
</tr>
<tr>
<td>tests/test-if6.out</td>
<td>2 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-local1.mc</td>
<td>13 +++</td>
<td></td>
</tr>
<tr>
<td>tests/test-local1.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-local2.mc</td>
<td>14 +++</td>
<td></td>
</tr>
<tr>
<td>tests/test-local2.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-ops1.mc</td>
<td>28 ++++</td>
<td></td>
</tr>
<tr>
<td>tests/test-ops1.out</td>
<td>24 ++++</td>
<td></td>
</tr>
<tr>
<td>tests/test-ops2.mc</td>
<td>17 +++</td>
<td></td>
</tr>
<tr>
<td>tests/test-ops2.out</td>
<td>14 +++</td>
<td></td>
</tr>
<tr>
<td>tests/test-printbig.mc</td>
<td>25 ++++</td>
<td></td>
</tr>
<tr>
<td>tests/test-printbig.out</td>
<td>88 +++++</td>
<td></td>
</tr>
<tr>
<td>tests/test-var1.mc</td>
<td>7 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-var1.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-var2.mc</td>
<td>13 +++</td>
<td></td>
</tr>
<tr>
<td>tests/test-var2.out</td>
<td>1 +</td>
<td></td>
</tr>
<tr>
<td>tests/test-while1.mc</td>
<td>11 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-while1.out</td>
<td>6 ++</td>
<td></td>
</tr>
<tr>
<td>tests/test-while2.mc</td>
<td>16 +++</td>
<td></td>
</tr>
<tr>
<td>tests/test-while2.out</td>
<td>1 +</td>
<td></td>
</tr>
</tbody>
</table>

172 files changed, 2331 insertions(+)