

# **bawk: a versatile text-processing language**

---

Ashley An, Christine Hsu, Melanie Sawyer, and Victoria Yang

# Contents

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Introduction</b>                            | <b>1</b>  |
| 1.1      | Background . . . . .                           | 1         |
| <b>2</b> | <b>Language Tutorial</b>                       | <b>1</b>  |
| 2.1      | Writing a simple bawk program . . . . .        | 1         |
| 2.2      | Compiling and executing a .bawk file . . . . . | 2         |
| 2.3      | Simple File Reading . . . . .                  | 3         |
| <b>3</b> | <b>Language Reference Manual</b>               | <b>3</b>  |
| 3.1      | Lexical Conventions . . . . .                  | 3         |
| 3.2      | Comments . . . . .                             | 4         |
| 3.3      | Types . . . . .                                | 4         |
| 3.4      | Special Variables . . . . .                    | 5         |
| 3.5      | Keywords . . . . .                             | 5         |
| 3.6      | Operators . . . . .                            | 6         |
| 3.7      | Structure of Program . . . . .                 | 9         |
| 3.8      | Functions . . . . .                            | 10        |
| 3.9      | Control Flow . . . . .                         | 10        |
| 3.10     | Standard Library Features . . . . .            | 11        |
| <b>4</b> | <b>Project Plan</b>                            | <b>13</b> |
| 4.1      | Team Roles & Responsibilities . . . . .        | 13        |
| 4.2      | Project Log . . . . .                          | 14        |

|          |                                      |            |
|----------|--------------------------------------|------------|
| 4.3      | Development Process . . . . .        | 99         |
| 4.4      | Software Development Tools . . . . . | 99         |
| 4.5      | Programming Style Guide . . . . .    | 99         |
| 4.6      | Timeline . . . . .                   | 100        |
| 4.7      | Language Evolution . . . . .         | 100        |
| 4.8      | Testing Process . . . . .            | 101        |
| <b>5</b> | <b>System Architecture</b>           | <b>101</b> |
| 5.1      | Scanner . . . . .                    | 102        |
| 5.2      | Parser . . . . .                     | 102        |
| 5.3      | Semantic Checker . . . . .           | 102        |
| 5.4      | Code Generation . . . . .            | 103        |
| 5.5      | C Libraries . . . . .                | 103        |
| <b>6</b> | <b>Test Plan</b>                     | <b>106</b> |
| 6.1      | Test Suite . . . . .                 | 106        |
| 6.2      | Automation . . . . .                 | 106        |
| 6.3      | Examples and .11 output . . . . .    | 106        |
| <b>7</b> | <b>Lessons Learned</b>               | <b>112</b> |
| 7.1      | Ashley . . . . .                     | 112        |
| 7.2      | Christine . . . . .                  | 113        |
| 7.3      | Melanie . . . . .                    | 113        |
| 7.4      | Victoria . . . . .                   | 114        |



# 1 Introduction

bawk is a versatile text processing language loosely inspired by awk, building upon it with imperative constructs such as infinitely nested mutable arrays and control flow structures. bawk streamlines the process of file reading and repeatedly executing operations over lines of a file. It syntactically resembles C and other similar imperative programming languages, but features awk-like features including `BEGIN`, `LOOP`, and `END` blocks, powerful string and regex matching, and intuitive file manipulation.

## 1.1 Background

Although text processing languages are relatively domain-specific, they hold a wide range of use cases. awk is a widely used, data-driven text processing language that primarily performs operations on text files. Its main function is to search through files for lines that contain specified patterns and perform operations once those lines are found.

While awk is designed to be terse and is adept at one-line programs, bawk adopts a more verbose, C-like syntax. bawk aims to make it easy to read, analyze, and write to text files in a simple and intuitive syntax mimicking that of awk.

## 2 Language Tutorial

This section will provide details about compiling and running a simple `bawk` program, and provides some examples of useful language constructs. Full documentation for our language is included in the following section.

### 2.1 Writing a simple bawk program

There are four parts to a bawk program, all separated into `blocks`:

- `BEGIN`:  
The begin block is executed before the file is read, and is the only block in which functions and global variables can be defined.
- `LOOP`:  
The loop block automatically loops through all the lines in the input file, and executes once for each line of the file. If the input file is empty, the `LOOP` block will not execute. The loop block can use functions and global variables defined in begin, but can only define local variables.

- **END:**  
The end block is something of a post-processor – it is for instructions that should be executed after the input file has been read. It too can utilize functions and global vars defined in begin; it can also define its own local variables.
- **CONFIG (optional):**  
The config block allows the programmer to optionally customize how the file will be read. It allows for the modification of 2 configuration variables: RS and FS. RS (record separator) defines how lines in a file will be separated. The default is a newline (””) symbol. Some files may have different line separators (i.e. Windows machines will often generate new lines with a carriage return ”” and a traditional new line symbol ””). The FS (field separator) variable defines how fields in a line are separated. Since there are a multitude of ways that files can be split into fields (TSV files, CSV files, etc.) its useful to be able to change this value to allow for adaptable file reading. The default FS value is a space ( ).

To write a simple hello world program, we’ll first need to write out the blocks. Blocks are defined with their name and open/closed curly brackets. The body of the block is placed inside the brackets.

```
1 BEGIN {}
2 LOOP {}
3 END {}
```

This in itself is a valid bawk program– it will simply output nothing. However, let’s add a print statement somewhere to ensure that everything works as expected.

As mentioned above, the *only* instructions that can be placed in the **BEGIN** block are globals and function definitions, so our print statement will have to be placed in either **LOOP** or **END**.

```
1 BEGIN {}
2 LOOP {
3     print("This is the loop block");
4 }
5 END {
6     print("This is the end block");
7 }
```

When this file is executed, it should print out "This is the loop block" for every line of the input file, and "This is the end block" once at the end.

## 2.2 Compiling and executing a .bawk file

Now that we’ve written a simple bawk program, let’s run it through the bawk compiler. Navigate to the **bawk** folder and run `make all`. This will build the source compiler. Then, to execute a simple bawk program, run

```
$ ./bawk.sh <program.bawk> <input.txt>
```

where `program.bawk` is the program file and `input.txt` is the input file (it doesn't have a `.txt` extension) If the build and compilation was successful, this should output something like

```
This is the loop block
This is the loop block
This is the loop block
This is the end block
```

## 2.3 Simple File Reading

The simplest and most important construct in bawk is to loop through a file. A simple program to loop through a file and print each line is the following:

```
1 BEGIN {}
2 LOOP {
3     print($0);
4 }
5 END {}
```

Execute the program similarly as above.

```
$ ./bawk.sh <program.bawk> <input.txt>
```

This program will output each line of the file.

Note that `$0` indicates that the entire line is being printed. Using `$n` will print the `n`th field of a line, so `$2` will print the second word in a line.

The language manual will cover in depth all syntactic constructs not discussed here.

# 3 Language Reference Manual

## 3.1 Lexical Conventions

Identifiers are primarily used for variable declaration or function declaration, and can be any combination of numbers, letters, and the `_` symbol as long as the resulting string is not an existing keyword.

| Valid:      | Invalid:    |
|-------------|-------------|
| myVariable  | 1variable   |
| my_variable | function    |
| myVariable1 | my.variable |
| -           |             |

Figure 1: Examples of valid and invalid identifiers

### 3.2 Comments

A single-line comment is preceded by `.` There are no multi-line comment symbols.

```
# hi i am a comment
```

### 3.3 Types

bawk is a strongly typed language. It has 5 types: booleans, integers, strings, regex, and arrays.

#### Booleans

A boolean is either true or false. Its type is denoted by the keyword `bool`:

```
1 bool b;
2 b = true;
```

#### Integers

An integer is defined as a sequence of one or more digits representing an integer, with the leading digit being non-zero (i.e.  $[1-9][0-9]^*$ ). Integer types are denoted with the keyword `int`:

```
1 int x;
2 x = 123;
```

#### Strings

String literals are defined as a sequence of ASCII characters enclosed by a pair of double quotation marks. String types are denoted with the keyword `string`:

```
1 string s;
2 s = "abc!!";
```

#### Regex

Regex (regular expression) literals are denoted with a `rgx` type and enclosed in a pair of single quotation marks. Regex types are denoted by the keyword `rgx`:

```
1 rgx example;
2 example = 'p.n';
```



## Arrays

Arrays must be initialized by their type and the number of dimensions. Each array must have elements of all the same type, even when nested. Arrays are dynamically-sized. There are int, bool, string, and rgx arrays.

```
1 int [][] arr;      # 2-dimensional array with type int
2 arr = [[1, 2, 3],[1, 2, 3]];
```

## 3.4 Special Variables

### RS

RS is a built-in string variable for the "record separator". The default value is a newline character, but this can be changed in the CONFIG block, e.g., RS = "\r\n"

### FS

FS is a built-in string variable for the field separator. The default value is a space, but this can be changed in the CONFIG block. Indicating that fields within a record should be separated by commas is FS= ",";

### NF

NF is a built-in special variable that retrieves the number of fields in the current record that is being read in LOOP. This variable is useful when iterating over the fields in a line.

## 3.5 Keywords

### function

**function** denotes how the user starts a function declaration; an explicit return type is required. Nested functions are not allowed.

```
1 function int add(int x, int y) {
2     return x + y;
3 }
```

### return

**return** returns the data type at the end of a function. If there is code after a return statement the parser will throw an error that halts execution.

```
1 function int add(int x, int y) {
2     return x + y;
3 }
```

### true

**true** is a boolean keyword that means something is true.

```
1 bool flag = true;
2 if (flag){
3     print("hello");
4 }
```

### **false**

**false** is a boolean keyword that means something is false.

```
1 bool flag = true;
2 int x = 6;
3 if (x>5){
4     flag = false;
5 }
6 if (!flag){
7     print ("x is greater than 5");
```

## **3.6 Operators**

### **\$ (Access)**

**\$n** retrieves the nth string field in the current record.

```
1 $0 # refers to the whole record
2 $1 # first field in current record
3 $2 # second field in current record
```

If n is greater than the number of fields in the line (NF), the access operator will return an empty string to allow continuous execution of the program.

**!**

**!** is the logical not operator.

### **& (Concatenation)**

**&** is used for string concatenation, e.g.

```
1 string s;
2 s = "hello" & "world."
```

**&&**

**&&** represents logical AND, e.g.

```
1 int i;
2 i = 5;
3 if (i == 5 && i < 6){
4     print (int_to_string(i));
```

```
5 }
```

This will output 5.

||

|| represents logical OR, e.g.

```
1 int i;  
2 i = 5;  
3 if (i < 6 || i > 10){  
4     print (int_to_string(i));  
5 }
```

This will output 5.

-

- is both a binary operator used for the subtraction of integers and a unary operator used to represent negative numbers.

+

+ is used for the addition of integers.

\*

\* is used for the multiplication of integers.

/

/ is used for the division of integers.

++

++ is used for the incrementation of integers by 1, e.g. x++.

--

-- is used for the decrementation of integers by 1, e.g. x--.

+=

+= is the incrementation and assignment of integers by a specified integer, e.g. x += 5

-=

-= is the decrementation and assignment of integers by a specified integer,

e.g. `x -= 5`

`[]`

`[]` is used to read from or write to arrays at a certain index, e.g.:

```
1 if (array[0] == 5){
2     return array[0];
3 }
```

`<=`

`<=` is used for string or integer comparison and returns a boolean that indicates if x less than or equal to y.

`<`

`<` is used for string or integer comparison and returns a boolean that indicates if x less than y.

`>`

`>` is used for string or integer comparison and returns a boolean that indicates if x greater than y.

`>=`

`>=` is used for string or integer comparison and returns a boolean that indicates if x greater than or equal to y.

`==`

`==` is used for string, integer, or boolean comparison and returns a boolean that indicates if x is equal to y.

`!=`

`!=` is used for string, integer, or boolean comparison and returns a boolean that indicates if x not equal to y.

**~: Regex matching**

`~`, when used as `x ~ y`, returns a boolean that indicates if string x matches the rgx denoted by y.

`!~`

`!~` , when used as `x!~ y`, returns a boolean that indicates if string `x` does not match the rgx denoted by `y`.

`%`  
`%`, when used as `x%y`, returns a boolean that indicates if rgx `x` equals the rgx denoted by `y`.

`!%`  
`!%`, when used as `x!%y`, returns a boolean that indicates if rgx `x` equals the rgx denoted by `y`.

### 3.7 Structure of Program

`bawk` programs are divided into four blocks: `BEGIN`, `LOOP`, `END`, and `CONFIG`:

```
1 BEGIN {...}
2 LOOP {...}
3 END {...}
4 CONFIG {...}
```

`BEGIN`, `LOOP`, and `END` blocks must be present in a `.bawk` program in that order, although they can be empty. The `BEGIN` block allows global variable declarations and function declarations, both of which are visible to the scope of the entire program. Function declarations can be done in the `BEGIN` block only. Functions can be called from within the declaration of a function if they have been defined already. The statements in a `LOOP` block are executed for each line of an inputted file. The `LOOP` and `END` blocks can contain variable declarations and statements, but no function declarations. They can use any variables declared in `BEGIN`. Variables declared in `LOOP` and `END` blocks exist within the scope of their respective blocks. Local variables declared in the `LOOP` and `END` blocks may have the same names as global variables, but they will override the globals within the scope of the block.

Variable declarations must all occur together at the beginning of a scope.

The `CONFIG` block is optional, but must be the last block of the program if it is there. The `CONFIG` blocks purpose is to set the record separator `RS` and the field separator `FS`, and can only take expressions that set `RS` and `FS`.

### 3.8 Functions

User-defined functions are allowed. Function declarations must be written inside the `BEGIN` block. They must have an explicit return type (which can be void). A function declaration looks like this:

```
1 function int add(int x, int y) {
2     return x + y;
3 }
```

Function declarations cannot be nested. Functions can be called from within a function declaration as long as the called function is already defined.

### 3.9 Control Flow

`bawk` has `if` statements, `for` loops, and `while` loops.

#### **if**

`if...else` denotes a conditional statement, e.g.:

```
1 int x = 0;
2 if (x < 10){
3     x++;
4 }
5 else{
6     print(int_to_string(x));
7 }
```

#### **for**

A `for` loop iterates through a block of code a number of times specified by a incrementation variable. The parentheses must contain 3 statements delimited by semicolons. The first segment contains the initialization of a counter variable, the second is the condition under which the `for` loop will continue

looping, and the third is what happens to the counter variable after each iteration.

```
1 int i;
2 for (i = 0; i < 10; i++) {
3     print(int_to_string(x));
4 }
```

## while

A while loop iterates through a block of code until its corresponding conditional expression is not met.

```
1 int x;
2 x = 0;
3 while (x < 10) {
4     print(int_to_string(x));
5     x++;
6 }
```

### 3.10 Standard Library Features

`int string_to_int(string a)`

This function converts a string into an int provided that the string follows the regex `[0-9]*`. The `string_to_int()` function is especially useful for parsing integers from the input file and performing arithmetic operations on them.

`string int_to_string(int a)`

This function converts an int into a string. The `int_to_string()` function is especially useful when printing out integers, as the `print` function only takes strings as parameters.

`string rgx_to_string(rgx a)`

This function converts a rgx into a string. As above, this function is useful when printing out the values of regex expressions.

`string bool_to_string(bool a)`

This function converts a boolean into a string. Also useful for printing out the value of a boolean.

**int length (arr[] a)**

This function returns the length of the array a.

**bool contains (arr[] a, val b)**

This function is used to check if b is present in the array, given that b is the same type as the values contained in a. This can also be used to determine whether an array index has been assigned or not. Returns **true** if item in array, **false** otherwise, e.g.:

```
1 if (!contains(array, $a)) {
2     array[length(array) - 1] = $0;
3 }
```

**int index\_of(arr[] a, val b)**

This function returns the index of the first instance of b within the array a by value, given that b is the same type as the values in a. If b is not in a, return -1.

```
1 arr[] a;
2 a = {1, 2, 3, 2};
3 contains(a, 2); # returns 0
4 contains(a, 0); # returns -1
```

**void insert(arr[] a, int b, val c)**

This function inserts c at a[b], given that b is between 0 and length(a), else it throws an error.

```
1 arr[] a;
2 a = {1, 2, 3, 2};
3 insert(a, 0, 0); # a = {0, 2, 3, 2};
4 insert(a, 7, 0); # throws error
```

**val delete(arr[] a, int b)**

This function deletes a[b], given that b is between 0 and length(a) - 1, else it throws an error. It returns the value previously at a[b] if it is a valid deletion.

```
1 arr[] a;
2 a = {1, 2, 3, 2};
3 delete(a, 0); # a = {2, 3, 2};
4 delete(a, 7); # throws error
```



### **void print(string a)**

This function prints a string input. In order to print other data types, they first need to be converted to strings using built-in functions. A new line is automatically printed at the end of the string.

```
1 print("hello");
2 print("world");
3 # Outputs:
4 # hello
5 # world
6
7 print(int_to_string(2));
8 # Outputs:
9 # 2
```

### **void nprint(string a)**

This function prints a string input without a new line at the end. To print other data types, they must be converted to strings using type conversion built-in functions.

```
1 nprint("hello ");
2 nprint("world");
3 # Output:
4 # hello world
```

## **4 Project Plan**

### **4.1 Team Roles & Responsibilities**

Each team member made significant contributions to the parts of the language listed below, but all members were in constant communication about every aspect of the language and each member worked on most parts of the compiler.

| Team Member    | Responsibilities  |
|----------------|---|
| Ashley An      | Compiler Front End, C Libraries, Code Generation, Testing |
| Christine Hsu  | Compiler Front End, Code Generation                       |
| Melanie Sawyer | Compiler Front End, C Libraries, Code Generation          |
| Victoria Yang  | Semantic Checking, Code Generation                        |

## 4.2 Project Log

```
1 commit f4bd2c0b3c3ba3bbca03696f871516f6d59eac1a
2 Author: Melanie Sawyer <melaniensawyer@gmail.com>
3 Date: Tue Dec 18 21:10:47 2018 -0500
4
5 Update README.md
6
7 commit 7ca7038084aea88554a425a2c668f19aa0aa71b9
8 Author: Ashley An <ashley.an@columbia.edu>
9 Date: Tue Dec 18 15:03:42 2018 -0500
10
11 edited demo and input.txt
12
13 commit 58eb8b92e1db8c3c92df16b7efcf11c958dad78
14 Author: Ashley An <ashley.an@columbia.edu>
15 Date: Tue Dec 18 14:46:54 2018 -0500
16
17 added matrix multiplication
18
19 commit 50ec7015d015b2abc819151d884124c5fdb41e84
20 Author: Ashley An <ashley.an@columbia.edu>
21 Date: Mon Dec 17 19:20:37 2018 -0500
22
23 created demo folder
24
25 commit d1f040521497701c46a0f3ad6295296ac4185d57
26 Author: Melanie Sawyer <melaniensawyer@gmail.com>
27 Date: Tue Dec 18 13:46:11 2018 -0500
28
29 add shuffled file
30
31 commit a5e45a196928a3b1c96fc158ad13da3a4f2fd4b9
32 Author: Melanie Sawyer <melaniensawyer@gmail.com>
33 Date: Tue Dec 18 13:43:17 2018 -0500
34
35 add finalized demo
36
37 commit 35899684fbffe3bb785841162315aff2a6fa2b12
38 Author: Christine Hsu <christine.hhsu@gmail.com>
39 Date: Tue Dec 18 01:11:11 2018 -0500
40
41 add array init to demo
42
43 commit 04248baa9f3290951e04860bc127defc01fd426c
44 Author: Christine Hsu <christine.hhsu@gmail.com>
45 Date: Tue Dec 18 00:56:59 2018 -0500
46
47 working on demo
48
49 commit 463408b5e171ded0de0bc1a6bd25d1f44bd5bf4a
50 Author: Christine Hsu <christine.hhsu@gmail.com>
51 Date: Mon Dec 17 19:09:55 2018 -0500
52
53 add shuffle and demo.bawk
54
```

```
55 | commit d4bf4f0c937d7d9f1888eb1a10aa9071714b2d05
56 | Merge: 24f638a e39ba97
57 | Author: Melanie Sawyer <melaniensawyer@gmail.com>
58 | Date: Mon Dec 17 19:00:35 2018 -0500
59 |
60 | Merge branch 'master' of https://github.com/soybean/PLT-f18
61 |
62 | commit 24f638a7ce46362ddf0fd4a0c7e2045296598731
63 | Author: Melanie Sawyer <melaniensawyer@gmail.com>
64 | Date: Mon Dec 17 19:00:06 2018 -0500
65 |
66 | add file
67 |
68 | commit e39ba97cc337812a4b756e1620e1b9be16c2b796
69 | Author: Ashley An <ashley.an@columbia.edu>
70 | Date: Mon Dec 17 18:40:54 2018 -0500
71 |
72 | fixed dollar and RS/FS tests
73 |
74 | commit 95acdb938ed8e338171c6f77b585455d63498924
75 | Author: Melanie Sawyer <melaniensawyer@gmail.com>
76 | Date: Mon Dec 17 18:37:06 2018 -0500
77 |
78 | fix $0
79 |
80 | commit 64aa638b0ef8d7131f7b7abad99e7ba8f259b874
81 | Merge: 6d901cd 5abf55e
82 | Author: Melanie Sawyer <melaniensawyer@gmail.com>
83 | Date: Mon Dec 17 18:29:31 2018 -0500
84 |
85 | Merge branch 'master' of https://github.com/soybean/PLT-f18
86 |
87 | commit 6d901cdd1d96c2db88ccf4f3ac8acc0e4d4d6956
88 | Author: Melanie Sawyer <melaniensawyer@gmail.com>
89 | Date: Mon Dec 17 18:29:06 2018 -0500
90 |
91 | fix dollar
92 |
93 | commit 5abf55e68643f46baccdda0eb7204e3afd2528b8
94 | Author: Christine Hsu <christine.hhsu@gmail.com>
95 | Date: Mon Dec 17 18:26:25 2018 -0500
96 |
97 | fix I00BE
98 |
99 | commit c64929f0b38f49070138d76c3217cf3f51e33392
100 | Merge: 604987c 476fcc3
101 | Author: Melanie Sawyer <melaniensawyer@gmail.com>
102 | Date: Mon Dec 17 17:48:22 2018 -0500
103 |
104 | Merge branch 'master' of https://github.com/soybean/PLT-f18
105 |
106 | commit 604987cace35ca14951cd255e5b25e4c6d4cf271
107 | Author: Melanie Sawyer <melaniensawyer@gmail.com>
108 | Date: Mon Dec 17 17:47:46 2018 -0500
109 |
110 | fix $
```

```
111
112 commit 476fcc35116efd48c6256ced67781f45d9ca86c5
113 Author: Ashley An <ashley.an@columbia.edu>
114 Date: Mon Dec 17 17:32:59 2018 -0500
115
116     fixed formatting of test
117
118 commit 921d0913bb3ec5a638ba73c10daad5bc0fa88b1e
119 Author: Ashley An <ashley.an@columbia.edu>
120 Date: Mon Dec 17 17:29:45 2018 -0500
121
122     switched RS to FS
123
124 commit 2b114337cfe8e6015469ad9fbfee8ac2933f4b7e
125 Merge: c6870be e1c358f
126 Author: Ashley An <ashley.an@columbia.edu>
127 Date: Mon Dec 17 17:28:13 2018 -0500
128
129     merge
130
131 commit c6870bed867f7a89f4753022fb5d0d24798698ad
132 Author: Ashley An <ashley.an@columbia.edu>
133 Date: Mon Dec 17 17:25:24 2018 -0500
134
135     cleaned up and commented code
136
137 commit e1c358ff49b5524a542b4575319f4644e077ad76
138 Author: Melanie Sawyer <melaniensawyer@gmail.com>
139 Date: Mon Dec 17 17:23:22 2018 -0500
140
141     add file with columns and constitution
142
143 commit a77c6858cc8fee5c9b4e700bf05baaa89aa8e513
144 Author: Christine Hsu <christine.hhsu@gmail.com>
145 Date: Mon Dec 17 17:17:33 2018 -0500
146
147     add shuffle.bawk
148
149 commit a737c26dd43768af98e908c0c4a7f5c6b6efcc94
150 Author: Christine Hsu <christine.hhsu@gmail.com>
151 Date: Mon Dec 17 17:17:05 2018 -0500
152
153     demo
154
155 commit a741435266c3f32ebf6aa00edb35826efd193094
156 Merge: 61a8e87 94e6b02
157 Author: Christine Hsu <christine.hhsu@gmail.com>
158 Date: Mon Dec 17 17:16:42 2018 -0500
159
160     Merge branch 'master' of https://github.com/soybean/bawk
161
162 commit 61a8e87fe4b9fb462ebd43738a932f211039e78f
163 Author: Christine Hsu <christine.hhsu@gmail.com>
164 Date: Mon Dec 17 17:16:34 2018 -0500
165
166     add nprint to codegen
```

```
167
168 commit 94e6b02c8cc85cc30eb7a356613f868d97005d90
169 Author: Melanie Sawyer <melaniensawyer@gmail.com>
170 Date: Mon Dec 17 17:13:45 2018 -0500
171
172     add constitution to ascii art
173
174 commit ec2c9cf5aed89d69de026df897a6fdad29ce0529
175 Author: Melanie Sawyer <melaniensawyer@gmail.com>
176 Date: Mon Dec 17 17:13:18 2018 -0500
177
178     remove comment
179
180 commit fd46c48df455b0e6195929f86750ea2afa7eb004
181 Author: Christine Hsu <christine.hhsu@gmail.com>
182 Date: Mon Dec 17 16:59:17 2018 -0500
183
184     add the constitution
185
186 commit caba497b9b6e2c324d4e3bba76b1922f8f202691
187 Merge: 7b18691 8e03324
188 Author: Christine Hsu <christine.hhsu@gmail.com>
189 Date: Mon Dec 17 16:59:06 2018 -0500
190
191     Merge branch 'master' of https://github.com/soybean/bawk
192
193 commit 7b18691a22fd138c305acc787ce30907fbfe56ad
194 Author: Christine Hsu <christine.hhsu@gmail.com>
195 Date: Mon Dec 17 16:56:37 2018 -0500
196
197     add the Constitution
198
199 commit 8e0332407b896fb0c97d72d59b945f11ebb135b3
200 Author: Melanie Sawyer <melaniensawyer@gmail.com>
201 Date: Mon Dec 17 04:37:13 2018 -0500
202
203     add demo text file
204
205 commit 95a328eb1727872a2b15ca6e7e29bf7f5ac440fa
206 Author: Ashley An <ashley.an@columbia.edu>
207 Date: Sun Dec 16 20:17:21 2018 -0500
208
209     removed unnecessary script
210
211 commit 842fdc02b05249011ec983d6adb559c58452e179
212 Author: Ashley An <ashley.an@columbia.edu>
213 Date: Sun Dec 16 20:16:34 2018 -0500
214
215     cleaned up files
216
217 commit b1d2645cf1975c964da82830a51db5bb2a487b48
218 Author: Ashley An <ashley.an@columbia.edu>
219 Date: Sun Dec 16 20:13:30 2018 -0500
220
221     added .out and .err files for tests
222
```

```

223 commit 55cf1296ecd4f9198db83e69ceb735e2256601f8
224 Merge: f02e5de c183cd1
225 Author: Melanie Sawyer <melaniensawyer@gmail.com>
226 Date: Sun Dec 16 19:59:37 2018 -0500
227
228 Merge branch 'master' of https://github.com/soybean/PLT-f18
229
230 commit f02e5de67b16b2f0062a58a4e99996d388a658f0
231 Author: Melanie Sawyer <melaniensawyer@gmail.com>
232 Date: Sun Dec 16 19:59:18 2018 -0500
233
234 remove warnings
235
236 commit c183cd1d668c600c2b211cae8a5a5c4cc3fa9937
237 Author: Ashley An <ashley.an@columbia.edu>
238 Date: Sun Dec 16 19:54:22 2018 -0500
239
240 fixed testall.sh so that it prints both stderr and stdout to .out file
241
242 commit 6f4c79e1874f7bf983ba86ea91955e19429a3b64
243 Author: Ashley An <ashley.an@columbia.edu>
244 Date: Sun Dec 16 19:39:15 2018 -0500
245
246 removed test and fixed generate-tests script to print out both stderr and
    stdout to .out file
247
248 commit 2cff306804050f5e8a81fcbfd07b396d1972a7c7
249 Merge: 97c1825 4ff24ee
250 Author: Ashley An <ashley.an@columbia.edu>
251 Date: Sun Dec 16 17:54:06 2018 -0500
252
253 Merge branch 'master' of https://github.com/soybean/PLT-f18
254
255 commit 97c1825ea42294bd096962a20eabeebf77dale1cc
256 Author: Ashley An <ashley.an@columbia.edu>
257 Date: Sun Dec 16 17:53:56 2018 -0500
258
259 moved runtime fail tests to pass tests
260
261 commit 4ff24eef455ff8baab198e2c4b82bae2c9978692
262 Author: Melanie Sawyer <melaniensawyer@gmail.com>
263 Date: Sun Dec 16 17:41:21 2018 -0500
264
265 remove print statement
266
267 commit 15cc960c0bf7d6b129693ae39fb859b56d599964
268 Merge: e7156d3 ca1d52f
269 Author: Melanie Sawyer <melaniensawyer@gmail.com>
270 Date: Sun Dec 16 17:36:42 2018 -0500
271
272 Merge branch 'master' of https://github.com/soybean/PLT-f18
273
274 commit e7156d39730099674cf5fea3b32ab6e2834776ec
275 Author: Melanie Sawyer <melaniensawyer@gmail.com>
276 Date: Sun Dec 16 17:36:25 2018 -0500
277

```

```
278     add RS/FS
279
280 commit 3952cf8cb41f71d4d88b7db408397f3461dd3a3d
281 Author: Melanie Sawyer <melaniensawyer@gmail.com>
282 Date:   Sun Dec 16 15:39:31 2018 -0500
283
284     broken stuff
285
286 commit ca1d52f91265627f72766b037592e18d321517f5
287 Author: Ashley An <ashley.an@columbia.edu>
288 Date:   Sun Dec 16 14:00:13 2018 -0500
289
290     print usage message to stderr
291
292 commit c4ecf6208c39083d7da2b6c608be4b492d55e230
293 Author: Christine Hsu <christine.hhsu@gmail.com>
294 Date:   Sun Dec 16 11:04:16 2018 -0500
295
296     print error to stderr
297
298 commit c3eff0255b80f756ead323351ec74e4563aee4c4
299 Author: Christine Hsu <christine.hhsu@gmail.com>
300 Date:   Sun Dec 16 10:29:40 2018 -0500
301
302     index out of bounds error
303
304 commit 04f5f3de3e06be976dc97dd9d0e83825258a3b78
305 Author: Christine Hsu <christine.hhsu@gmail.com>
306 Date:   Sat Dec 15 19:58:37 2018 -0500
307
308     add index out of bounds error for arrays
309
310 commit 404b40317d70797f90174aa49faa3da89623c16c
311 Author: Ashley An <ashley.an@columbia.edu>
312 Date:   Thu Dec 13 23:55:38 2018 -0500
313
314     added usage message for when input file isn't given
315
316 commit 82d8aa5b9a401e5022faf49e4c38c1e8a3cd0597
317 Author: Melanie Sawyer <melaniensawyer@gmail.com>
318 Date:   Fri Dec 14 16:30:30 2018 -0500
319
320     fix bool operators
321
322 commit 1adf3ca74b42950e0ae0f2baec15d253d9ef2814
323 Author: Victoria Yang <victoria.j.yang@gmail.com>
324 Date:   Thu Dec 13 23:39:06 2018 -0500
325
326     fix syntax
327
328 commit 5f54e2a52e0611ef924aac9219277707d86ae518
329 Author: Ashley An <ashley.an@columbia.edu>
330 Date:   Thu Dec 13 23:37:18 2018 -0500
331
332     added config tests back
333
```

```
334 | commit 00593446fcdca8364f5b09eb019b020138da8532
335 | Merge: 95026ff 32ed67f
336 | Author: Ashley An <ashley.an@columbia.edu>
337 | Date: Thu Dec 13 23:23:41 2018 -0500
338 |
339 | Merge branch 'master' of https://github.com/soybean/PLT-f18
340 |
341 | commit 95026ffd23e441f3b9aacf110f1d5501d8143336
342 | Author: Ashley An <ashley.an@columbia.edu>
343 | Date: Thu Dec 13 23:23:36 2018 -0500
344 |
345 | creating testing suite script
346 |
347 | commit 32ed67f5d4b320b706c71a74324f11bfe0f0e065
348 | Merge: 12bc74e 47c6140
349 | Author: Christine Hsu <christine.hhsu@gmail.com>
350 | Date: Thu Dec 13 21:14:19 2018 -0500
351 |
352 | fix merge conflict
353 |
354 | commit 12bc74ec1e4577cbad2a2fdfed874d4cda0e833c
355 | Author: Christine Hsu <christine.hhsu@gmail.com>
356 | Date: Thu Dec 13 21:13:30 2018 -0500
357 |
358 | attempt index out of bounds check
359 |
360 | commit 4bdaff44e9f85c7bda2aa1d954a0fd0b78f4da63
361 | Author: Christine Hsu <christine.hhsu@gmail.com>
362 | Date: Thu Dec 13 21:13:04 2018 -0500
363 |
364 | attempt index out of bounds check
365 |
366 | commit 47c6140b1afae2be6aaecf2e9f1f997470173b9a
367 | Merge: ebce545 94ac77d
368 | Author: Victoria Yang <victoria.j.yang@gmail.com>
369 | Date: Thu Dec 13 21:10:17 2018 -0500
370 |
371 | Merge branch 'master' of https://github.com/soybean/PLT-f18
372 |
373 | commit ebce545674f428b06f0632cb9c42d347ddde764d
374 | Author: Victoria Yang <victoria.j.yang@gmail.com>
375 | Date: Thu Dec 13 21:10:06 2018 -0500
376 |
377 | fixed formatting
378 |
379 | commit 94ac77d41f7853861fd7ec5c64cd02128d10009b
380 | Author: Ashley An <ashley.an@columbia.edu>
381 | Date: Thu Dec 13 20:52:39 2018 -0500
382 |
383 | renamed test.txt for demo
384 |
385 | commit f8f2bc92e4d2a7a1ce13468c93eeec5ce913f002
386 | Author: Ashley An <ashley.an@columbia.edu>
387 | Date: Thu Dec 13 20:33:51 2018 -0500
388 |
389 | removed test
```



```
390
391 commit c5afa988328574c041068f6db8145ee408a99b79
392 Author: Ashley An <ashley.an@columbia.edu>
393 Date: Thu Dec 13 20:32:59 2018 -0500
394
395     fixed formatting of test
396
397 commit 2687261e2d6fcda9d446839898e6d8665cb55c81
398 Merge: 5ac0cf7 19010a6
399 Author: Victoria Yang <victoria.j.yang@gmail.com>
400 Date: Thu Dec 13 20:27:53 2018 -0500
401
402     Merge branch 'master' of https://github.com/soybean/PLT-f18
403
404 commit 5ac0cf76d5453cf84fad6da6dba7017d835b6dd7
405 Author: Victoria Yang <victoria.j.yang@gmail.com>
406 Date: Thu Dec 13 20:27:39 2018 -0500
407
408     fix delete for arrays
409
410 commit 19010a6ead0ac3688c0a8ab53f419345cdcea7c9
411 Merge: 5d298c6 9a575a8
412 Author: Ashley An <ashley.an@columbia.edu>
413 Date: Thu Dec 13 20:26:18 2018 -0500
414
415     Merge branch 'master' of https://github.com/soybean/PLT-f18
416
417 commit 5d298c60a3290ab22c39f7c563a31e35359be53d
418 Author: Ashley An <ashley.an@columbia.edu>
419 Date: Thu Dec 13 20:26:14 2018 -0500
420
421     fixed dynamic arr tests
422
423 commit 9a575a880e0f73d7bb51423ff69df3b03ddb1855
424 Merge: 0804277 e2fb5a2
425 Author: Melanie Sawyer <melaniensawyer@gmail.com>
426 Date: Thu Dec 13 20:19:38 2018 -0500
427
428     Merge branch 'master' of https://github.com/soybean/PLT-f18
429
430 commit 08042770bed615dc896164550b1770478cd08737
431 Author: Melanie Sawyer <melaniensawyer@gmail.com>
432 Date: Thu Dec 13 20:19:18 2018 -0500
433
434     remove weird files and add stephen edwards in ascii art
435
436 commit e2fb5a23518904b121912df866bd60c9b0c06d0d
437 Author: Victoria Yang <victoria.j.yang@gmail.com>
438 Date: Thu Dec 13 20:16:42 2018 -0500
439
440     get rid of unused code in semant
441
442 commit 6ee28f6f6492ccfe7a9919a6946ccefcae8075048
443 Author: Ashley An <ashley.an@columbia.edu>
444 Date: Thu Dec 13 20:14:26 2018 -0500
445
```

```
446     deleted config tests
447
448 commit a52c1f116d43236dcbea3eaa2d6ef7270c486827
449 Author: Victoria Yang <victoria.j.yang@gmail.com>
450 Date:   Thu Dec 13 20:04:33 2018 -0500
451
452     fixed delete in semant
453
454 commit 33717d6a96c9ee10a7de91175f245ca97d0d77c9
455 Author: Melanie Sawyer <melaniensawyer@gmail.com>
456 Date:   Thu Dec 13 20:03:23 2018 -0500
457
458     fix remaining warnings
459
460 commit f4b13394088b1958b23b32d4f90f80b156045e17
461 Merge:  c12c0ad 15fe3ee
462 Author: Melanie Sawyer <melaniensawyer@gmail.com>
463 Date:   Thu Dec 13 20:00:45 2018 -0500
464
465     Merge branch 'master' of https://github.com/soybean/PLT-f18
466
467 commit c12c0ad894183167e6c42172528eeb6afa56505b
468 Merge:  32b0e72 73fd684
469 Author: Melanie Sawyer <melaniensawyer@gmail.com>
470 Date:   Thu Dec 13 20:00:22 2018 -0500
471
472     Merge branch 'master' of https://github.com/soybean/PLT-f18
473
474 commit 32b0e72a20cf83444c9ed377b3904fb8a5546ab3
475 Author: Melanie Sawyer <melaniensawyer@gmail.com>
476 Date:   Thu Dec 13 20:00:04 2018 -0500
477
478     get rid of some warnings
479
480 commit 15fe3ee3825f5355970104bf9dc0477e17e1d687
481 Author: Christine Hsu <christine.hhsu@gmail.com>
482 Date:   Thu Dec 13 19:58:22 2018 -0500
483
484     Clean error messages in codegen
485
486 commit 73fd6844301f12dc8df7ec86b305b95b16ecadda
487 Author: Ashley An <ashley.an@columbia.edu>
488 Date:   Thu Dec 13 19:56:26 2018 -0500
489
490     fixed for2 test
491
492 commit 3da5d6ee016f42470810cc59652349d832f2492a
493 Author: Victoria Yang <victoria.j.yang@gmail.com>
494 Date:   Thu Dec 13 19:45:59 2018 -0500
495
496     fixed loop/end name
497
498 commit 862608b1a3ab13946a6aea54a45e33d70bf7280f
499 Author: Christine Hsu <christine.hhsu@gmail.com>
500 Date:   Thu Dec 13 19:43:14 2018 -0500
501
```

```
502     remove comparelists
503
504 commit cb61dbe1e1d0cf0ea1a5d466077e65413c01b7f2
505 Merge: cdddf5c 4ee85f1
506 Author: Christine Hsu <christine.hhsu@gmail.com>
507 Date: Thu Dec 13 19:37:22 2018 -0500
508
509     Merge branch 'master' of https://github.com/soybean/PLT-f18
510
511 commit cdddf5c3f26cb727dec8f80dbe86a011be8b6248
512 Author: Christine Hsu <christine.hhsu@gmail.com>
513 Date: Thu Dec 13 19:37:11 2018 -0500
514
515     add index_of
516
517 commit 4ee85f18c9069d220d39dbe53a141f2b93b73d7e
518 Merge: 3bff4ad 12393e0
519 Author: Ashley An <ashley.an@columbia.edu>
520 Date: Thu Dec 13 19:33:47 2018 -0500
521
522     Merge branch 'master' of https://github.com/soybean/PLT-f18
523
524 commit 3bff4ad39c7730d40879b9fca150ab2415ad76c8
525 Author: Ashley An <ashley.an@columbia.edu>
526 Date: Thu Dec 13 19:33:43 2018 -0500
527
528     added loop and end fail tests
529
530 commit 12393e00f29c2ba7ce23e7baa952da58cec1d39b
531 Merge: a71afd6 031bc22
532 Author: Christine Hsu <christine.hhsu@gmail.com>
533 Date: Thu Dec 13 19:30:14 2018 -0500
534
535     contains works
536
537 commit a71afd646ad10aa2fef2755797eaeec9ab7540
538 Author: Christine Hsu <christine.hhsu@gmail.com>
539 Date: Thu Dec 13 19:28:47 2018 -0500
540
541     contains works
542
543 commit 031bc2200b984a880b3e90074c7f92ccb4b689db
544 Merge: 71b0bea bdef67d
545 Author: Melanie Sawyer <melaniensawyer@gmail.com>
546 Date: Thu Dec 13 19:21:37 2018 -0500
547
548     Merge branch 'master' of https://github.com/soybean/PLT-f18
549
550 commit 71b0beab990e85d27aba51a36bc054aa5d61e91d
551 Author: Melanie Sawyer <melaniensawyer@gmail.com>
552 Date: Thu Dec 13 19:18:29 2018 -0500
553
554     fix some warnings
555
556 commit bdef67d70875919c0ec76d3a01b3ec1db73a900a
557 Author: Ashley An <ashley.an@columbia.edu>
```

558 Date: Thu Dec 13 19:17:04 2018 -0500  
559  
560 removed main function for mylist.c  
561  
562 commit 499b4d0c57377fbaea5f88e95b406cbb6e36e3fb  
563 Author: Ashley An <ashley.an@columbia.edu>  
564 Date: Thu Dec 13 19:08:42 2018 -0500  
565  
566 fixed comparators to take in unsigned longs instead of const void \*  
567  
568 commit f98ea40080c7b81d89fd0236c439f48f1966f994  
569 Author: Ashley An <ashley.an@columbia.edu>  
570 Date: Thu Dec 13 18:44:37 2018 -0500  
571  
572 edited README  
573  
574 commit 993c72da72b557b84f05ddbf6d8e266360ed3c79  
575 Author: Ashley An <ashley.an@columbia.edu>  
576 Date: Thu Dec 13 18:42:45 2018 -0500  
577  
578 edited README  
579  
580 commit c5fee69b7b3202a3ff534be78c3cd72e34f780d9  
581 Author: Ashley An <ashley.an@columbia.edu>  
582 Date: Thu Dec 13 18:41:30 2018 -0500  
583  
584 edited README  
585  
586 commit b98fb56c0d353118019b2777ff3b98d76f5d3419  
587 Author: Ashley An <ashley.an@columbia.edu>  
588 Date: Thu Dec 13 18:40:06 2018 -0500  
589  
590 edited README  
591  
592 commit 3e2118083ba7eb904dac22df11eab72fc9e9bc26  
593 Author: Ashley An <ashley.an@columbia.edu>  
594 Date: Thu Dec 13 18:37:50 2018 -0500  
595  
596 edited Makefile and bawk.sh to be cleaner  
597  
598 commit 650b325678c96123caef7b6f1acd828a9403c230  
599 Author: Ashley An <ashley.an@columbia.edu>  
600 Date: Thu Dec 13 18:27:04 2018 -0500  
601  
602 removed unnecessary script  
603  
604 commit 145d9b3aad73a42c60a9c4d77e0f022ccd7b1817  
605 Author: Ashley An <ashley.an@columbia.edu>  
606 Date: Thu Dec 13 17:35:07 2018 -0500  
607  
608 fixed contains and indexOf  
609  
610 commit 89951de56381258d31dd80878997666f17587003  
611 Author: Ashley An <ashley.an@columbia.edu>  
612 Date: Thu Dec 13 17:18:03 2018 -0500  
613

```
614     fixed bawk.sh script so that it stops execution when there's an error
615
616 commit df46c68f6c444e23ada5ab81375ef2a2ea4fda71
617 Author: Ashley An <ashley.an@columbia.edu>
618 Date:   Thu Dec 13 17:09:18 2018 -0500
619
620     fixed error message for insert in semant
621
622 commit af4295347fa2dce81ea25156f44d331119d6ee3d
623 Author: Ashley An <ashley.an@columbia.edu>
624 Date:   Thu Dec 13 16:56:47 2018 -0500
625
626     fixed minor error in semant
627
628 commit aba9e1e6aafba41be6f779dea620ceedae8ec303
629 Author: Ashley An <ashley.an@columbia.edu>
630 Date:   Thu Dec 13 16:49:23 2018 -0500
631
632     fixed formatting of test
633
634 commit 47a37c89b9c4b3020c0a08f5f6503c0fc7c23bb2
635 Merge: 816ea6a f41973a
636 Author: Ashley An <ashley.an@columbia.edu>
637 Date:   Thu Dec 13 16:38:50 2018 -0500
638
639     Merge branch 'master' of https://github.com/soybean/PLT-f18
640
641 commit 816ea6ad2f05602e44e70af763ed5cb2eb78fc55
642 Author: Ashley An <ashley.an@columbia.edu>
643 Date:   Thu Dec 13 16:38:44 2018 -0500
644
645     fixed regex assign test
646
647 commit f41973ae64d3999647fb04752ece039ddef36e82
648 Author: Melanie Sawyer <melaniensawyer@gmail.com>
649 Date:   Thu Dec 13 16:36:28 2018 -0500
650
651     Update README.md
652
653 commit f3ec5b4a8a0f2042f0756e6f52203a8479beaa11
654 Merge: 862afa9 676628e
655 Author: Melanie Sawyer <melaniensawyer@gmail.com>
656 Date:   Thu Dec 13 16:29:18 2018 -0500
657
658     Merge branch 'master' of https://github.com/soybean/PLT-f18
659
660 commit 862afa9a9c970be2d4b83ec41b44078f4df82c50
661 Author: Melanie Sawyer <melaniensawyer@gmail.com>
662 Date:   Thu Dec 13 16:27:55 2018 -0500
663
664     add single run executable
665
666 commit 676628e9e0faa344b65a125f7de15fe077d68a4d
667 Author: Victoria Yang <victoria.j.yang@gmail.com>
668 Date:   Thu Dec 13 16:08:38 2018 -0500
669
```

```
670 cleared warning
671
672 commit ca7e71d700af84d7ed3876eb331e4a864d724bcd
673 Author: Victoria Yang <victoria.j.yang@gmail.com>
674 Date: Thu Dec 13 15:12:17 2018 -0500
675
676 clean up warnings
677
678 commit 80fd93779db4f8f8de2e9f43159ff8aeaa64e22d
679 Author: Melanie Sawyer <melaniensawyer@gmail.com>
680 Date: Thu Dec 13 13:55:05 2018 -0500
681
682 remove print statements in test script
683
684 commit e3d020bba6af80ece31ca077d5e25dbd854cfad6
685 Author: Melanie Sawyer <melaniensawyer@gmail.com>
686 Date: Thu Dec 13 13:30:28 2018 -0500
687
688 string operators
689
690 commit 6af4508e5586421e3cfbe409a2c4b020b5bd98a2
691 Merge: 061167a 9884e63
692 Author: Victoria Yang <victoria.j.yang@gmail.com>
693 Date: Thu Dec 13 12:33:55 2018 -0500
694
695 merge
696
697 commit 061167ac79c78f9c98cf844dadb77322011400da
698 Author: Victoria Yang <victoria.j.yang@gmail.com>
699 Date: Thu Dec 13 12:28:09 2018 -0500
700
701 clean up warnings
702
703 commit 9884e6372114d368a0350ce648decdd4c98190
704 Author: Ashley An <ashley.an@columbia.edu>
705 Date: Thu Dec 13 12:19:02 2018 -0500
706
707 cleaned up Makefile
708
709 commit 7c1789f565b65c3a2dca24c66855f23b8abc15f3
710 Merge: 7d57972 75f957d
711 Author: Ashley An <ashley.an@columbia.edu>
712 Date: Thu Dec 13 12:12:59 2018 -0500
713
714 Merge branch 'master' of https://github.com/soybean/PLT-f18
715
716 commit 7d579729c6faf36a9ab73edd39bc3438b382207e
717 Author: Ashley An <ashley.an@columbia.edu>
718 Date: Thu Dec 13 12:12:53 2018 -0500
719
720 fixed make clean
721
722 commit 75f957db467c701718dd2b8d1739d4b9da8ca54b
723 Merge: 9b74c90 2d61756
724 Author: Christine Hsu <christine.hhsu@gmail.com>
725 Date: Thu Dec 13 12:11:40 2018 -0500
```

726  
727 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
728  
729 commit 9b74c90bb0a998e51d51a1b8b81d61fd86781673  
730 Author: Christine Hsu <christine.hhsu@gmail.com>  
731 Date: Thu Dec 13 12:11:27 2018 -0500  
732  
733 take care of some warnings  
734  
735 commit 2d61756b46c4c6347df15b7f63b3bddd28adf8ec  
736 Author: Ashley An <ashley.an@columbia.edu>  
737 Date: Thu Dec 13 12:04:22 2018 -0500  
738  
739 removed unnecessary script  
740  
741 commit 3afa53dafa6690f8ab9dde07b83ef1b5c51d6298  
742 Author: Ashley An <ashley.an@columbia.edu>  
743 Date: Thu Dec 13 11:43:52 2018 -0500  
744  
745 removed unnecessary test files  
746  
747 commit cf666ef3f8b8f737f342c9d7a16c0a5f8efa5f8d  
748 Author: Christine Hsu <christine.hhsu@gmail.com>  
749 Date: Thu Dec 13 11:37:56 2018 -0500  
750  
751 add rgx comparator  
752  
753 commit 15f319f491df1d4f680d8b6b7ee491ff12217f68  
754 Merge: d4b02bf 1e5c8c6  
755 Author: Christine Hsu <christine.hhsu@gmail.com>  
756 Date: Thu Dec 13 11:30:04 2018 -0500  
757  
758 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
759  
760 commit d4b02bf90a0d50e0222f81c3d5d7ad2a6b687b5b  
761 Author: Christine Hsu <christine.hhsu@gmail.com>  
762 Date: Thu Dec 13 11:29:55 2018 -0500  
763  
764 index\_of works for string arrays  
765  
766 commit 1e5c8c6af3e064027a2af637169a8f9f66598409  
767 Author: Ashley An <ashley.an@columbia.edu>  
768 Date: Thu Dec 13 11:29:38 2018 -0500  
769  
770 fixed formatting of tests  
771  
772 commit 508cc1f5375d1a83edcc8ce2479be95a353aeacf  
773 Author: Ashley An <ashley.an@columbia.edu>  
774 Date: Thu Dec 13 11:28:07 2018 -0500  
775  
776 edited contains tests  
777  
778 commit 2f82fb9bf24352f8876e115f2d71a58d1199368d  
779 Merge: cabe654 805e444  
780 Author: Victoria Yang <victoria.j.yang@gmail.com>  
781 Date: Thu Dec 13 11:22:00 2018 -0500

```
782
783 Merge branch 'master' of https://github.com/soybean/PLT-f18
784
785 commit cabe654c1caf93c570d60dea93a933cbe9f1965f
786 Author: Victoria Yang <victoria.j.yang@gmail.com>
787 Date: Thu Dec 13 11:21:41 2018 -0500
788
789 arrayderef assigngit add codegen.ml
790
791 commit 805e4447089781ee430e9bbba89ee8911c5ed53c
792 Author: Christine Hsu <christine.hhsu@gmail.com>
793 Date: Thu Dec 13 10:59:12 2018 -0500
794
795 contains works for string arrays of any dimension
796
797 commit 734f0fe20524437c3eaec7a8fbb6c700b5e2e4f7
798 Author: Christine Hsu <christine.hhsu@gmail.com>
799 Date: Thu Dec 13 10:51:52 2018 -0500
800
801 Use legit comparator
802
803 commit d8079d52090d01419c92a951121a6ff203932130
804 Author: Christine Hsu <christine.hhsu@gmail.com>
805 Date: Thu Dec 13 10:46:01 2018 -0500
806
807 string contains works
808
809 commit c0cadedc3299c2911602d024ef9972f0b3b05a8f
810 Merge: 10b7d42 02675a1
811 Author: Christine Hsu <christine.hhsu@gmail.com>
812 Date: Wed Dec 12 23:19:59 2018 -0500
813
814 Merge branch 'master' of https://github.com/soybean/PLT-f18
815
816 commit 10b7d423c8c93706ab0bef99b9d04305881d7002
817 Author: Christine Hsu <christine.hhsu@gmail.com>
818 Date: Wed Dec 12 23:19:46 2018 -0500
819
820 change ret type of contains
821
822 commit 02675a1f5ab975ce12d5804ef9164dd3c75e98ba
823 Merge: 1b4b93c 4731ba7
824 Author: Victoria Yang <victoria.j.yang@gmail.com>
825 Date: Wed Dec 12 22:10:19 2018 -0500
826
827 Merge branch 'master' of https://github.com/soybean/PLT-f18
828
829 commit 4731ba7ef29ffe719702cd19806fef6675a0c886
830 Merge: 6fce76e 86952f3
831 Author: Melanie Sawyer <melaniensawyer@gmail.com>
832 Date: Wed Dec 12 23:01:00 2018 -0500
833
834 add string stuff
835
836 commit 1b4b93cdc6a72221fb888252a89278fc8caeb7b3
837 Author: Victoria Yang <victoria.j.yang@gmail.com>
```



```
838 Date: Wed Dec 12 22:10:07 2018 -0500
839
840 added rgx
841
842 commit d04dddc7c73adf93a41bbb25f3f23f34277336e4
843 Author: Victoria Yang <victoria.j.yang@gmail.com>
844 Date: Wed Dec 12 22:08:35 2018 -0500
845
846 Revert "added regex and working on assign"
847
848 This reverts commit 5bff3248402a86d4a62f3a89c98dce8536c9a409.
849
850 commit 86952f3d8b8bbbc470558361ddc2f0858743deb4b
851 Merge: 5bff324 5a4c312
852 Author: Victoria Yang <victoria.j.yang@gmail.com>
853 Date: Wed Dec 12 21:56:16 2018 -0500
854
855 Merge branch 'master' of https://github.com/soybean/PLT-f18
856
857 commit 5a4c312be451c193253a42f6f6ba4d164125237a
858 Author: Ashley An <ashley.an@columbia.edu>
859 Date: Wed Dec 12 22:37:02 2018 -0500
860
861 fixed formatting of test
862
863 commit 57a93defe69ea159ce89c089b96751ae5ceddcf3
864 Author: Ashley An <ashley.an@columbia.edu>
865 Date: Wed Dec 12 22:35:32 2018 -0500
866
867 added increment back into refactored codegen
868
869 commit 5bff3248402a86d4a62f3a89c98dce8536c9a409
870 Author: Victoria Yang <victoria.j.yang@gmail.com>
871 Date: Wed Dec 12 21:55:59 2018 -0500
872
873 added regex and working on assign
874
875 commit 8521a7683aa142b55c8bc052b787838451a181cb
876 Author: Ashley An <ashley.an@columbia.edu>
877 Date: Wed Dec 12 21:31:12 2018 -0500
878
879 fixed tests
880
881 commit cc629562d8614f8941ea37be3f9619eae288b1d6
882 Author: Ashley An <ashley.an@columbia.edu>
883 Date: Wed Dec 12 21:28:27 2018 -0500
884
885 fixed tests
886
887 commit ec03b397940122ee247337e3737057741289fbf2
888 Author: Ashley An <ashley.an@columbia.edu>
889 Date: Wed Dec 12 21:15:05 2018 -0500
890
891 created a variable called rs to hold RS that we want to string from each read-
892 in line
```

```
893 commit a3d61560e223bcb8913e43f17e181581dd9b0913
894 Author: Ashley An <ashley.an@columbia.edu>
895 Date: Wed Dec 12 21:13:07 2018 -0500
896
897     fixed dollar so that it now strips \n from string
898
899 commit e5541e7306243746d3387579c9d2c733f55e8ea2
900 Author: Ashley An <ashley.an@columbia.edu>
901 Date: Wed Dec 12 21:04:44 2018 -0500
902
903     deleted redundant test
904
905 commit 6b770e69952d9f027d429ffc538293a37536c0ae
906 Author: Ashley An <ashley.an@columbia.edu>
907 Date: Wed Dec 12 21:03:10 2018 -0500
908
909     changed name of test to be grouped into all dollar tests
910
911 commit d67e45e118f790c5741197a2bd1a5c8fe0d2947d
912 Merge: 7430a02 3544e38
913 Author: Ashley An <ashley.an@columbia.edu>
914 Date: Wed Dec 12 21:01:40 2018 -0500
915
916     Merge branch 'master' of https://github.com/soybean/PLT-f18
917
918 commit 7430a02516c30e10343e67d0065fc81f484b2aa2
919 Author: Ashley An <ashley.an@columbia.edu>
920 Date: Wed Dec 12 21:01:26 2018 -0500
921
922     fixed config tests
923
924 commit 3544e386e65f9b4c2310cab1cb5b8993e63b427e
925 Author: Victoria Yang <victoria.j.yang@gmail.com>
926 Date: Wed Dec 12 20:49:45 2018 -0500
927
928     set assign
929
930 commit d5ec0b319bd3d44322ac7d181c30a91ae3a8d2e3
931 Author: Ashley An <ashley.an@columbia.edu>
932 Date: Wed Dec 12 20:49:23 2018 -0500
933
934     changed True and False to true and false for bool_to_string
935
936 commit fc47e0d01928d0fb8feefde14a1c434ee2e9a4f9
937 Merge: d26dd13 e62a16c
938 Author: Ashley An <ashley.an@columbia.edu>
939 Date: Wed Dec 12 20:43:44 2018 -0500
940
941     Merge branch 'master' of https://github.com/soybean/PLT-f18
942
943 commit d26dd1351469f57d856b9baff5f9296c185cf1e9
944 Author: Ashley An <ashley.an@columbia.edu>
945 Date: Wed Dec 12 20:43:40 2018 -0500
946
947     fixed tests
948
```

```
949 | commit e62a16ce2ec59ca74a1c1de7f36c8dfa405b1faa
950 | Author: Christine Hsu <christine.hhsu@gmail.com>
951 | Date: Wed Dec 12 18:21:56 2018 -0500
952 |
953 |     array_test
954 |
955 | commit 41b0fa6816a06dd933432c516421f66224cf9fe0
956 | Author: Christine Hsu <christine.hhsu@gmail.com>
957 | Date: Wed Dec 12 16:06:16 2018 -0500
958 |
959 |     contains changes
960 |
961 | commit 6fce76ec318c617b61d1d6a57c8140208d550185
962 | Merge: d221056 a97e23c
963 | Author: Melanie Sawyer <melaniensawyer@gmail.com>
964 | Date: Wed Dec 12 15:47:52 2018 -0500
965 |
966 |     Merge branch 'master' of https://github.com/soybean/PLT-f18
967 |
968 | commit d221056d363394b3bea2a31aa3e261dfbc6b033b
969 | Author: Melanie Sawyer <melaniensawyer@gmail.com>
970 | Date: Wed Dec 12 15:47:21 2018 -0500
971 |
972 |     equals
973 |
974 | commit a97e23cd01fb5616a55b2c2552045536d2ae56f5
975 | Author: Christine Hsu <christine.hhsu@gmail.com>
976 | Date: Wed Dec 12 14:46:11 2018 -0500
977 |
978 |     new contains attempt
979 |
980 | commit ad6d4d1f8c6066591d536d73bbbabc5dad0f3ed1
981 | Author: Ashley An <ashley.an@columbia.edu>
982 | Date: Wed Dec 12 14:14:08 2018 -0500
983 |
984 |     created contains and indexOf wrapper methods
985 |
986 | commit 2597b15b8c0721d436487c814986db881d89841f
987 | Author: Melanie Sawyer <melaniensawyer@gmail.com>
988 | Date: Wed Dec 12 13:26:45 2018 -0500
989 |
990 |     add rgx_to_str
991 |
992 | commit fa32cbacd614948091037b4c1bfdaefa44e71123
993 | Author: Melanie Sawyer <melaniensawyer@gmail.com>
994 | Date: Wed Dec 12 13:18:11 2018 -0500
995 |
996 |     fix NF for empty lines
997 |
998 | commit 9e1c2f1ed7da2f85a37c11886748bff9a3cdb61f
999 | Author: Melanie Sawyer <melaniensawyer@gmail.com>
1000 | Date: Wed Dec 12 13:16:13 2018 -0500
1001 |
1002 |     add NF
1003 |
1004 | commit 38f857cd74ab392a9784ba60bb40131840c3e807
```

```
1005 Merge: c071142 49cb85c
1006 Author: Christine Hsu <christine.hhsu@gmail.com>
1007 Date: Wed Dec 12 12:17:30 2018 -0500
1008
1009 Merge branch 'master' of https://github.com/soybean/PLT-f18
1010
1011 commit c071142dbb89e8e62ab89964bfe6efd5ec7a2a0f
1012 Author: Christine Hsu <christine.hhsu@gmail.com>
1013 Date: Wed Dec 12 12:16:55 2018 -0500
1014
1015 add insert, delete, length built-ins
1016
1017 commit 49cb85c713e9129332feb290b48f5c2f9b2a59dd
1018 Author: Christine Hsu <christine.hhsu@gmail.com>
1019 Date: Wed Dec 12 09:02:37 2018 -0500
1020
1021 clean indents
1022
1023 commit 4f8e25d5cd5cdf8d96872d55105261f7b33e452e
1024 Author: Ashley An <ashley.an@columbia.edu>
1025 Date: Tue Dec 11 23:32:50 2018 -0500
1026
1027 removed test
1028
1029 commit bf23c9fc69175bf06661c5eb01d3dc471796feb9
1030 Author: Christine Hsu <christine.hhsu@gmail.com>
1031 Date: Tue Dec 11 22:57:48 2018 -0500
1032
1033 Add arrays to restructured codegen
1034
1035 commit de9f16e04b4aa2a7ec93ca226c57b1b1a0d340c2
1036 Merge: 5871087 56812fc
1037 Author: Ashley An <ashley.an@columbia.edu>
1038 Date: Tue Dec 11 22:29:55 2018 -0500
1039
1040 Merge pull request #13 from soybean/ashley
1041
1042 Ashley
1043
1044 commit 56812fcd486cbb631bd28c13b52ec8bb8c3db379
1045 Author: Ashley An <ashley.an@columbia.edu>
1046 Date: Tue Dec 11 22:28:50 2018 -0500
1047
1048 added other stuff from loop and end back into func
1049
1050 commit ce0d72f2ff5e80d206874beb55c6bfc06fba1e06
1051 Author: Ashley An <ashley.an@columbia.edu>
1052 Date: Tue Dec 11 22:07:39 2018 -0500
1053
1054 fixed assign
1055
1056 commit 119b59927bb0120809949220be4473529367fa60
1057 Author: Ashley An <ashley.an@columbia.edu>
1058 Date: Tue Dec 11 21:54:22 2018 -0500
1059
1060 if, for, while work in loop and end blocks now
```

1061  
1062 commit a1d5f0a99855f95a7250fca03902f5655d141ca6  
1063 Author: Ashley An <ashley.an@columbia.edu>  
1064 Date: Tue Dec 11 21:53:40 2018 -0500  
1065  
1066 removed preventing dollar in all functions from semant for now  
1067  
1068 commit e9d4aa9bf961722c4fe12207446a5118fc3aa4a1  
1069 Author: Ashley An <ashley.an@columbia.edu>  
1070 Date: Tue Dec 11 21:27:47 2018 -0500  
1071  
1072 if statements now work in loopgit add sadtest.bawk  
1073  
1074 commit bed0025852a7bc2bf424972a6c01782236220018  
1075 Author: Christine Hsu <christine.hhsu@gmail.com>  
1076 Date: Tue Dec 11 21:06:39 2018 -0500  
1077  
1078 edit function declarations  
1079  
1080 commit 845738dbe7435d30576d979581960362450184a4  
1081 Author: Ashley An <ashley.an@columbia.edu>  
1082 Date: Tue Dec 11 20:04:57 2018 -0500  
1083  
1084 refactoring codegen to make loop and end functions instead  
1085  
1086 commit d7900c2c87cbe14ad24ed9bd622503f19a049327  
1087 Author: Ashley An <ashley.an@columbia.edu>  
1088 Date: Tue Dec 11 19:15:00 2018 -0500  
1089  
1090 edited sadtest.bawk  
1091  
1092 commit a07f5040569f24e707bf610d972b1cddb569a405d  
1093 Author: Ashley An <ashley.an@columbia.edu>  
1094 Date: Tue Dec 11 19:14:46 2018 -0500  
1095  
1096 turned loop and end into functions in semant and sast  
1097  
1098 commit 58710879080b2b98ab5724c57a7f21651636680f  
1099 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1100 Date: Mon Dec 10 22:06:03 2018 -0500  
1101  
1102 Delete array.s  
1103  
1104 commit 564530b48cdc6385741e160ed054205d91b70058  
1105 Author: Ashley An <ashley.an@columbia.edu>  
1106 Date: Mon Dec 10 21:19:29 2018 -0500  
1107  
1108 edited Makefile  
1109  
1110 commit 185fe79e119bd3a8eec97a87f2025f2a2067050f  
1111 Author: Ashley An <ashley.an@columbia.edu>  
1112 Date: Mon Dec 10 21:18:54 2018 -0500  
1113  
1114 changes to Makefile  
1115  
1116 commit cdc6775a0b424e3a868542fb7081b078621ecfa2

```
1117 Author: Ashley An <ashley.an@columbia.edu>
1118 Date: Mon Dec 10 21:16:24 2018 -0500
1119
1120 added compareStrs method
1121
1122 commit 80e47b54181334a49f863a7e66223b38bf589b2f
1123 Author: Ashley An <ashley.an@columbia.edu>
1124 Date: Mon Dec 10 21:15:41 2018 -0500
1125
1126 added compareStrs method
1127
1128 commit bfb7528a9d3a3df39dea6400fc7f7a3e621e9b0
1129 Author: Ashley An <ashley.an@columbia.edu>
1130 Date: Sat Dec 8 16:50:33 2018 -0500
1131
1132 added contains method that returns int
1133
1134 commit 13d48e879a8532fe759a6ed4f9d9a3c27b5b848d
1135 Author: Ashley An <ashley.an@columbia.edu>
1136 Date: Sat Dec 8 14:31:48 2018 -0500
1137
1138 testing loop block if statements
1139
1140 commit dd907142fdf448670e95d48f77e96748329f0568
1141 Merge: 9009905 3bfc962
1142 Author: Ashley An <ashley.an@columbia.edu>
1143 Date: Thu Dec 6 14:54:12 2018 -0500
1144
1145 Merge branch 'master' of https://github.com/soybean/PLT-f18
1146
1147 commit 3bfc96283a5ff36e0dab992fd1f820c92f099480
1148 Author: Victoria Yang <victoria.j.yang@gmail.com>
1149 Date: Fri Dec 7 12:42:38 2018 -0500
1150
1151 update begin with some exprs
1152
1153 commit 900990564128b5f077dcff88d066b1e33bbba4f0
1154 Author: Ashley An <ashley.an@columbia.edu>
1155 Date: Thu Dec 6 14:54:04 2018 -0500
1156
1157 created string_to_int test
1158
1159 commit 9ecd96bc4d88d14e7765dea692573ca12de91eab
1160 Author: Ashley An <ashley.an@columbia.edu>
1161 Date: Thu Dec 6 13:58:42 2018 -0500
1162
1163 fixed regex test
1164
1165 commit ab738de82bdf9ab4837731efd32170b7998a1193
1166 Author: Ashley An <ashley.an@columbia.edu>
1167 Date: Thu Dec 6 13:56:57 2018 -0500
1168
1169 fixed regex tests
1170
1171 commit 8d333ca1710cb03b8d842bfd3ff8ec8d1c9394f7
1172 Author: Ashley An <ashley.an@columbia.edu>
```

```
1173 Date: Thu Dec 6 13:44:50 2018 -0500
1174
1175 fixed increment and decrement tests
1176
1177 commit e95791136a92b4f1187912760d162df400468dd9
1178 Author: Ashley An <ashley.an@columbia.edu>
1179 Date: Thu Dec 6 13:25:46 2018 -0500
1180
1181 edited makefile clean to remove .o files and removed duplicate int_to_string
in convert.c
1182
1183 commit 7a7756fffb5fdc5437ea1ddbdb24a0b413e68ca8
1184 Author: Ashley An <ashley.an@columbia.edu>
1185 Date: Thu Dec 6 12:09:06 2018 -0500
1186
1187 made dollar test print to test it
1188
1189 commit 6832811e2d0cb840c8f874486b56654de0bdae1a
1190 Merge: c719e4c b966e03
1191 Author: Ashley An <ashley.an@columbia.edu>
1192 Date: Thu Dec 6 11:21:29 2018 -0500
1193
1194 Merge branch 'master' of https://github.com/soybean/PLT-f18
1195
1196 commit c719e4c56534ad015e18e9f99ccc2bd4540840ed
1197 Author: Ashley An <ashley.an@columbia.edu>
1198 Date: Thu Dec 6 11:21:25 2018 -0500
1199
1200 added test for rgx to string comparator
1201
1202 commit b966e03845e8e4c3e4a1ef4f7a00ae0e1083d2a5
1203 Author: Victoria Yang <victoria.j.yang@gmail.com>
1204 Date: Thu Dec 6 11:20:28 2018 -0500
1205
1206 add semicolon
1207
1208 commit 74fa4e51720a450e1e00884d9fcfdaf0b79ca299
1209 Merge: f375848 416bbde
1210 Author: Victoria Yang <victoria.j.yang@gmail.com>
1211 Date: Thu Dec 6 11:16:51 2018 -0500
1212
1213 Merge branch 'master' of https://github.com/soybean/PLT-f18
1214
1215 commit 416bbdee3ec404990e9ba115bb11380ef30c0198
1216 Author: Ashley An <ashley.an@columbia.edu>
1217 Date: Thu Dec 6 11:16:38 2018 -0500
1218
1219 fixed regex tests
1220
1221 commit f37584837cd66ecb07ca4d80e6a3fc1c07e39399
1222 Merge: 804e0e5 00ce943
1223 Author: Victoria Yang <victoria.j.yang@gmail.com>
1224 Date: Thu Dec 6 11:16:15 2018 -0500
1225
1226 Merge branch 'master' of https://github.com/soybean/PLT-f18
1227
```

```
1228 commit 00ce9434af2f1a295a835e5e322a0c442d6fdb73
1229 Author: Ashley An <ashley.an@columbia.edu>
1230 Date: Thu Dec 6 11:13:33 2018 -0500
1231
1232     fixed regex tests
1233
1234 commit 7e4a214dc125601230d9516b3b602f1e78c7d8db
1235 Merge: 73d8f15 36c3e0b
1236 Author: Ashley An <ashley.an@columbia.edu>
1237 Date: Thu Dec 6 11:05:10 2018 -0500
1238
1239     Merge branch 'master' of https://github.com/soybean/PLT-f18
1240
1241 commit 73d8f152459f98aa1d333a79fa22a81767d497e2
1242 Author: Ashley An <ashley.an@columbia.edu>
1243 Date: Thu Dec 6 11:05:05 2018 -0500
1244
1245     added a bunch of missing fail tests
1246
1247 commit 804e0e53cfff619936db3a3f831b9f57b52a30de
1248 Author: Victoria Yang <victoria.j.yang@gmail.com>
1249 Date: Tue Dec 4 16:12:01 2018 -0500
1250
1251     need this for hw
1252
1253 commit 36c3e0b030a21fa51e74e2c1ae99d5b7ae39e8e4
1254 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1255 Date: Tue Dec 4 15:19:50 2018 -0500
1256
1257     uncomment string to int
1258
1259 commit e30c4be45c44b6653bc266caca1c7327c9bdff60
1260 Merge: c95edb0 6ae6907
1261 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1262 Date: Tue Dec 4 15:14:21 2018 -0500
1263
1264     Merge pull request #12 from soybean/regex-stuff
1265
1266     Regex stuff
1267
1268 commit 6ae690784fcd42384dfbee3b2dc50cd5172a4083
1269 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1270 Date: Tue Dec 4 15:13:34 2018 -0500
1271
1272     add regex equals and not equals
1273
1274 commit 26739dbc4939f7cd11558ef0c549a9048382c849
1275 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1276 Date: Tue Dec 4 15:03:43 2018 -0500
1277
1278     add not comp
1279
1280 commit 2c674551855ed1d03a80481db9532c351e8c1f6e
1281 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1282 Date: Tue Dec 4 14:47:16 2018 -0500
1283
```



```
1284     change regex function name
1285
1286 commit c95edb0064ca2a4820b5831d0d52e7ac0a4418e9
1287 Merge: 04ab7a5 c84941a
1288 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1289 Date: Tue Dec 4 14:40:38 2018 -0500
1290
1291     Merge branch 'master' of https://github.com/soybean/PLT-f18
1292
1293 commit 04ab7a586c5bb80a1896b3dfc30d6eb154aa6e94
1294 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1295 Date: Tue Dec 4 14:40:32 2018 -0500
1296
1297     add to gitignore
1298
1299 commit 8e958170a46fbc1b33b1530936c6cce6d14e1080
1300 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1301 Date: Tue Dec 4 01:36:47 2018 -0500
1302
1303     remove leading quote: NEEDS FIXING!
1304
1305 commit 4b01498dcebfed58284901e8cee7c7c17ac4f909
1306 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1307 Date: Tue Dec 4 01:26:16 2018 -0500
1308
1309     add rgx
1310
1311 commit c84941aaa1eefb81c63a07b24e1f4ceb4ff9de8c
1312 Author: Ashley An <ashley.an@columbia.edu>
1313 Date: Mon Dec 3 22:58:31 2018 -0500
1314
1315     tested 3D arrays
1316
1317 commit a077631ee32c81c5ae9a8bd35ac8104362997e47
1318 Author: Ashley An <ashley.an@columbia.edu>
1319 Date: Mon Dec 3 22:50:50 2018 -0500
1320
1321     removed main method from mylist.c
1322
1323 commit 1ddfacc8fca9fd9ba3383d7b19f353a378a4fe95b
1324 Merge: 2d1023f dffa5c5
1325 Author: Ashley An <ashley.an@columbia.edu>
1326 Date: Mon Dec 3 22:48:33 2018 -0500
1327
1328     Merge branch 'master' of https://github.com/soybean/PLT-f18
1329
1330 commit 2d1023f166678d6c3f9dd5833168ad1b93bd7afa
1331 Author: Ashley An <ashley.an@columbia.edu>
1332 Date: Mon Dec 3 22:48:29 2018 -0500
1333
1334     changed struct Node to have unsigned long data instead of void * data
1335
1336 commit dffa5c501217969a8d6e7a6fd142b35dc29643dd
1337 Author: Victoria Yang <victoria.j.yang@gmail.com>
1338 Date: Mon Dec 3 22:19:37 2018 -0500
1339
```

```
1340     codegen pluseq/minuseq
1341
1342 commit af72c36774581a770ab5a538250e8b4bbc1f1154
1343 Author: Christine Hsu <christine.hhsu@gmail.com>
1344 Date:   Mon Dec 3 19:26:31 2018 -0500
1345
1346     Update scanner.mll
1347
1348 commit 03411832dca0eb6b1ac20d54f5dc3b3d4222881c
1349 Author: Victoria Yang <victoria.j.yang@gmail.com>
1350 Date:   Mon Dec 3 16:38:26 2018 -0500
1351
1352     change insert return to be oid
1353
1354 commit c58b5568cc86dc71d6ac28c92ad547745958838e
1355 Author: Victoria Yang <victoria.j.yang@gmail.com>
1356 Date:   Mon Dec 3 16:32:12 2018 -0500
1357
1358     fixed test so inc/dec is postfix
1359
1360 commit a3b6f32c8442944ce42033ecf135a8d7358bb1dd
1361 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1362 Date:   Mon Dec 3 15:48:06 2018 -0500
1363
1364     get concatenation working
1365
1366 commit 93f3902bc67b0946baceb8655ae1fb867f066c1a
1367 Merge:  f8376de 3d0351e
1368 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1369 Date:   Mon Dec 3 15:18:48 2018 -0500
1370
1371     fix merge conflicts
1372
1373 commit f8376dea35daef27931eb775cb45769c1143e3b7
1374 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1375 Date:   Mon Dec 3 15:14:14 2018 -0500
1376
1377     add bool_to_string built-in
1378
1379 commit 3d0351e9a30433ae323d112e1b780cbd1679066d
1380 Author: Victoria Yang <victoria.j.yang@gmail.com>
1381 Date:   Mon Dec 3 00:47:32 2018 -0500
1382
1383     update codegen to work with sast
1384
1385 commit 998511f3a39510908c1e3675abe6191faece98f1
1386 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1387 Date:   Mon Dec 3 00:36:35 2018 -0500
1388
1389     fix inc and dec
1390
1391 commit a83568780a96545530e4b1dc14c6ac218ec3e5b1
1392 Author: Victoria Yang <victoria.j.yang@gmail.com>
1393 Date:   Sun Dec 2 23:42:01 2018 -0500
1394
1395     revert
```

```
1396
1397 commit e470159b451688f8c7a7f420a308503981dec16f
1398 Author: Victoria Yang <victoria.j.yang@gmail.com>
1399 Date: Sun Dec 2 23:36:04 2018 -0500
1400
1401     made config clean
1402
1403 commit f83a55b16090cc74e66512208e7857f980788401
1404 Author: Victoria Yang <victoria.j.yang@gmail.com>
1405 Date: Sun Dec 2 23:29:09 2018 -0500
1406
1407     fixed config block
1408
1409 commit d5d7964b3e4a3c562bf0ca4cbd1b05bc1640f779
1410 Author: Victoria Yang <victoria.j.yang@gmail.com>
1411 Date: Sun Dec 2 22:56:44 2018 -0500
1412
1413     working on config
1414
1415 commit 7f818017d307337d82a3c572d741c4d1f9c30cc0
1416 Author: Ashley An <ashley.an@columbia.edu>
1417 Date: Sun Dec 2 20:46:54 2018 -0500
1418
1419     starting testing 3D arrays
1420
1421 commit 4705d0c7b2ee53aa443ddc6e244dcc4bff2ac2df
1422 Merge: 958dbdb 7df597d
1423 Author: Ashley An <ashley.an@columbia.edu>
1424 Date: Sun Dec 2 20:12:10 2018 -0500
1425
1426     Merge branch 'master' of https://github.com/soybean/PLT-f18
1427
1428 commit 958dbdbac8411864adab5d9b31bcfd64605c779d
1429 Author: Ashley An <ashley.an@columbia.edu>
1430 Date: Sun Dec 2 20:12:07 2018 -0500
1431
1432     fixed compareLists method
1433
1434 commit 7df597d9dc682128d78f26f3939fd1af5ccee7c8
1435 Merge: a5fb40a 9f989ae
1436 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1437 Date: Sun Dec 2 20:07:40 2018 -0500
1438
1439     fix merge conflicts
1440
1441 commit a5fb40a073f0e373dbe086776f114f4c5ebce673
1442 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1443 Date: Sun Dec 2 20:06:48 2018 -0500
1444
1445     get increment and decrement working
1446
1447 commit 9f989ae9d24a1c00083209c6d6af40f8e997b693
1448 Merge: 6729dc5 0be4872
1449 Author: Victoria Yang <victoria.j.yang@gmail.com>
1450 Date: Sun Dec 2 19:31:59 2018 -0500
1451
```

```
1452 Merge branch 'master' of https://github.com/soybean/PLT-f18
1453
1454 commit 6729dc52f9a2d3c1fd2e4e13fc27e98b93f8cd06
1455 Author: Victoria Yang <victoria.j.yang@gmail.com>
1456 Date: Sun Dec 2 19:31:47 2018 -0500
1457
1458 fixed dollar sign in end
1459
1460 commit 0be487229986b8a22e27044ce223d710101c5ca8
1461 Author: Christine Hsu <christine.hhsu@gmail.com>
1462 Date: Sun Dec 2 18:54:02 2018 -0500
1463
1464 errors caused by merge conflict
1465
1466 commit bbce643c0bc66c6572d2b0f42d2662952427ea22
1467 Merge: 42363f5 2b35569
1468 Author: Christine Hsu <christine.hhsu@gmail.com>
1469 Date: Sun Dec 2 18:51:29 2018 -0500
1470
1471 Merge pull request #11 from soybean/assignbranch
1472
1473 assignbranch: 1d array literals working
1474
1475 commit 2b355693ac6d516e25a587770ddd0506172876dd
1476 Merge: bffcba6 42363f5
1477 Author: Christine Hsu <christine.hhsu@gmail.com>
1478 Date: Sun Dec 2 18:50:52 2018 -0500
1479
1480 Merge branch 'master' into assignbranch
1481
1482 commit bffcba6312e6449a914da044ff8874ad94a01a08
1483 Author: Christine Hsu <christine.hhsu@gmail.com>
1484 Date: Sun Dec 2 18:49:00 2018 -0500
1485
1486 add array_test, works with 1D arrays
1487
1488 commit 6f175c9b99aaca1fa4c7249e45c5dc5344e51e8
1489 Author: Christine Hsu <christine.hhsu@gmail.com>
1490 Date: Sun Dec 2 18:44:06 2018 -0500
1491
1492 fixed locals in codegen
1493
1494 commit 42363f5017bdc15bc22d7dcf46436dbbac02311b
1495 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1496 Date: Sun Dec 2 16:40:11 2018 -0500
1497
1498 add increment and decrement
1499
1500 commit 1366da86174305701b1a8cb24100d62c3e4f16fa
1501 Author: Christine Hsu <christine.hhsu@gmail.com>
1502 Date: Sat Dec 1 23:49:44 2018 -0500
1503
1504 attempting to fix locals in loop/end
1505
1506 commit 0fb4c78d61cc60dd2c3d0eb35bcad81a974809f4
1507 Author: Christine Hsu <christine.hhsu@gmail.com>
```

1508 Date: Sat Dec 1 17:56:46 2018 -0500  
1509  
1510 fix scanner quotes and other small fixes  
1511  
1512 commit 4a4867b1e37c8530cb4e61b683d6d5a93f5060d5  
1513 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1514 Date: Sat Dec 1 17:47:53 2018 -0500  
1515  
1516 rly hope empty array assign works  
1517  
1518 commit eb9b798f0d0c123290a6bb53d699c44e4cea0fec  
1519 Author: Ashley An <ashley.an@columbia.edu>  
1520 Date: Sat Dec 1 16:57:59 2018 -0500  
1521  
1522 now can create nested arrays with linked lists  
1523  
1524 commit 136f327c224142326a06c40c60b1e310347e55f0  
1525 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1526 Date: Fri Nov 30 16:20:59 2018 -0500  
1527  
1528 FIXED EMPTY PROBLEM?????  
1529  
1530 commit 842c179c8e67740010cbc13fd59f4b02b17bb188  
1531 Author: Ashley An <ashley.an@columbia.edu>  
1532 Date: Fri Nov 30 14:11:23 2018 -0500  
1533  
1534 pushed index0f test in main but it's not working  
1535  
1536 commit 49d2f846768620a72e881e48d74c4780548d831b  
1537 Merge: 89ed76d 7051fe2  
1538 Author: Ashley An <ashley.an@columbia.edu>  
1539 Date: Fri Nov 30 14:09:29 2018 -0500  
1540  
1541 Merge branch 'master' of https://github.com/soybean/PLT-f18  
1542  
1543 commit 89ed76dd639bbb4c4bab2f238089c3ba72ca70f6  
1544 Author: Ashley An <ashley.an@columbia.edu>  
1545 Date: Fri Nov 30 14:09:23 2018 -0500  
1546  
1547 added ability to initialize and populate a nested array  
1548  
1549 commit a72d977a11920d8af5d051d2f94aa3bdd7c351ed  
1550 Author: Ashley An <ashley.an@columbia.edu>  
1551 Date: Fri Nov 30 13:33:02 2018 -0500  
1552  
1553 fixed code to not take in pointers to positions in list  
1554  
1555 commit cfab24d4fcf5c8a98723a8fd0d17eb3117a4602a  
1556 Author: Ashley An <ashley.an@columbia.edu>  
1557 Date: Fri Nov 30 13:18:45 2018 -0500  
1558  
1559 create copy of working arrays so far  
1560  
1561 commit 59f8fe9193c572b62ab41437529fc3d75fdc1d1f  
1562 Author: Ashley An <ashley.an@columbia.edu>  
1563 Date: Fri Nov 30 13:14:42 2018 -0500

```
1564
1565     made it so you give actual values not pointers
1566
1567 commit 54196dcb3a310f508d43f9f1ba221697d03c268b
1568 Merge: a5bccbc 5f0d948
1569 Author: Ashley An <ashley.an@columbia.edu>
1570 Date:   Fri Nov 30 12:59:49 2018 -0500
1571
1572     Merge branch 'master' of https://github.com/soybean/PLT-f18
1573
1574 commit a5bccbc4701f687dc4f26020148daf7ca7a58251
1575 Author: Ashley An <ashley.an@columbia.edu>
1576 Date:   Fri Nov 30 12:59:45 2018 -0500
1577
1578     got rid of global variable
1579
1580 commit 7051fe21e52c9576889aa32ccdb851dbfe37501f
1581 Author: Victoria Yang <victoria.j.yang@gmail.com>
1582 Date:   Fri Nov 30 12:55:43 2018 -0500
1583
1584     changed literal test from fail to pass
1585
1586 commit 3c0a0f9d576d83e8fdb31683f00120e8adf3ab34
1587 Merge: 7085f44 a72d977
1588 Author: Victoria Yang <victoria.j.yang@gmail.com>
1589 Date:   Fri Nov 30 12:54:05 2018 -0500
1590
1591     Merge branch 'master' of https://github.com/soybean/PLT-f18
1592
1593 commit 7085f447ba9263fcd725c8b3f5540fb175e07d0
1594 Author: Victoria Yang <victoria.j.yang@gmail.com>
1595 Date:   Fri Nov 30 12:51:36 2018 -0500
1596
1597     fixed not equal in arraylit
1598
1599 commit 5f0d948cb79c3d75ccf06d139db11e0b54689173
1600 Author: Victoria Yang <victoria.j.yang@gmail.com>
1601 Date:   Fri Nov 30 11:34:32 2018 -0500
1602
1603     added tests and cleaned up code
1604
1605 commit 131106371a19f3b4ed4188033fd68d3d450d59d4
1606 Author: Ashley An <ashley.an@columbia.edu>
1607 Date:   Thu Nov 29 22:27:09 2018 -0500
1608
1609     fixed make all
1610
1611 commit 8f39826475c46890f893937443caa761fad5e3d0
1612 Author: Ashley An <ashley.an@columbia.edu>
1613 Date:   Thu Nov 29 22:25:43 2018 -0500
1614
1615     edited main so there are no memory errors
1616
1617 commit f2425927ce589da9a21974d8675dea404a5cf65b
1618 Author: Victoria Yang <victoria.j.yang@gmail.com>
1619 Date:   Thu Nov 29 16:47:47 2018 -0500
```

1620  
1621 refactor enhanced for  
1622  
1623 commit db1c2c434a792b51b11adab9b7d51afbf3485193  
1624 Merge: 7802bf1 7cc49bf  
1625 Author: Ashley An <ashley.an@columbia.edu>  
1626 Date: Thu Nov 29 15:20:32 2018 -0500  
1627  
1628 Merge branch 'master' of https://github.com/soybean/PLT-f18  
1629  
1630 commit 7802bf196612121c5a924ed160289c43e5b025dc  
1631 Author: Ashley An <ashley.an@columbia.edu>  
1632 Date: Thu Nov 29 15:20:26 2018 -0500  
1633  
1634 tested that you can create nested arrays  
1635  
1636 commit 7cc49bfa752aa356d8a5099744d29497a9e1cb8a  
1637 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1638 Date: Thu Nov 29 11:59:53 2018 -0500  
1639  
1640 fixed syntax error  
1641  
1642 commit e4b6081411bb0c4bbaf375f1db20008867e488f9  
1643 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1644 Date: Thu Nov 29 11:57:12 2018 -0500  
1645  
1646 refactoring call  
1647  
1648 commit efc8f6df905308e1065334bbaf5c4143167d9c90  
1649 Merge: 1c8281c 99538eb  
1650 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1651 Date: Thu Nov 29 11:55:01 2018 -0500  
1652  
1653 Merge branch 'master' of https://github.com/soybean/PLT-f18  
1654  
1655 commit 1c8281c0ba70a4da0eccc4b78069d355c58efaed  
1656 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1657 Date: Thu Nov 29 11:54:42 2018 -0500  
1658  
1659 refactoring  
1660  
1661 commit 2f22a64259d855e7ec45735467e4e1360266f48d  
1662 Merge: 8a21715 f6c5d01  
1663 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1664 Date: Thu Nov 29 11:41:18 2018 -0500  
1665  
1666 Merge branch 'master' of https://github.com/soybean/PLT-f18  
1667  
1668 commit 8a21715302e2cca40924985e6a137f14578adc75  
1669 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1670 Date: Thu Nov 29 11:41:04 2018 -0500  
1671  
1672 revert  
1673  
1674 commit 99538ebb8b285dc80a7a61c20474c962c982e595  
1675 Author: Ashley An <ashley.an@columbia.edu>

1676 Date: Thu Nov 29 00:52:04 2018 -0500  
1677  
1678 minor typo fix  
1679  
1680 commit f6c5d014e3538c88e9060549100e6f419f0de5e4  
1681 Author: Ashley An <ashley.an@columbia.edu>  
1682 Date: Wed Nov 28 21:21:21 2018 -0500  
1683  
1684 created more files to try to implement generic methods for nested arrays  
1685  
1686 commit 8a4d23e1f3fa9b35c7d5677f3d6372905267c602  
1687 Author: Ashley An <ashley.an@columbia.edu>  
1688 Date: Wed Nov 28 21:14:18 2018 -0500  
1689  
1690 made init return pointer to list, made contains and indexOf generic for  
primitives  
1691  
1692 commit c166d3760e8d2e283254d63d562787cbe21e18de  
1693 Merge: 4356679 081d736  
1694 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1695 Date: Wed Nov 28 18:52:21 2018 -0500  
1696  
1697 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
1698  
1699 commit 081d73614bb13ee3483c492acfb8b19247aa0c8f  
1700 Author: Christine Hsu <christine.hhsu@gmail.com>  
1701 Date: Tue Nov 27 23:24:55 2018 -0500  
1702  
1703 strlit quotes, escape chars in scanner  
1704  
1705 commit 435667944f1bf86cba2e70ebdb7c0d73076af6e6  
1706 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1707 Date: Tue Nov 27 18:38:14 2018 -0500  
1708  
1709 Revert "refactoring"  
1710  
1711 This reverts commit 6e01d637372e96b1d2e3b7a520663dadbcbfaccc.  
1712  
1713 commit 6e01d637372e96b1d2e3b7a520663dadbcbfaccc  
1714 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1715 Date: Tue Nov 27 18:33:12 2018 -0500  
1716  
1717 refactoring  
1718  
1719 commit c38fdbab4fd8dd2ce9f210c11fa9dfda99696c37  
1720 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1721 Date: Tue Nov 27 18:27:51 2018 -0500  
1722  
1723 updated error statement  
1724  
1725 commit df794b9bb5a1006fa45ecda89aa099b7eb14d9ee  
1726 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1727 Date: Tue Nov 27 17:25:29 2018 -0500  
1728  
1729 adds prettyprinting  
1730



1731 commit 2d331588f6f6318cfb092c3c9ac33ce512a669d1  
1732 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1733 Date: Tue Nov 27 17:10:44 2018 -0500  
1734  
1735 change arrayderef in loop  
1736  
1737 commit f9f37a92d2c155d9101ba3cd1ba9f773ce3c2220  
1738 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1739 Date: Tue Nov 27 16:14:25 2018 -0500  
1740  
1741 changed deref  
1742  
1743 commit c935b816e004eeae18195a1c85b7da089bbeef88  
1744 Merge: d0e2b6f 4cbb8e0  
1745 Author: Ashley An <ashley.an@columbia.edu>  
1746 Date: Tue Nov 27 00:15:41 2018 -0500  
1747  
1748 Merge branch 'master' of https://github.com/soybean/PLT-f18  
1749  
1750 commit d0e2b6f10ad12b4f4ab66d58339b71774f1f739a  
1751 Author: Ashley An <ashley.an@columbia.edu>  
1752 Date: Tue Nov 27 00:15:32 2018 -0500  
1753  
1754 fixed insert element bug  
1755  
1756 commit 4cbb8e0eb743b24415553a4ed16309abd7e28c43  
1757 Merge: d4e8eb8 99ebebe  
1758 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1759 Date: Mon Nov 26 23:57:39 2018 -0500  
1760  
1761 Merge branch 'master' of https://github.com/soybean/PLT-f18  
1762  
1763 commit d4e8eb805720d9ceb37bc9043252d91ecd116101  
1764 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1765 Date: Mon Nov 26 23:57:27 2018 -0500  
1766  
1767 fix sast and syntax errors in semant  
1768  
1769 commit 99ebebe154628949170180f1a76b8bc67c7b8359  
1770 Merge: 229d460 6d07908  
1771 Author: Ashley An <ashley.an@columbia.edu>  
1772 Date: Mon Nov 26 23:44:57 2018 -0500  
1773  
1774 Merge branch 'master' of https://github.com/soybean/PLT-f18  
1775  
1776 commit 229d460165971b89e1847f7bff403134c2810a40  
1777 Author: Ashley An <ashley.an@columbia.edu>  
1778 Date: Mon Nov 26 23:44:46 2018 -0500  
1779  
1780 added functionalities for boolean arrays  
1781  
1782 commit 6d0790819d2c896158748de69f4318d895080027  
1783 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1784 Date: Mon Nov 26 23:42:01 2018 -0500  
1785  
1786 change ops in begin

```
1787
1788 commit af6e7fc1e711fee9e20c77304cf505c91688b90c
1789 Merge: 8ef6f68 3b53883
1790 Author: Victoria Yang <victoria.j.yang@gmail.com>
1791 Date: Mon Nov 26 23:40:39 2018 -0500
1792
1793 Merge branch 'master' of https://github.com/soybean/PLT-f18
1794
1795 commit 8ef6f6802df2f9fc1a50e1653f24f6988b9d10a1
1796 Author: Victoria Yang <victoria.j.yang@gmail.com>
1797 Date: Mon Nov 26 23:40:16 2018 -0500
1798
1799 changed increment and decrement
1800
1801 commit 3b5388318e10edecd870d7f8e65cd084d15d5451
1802 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1803 Date: Mon Nov 26 23:39:35 2018 -0500
1804
1805 add regex file
1806
1807 commit 2035771724a9a1e2b24aa779ca033a5d13b1e3ef
1808 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1809 Date: Mon Nov 26 23:35:50 2018 -0500
1810
1811 update sast with new binops and uops
1812
1813 commit 5b7c825eea0712f42640b34b2b7239d6df7ed951
1814 Merge: e36fff6 67f1ac7
1815 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1816 Date: Mon Nov 26 23:30:26 2018 -0500
1817
1818 Merge branch 'master' of https://github.com/soybean/PLT-f18
1819
1820 commit e36fff6b63227e1c1d7f8164df1ed0a4822f9346
1821 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1822 Date: Mon Nov 26 23:29:46 2018 -0500
1823
1824 fix wrong binops and uops
1825
1826 commit 67f1ac7e1ea0e34c6e4652a2226df714811429f5
1827 Author: Victoria Yang <victoria.j.yang@gmail.com>
1828 Date: Mon Nov 26 23:27:09 2018 -0500
1829
1830 change ops to expr
1831
1832 commit cee23e85f62917f59d6da276720a5e93cb273157
1833 Merge: 92a67e4 13c980e
1834 Author: Victoria Yang <victoria.j.yang@gmail.com>
1835 Date: Mon Nov 26 23:22:46 2018 -0500
1836
1837 Merge branch 'master' of https://github.com/soybean/PLT-f18
1838
1839 commit 92a67e43fe1a7992fd9e5d3d6377c99a39fdc3bd
1840 Author: Victoria Yang <victoria.j.yang@gmail.com>
1841 Date: Mon Nov 26 23:22:29 2018 -0500
1842
```

1843 change strcat to expression  
1844  
1845 commit 13c980e58592d38d97176ea62d26fdc0e0a14694  
1846 Author: Christine Hsu <christine.hhsu@gmail.com>  
1847 Date: Mon Nov 26 23:22:00 2018 -0500  
1848  
1849 remove builder from string format  
1850  
1851 commit bdeddd5eb0ee770de688844ac623691db6b6811c  
1852 Merge: 51b4ab8 028982c  
1853 Author: Christine Hsu <christine.hhsu@gmail.com>  
1854 Date: Mon Nov 26 23:19:14 2018 -0500  
1855  
1856 Merge branch 'master' of https://github.com/soybean/PLT-f18  
1857  
1858 commit 51b4ab8f17cc2e0916f6b8a2f215d260b74e0533  
1859 Author: Christine Hsu <christine.hhsu@gmail.com>  
1860 Date: Mon Nov 26 23:18:49 2018 -0500  
1861  
1862 add binop and unop to loop and end  
1863  
1864 commit 028982c7588b7cb6582c47571d7aadbfc4eaad36  
1865 Merge: cef698a 7e28721  
1866 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1867 Date: Mon Nov 26 23:16:22 2018 -0500  
1868  
1869 xMerge branch 'master' of https://github.com/soybean/PLT-f18  
1870  
1871 commit cef698a20c92173db7d513fe2693cbbbc9c2f14d1  
1872 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1873 Date: Mon Nov 26 23:16:03 2018 -0500  
1874  
1875 cleaned up code and let \$ not be called in begin  
1876  
1877 commit 7e2872130fff19d1bec91b44ea8988c71e538f31  
1878 Merge: fd37e12 e7c8bb7  
1879 Author: Ashley An <ashley.an@columbia.edu>  
1880 Date: Mon Nov 26 23:09:46 2018 -0500  
1881  
1882 Merge branch 'master' of https://github.com/soybean/PLT-f18  
1883  
1884 commit fd37e12c619d452bda4bfbef7783fc9736c45df6  
1885 Author: Ashley An <ashley.an@columbia.edu>  
1886 Date: Mon Nov 26 23:09:36 2018 -0500  
1887  
1888 added functionalities for string arrays  
1889  
1890 commit e7c8bb7572117707651b6eed5ea8e8e0d25a63b2  
1891 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
1892 Date: Mon Nov 26 23:00:47 2018 -0500  
1893  
1894 add gitignore  
1895  
1896 commit 9b5d56256965b95dc8b9e1467ff7c26240e41d48  
1897 Merge: e2f0bd0 40c3b3f  
1898 Author: Melanie Sawyer <melaniensawyer@gmail.com>

```
1899 Date: Mon Nov 26 22:58:12 2018 -0500
1900
1901 Merge branch 'master' of https://github.com/soybean/PLT-f18
1902
1903 commit e2f0bd0424fef306ed10297adc8ff5f6a2e81a38
1904 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1905 Date: Mon Nov 26 22:57:39 2018 -0500
1906
1907 get $ working
1908
1909 commit 40c3b3f1bf95f3b5f9ed58dc726235eb77c75b82
1910 Author: Christine Hsu <christine.hhsu@gmail.com>
1911 Date: Mon Nov 26 20:54:48 2018 -0500
1912
1913 add for, if, while to codegen
1914
1915 commit 29cea60bd717f44ffa67451963179ca6ac55d3ab
1916 Author: Victoria Yang <victoria.j.yang@gmail.com>
1917 Date: Mon Nov 26 19:16:23 2018 -0500
1918
1919 cleaned up deref
1920
1921 commit 07ab23d6fe31c8c87492e11576984b93abe7a7b0
1922 Author: Victoria Yang <victoria.j.yang@gmail.com>
1923 Date: Mon Nov 26 16:14:58 2018 -0500
1924
1925 fix return types for insert and delete
1926
1927 commit a7625fcd272c37211411f497a1287c89fc631fe6
1928 Author: Victoria Yang <victoria.j.yang@gmail.com>
1929 Date: Mon Nov 26 00:25:07 2018 -0500
1930
1931 update access in loop
1932
1933 commit 4cfe18b560f6735e9066576ff402b172140fe643
1934 Author: Victoria Yang <victoria.j.yang@gmail.com>
1935 Date: Mon Nov 26 00:24:16 2018 -0500
1936
1937 rewrite $
1938
1939 commit a52a4fb184e92e38aa7f51af7ef40973eff7988e
1940 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1941 Date: Mon Nov 26 00:22:21 2018 -0500
1942
1943 add parser
1944
1945 commit 4db4fb665a52b37e02baed1259843f398788181c
1946 Author: Melanie Sawyer <melaniensawyer@gmail.com>
1947 Date: Mon Nov 26 00:16:58 2018 -0500
1948
1949 add access as expr
1950
1951 commit acd6a9d07e865368d4a8b0ec4521a2f407984f11
1952 Author: Victoria Yang <victoria.j.yang@gmail.com>
1953 Date: Sun Nov 25 23:29:55 2018 -0500
1954
```

1955 fixed  
1956  
1957 commit 33a8970b2c61da67339950b01317d957f69f9f28  
1958 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1959 Date: Sun Nov 25 23:27:48 2018 -0500  
1960  
1961 add access as expr  
1962  
1963 commit 2c5834de7c398a928c7689acb16747652c30bb73  
1964 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1965 Date: Sun Nov 25 22:01:32 2018 -0500  
1966  
1967 add delete to loop  
1968  
1969 commit 877eb733243749302b75060069915ff42510c091  
1970 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1971 Date: Sun Nov 25 22:00:47 2018 -0500  
1972  
1973 syntax error  
1974  
1975 commit 251f4c91d6bd66c292e0d67cbb5c3a8a4fe81f9a  
1976 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1977 Date: Sun Nov 25 21:59:22 2018 -0500  
1978  
1979 prelim delete  
1980  
1981 commit bd25e6d88d236f7ea8edfd3dd252fbafbb4267f4  
1982 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1983 Date: Sun Nov 25 21:55:34 2018 -0500  
1984  
1985 temp ty  
1986  
1987 commit a82a11ed48b80b0066d9988199b11a943a171e25  
1988 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1989 Date: Sun Nov 25 21:53:53 2018 -0500  
1990  
1991 clean up  
1992  
1993 commit a3dc9504519890f85a10c8d3d7995f708c5f504d  
1994 Author: Victoria Yang <victoria.j.yang@gmail.com>  
1995 Date: Sun Nov 25 21:50:59 2018 -0500  
1996  
1997 cleaning up unused arguments  
1998  
1999 commit 0ec0d495d7fd0770dd42bd08ad0cf4274bedf7ad  
2000 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2001 Date: Sun Nov 25 21:48:02 2018 -0500  
2002  
2003 added insert to loop  
2004  
2005 commit 1e5cca68f3b88f1e00a46a0d26ce1ccc3938074d  
2006 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2007 Date: Sun Nov 25 21:46:43 2018 -0500  
2008  
2009 fixed insert  
2010

```
2011 | commit 4601946dfaa747299f2c49f98eb4e9dd2c83efc5
2012 | Author: Victoria Yang <victoria.j.yang@gmail.com>
2013 | Date: Sun Nov 25 21:33:54 2018 -0500
2014 |
2015 |     wrote insert
2016 |
2017 | commit 95ee1663dc8229aacdce9ef33834a6bee5a2d367
2018 | Author: Victoria Yang <victoria.j.yang@gmail.com>
2019 | Date: Sun Nov 25 21:30:42 2018 -0500
2020 |
2021 |     WIP insert
2022 |
2023 | commit 54c14e4770c9108f2cf2e993c3f4e54da9f3f6f5
2024 | Author: Victoria Yang <victoria.j.yang@gmail.com>
2025 | Date: Sun Nov 25 21:20:11 2018 -0500
2026 |
2027 |     remove access as unop
2028 |
2029 | commit 3f37b75c7a9959f3bfaeffc322364f52e0ed96d7
2030 | Author: Victoria Yang <victoria.j.yang@gmail.com>
2031 | Date: Sun Nov 25 20:00:14 2018 -0500
2032 |
2033 |     change arraylit in begin
2034 |
2035 | commit b5035672401e7636a8f35807f671e78e42bc72f1
2036 | Author: Victoria Yang <victoria.j.yang@gmail.com>
2037 | Date: Sun Nov 25 19:58:26 2018 -0500
2038 |
2039 |     checking array of different types
2040 |
2041 | commit fac88dbebde940a68f78844654c1a6a8ee2f5d50
2042 | Author: Victoria Yang <victoria.j.yang@gmail.com>
2043 | Date: Sun Nov 25 18:09:26 2018 -0500
2044 |
2045 |     update arraylit in begin
2046 |
2047 | commit 0a737c43aa975c3f4171e21d7110c5f48580c043
2048 | Author: Victoria Yang <victoria.j.yang@gmail.com>
2049 | Date: Sun Nov 25 18:06:04 2018 -0500
2050 |
2051 |     trying to fix type checking in arraylit
2052 |
2053 | commit 500f710f01a91fb3aac852e13e4485ad6658bce6
2054 | Author: Christine Hsu <christine.hhsu@gmail.com>
2055 | Date: Sun Nov 25 11:46:32 2018 -0500
2056 |
2057 |     add default values to rs, fs
2058 |
2059 | commit bb84ac3aa51228fd8284859f15c0b7e2a4b8be8e
2060 | Author: Christine Hsu <christine.hhsu@gmail.com>
2061 | Date: Sun Nov 25 09:02:04 2018 -0500
2062 |
2063 |     add rs, fs assign to codegen
2064 |
2065 | commit 71f407d9b3db81b2f51a3cb59efe60ceefb04703
2066 | Merge: 90806df 015fe16
```

2067 Author: Christine Hsu <christine.hhsu@gmail.com>  
2068 Date: Sun Nov 25 08:59:31 2018 -0500  
2069  
2070 Merge branch 'configbranch'  
2071  
2072 commit 015fe16a103d8463e522b49bd0af877df28f975b  
2073 Author: Christine Hsu <christine.hhsu@gmail.com>  
2074 Date: Sun Nov 25 00:49:39 2018 -0500  
2075  
2076 add rs, fsassign to codegen  
2077  
2078 commit 90806dfb43e2060c7b9423be47f2c3d6bf3b45cc  
2079 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2080 Date: Sun Nov 25 00:09:18 2018 -0500  
2081  
2082 update arraylit  
2083  
2084 commit 3c07c6302d65a6f911d164a8851d4dcbf994bf3c  
2085 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2086 Date: Sun Nov 25 00:08:04 2018 -0500  
2087  
2088 fixed nested arrays type failure  
2089  
2090 commit f46bcc761466bacdd9410a02433bc95904961694  
2091 Author: Ashley An <ashley.an@columbia.edu>  
2092 Date: Sun Nov 25 00:00:30 2018 -0500  
2093  
2094 cleaned up code  
2095  
2096 commit 5d658eeb9536ab2a7c8584f8a73a350e7aa83879  
2097 Author: Christine Hsu <christine.hhsu@gmail.com>  
2098 Date: Sat Nov 24 23:50:44 2018 -0500  
2099  
2100 trying to add config, modified ast  
2101  
2102 commit 61c0d3909f96189aafd49ab51a0d7fbcc0b27ee3  
2103 Merge: e70b8ae 8bcfe77  
2104 Author: Ashley An <ashley.an@columbia.edu>  
2105 Date: Sat Nov 24 23:42:47 2018 -0500  
2106  
2107 Merge branch 'master' of https://github.com/soybean/PLT-f18  
2108  
2109 commit e70b8ae2824e1541fc32037515455de693498808  
2110 Author: Ashley An <ashley.an@columbia.edu>  
2111 Date: Sat Nov 24 23:42:36 2018 -0500  
2112  
2113 fixed some array tests  
2114  
2115 commit 8bcfe779da04070903e108817baa3fe1f088964d  
2116 Author: Christine Hsu <christine.hhsu@gmail.com>  
2117 Date: Fri Nov 23 21:23:27 2018 -0500  
2118  
2119 remove Rgx from expr  
2120  
2121 commit 60b8e76d6364ac9e703c0fece08ddb156eece955  
2122 Author: Ashley An <ashley.an@columbia.edu>

2123 Date: Fri Nov 23 17:15:55 2018 -0500  
2124  
2125 edited array implementation to be in one file and added array built-in  
functions  
2126  
2127 commit fda1ffb2da8c23cf3647359cac286b7689f0dbbb  
2128 Author: Ashley An <ashley.an@columbia.edu>  
2129 Date: Fri Nov 23 14:16:05 2018 -0500  
2130  
2131 created our own array data structure in C  
2132  
2133 commit adcbd583d6e0f3764e0526a5f88a14ef7a161f98  
2134 Author: Christine Hsu <christine.hhsu@gmail.com>  
2135 Date: Fri Nov 23 10:47:34 2018 -0500  
2136  
2137 test script remove input.txt command arg  
2138  
2139 commit ef75916d79abb112a290fd958e081fb4aa805a0b  
2140 Author: Christine Hsu <christine.hhsu@gmail.com>  
2141 Date: Fri Nov 23 10:46:32 2018 -0500  
2142  
2143 remove -f and input file name from test script  
2144  
2145 commit 7c0b36c4bd109d9c301df514caea2dad2870c181  
2146 Author: Ashley An <ashley.an@columbia.edu>  
2147 Date: Fri Nov 23 09:33:06 2018 -0500  
2148  
2149 added test that checks if functions can call other user-defined functions  
2150  
2151 commit e8c3290dcdb9fbd3ac11d2932cfcbad5bb2f5ea2  
2152 Author: Christine Hsu <christine.hhsu@gmail.com>  
2153 Date: Thu Nov 22 23:10:17 2018 -0500  
2154  
2155 add assign to codegen  
2156  
2157 commit f8e0585f51894f820f41bddf709e8409beda470d  
2158 Author: Christine Hsu <christine.hhsu@gmail.com>  
2159 Date: Thu Nov 22 10:17:01 2018 -0500  
2160  
2161 refactor loop and end blocks  
2162  
2163 commit 36345bd405c6fab97cc8cf9ed0d7d28057a6ba5a  
2164 Author: Christine Hsu <christine.hhsu@gmail.com>  
2165 Date: Thu Nov 22 10:08:36 2018 -0500  
2166  
2167 add locals to loop, end  
2168  
2169 commit accc673a8fd312ac0eb6175e4a2a3236895b42b0  
2170 Merge: c19f691 f7d11d0  
2171 Author: Christine Hsu <christine.hhsu@gmail.com>  
2172 Date: Wed Nov 21 20:49:23 2018 -0500  
2173  
2174 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
2175  
2176 commit c19f6914d542ce6148549b6ed9c28d5fca41ff61  
2177 Author: Christine Hsu <christine.hhsu@gmail.com>



```

2178 Date:   Wed Nov 21 20:49:03 2018 -0500
2179
2180     codegen is now structured correctly
2181
2182 commit f7d11d064211c9b36338e1be7b79485c217f3e0c
2183 Author: Melanie Sawyer <melaniensawyer@gmail.com>
2184 Date:   Wed Nov 21 17:12:29 2018 -0500
2185
2186     add line parameter to loop function
2187
2188 commit 21145b8a9e6c6506d44ace534bb57c53712237c6
2189 Author: Christine Hsu <christine.hhsu@gmail.com>
2190 Date:   Wed Nov 21 16:56:49 2018 -0500
2191
2192     add int_func test
2193
2194 commit dcf6db54884860998a0eb5495ccf514127112780
2195 Author: Christine Hsu <christine.hhsu@gmail.com>
2196 Date:   Wed Nov 21 15:51:08 2018 -0500
2197
2198     begin block works for some cases, need to test more
2199
2200 commit c4bb4e6b30ca388f356981a71685fd8ae36cf92d
2201 Author: Christine Hsu <christine.hhsu@gmail.com>
2202 Date:   Wed Nov 21 09:42:41 2018 -0500
2203
2204     begin block error
2205
2206 commit eee34d7c152c1b54695cc8039d4ba3b91048a240
2207 Merge: 62ba5f7 bcddef4
2208 Author: Melanie Sawyer <melaniensawyer@gmail.com>
2209 Date:   Wed Nov 21 12:52:21 2018 -0500
2210
2211     Merge branch 'master' of https://github.com/soybean/PLT-f18
2212
2213 commit 62ba5f7b2d4e3da24baa461a100e13bea04e1273
2214 Author: Melanie Sawyer <melaniensawyer@gmail.com>
2215 Date:   Wed Nov 21 12:51:32 2018 -0500
2216
2217     fix string_to_int
2218
2219 commit bcddef42cabf8404cd48d10fee6f38b645bb4cb1
2220 Merge: de0d273 a9b4529
2221 Author: Christine Hsu <christine.hhsu@gmail.com>
2222 Date:   Wed Nov 21 09:03:48 2018 -0500
2223
2224     Merge branch 'master' of https://github.com/soybean/PLT-f18
2225
2226 commit de0d2737997657a6173368c8f10712e9dc1c712b
2227 Author: Christine Hsu <christine.hhsu@gmail.com>
2228 Date:   Wed Nov 21 09:03:30 2018 -0500
2229
2230     restructured codegen to work with structure for loop/end
2231
2232 commit a9b4529fb96cc1f227c2a0ac746edf16f48c3983
2233 Author: Victoria Yang <victoria.j.yang@gmail.com>

```

```
2234 Date: Wed Nov 21 00:01:08 2018 -0500
2235
2236 added arrayderef to begin
2237
2238 commit 927b14fc571a49af6e38eb870896b950947a7683
2239 Author: Victoria Yang <victoria.j.yang@gmail.com>
2240 Date: Tue Nov 20 23:59:51 2018 -0500
2241
2242 recursive arrayderef for arrays
2243
2244 commit 42dc61a92f9edcb74b9db5f18ad48c5f87cd3941
2245 Author: Melanie Sawyer <melaniensawyer@gmail.com>
2246 Date: Tue Nov 20 18:01:42 2018 -0500
2247
2248 remove -f flag and input arg
2249
2250 commit 422f39097ef34b66f587d0d9fd5955917d62b801
2251 Author: Melanie Sawyer <melaniensawyer@gmail.com>
2252 Date: Tue Nov 20 17:57:49 2018 -0500
2253
2254 edit test
2255
2256 commit ae6ba275ea36d8151ec4e2286b7088f448e032aa
2257 Author: Melanie Sawyer <melaniensawyer@gmail.com>
2258 Date: Tue Nov 20 17:42:48 2018 -0500
2259
2260 change test and input file
2261
2262 commit c02acc6dc5f93339806ef173128ccb09e69c54c5
2263 Author: Melanie Sawyer <melaniensawyer@gmail.com>
2264 Date: Tue Nov 20 17:24:54 2018 -0500
2265
2266 add structure file and looping
2267
2268 commit a40aa57195a7fb9d3142769bdff0f6b294aa731e
2269 Author: Melanie Sawyer <melaniensawyer@gmail.com>
2270 Date: Tue Nov 20 14:53:22 2018 -0500
2271
2272 add structure file
2273
2274 commit 1ae1922f30a81dd5309e80f9d71195a428843b1d
2275 Merge: f4cdf47 ea6cf60
2276 Author: Melanie Sawyer <melaniensawyer@gmail.com>
2277 Date: Tue Nov 20 14:12:46 2018 -0500
2278
2279 Merge branch 'master' of https://github.com/soybean/PLT-f18
2280
2281 commit f4cdf4796455636e3cdeb8d2762e4bc044d715e9
2282 Author: Melanie Sawyer <melaniensawyer@gmail.com>
2283 Date: Tue Nov 20 14:12:16 2018 -0500
2284
2285 remove quotation marks from string
2286
2287 commit 6a35dfc0a1f6e91dab6d30e3ee5dbd48633491e5
2288 Author: Melanie Sawyer <melaniensawyer@gmail.com>
2289 Date: Tue Nov 20 00:43:28 2018 -0500
```

```

2290
2291     get linking to work
2292
2293 commit ea6cf60447589a3eb97ff88e6cffe5f3d3773f8c
2294 Author: Victoria Yang <victoria.j.yang@gmail.com>
2295 Date:   Mon Nov 19 23:18:50 2018 -0500
2296
2297     adding arrayderef to begin
2298
2299 commit 364a0321a3a1c322a0b1a8a60225f888f537c3b9
2300 Merge: 9aef102 a272ef9
2301 Author: Victoria Yang <victoria.j.yang@gmail.com>
2302 Date:   Mon Nov 19 23:17:13 2018 -0500
2303
2304     Merge branch 'master' of https://github.com/soybean/PLT-f18
2305
2306 commit 9aef10242748a14a8d258f84de3b6178455f0a3b
2307 Author: Victoria Yang <victoria.j.yang@gmail.com>
2308 Date:   Mon Nov 19 23:16:32 2018 -0500
2309
2310     array deref
2311
2312 commit a272ef9f1bd394e9f6c02195db7e9745e17e92bc
2313 Merge: df3f56b bbc287d
2314 Author: Christine Hsu <christine.hhsu@gmail.com>
2315 Date:   Mon Nov 19 22:11:30 2018 -0500
2316
2317     Merge branch 'master' of https://github.com/soybean/PLT-f18
2318
2319 commit df3f56b5624aed43bf9784f9e76ffc07f0537f0b
2320 Merge: 1ae4dc1 52eb8c3
2321 Author: Christine Hsu <christine.hhsu@gmail.com>
2322 Date:   Mon Nov 19 22:11:28 2018 -0500
2323
2324     merge changes
2325
2326 commit 1ae4dc14d8e37023e9ecdac181831948c8e78931
2327 Author: Christine Hsu <christine.hhsu@gmail.com>
2328 Date:   Mon Nov 19 22:05:46 2018 -0500
2329
2330     edit codegen
2331
2332 commit 52eb8c3c453b0ac6a62e70f437c2ca0a03cdb862
2333 Author: Victoria Yang <victoria.j.yang@gmail.com>
2334 Date:   Mon Nov 19 22:04:32 2018 -0500
2335
2336     put arraylit
2337
2338 commit 0669e9b2e4fddfcc775c369944a4dfbf162ea082
2339 Author: Victoria Yang <victoria.j.yang@gmail.com>
2340 Date:   Mon Nov 19 22:00:47 2018 -0500
2341
2342     fixing arraylit
2343
2344 commit bbc287df6413611bb356ddce67c72ff4ddbab2b9
2345 Merge: beccfe2 52eb8c3

```

2346 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
2347 Date: Mon Nov 19 21:57:29 2018 -0500  
2348  
2349 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
2350  
2351 commit beccfe25d71a9bafd80f471c1431c2681602b636  
2352 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
2353 Date: Mon Nov 19 21:56:53 2018 -0500  
2354  
2355 add string\_to\_int  
2356  
2357 commit 8aa00cc9bab5adc3ccfde057ce00f480aa732d21  
2358 Merge: 280f26f a2b6adf  
2359 Author: Ashley An <ashley.an@columbia.edu>  
2360 Date: Mon Nov 19 19:34:05 2018 -0500  
2361  
2362 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
2363  
2364 commit a2b6adfe577acdcdbd9868adbaf089050f59b9cb  
2365 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2366 Date: Mon Nov 19 19:56:10 2018 -0500  
2367  
2368 added index\_of  
2369  
2370 commit 280f26f5e25606a95a3eb3d51958f09b9827a0ce  
2371 Author: Ashley An <ashley.an@columbia.edu>  
2372 Date: Mon Nov 19 19:33:55 2018 -0500  
2373  
2374 added more tests for contains and index\_of  
2375  
2376 commit b85600cebfc15751f1ae96d2a6047a50ea093d6a  
2377 Merge: dbf0368 d0e41a7  
2378 Author: Ashley An <ashley.an@columbia.edu>  
2379 Date: Mon Nov 19 19:25:26 2018 -0500  
2380  
2381 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
2382  
2383 commit d0e41a76aa328d8b72ed9917a2511e50014f48dc  
2384 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2385 Date: Mon Nov 19 19:47:59 2018 -0500  
2386  
2387 put contains in both blocks  
2388  
2389 commit 4fa00b3c09b8366bb780d18d8bc6371e9a808396  
2390 Merge: b2dccdf 80cc2a5  
2391 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2392 Date: Mon Nov 19 19:27:31 2018 -0500  
2393  
2394 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
2395  
2396 commit b2dccdf718604d7ecb386dd921ee95cdfb25576c  
2397 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2398 Date: Mon Nov 19 19:27:17 2018 -0500  
2399  
2400 wrote contains  
2401

```
2402 commit dbf03688c0c93875a573e292aed6164ef6759c15
2403 Author: Ashley An <ashley.an@columbia.edu>
2404 Date: Mon Nov 19 19:25:15 2018 -0500
2405
2406     added tests for contains and index_of
2407
2408 commit 80cc2a57e2f02c31fa0bde5652b5908b366a00d6
2409 Merge: ab2958a 256207d
2410 Author: Melanie Sawyer <melaniensawyer@gmail.com>
2411 Date: Mon Nov 19 19:19:10 2018 -0500
2412
2413     Merge branch 'master' of https://github.com/soybean/PLT-f18
2414
2415 commit ab2958aef825bf5ac5f2fab74f85da56a4d3f6cf
2416 Author: Melanie Sawyer <melaniensawyer@gmail.com>
2417 Date: Mon Nov 19 19:17:49 2018 -0500
2418
2419     fix codegen
2420
2421 commit 256207d235192d37f67f5cf27a0e981c59ae3023
2422 Author: Ashley An <ashley.an@columbia.edu>
2423 Date: Mon Nov 19 19:15:38 2018 -0500
2424
2425     added int_to_string test
2426
2427 commit 7253cbd1cec0891dfa7291a6cc0239a1aa4515a5
2428 Merge: 1d31eb9 9449603
2429 Author: Christine Hsu <christine.hhsu@gmail.com>
2430 Date: Mon Nov 19 19:11:12 2018 -0500
2431
2432     Merge branch 'master' of https://github.com/soybean/PLT-f18
2433
2434 commit 9449603b5a143c23cc3dd93553de39871a53ed0a
2435 Merge: 3e145df 16cc561
2436 Author: Ashley An <ashley.an@columbia.edu>
2437 Date: Mon Nov 19 18:49:00 2018 -0500
2438
2439     Merge branch 'master' of https://github.com/soybean/PLT-f18
2440
2441 commit 3e145df8589246326b4cea5b705c634334bb6e27
2442 Author: Ashley An <ashley.an@columbia.edu>
2443 Date: Mon Nov 19 18:48:49 2018 -0500
2444
2445     edited dynamic array tests
2446
2447 commit 16cc561389bf20567dd81d3b9f7051abfcb94030
2448 Author: Victoria Yang <victoria.j.yang@gmail.com>
2449 Date: Mon Nov 19 18:46:52 2018 -0500
2450
2451     updated length in loop block
2452
2453 commit 2ea855215a41e07e69c322db2f4ae4ab8d9e92ba
2454 Merge: 7aa8fe8 324732f
2455 Author: Victoria Yang <victoria.j.yang@gmail.com>
2456 Date: Mon Nov 19 18:45:26 2018 -0500
2457
```

```

2458 Merge branch 'master' of https://github.com/soybean/PLT-f18
2459
2460 commit 7aa8fe816ab83bc2b91334d8fd21ba79bb7cb89e
2461 Author: Victoria Yang <victoria.j.yang@gmail.com>
2462 Date: Mon Nov 19 18:44:59 2018 -0500
2463
2464 restructured length and removed wrong tests
2465
2466 commit 1d31eb9745c99b7b1bd02634a29b6038931d480c
2467 Merge: cea2575 324732f
2468 Author: Christine Hsu <christine.hhsu@gmail.com>
2469 Date: Mon Nov 19 18:43:50 2018 -0500
2470
2471 Merge branch 'master' of https://github.com/soybean/PLT-f18
2472
2473 commit 324732f884cbe260aa31d406ad8ac8e35dd376a5
2474 Merge: e36cc34 d6fd3d3
2475 Author: Ashley An <ashley.an@columbia.edu>
2476 Date: Mon Nov 19 18:43:09 2018 -0500
2477
2478 Merge branch 'master' of https://github.com/soybean/PLT-f18
2479
2480 commit e36cc3425db595e4f98f694715feccc5c895a4fc
2481 Author: Ashley An <ashley.an@columbia.edu>
2482 Date: Mon Nov 19 18:42:57 2018 -0500
2483
2484 edited Makefile so it will remove convert.o when running make all
2485
2486 commit d6fd3d3239fadb6eb66471d11f8461e19e649c47
2487 Merge: c2b4686 d2a2086
2488 Author: Victoria Yang <victoria.j.yang@gmail.com>
2489 Date: Mon Nov 19 18:38:25 2018 -0500
2490
2491 Merge branch 'master' of https://github.com/soybean/PLT-f18
2492
2493 commit c2b4686172dba118014a1337749c97e467c0e79f
2494 Author: Victoria Yang <victoria.j.yang@gmail.com>
2495 Date: Mon Nov 19 18:37:49 2018 -0500
2496
2497 fixed syntax error for enhanced for
2498
2499 commit 6643a33268453ef2b9376b01b547dc0339b36b69
2500 Author: Victoria Yang <victoria.j.yang@gmail.com>
2501 Date: Mon Nov 19 18:33:01 2018 -0500
2502
2503 update enhanced for in both places
2504
2505 commit 3891c3d763d6ece91bda2002edf947289cdbc882
2506 Author: Victoria Yang <victoria.j.yang@gmail.com>
2507 Date: Mon Nov 19 18:31:14 2018 -0500
2508
2509 fixed enhanced for loops git add semant.ml
2510
2511 commit 5c588fd5deaa8f042da19c71528d78c9ec834198
2512 Author: Victoria Yang <victoria.j.yang@gmail.com>
2513 Date: Mon Nov 19 18:09:47 2018 -0500

```

2514  
2515 updating both blocks  
2516  
2517 commit cea2575b2b7d9579f505220e60d32e6ad27b410a  
2518 Merge: b0ecd2a e2533b2  
2519 Author: Christine Hsu <christine.hhsu@gmail.com>  
2520 Date: Mon Nov 19 18:07:58 2018 -0500  
2521  
2522 Merge branch 'master' of https://github.com/soybean/PLT-f18  
2523  
2524 commit b0ecd2a667bf2f848c6e599a62076156859f826a  
2525 Merge: 7863a29 27dbb12  
2526 Author: Christine Hsu <christine.hhsu@gmail.com>  
2527 Date: Mon Nov 19 18:07:51 2018 -0500  
2528  
2529 Merge branch 'master' of https://github.com/soybean/PLT-f18  
2530  
2531 commit e2533b28d29e9f3770b6ee9db7034ce3dfbab16c  
2532 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2533 Date: Mon Nov 19 18:07:40 2018 -0500  
2534  
2535 commenting in arraylit  
2536  
2537 commit 7863a293d1da96a8f22469c90b8e2fdf89868dc4  
2538 Author: Christine Hsu <christine.hhsu@gmail.com>  
2539 Date: Mon Nov 19 18:07:40 2018 -0500  
2540  
2541 add wildcard pattern matching to codegen  
2542  
2543 commit 27dbb12c1846248d4a72bd3ee1340b23977a0f1d  
2544 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2545 Date: Mon Nov 19 18:06:59 2018 -0500  
2546  
2547 fixing enhanced for  
2548  
2549 commit e2153498e790ffeae3f2e6ac392672dbca9c5d5e  
2550 Merge: e779dcd a1c4c38  
2551 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
2552 Date: Mon Nov 19 16:50:18 2018 -0500  
2553  
2554 Merge branch 'master' of https://github.com/soybean/PLT-f18  
2555  
2556 commit e779dcd009b60642c8087ebc3f960113ce8d6970  
2557 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
2558 Date: Mon Nov 19 16:49:53 2018 -0500  
2559  
2560 add convert.c  
2561  
2562 commit a1c4c3879724fa7c6fd66a490bd1227c3010f0a4  
2563 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2564 Date: Mon Nov 19 16:49:09 2018 -0500  
2565  
2566 put length in both blocks  
2567  
2568 commit 25d551ab108e9b28a25f7d21fea86caf958495ea  
2569 Author: Victoria Yang <victoria.j.yang@gmail.com>

```
2570 Date: Mon Nov 19 16:47:06 2018 -0500
2571
2572 got length to compile
2573
2574 commit e41551778d6b053f27c2838ded61eb7c91c9d394
2575 Author: Victoria Yang <victoria.j.yang@gmail.com>
2576 Date: Mon Nov 19 16:15:46 2018 -0500
2577
2578 trying to fix length
2579
2580 commit 86b36f38754881f36a03fffd32287db9eda916f1
2581 Merge: a6fa325 eb9beba
2582 Author: Victoria Yang <victoria.j.yang@gmail.com>
2583 Date: Mon Nov 19 12:36:12 2018 -0500
2584
2585 Merge branch 'master' of https://github.com/soybean/PLT-f18
2586
2587 commit a6fa32564c7cffff3758d89384b88e148aec2586
2588 Author: Victoria Yang <victoria.j.yang@gmail.com>
2589 Date: Mon Nov 19 12:35:58 2018 -0500
2590
2591 added tests
2592
2593 commit eb9beba1e21c618692edb500273b9874ab0023c3
2594 Merge: ab77308 43b6cc5
2595 Author: Melanie Sawyer <melaniensawyer@gmail.com>
2596 Date: Mon Nov 19 11:51:26 2018 -0500
2597
2598 add convert to makefile
2599
2600 commit ab7730829c88ceffd92cf671ddf8bd47927c703c
2601 Author: Melanie Sawyer <melaniensawyer@gmail.com>
2602 Date: Mon Nov 19 11:49:54 2018 -0500
2603
2604 dis makefile
2605
2606 commit d2a20865aa4e42dc42c6f91823a638b2e8bbcd9bb
2607 Merge: 5ac9407 6643a33
2608 Author: Ashley An <ashley.an@columbia.edu>
2609 Date: Sun Nov 18 23:10:28 2018 -0500
2610
2611 Merge branch 'master' of https://github.com/soybean/PLT-f18
2612
2613 commit 5ac94078e7ebace57c129946fca78c979b94b3cb
2614 Author: Ashley An <ashley.an@columbia.edu>
2615 Date: Sun Nov 18 23:10:13 2018 -0500
2616
2617 created enhanced for loop fail test and 2D array assign fail test
2618
2619 commit 43b6cc5feec8bb2f9720459f39316575a508d951
2620 Author: Ashley An <ashley.an@columbia.edu>
2621 Date: Sun Nov 18 22:43:58 2018 -0500
2622
2623 added target to Makefile that allows for clearing .out and .err files
2624
2625 commit 3a04c7969c7ff62d53159c8dd92c38a7ab7a8049
```



2626 Author: Ashley An <ashley.an@columbia.edu>  
2627 Date: Sun Nov 18 22:13:05 2018 -0500  
2628  
2629 added more array tests  
2630  
2631 commit 8bbf6937fc9850557a3894019cbf2f3bf825e8c3  
2632 Author: Ashley An <ashley.an@columbia.edu>  
2633 Date: Sun Nov 18 22:02:13 2018 -0500  
2634  
2635 added more void tests and edited regex tests  
2636  
2637 commit c77121ce40443cb700893e5ae764bf1d53402220  
2638 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2639 Date: Sun Nov 18 21:32:14 2018 -0500  
2640  
2641 trying to fix arrays  
2642  
2643 commit b13a89016cd53ff8f9b16363aeab9aebeefcc183  
2644 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2645 Date: Sun Nov 18 20:53:46 2018 -0500  
2646  
2647 commenting out builtin array functions  
2648  
2649 commit 5dcc76160524ebb691dfe31c5208531606b97fbd  
2650 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2651 Date: Sun Nov 18 20:46:57 2018 -0500  
2652  
2653 rewrote length  
2654  
2655 commit c16a243505a63a4de3d84f6c7dc28658472bc471  
2656 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2657 Date: Sun Nov 18 20:43:07 2018 -0500  
2658  
2659 fix empty array problem  
2660  
2661 commit 2783b1bd675563e9990c8446c2cffbead2f16ad6  
2662 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2663 Date: Sun Nov 18 20:40:26 2018 -0500  
2664  
2665 fix length  
2666  
2667 commit 90c82407bfb5a6cb2be42f74f6ad8cd1acbf941f  
2668 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2669 Date: Sun Nov 18 20:16:22 2018 -0500  
2670  
2671 Add length  
2672  
2673 commit e426c4e275ecd5715ef90079b606918d3cb258c1  
2674 Merge: 5ff59c1 76bc3a1  
2675 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
2676 Date: Fri Nov 16 15:44:47 2018 -0500  
2677  
2678 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
2679  
2680 commit 1ef338731908d9b584660fd95c6df90d3087dab0  
2681 Author: Victoria Yang <victoria.j.yang@gmail.com>

2682 Date: Fri Nov 16 13:03:34 2018 -0500  
2683  
2684 temporary stash  
2685  
2686 commit 76bc3a13658bee8fb2b2bdf9d14c45ed0555546c  
2687 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2688 Date: Fri Nov 16 12:58:44 2018 -0500  
2689  
2690 Make things the same  
2691  
2692 commit 5686d6f9ac273d92b83966fe45487f685d183212  
2693 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2694 Date: Fri Nov 16 12:53:41 2018 -0500  
2695  
2696 fixed NF  
2697  
2698 commit c96045e7e68b742fd145b5bc6d1225be7856057b  
2699 Author: Ashley An <ashley.an@columbia.edu>  
2700 Date: Fri Nov 16 12:20:31 2018 -0500  
2701  
2702 changed functions back to length in array tests  
2703  
2704 commit 32894a3b031fd0e6577f30807979756c4e3ab71e  
2705 Author: Ashley An <ashley.an@columbia.edu>  
2706 Date: Fri Nov 16 12:13:49 2018 -0500  
2707  
2708 edited test script to create .out and .err files  
2709  
2710 commit c68129a5d42c2e78d90ec08b476bbb7be01a564b  
2711 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2712 Date: Fri Nov 16 12:13:43 2018 -0500  
2713  
2714 fixing NF  
2715  
2716 commit ca602b25f565b83098fe65ef0cd967c52d3472b1  
2717 Merge: 70b995d 9ec7353  
2718 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2719 Date: Fri Nov 16 12:03:55 2018 -0500  
2720  
2721 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
2722  
2723 commit 70b995d08132b10891959d909fa8179f5855b087  
2724 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2725 Date: Fri Nov 16 12:03:33 2018 -0500  
2726  
2727 fix NF  
2728  
2729 commit 5ff59c1ac581fc323a95664b17fa019ded8f96c0  
2730 Merge: 1bada60 9ec7353  
2731 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
2732 Date: Fri Nov 16 12:01:58 2018 -0500  
2733  
2734 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
2735  
2736 commit 9ec7353d36f51e84337c7231f551f148abf60db2  
2737 Author: Ashley An <ashley.an@columbia.edu>

2738 Date: Fri Nov 16 11:01:41 2018 -0500  
2739  
2740 added RS test  
2741  
2742 commit 378a27e7d88267ae433fb9c27b947f27c9485bbe  
2743 Author: Ashley An <ashley.an@columbia.edu>  
2744 Date: Fri Nov 16 10:59:07 2018 -0500  
2745  
2746 added tests for FS and RS  
2747  
2748 commit 9f31759d3d32a4d6414b7834068bbd9fed1684c2  
2749 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2750 Date: Fri Nov 16 10:34:56 2018 -0500  
2751  
2752 fixing errors in semant  
2753  
2754 commit b8526586f65973d66a9ce7c92c70aa903d334fac  
2755 Merge: ecf7b54 f4c3bfa  
2756 Author: Ashley An <ashley.an@columbia.edu>  
2757 Date: Fri Nov 16 00:19:09 2018 -0500  
2758  
2759 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
2760  
2761 commit ecf7b545e8d4f2b729137318526a10d637d22be4  
2762 Author: Ashley An <ashley.an@columbia.edu>  
2763 Date: Fri Nov 16 00:18:53 2018 -0500  
2764  
2765 added more tests  
2766  
2767 commit f4c3bfaccbed47ee1fcafe7b8a522dd9fd174d0d  
2768 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2769 Date: Thu Nov 15 23:57:46 2018 -0500  
2770  
2771 fixed printing in tests  
2772  
2773 commit 25b174a94cdde9d5791ac1ee6c62e564dfb837d3  
2774 Author: Ashley An <ashley.an@columbia.edu>  
2775 Date: Thu Nov 15 23:33:21 2018 -0500  
2776  
2777 Update pass-fib.bawk  
2778  
2779 commit e29f8e642acf38c610a4e7df329ddb724be019a7  
2780 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2781 Date: Thu Nov 15 23:16:11 2018 -0500  
2782  
2783 fixed enhanced for loop stuff  
2784  
2785 commit 2fb7fed00960d9a3f4fd99b93d6fbbeb5694f8bbd  
2786 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2787 Date: Thu Nov 15 23:13:19 2018 -0500  
2788  
2789 match sast to ast  
2790  
2791 commit 66be0df4a149d99238defba968099d8c7d7f18af  
2792 Merge: 2c06283 7f0bedd  
2793 Author: Victoria Yang <victoria.j.yang@gmail.com>

```

2794 Date: Thu Nov 15 23:10:12 2018 -0500
2795
2796 Merge branch 'master' of https://github.com/soybean/PLT-f18
2797
2798 commit 2c062830e9e3a8b80116175c5aaa6914622e6675
2799 Author: Victoria Yang <victoria.j.yang@gmail.com>
2800 Date: Thu Nov 15 23:09:54 2018 -0500
2801
2802 idk if this works?
2803
2804 commit 7f0bedd355ff78a005ea973ef8bc202e7e8aa9fd
2805 Author: Ashley An <ashley.an@columbia.edu>
2806 Date: Thu Nov 15 22:56:52 2018 -0500
2807
2808 edited bawk.ml
2809
2810 commit 766d6af0e17c0687fc4daeb8db8d0b458ffd1dcf
2811 Merge: 025891e 408f4dc
2812 Author: Ashley An <ashley.an@columbia.edu>
2813 Date: Thu Nov 15 22:55:13 2018 -0500
2814
2815 Merge branch 'master' of https://github.com/soybean/PLT-f18
2816
2817 commit 025891e3c733464f1cf9f0120f4071fecc6f3659
2818 Author: Ashley An <ashley.an@columbia.edu>
2819 Date: Thu Nov 15 22:55:00 2018 -0500
2820
2821 fixed increment and decrement
2822
2823 commit 408f4dc45a218eb35ec0103ab27e78ac0a1e8054
2824 Author: Victoria Yang <victoria.j.yang@gmail.com>
2825 Date: Thu Nov 15 22:38:45 2018 -0500
2826
2827 fixed line 79
2828
2829 commit 3bb199c92aabcd5f4d141b3ea881279d5fc10bb1
2830 Author: Victoria Yang <victoria.j.yang@gmail.com>
2831 Date: Thu Nov 15 22:33:50 2018 -0500
2832
2833 fixed locals
2834
2835 commit fc6812c15c2705c8021617bdda19823562aa48af
2836 Author: Victoria Yang <victoria.j.yang@gmail.com>
2837 Date: Thu Nov 15 22:32:09 2018 -0500
2838
2839 fix
2840
2841 commit 0575c1b3c74c728cc5fdfdb7534ac5c4167fdd7
2842 Merge: 2ac8735 d992cb3
2843 Author: Victoria Yang <victoria.j.yang@gmail.com>
2844 Date: Thu Nov 15 22:29:35 2018 -0500
2845
2846 Merge pull request #10 from soybean/victoriajyang-patch-1
2847
2848 fix
2849

```

2850 commit d992cb3123b2349a88578bfcef6c0c6eb2bbcd8f  
2851 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2852 Date: Thu Nov 15 22:26:44 2018 -0500  
2853  
2854 fix  
2855  
2856 commit 2ac8735a6b51626b051080a43993cd3342fb9157  
2857 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2858 Date: Thu Nov 15 22:23:18 2018 -0500  
2859  
2860 fixing syntax  
2861  
2862 commit 205292703f582576157734798782bb3f8c815202  
2863 Merge: 6550c4f a1b780d  
2864 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2865 Date: Thu Nov 15 22:14:42 2018 -0500  
2866  
2867 Merge pull request #9 from soybean/victoriajyang-patch-1  
2868  
2869 testing  
2870  
2871 commit a1b780d1df44cd4d332f0f3fea80e185d330b79a  
2872 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2873 Date: Thu Nov 15 22:10:29 2018 -0500  
2874  
2875 testing  
2876  
2877 commit 6550c4fad59316461a4ffbfafd56e6804ef54805  
2878 Author: Ashley An <ashley.an@columbia.edu>  
2879 Date: Thu Nov 15 22:05:12 2018 -0500  
2880  
2881 fixed regex tests  
2882  
2883 commit 23bb3dcf15252a492458767b2eaf75537b360334  
2884 Author: Ashley An <ashley.an@columbia.edu>  
2885 Date: Thu Nov 15 21:41:33 2018 -0500  
2886  
2887 Update pass-ops4.bawk  
2888  
2889 commit b0ee8e8acbcd73ff14451e28baeed60a307401c2  
2890 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2891 Date: Thu Nov 15 20:56:29 2018 -0500  
2892  
2893 match sast to ast  
2894  
2895 commit 8dae8d6f1547855ea2e37da6e5dff898cc0dc5fa  
2896 Merge: b8c3936 26942bc  
2897 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2898 Date: Thu Nov 15 20:54:04 2018 -0500  
2899  
2900 Merge branch 'master' of https://github.com/soybean/PLT-f18  
2901  
2902 commit b8c393687473b4fe03a797887f686d20e3b24a77  
2903 Author: Victoria Yang <victoria.j.yang@gmail.com>  
2904 Date: Thu Nov 15 20:52:12 2018 -0500  
2905

```
2906      uncommented sast from bawk
2907
2908 commit 26942bc310ab5981a209e4760f6933fb932622ad
2909 Merge: 58f7fb5 bbe7118
2910 Author: Ashley An <ashley.an@columbia.edu>
2911 Date: Thu Nov 15 19:14:15 2018 -0500
2912
2913      Merge branch 'master' of https://github.com/soybean/PLT-f18
2914
2915 commit 58f7fb553ae412c61e6a5c2b1f582e4a9a3fd502
2916 Author: Ashley An <ashley.an@columbia.edu>
2917 Date: Thu Nov 15 19:14:05 2018 -0500
2918
2919      fixed else stmts
2920
2921 commit bbe7118478793bff353962a87aa04399fe05ed57
2922 Author: Ashley An <ashley.an@columbia.edu>
2923 Date: Thu Nov 15 19:11:37 2018 -0500
2924
2925      Update pass-func3.bawk
2926
2927 commit 8d2d042b803a7ec3085f551ed09b8a6dab2259ae
2928 Author: Ashley An <ashley.an@columbia.edu>
2929 Date: Thu Nov 15 19:03:11 2018 -0500
2930
2931      Update pass-boolarr4.bawk
2932
2933 commit 6422d58e4db7715bafb555ab796da373c0caf1cf
2934 Author: Ashley An <ashley.an@columbia.edu>
2935 Date: Thu Nov 15 18:16:18 2018 -0500
2936
2937      fixed problems with order of statements, vars, functions in parser
2938
2939 commit ec0d36edbf23dafa955280451f7c1a01c171888c
2940 Author: Ashley An <ashley.an@columbia.edu>
2941 Date: Thu Nov 15 18:02:14 2018 -0500
2942
2943      added to pretty printing
2944
2945 commit 4df85e256166881de1779bb65570c81087c54962
2946 Author: Christine Hsu <christine.hhsu@gmail.com>
2947 Date: Thu Nov 15 00:22:16 2018 -0500
2948
2949      creating blocks in codegen works
2950
2951 commit e298917446c00321d3b0f353ca2808d60fac0531
2952 Author: Ashley An <ashley.an@columbia.edu>
2953 Date: Thu Nov 15 00:18:29 2018 -0500
2954
2955      edited bawk.ml
2956
2957 commit aab137a7da6984c3512e7a0a5385291a727aa48a
2958 Merge: 8f8f609 85a594e
2959 Author: Ashley An <ashley.an@columbia.edu>
2960 Date: Thu Nov 15 00:15:33 2018 -0500
2961
```

```

2962 Merge branch 'master' of https://github.com/soybean/PLT-f18
2963
2964 commit 8f8f609bae1399daa2abb1ce43404ccc736991cd
2965 Author: Ashley An <ashley.an@columbia.edu>
2966 Date: Thu Nov 15 00:15:22 2018 -0500
2967
2968     fixed arraysvim parser.mly !
2969
2970 commit 85a594e16eac276fd18cb6993a7ed12092af383d
2971 Author: Victoria Yang <victoria.j.yang@gmail.com>
2972 Date: Thu Nov 15 00:11:44 2018 -0500
2973
2974     fixed pretty printing syntax errors
2975
2976 commit c5438999627f31843145423bedfa1dc26d1c1b57
2977 Author: Victoria Yang <victoria.j.yang@gmail.com>
2978 Date: Thu Nov 15 00:08:25 2018 -0500
2979
2980     Putting pretty-print back in
2981
2982 commit bd8ad2a0ee18b072f8def8e67d42be2ec8d92741
2983 Merge: ca1cbc2 bf80f27
2984 Author: Victoria Yang <victoria.j.yang@gmail.com>
2985 Date: Thu Nov 15 00:05:10 2018 -0500
2986
2987 Merge branch 'master' of https://github.com/soybean/PLT-f18
2988
2989 commit ca1cbc29791fbbf686d9af07098b53cba7b9aa15c
2990 Author: Victoria Yang <victoria.j.yang@gmail.com>
2991 Date: Thu Nov 15 00:04:53 2018 -0500
2992
2993     added block checking
2994
2995 commit bf80f2769d6254d9add5c34a574c0d40db3b438f
2996 Author: Victoria Yang <victoria.j.yang@gmail.com>
2997 Date: Wed Nov 14 23:59:08 2018 -0500
2998
2999     Cleaned up unnecessary comments
3000
3001 commit c6ccc613d3a76cc1f2b0e7e88e687341bb0803f9
3002 Author: Ashley An <ashley.an@columbia.edu>
3003 Date: Wed Nov 14 23:16:49 2018 -0500
3004
3005     added pretty printing for ast except for array types (like int[] or int[][] )
3006
3007 commit 3c252d6abe0baa2068fe0194dbb9d3d5c622a05b
3008 Author: Ashley An <ashley.an@columbia.edu>
3009 Date: Wed Nov 14 21:35:00 2018 -0500
3010
3011     clean up code
3012
3013 commit b3ef0f10f8c00eedfcbb8ee814e71e5a794349a4
3014 Author: Ashley An <ashley.an@columbia.edu>
3015 Date: Wed Nov 14 21:30:26 2018 -0500
3016
3017     cleaned up code

```

3018  
3019 commit a4bc32564118eb6b7647300cb522256257b4e0e6  
3020 Author: Ashley An <ashley.an@columbia.edu>  
3021 Date: Wed Nov 14 21:25:19 2018 -0500  
3022  
3023 cleaned up code  
3024  
3025 commit 96b7ae391d837cd548776cabfdb7623f378f0f06  
3026 Author: Ashley An <ashley.an@columbia.edu>  
3027 Date: Wed Nov 14 21:22:04 2018 -0500  
3028  
3029 more cleanup  
3030  
3031 commit 37a8853908832221ae6f3a34e18accf08b87e4be  
3032 Author: Ashley An <ashley.an@columbia.edu>  
3033 Date: Wed Nov 14 21:20:57 2018 -0500  
3034  
3035 cleaned up code  
3036  
3037 commit 7bb05e4368c3db3d4b4b58eb180141256961f555  
3038 Merge: 3619ed3 78e710a  
3039 Author: Ashley An <ashley.an@columbia.edu>  
3040 Date: Wed Nov 14 21:13:40 2018 -0500  
3041  
3042 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
3043  
3044 commit 3619ed31fd3de10730d1f11668535deb06ecc610  
3045 Author: Ashley An <ashley.an@columbia.edu>  
3046 Date: Wed Nov 14 21:13:28 2018 -0500  
3047  
3048 cleaned up code  
3049  
3050 commit 78e710a975d473172b0343240d289cfaa7e714eb  
3051 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3052 Date: Wed Nov 14 00:54:40 2018 -0500  
3053  
3054 Reorder  
3055  
3056 commit bc8c8b9d544f8ca90b606e4df17b69f9435253d4  
3057 Merge: 48151fa 5d64af6  
3058 Author: Ashley An <ashley.an@columbia.edu>  
3059 Date: Wed Nov 14 00:53:58 2018 -0500  
3060  
3061 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
3062  
3063 commit 48151fad979d4ccfb48f04194c6ef6b4a4f1aff6  
3064 Author: Ashley An <ashley.an@columbia.edu>  
3065 Date: Wed Nov 14 00:53:46 2018 -0500  
3066  
3067 created even more tests  
3068  
3069 commit 5d64af6b47d82660cfadbd070881d796034e106c  
3070 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3071 Date: Wed Nov 14 00:44:06 2018 -0500  
3072  
3073 Remove old comments



3074  
3075 commit 4aace0fb9fea56fadd99a04ecc2d3ab29a6ec6a4  
3076 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3077 Date: Wed Nov 14 00:43:05 2018 -0500  
3078  
3079 added type checking for globals/locals  
3080  
3081 commit 753f874934e48228c1137837d1bd879947e345c0  
3082 Author: Ashley An <ashley.an@columbia.edu>  
3083 Date: Tue Nov 13 23:31:32 2018 -0500  
3084  
3085 modify test-hello.sh to run pass-helloworldloopend.bawk instead  
3086  
3087 commit 65c49c4fdbbc0055da6b3c963c6f2af091167358  
3088 Author: Ashley An <ashley.an@columbia.edu>  
3089 Date: Tue Nov 13 23:23:00 2018 -0500  
3090  
3091 adjusted spacing and indentation of test  
3092  
3093 commit 1bada60a830c2f713792bd646ebe25e8a222bfc2  
3094 Merge: 64dec61 153366a  
3095 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3096 Date: Tue Nov 13 23:22:13 2018 -0500  
3097  
3098 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
3099  
3100 commit 153366a00eb10a55fd6aecc247ed590742af354d  
3101 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3102 Date: Tue Nov 13 23:17:17 2018 -0500  
3103  
3104 testing arraylist  
3105  
3106 commit fd99f2a97abac146908d5a45f3013c38254eeaf7  
3107 Merge: 1d14059 abe5513  
3108 Author: Christine Hsu <christine.hhsu@gmail.com>  
3109 Date: Tue Nov 13 23:17:15 2018 -0500  
3110  
3111 Merge pull request #8 from soybean/christines-branch  
3112  
3113 includes codegen that supports print in loop and end blocks  
3114  
3115 commit 1d14059f28810930aff0f1bf15406fefbd621e52  
3116 Merge: 5876d99 38a812f  
3117 Author: Ashley An <ashley.an@columbia.edu>  
3118 Date: Tue Nov 13 23:10:10 2018 -0500  
3119  
3120 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
3121  
3122 commit 5876d99b84be061f206bd1a6332098be2018840b  
3123 Author: Ashley An <ashley.an@columbia.edu>  
3124 Date: Tue Nov 13 23:09:59 2018 -0500  
3125  
3126 added more tests  
3127  
3128 commit 9c6c7a70c5b649450cbdd007408bc36e42eef938  
3129 Author: Ashley An <ashley.an@columbia.edu>

3130 Date: Tue Nov 13 23:00:03 2018 -0500  
3131  
3132 added some tests  
3133  
3134 commit d55c391c67d346f513a9594dae824cb4d235f75d  
3135 Merge: 02c0fa5 df0476d  
3136 Author: Ashley An <ashley.an@columbia.edu>  
3137 Date: Tue Nov 13 22:58:37 2018 -0500  
3138  
3139 Merge branch 'master' of https://github.com/soybean/PLT-f18  
3140  
3141 commit 02c0fa591eaff6b46678e3e5240da8d0fc631771  
3142 Author: Ashley An <ashley.an@columbia.edu>  
3143 Date: Tue Nov 13 22:58:23 2018 -0500  
3144  
3145 edited sast for return  
3146  
3147 commit 38a812f0dc72c61ddc82eb1455cce57b1808ab90  
3148 Merge: 088074c 3d3693b  
3149 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3150 Date: Tue Nov 13 22:43:35 2018 -0500  
3151  
3152 Merge branch 'master' of https://github.com/soybean/PLT-f18  
3153  
3154 commit 3d3693b5f56fcf9b3a2a52ea216e09a0c33ae299  
3155 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3156 Date: Tue Nov 13 23:01:32 2018 -0500  
3157  
3158 Added noexpr  
3159  
3160 commit 088074c5eab879551f426b9d48a8ce453dfc6856  
3161 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3162 Date: Tue Nov 13 22:41:39 2018 -0500  
3163  
3164 add changes  
3165  
3166 commit df0476d8b4b696dc70db3ab84da6da77a51688a5  
3167 Author: Christine Hsu <christine.hhsu@gmail.com>  
3168 Date: Tue Nov 13 22:31:55 2018 -0500  
3169  
3170 add make clean to test hello script  
3171  
3172 commit abe551358e00443aa69d79c8dee347eb65a8121d  
3173 Author: Christine Hsu <christine.hhsu@gmail.com>  
3174 Date: Tue Nov 13 22:05:11 2018 -0500  
3175  
3176 add comments to codegen  
3177  
3178 commit 31626f01333aee55fe07285d7eaa93c3c90c7032  
3179 Author: Christine Hsu <christine.hhsu@gmail.com>  
3180 Date: Tue Nov 13 21:57:34 2018 -0500  
3181  
3182 got print working in loop and end blocks  
3183  
3184 commit 94965e8c94cdb61bd52f13736b23194a321c6a72  
3185 Author: Ashley An <ashley.an@columbia.edu>

3186 Date: Tue Nov 13 18:39:37 2018 -0500  
3187  
3188 added ability to write return ;  
3189  
3190 commit d6b9b7f8788449ce0a5d041a2738d2393ad69db8  
3191 Author: Ashley An <ashley.an@columbia.edu>  
3192 Date: Tue Nov 13 18:23:48 2018 -0500  
3193  
3194 modified some tests  
3195  
3196 commit ca0a491b05c584a4b93d2e117f78d0c02cc69cf2  
3197 Author: Ashley An <ashley.an@columbia.edu>  
3198 Date: Tue Nov 13 18:00:38 2018 -0500  
3199  
3200 modified tests to be in end block rather than loop block so testing is easier  
3201  
3202 commit 7e323d8f0526451c9902f127c1df709aa2474bab  
3203 Merge: 87912b0 b0a11de  
3204 Author: Ashley An <ashley.an@columbia.edu>  
3205 Date: Tue Nov 13 17:59:29 2018 -0500  
3206  
3207 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
3208  
3209 commit 87912b0b2aa6ffe039972a7e27ef13fdec1b89fe  
3210 Author: Ashley An <ashley.an@columbia.edu>  
3211 Date: Tue Nov 13 17:59:16 2018 -0500  
3212  
3213 modified and added some more tests  
3214  
3215 commit b0a11dea6ddfc2b0e51854db319c9f4782a9135f  
3216 Author: Christine Hsu <christine.hhsu@gmail.com>  
3217 Date: Tue Nov 13 11:15:50 2018 -0500  
3218  
3219 add make to test hello script  
3220  
3221 commit e17cc59e5528f200259f9a949163df186c9748c9  
3222 Author: Christine Hsu <christine.hhsu@gmail.com>  
3223 Date: Tue Nov 13 08:47:52 2018 -0500  
3224  
3225 add test script for hello world  
3226  
3227 commit 2aad7d77b591a56e521d19d98257e9f64291790  
3228 Author: Christine Hsu <christine.hhsu@gmail.com>  
3229 Date: Tue Nov 13 08:43:19 2018 -0500  
3230  
3231 modified codegen  
3232  
3233 commit a119b50867ba2df843fe1c2928b53420d9d44219  
3234 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3235 Date: Mon Nov 12 23:44:47 2018 -0500  
3236  
3237 Update sast to match ast  
3238  
3239 commit 66deb6be0baba1613fb1662b15f8be2ed69b1b0a  
3240 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3241 Date: Mon Nov 12 23:26:53 2018 -0500

3242  
3243 version of semant with NO ERRORS  
3244  
3245 commit fa7971fde838407306515204540377a8d46ec3c8  
3246 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3247 Date: Mon Nov 12 23:22:14 2018 -0500  
3248  
3249 assign update  
3250  
3251 commit 5c007c1054d73c4276a6f4fadcd2311afa76f4b44  
3252 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3253 Date: Mon Nov 12 23:12:57 2018 -0500  
3254  
3255 Syntax error  
3256  
3257 commit 0d25831317f84d4bc5bb3e4ab9c382c037cdeaea  
3258 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3259 Date: Mon Nov 12 22:01:37 2018 -0500  
3260  
3261 Commenting out pretty printing  
3262  
3263 commit 9f653986f5527465dc248b167254a61350067e84  
3264 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3265 Date: Mon Nov 12 21:51:34 2018 -0500  
3266  
3267 assign is back to expr  
3268  
3269 commit e308c711b2643b7a4e6663079fe840e420597760  
3270 Author: Ashley An <ashley.an@columbia.edu>  
3271 Date: Mon Nov 12 21:28:28 2018 -0500  
3272  
3273 remove unnecessary test case  
3274  
3275 commit 35ce5f314bd62dd9799d9a86c2ecad2d01b6f87b  
3276 Author: Christine Hsu <christine.hhsu@gmail.com>  
3277 Date: Mon Nov 12 21:10:38 2018 -0500  
3278  
3279 add print test for loop and end blocks  
3280  
3281 commit a3adcfdcb170fa6d186846441b8b291c17b87f89  
3282 Author: Christine Hsu <christine.hhsu@gmail.com>  
3283 Date: Mon Nov 12 19:05:47 2018 -0500  
3284  
3285 testing branch commits  
3286  
3287 commit 64dec61a71730db43df3b87847312f50a9157c35  
3288 Merge: 1c57896 35ce5f3  
3289 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3290 Date: Mon Nov 12 02:07:47 2018 -0500  
3291  
3292 Merge branch 'christines-branch' of <https://github.com/soybean/PLT-f18>  
3293  
3294 commit 1c57896471133e8f749bd11a6a8f370dd270f6e4  
3295 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3296 Date: Mon Nov 12 02:02:51 2018 -0500  
3297

```
3298     revert to id assign expr
3299
3300 commit 9577fec6f73757518e95a44ab81e6f05d06aba1f
3301 Merge: a79b58e 953bd28
3302 Author: Melanie Sawyer <melaniensawyer@gmail.com>
3303 Date:   Mon Nov 12 01:59:02 2018 -0500
3304
3305     Merge branch 'master' of https://github.com/soybean/PLT-f18
3306
3307 commit 953bd28cd3c863f50bb93c21e33a5650b3f00267
3308 Merge: 6e8afce e023701
3309 Author: Christine Hsu <christine.hhsu@gmail.com>
3310 Date:   Mon Nov 12 21:04:44 2018 -0500
3311
3312     Merge branch 'master' of https://github.com/soybean/PLT-f18
3313
3314 commit 6e8afced52e5307385d8b36de9a8a5a646e90afa
3315 Author: Christine Hsu <christine.hhsu@gmail.com>
3316 Date:   Mon Nov 12 21:04:37 2018 -0500
3317
3318     fixed bugs in sast
3319
3320 commit e0237012f5ccbc60892ed389d17ea16606f09c19
3321 Author: Victoria Yang <victoria.j.yang@gmail.com>
3322 Date:   Mon Nov 12 21:03:32 2018 -0500
3323
3324     Changing assign back
3325
3326 commit c89af10bf58843070ffe782497fdd00048181273
3327 Merge: cd42e82 8d6b98e
3328 Author: Christine Hsu <christine.hhsu@gmail.com>
3329 Date:   Mon Nov 12 21:01:27 2018 -0500
3330
3331     Merge branch 'master' of https://github.com/soybean/PLT-f18
3332
3333 commit cd42e8242b180322e1b757d5894692448d1ad16f
3334 Author: Christine Hsu <christine.hhsu@gmail.com>
3335 Date:   Mon Nov 12 21:00:27 2018 -0500
3336
3337     add fail assign test
3338
3339 commit 8d6b98e3655d69497fb646a1bcb33686655a4f8a
3340 Author: Victoria Yang <victoria.j.yang@gmail.com>
3341 Date:   Mon Nov 12 20:57:43 2018 -0500
3342
3343     changed assign to ID
3344
3345 commit f298d7e6a27742681b1317404147e04720a7e8be
3346 Author: Victoria Yang <victoria.j.yang@gmail.com>
3347 Date:   Mon Nov 12 20:51:18 2018 -0500
3348
3349     no keyword checking lol
3350
3351 commit cd13522c0f98bb88521c26a5d94de47bc4056c1b
3352 Author: Victoria Yang <victoria.j.yang@gmail.com>
3353 Date:   Mon Nov 12 20:46:41 2018 -0500
```

3354  
3355       reverting assign  
3356  
3357 commit 5670d61be3524cb2265bd4d51b1574efc0b161f2  
3358 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3359 Date:    Mon Nov 12 20:34:24 2018 -0500  
3360  
3361       fix assign  
3362  
3363 commit ef1fab20e19460ec6b54e2cdb2f5037854b388a2  
3364 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3365 Date:    Mon Nov 12 20:19:11 2018 -0500  
3366  
3367       added different types for contains and index\_of  
3368  
3369 commit 99e45b20336c0a18c2f0fbefc624121a0a83c36d  
3370 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3371 Date:    Mon Nov 12 20:16:54 2018 -0500  
3372  
3373       testing contains overload  
3374  
3375 commit 8eb3715e79ffc8af7d8fa91bd0d395d997edbf42  
3376 Author: Ashley An <ashley.an@columbia.edu>  
3377 Date:    Mon Nov 12 14:16:13 2018 -0500  
3378  
3379       helloworldbegin is supposed to be a fail test not a pass test  
3380  
3381 commit a79b58e92904ea1bd42e6b96713fbb45942f481d  
3382 Merge: 489d5b6 5b39b96  
3383 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3384 Date:    Mon Nov 12 00:54:29 2018 -0500  
3385  
3386       Merge branch 'master' of https://github.com/soybean/PLT-f18  
3387  
3388 commit fbbe2f5a7e833c658726e7c089b7b1b7b1e7c353  
3389 Merge: 85aac4e 5b39b96  
3390 Author: Ashley An <ashley.an@columbia.edu>  
3391 Date:    Mon Nov 12 00:52:38 2018 -0500  
3392  
3393       Merge branch 'master' of https://github.com/soybean/PLT-f18  
3394  
3395 commit 489d5b665236080c50611197fabdbf82c1df8c3d  
3396 Merge: 2d76716 9b0165c  
3397 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3398 Date:    Mon Nov 12 00:51:04 2018 -0500  
3399  
3400       Merge branch 'master' of https://github.com/soybean/PLT-f18  
3401  
3402 commit 85aac4e982e2e86acf88823298eb834002304d15  
3403 Merge: 25864d3 9b0165c  
3404 Author: Ashley An <ashley.an@columbia.edu>  
3405 Date:    Mon Nov 12 00:50:28 2018 -0500  
3406  
3407       Merge branch 'master' of https://github.com/soybean/PLT-f18  
3408  
3409 commit 5b39b96ca23b30e8a3311d77ff08c782eeb62875

3410 Merge: b8e9582 9b0165c  
3411 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3412 Date: Mon Nov 12 00:49:49 2018 -0500  
3413  
3414 Merge branch 'master' of https://github.com/soybean/PLT-f18  
3415  
3416 commit 9b0165c2690eed8467d74a09aad57410a97c094  
3417 Author: Christine Hsu <christine.hhsu@gmail.com>  
3418 Date: Mon Nov 12 00:49:18 2018 -0500  
3419  
3420 test Christine's email linking to git  
3421  
3422 commit 25864d338a2dc73000dd0bfaf7e9a7ab5ef39be4  
3423 Merge: d832c31 4823add  
3424 Author: Ashley An <ashley.an@columbia.edu>  
3425 Date: Mon Nov 12 00:49:05 2018 -0500  
3426  
3427 Merge branch 'master' of https://github.com/soybean/PLT-f18  
3428  
3429 commit d832c3198444271d10b35bef804f46793222cb84  
3430 Author: Ashley An <ashley.an@columbia.edu>  
3431 Date: Mon Nov 12 00:48:49 2018 -0500  
3432  
3433 added basic test for dollar and basic test for nf  
3434  
3435 commit b8e95823e53c9804af2e15f2f4a4178794fab48  
3436 Merge: 45e0f7f 4823add  
3437 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3438 Date: Mon Nov 12 00:48:17 2018 -0500  
3439  
3440 Merge branch 'master' of https://github.com/soybean/PLT-f18  
3441  
3442 commit 45e0f7f252b5d9e7d00de4671da14f7c39c562bf  
3443 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3444 Date: Mon Nov 12 00:46:51 2018 -0500  
3445  
3446 made change  
3447  
3448 commit 4823addc22d5cb7398f0eda4282d30105ead5a54  
3449 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>  
3450 Date: Mon Nov 12 00:45:58 2018 -0500  
3451  
3452 added more patterns to expr in codegen  
3453  
3454 commit 2d76716be73798c6ef9ca047c72fdce817f7fe38  
3455 Merge: a4d8f9b b89ef51  
3456 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3457 Date: Mon Nov 12 00:44:59 2018 -0500  
3458  
3459 Merge branch 'master' of https://github.com/soybean/PLT-f18  
3460  
3461 commit 9632aaf623f1eac7423c3aa5b00e55702fba6a5b  
3462 Merge: b9006fb 0bb1ca2  
3463 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3464 Date: Mon Nov 12 00:44:42 2018 -0500  
3465

```

3466 Merge branch 'master' of https://github.com/soybean/PLT-f18
3467
3468 commit b89ef51279ab479be8cfd998cb788835f58d508b
3469 Author: Ashley An <ashley.an@columbia.edu>
3470 Date: Mon Nov 12 00:44:35 2018 -0500
3471
3472 added makefile documentation
3473
3474 commit a4d8f9bfa036b6797ab8862f98f4cae12e2729bb
3475 Merge: ea0b8ce 0bb1ca2
3476 Author: Melanie Sawyer <melaniensawyer@gmail.com>
3477 Date: Mon Nov 12 00:43:38 2018 -0500
3478
3479 Merge branch 'master' of https://github.com/soybean/PLT-f18
3480
3481 commit 0bb1ca21bd9e884e1b7eef5bb9c9cf9b974da0d0
3482 Author: Ashley An <ashley.an@columbia.edu>
3483 Date: Mon Nov 12 00:42:51 2018 -0500
3484
3485 edited Makefile
3486
3487 commit b9006fb31c1039ef81f65160b51323161a77c664
3488 Merge: f2ab387 78688ab
3489 Author: Victoria Yang <victoria.j.yang@gmail.com>
3490 Date: Mon Nov 12 00:28:19 2018 -0500
3491
3492 Merge branch 'master' of https://github.com/soybean/PLT-f18
3493
3494 commit 78688abd8efa39651bba19006e17e0334c1232e4
3495 Author: Ashley An <ashley.an@columbia.edu>
3496 Date: Mon Nov 12 00:25:54 2018 -0500
3497
3498 created complete Makefile
3499
3500 commit f9c13f58a51dcbb2cca56fde045482dfedbe8745
3501 Merge: 49f02b5 43448eb
3502 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>
3503 Date: Mon Nov 12 00:14:46 2018 -0500
3504
3505 Merge branch 'master' of https://github.com/soybean/PLT-f18
3506
3507 commit 43448ebd6b6932ed747f3951bc2d9f406b7e9625
3508 Author: Melanie Sawyer <melaniensawyer@gmail.com>
3509 Date: Mon Nov 12 00:13:17 2018 -0500
3510
3511 add dollar and NF to scanner
3512
3513 commit f2ab3876a864a492e6baee9dcd15bed35c66ef86
3514 Author: Victoria Yang <victoria.j.yang@gmail.com>
3515 Date: Mon Nov 12 00:11:36 2018 -0500
3516
3517 fix
3518
3519 commit b030197e55f4aa80f155c0a4e856acb694935002
3520 Author: Victoria Yang <victoria.j.yang@gmail.com>
3521 Date: Sun Nov 11 23:47:35 2018 -0500

```



```

3522
3523     fixed syntax error
3524
3525 commit 4b2cd8e7690b0f51eddec30b304c3b004689186c
3526 Author: Victoria Yang <victoria.j.yang@gmail.com>
3527 Date:    Sun Nov 11 23:41:35 2018 -0500
3528
3529     Revert
3530
3531 commit fd481f0007957ce2c029604077d08f748231690b
3532 Author: Victoria Yang <victoria.j.yang@gmail.com>
3533 Date:    Sun Nov 11 23:39:14 2018 -0500
3534
3535     Update sast.ml
3536
3537 commit 49f02b58b96c4b539d892ad2e009a9d3e790553d
3538 Merge: 9c4925e 8683378
3539 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>
3540 Date:    Sun Nov 11 23:33:48 2018 -0500
3541
3542     Merge branch 'master' of https://github.com/soybean/PLT-f18
3543
3544 commit 8683378c472b8e52e81049472acbd9ec029c49fe
3545 Author: Ashley An <ashley.an@columbia.edu>
3546 Date:    Sun Nov 11 23:32:15 2018 -0500
3547
3548     Note:  is supposed to be Ashley An (error from using git in VM)
3549
3550 commit 9c4925eff397559053e0c3601e053f2d987aaace
3551 Merge: 8a10797 fb465df
3552 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>
3553 Date:    Sun Nov 11 23:31:07 2018 -0500
3554
3555     Merge branch 'master' of https://github.com/soybean/PLT-f18
3556
3557 commit fb465df3a1c0acec00ab87185534fc25d647dd56
3558 Author: Ashley An <ashley.an@columbia.edu>
3559 Date:    Sun Nov 11 23:27:38 2018 -0500
3560
3561     removed line to test because name Alfred Aho is coming up instead of name
        Ashley An
3562
3563 commit 8a107976740903bc0e4c41d1acb84967a2b40ebf
3564 Merge: f2e8269 f33ef97
3565 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>
3566 Date:    Sun Nov 11 23:23:06 2018 -0500
3567
3568     Merge branch 'master' of https://github.com/soybean/PLT-f18
3569
3570 commit f33ef977dafee702a1b2e49c118a67872b13d141
3571 Author: Alfred Aho <Ashley An>
3572 Date:    Sun Nov 11 23:22:54 2018 -0500
3573
3574     added line to input.txt
3575
3576 commit f2e82699803e21cdfd0a2571576d0885e80ea12b

```

3577 Merge: 092c929 dd3b216  
3578 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>  
3579 Date: Sun Nov 11 23:20:09 2018 -0500  
3580  
3581 Merge branch 'master' of https://github.com/soybean/PLT-f18  
3582  
3583 commit dd3b216d8eebaa23898524bd751d94cce808402f  
3584 Merge: 2a47441 2f248fe  
3585 Author: Alfred Aho <Ashley An>  
3586 Date: Sun Nov 11 23:19:13 2018 -0500  
3587  
3588 Merge branch 'master' of https://github.com/soybean/PLT-f18  
3589  
3590 commit 2a47441321dd6b661b3b8664ae79a8f7222abf9b  
3591 Author: Alfred Aho <Ashley An>  
3592 Date: Sun Nov 11 23:18:52 2018 -0500  
3593  
3594 created tests for nested arrays  
3595  
3596 commit 2f248fe44ae9df36847df6174c198b556cbc70e5  
3597 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3598 Date: Sun Nov 11 23:17:33 2018 -0500  
3599  
3600 trying new builtin decls  
3601  
3602 commit 092c929d521665955d5a39fb6394da77b9a71727  
3603 Merge: 5f45972 609c8ab  
3604 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>  
3605 Date: Sun Nov 11 23:04:44 2018 -0500  
3606  
3607 Merge branch 'master' of https://github.com/soybean/PLT-f18  
3608  
3609 commit 609c8ab563fab9534b77688540bd26f0718f2509  
3610 Merge: d3953b0 61229b5  
3611 Author: Ashley An <ashley.an@columbia.edu>  
3612 Date: Sun Nov 11 23:03:14 2018 -0500  
3613  
3614 Merge pull request #7 from soybean/ashleys-branch  
3615  
3616 Ashleys branch  
3617  
3618 commit 61229b58866cbb02d0d62ea2ae2902278291554a  
3619 Merge: ec163db d3953b0  
3620 Author: Ashley An <ashley.an@columbia.edu>  
3621 Date: Sun Nov 11 23:02:54 2018 -0500  
3622  
3623 Merge branch 'master' into ashleys-branch  
3624  
3625 commit ec163dbb51048dbbe0082acf980e330f38595bfb  
3626 Author: Alfred Aho <Ashley An>  
3627 Date: Sun Nov 11 23:00:53 2018 -0500  
3628  
3629 edited bawk.ml  
3630  
3631 commit 3701ce5db31de8deea874649bac8c1fd6417d353  
3632 Merge: 6d24f20 7383211

3633 Author: Alfred Aho <Ashley An>  
3634 Date: Sun Nov 11 22:58:55 2018 -0500  
3635  
3636 merge  
3637  
3638 commit 6d24f20476ce564784d5db2bf9f733ffef7a2ece  
3639 Author: Alfred Aho <Ashley An>  
3640 Date: Sun Nov 11 22:56:03 2018 -0500  
3641  
3642 implemented nested arrays  
3643  
3644 commit d3953b090e42a46b58e338711dbc50fa13fd8023  
3645 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3646 Date: Sun Nov 11 22:47:33 2018 -0500  
3647  
3648 Remove map  
3649  
3650 commit 2eafa3866b6bcd81826f4530c74685cf50de70e1  
3651 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3652 Date: Sun Nov 11 22:46:59 2018 -0500  
3653  
3654 Removed maps  
3655  
3656 commit b3629016630d5f0f6a0d5904fb276005ee69c13c  
3657 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3658 Date: Sun Nov 11 22:46:09 2018 -0500  
3659  
3660 Update expr  
3661  
3662 commit 03d76710bd4c446636df4023cba5ed3f306492b9  
3663 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3664 Date: Sun Nov 11 22:44:36 2018 -0500  
3665  
3666 Removed map  
3667  
3668 commit ea0b8ce84d52c80b1cd042b7b9330d182480695d  
3669 Merge: 66ec1a6 d2bb3f0  
3670 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3671 Date: Sun Nov 11 22:36:58 2018 -0500  
3672  
3673 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
3674  
3675 commit 66ec1a606a1409daeba7303f54d44486136189c1  
3676 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3677 Date: Sun Nov 11 22:36:05 2018 -0500  
3678  
3679 pass in input file to translate  
3680  
3681 commit d2bb3f07a73ce2b4e2304ede3578f69d20db8c06  
3682 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3683 Date: Sun Nov 11 22:35:43 2018 -0500  
3684  
3685 fixed mistake  
3686  
3687 commit 956a8f12af38327714f204924bed00bda32ea2df  
3688 Author: Victoria Yang <victoria.j.yang@gmail.com>

3689 Date: Sun Nov 11 22:28:51 2018 -0500  
3690  
3691 added begin test  
3692  
3693 commit 4a0092711d8bfb9448db6c73357954c1820f263f  
3694 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3695 Date: Sun Nov 11 22:27:53 2018 -0500  
3696  
3697 Added loop test.  
3698  
3699 commit ad18ed964981f6682e8c49a516eed5f457b03ea2  
3700 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3701 Date: Sun Nov 11 22:24:24 2018 -0500  
3702  
3703 Added keywords as "built in functions"  
3704  
3705 commit 5f45972757e4a29c8c262eb62aadd4448bd19c2a  
3706 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>  
3707 Date: Sun Nov 11 22:22:27 2018 -0500  
3708  
3709 first attempt in many to organize codegen for blocks  
3710  
3711 commit 27ea0cc7b475d85080cfe7c09b9340b7f3702043  
3712 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3713 Date: Sun Nov 11 22:03:25 2018 -0500  
3714  
3715 add stuff  
3716  
3717 commit 7383211be99af5c189b46f4bb96ee9e92c09b075  
3718 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>  
3719 Date: Sun Nov 11 15:41:08 2018 -0500  
3720  
3721 cleaned codegen  
3722  
3723 commit 99010b5d63b1e4ee66a608d12d85b5453cfbfacf  
3724 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>  
3725 Date: Sun Nov 11 15:30:27 2018 -0500  
3726  
3727 fixed strings and strlits for codegen  
3728  
3729 commit 48f32e15a673a9b7578defae6b1ea554bd49ad95  
3730 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>  
3731 Date: Sun Nov 11 09:50:33 2018 -0500  
3732  
3733 fixed stmt list in codegen  
3734  
3735 commit d3f58f97f2e32f6191cd0119bd5e0419acbc8bb7  
3736 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3737 Date: Sun Nov 11 02:04:50 2018 -0500  
3738  
3739 add input file command line arg and makefile  
3740  
3741 commit 98a096b34514ed38405c39b9a5a1ba03b73a9ae5  
3742 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3743 Date: Sun Nov 11 01:58:56 2018 -0500  
3744

```
3745     add stupid hello world
3746
3747 commit 531f74c53d254715734622597c6e7981b9eb53d9
3748 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>
3749 Date:   Sun Nov 11 01:14:23 2018 -0500
3750
3751     codegen produces LLVM IR, doesn't call function in block
3752
3753 commit 70fa1656e0e28696a390ac61497e90263700eb84
3754 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>
3755 Date:   Sat Nov 10 22:35:43 2018 -0500
3756
3757     added main function to codegen
3758
3759 commit 4b1cc1325683dcfed6f7b7092c644852ee4ff969
3760 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>
3761 Date:   Sat Nov 10 20:27:12 2018 -0500
3762
3763     broken codegen.ml
3764
3765 commit 3a6726ecbde01e82138d73ed5b928fbe006d0ba5
3766 Merge: 3eea685 5d8f2f6
3767 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>
3768 Date:   Sat Nov 10 20:09:58 2018 -0500
3769
3770     fix merge conflicts
3771
3772 commit 3eea685b83c9dda9f51cda87852eba7515a30f21
3773 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>
3774 Date:   Sat Nov 10 19:42:34 2018 -0500
3775
3776     re-add codegen.ml
3777
3778 commit 5d8f2f613dd9a37c423cea13e3bb8e71e9486266
3779 Author: Christine Hsu <christine.hhsu@gmail.com>
3780 Date:   Sat Nov 10 19:39:47 2018 -0500
3781
3782     comment out string_of_sprogram
3783
3784 commit 74c81549ef5aa15ff7022d4395a7d30d6592df8e
3785 Author: Christine Hsu <christine.hhsu@gmail.com>
3786 Date:   Sat Nov 10 19:37:21 2018 -0500
3787
3788     add flags to bawk.ml
3789
3790 commit 8f7eb0e8cca0d45efac2b38154cb46874087eab4
3791 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>
3792 Date:   Sat Nov 10 19:36:35 2018 -0500
3793
3794     add bawk.ml and sast.ml
3795
3796 commit 547a0e4ab7386b2e9785e6d98cff95b9cf59a377
3797 Author: Christine Hsu <christine.hhsu@gmail.com>
3798 Date:   Sat Nov 10 19:28:35 2018 -0500
3799
3800     codegen builds but is definitely very wong
```

```
3801
3802 commit 1f035b70cc5992c26987c8a611f627e6efaf953f
3803 Author: Victoria Yang <victoria.j.yang@gmail.com>
3804 Date: Sat Nov 10 18:31:08 2018 -0500
3805
3806 Remove float
3807
3808 commit fa4efaf1d07499bf508a36b9ae0846bf84c9d0a2
3809 Author: Victoria Yang <victoria.j.yang@gmail.com>
3810 Date: Sat Nov 10 17:32:33 2018 -0500
3811
3812 Cleaned up comments
3813
3814 commit 0cede7240d81c80f2d70294f6e1b370f265425ea
3815 Author: Victoria Yang <victoria.j.yang@gmail.com>
3816 Date: Sat Nov 10 17:07:02 2018 -0500
3817
3818 Added enhanced for
3819
3820 commit 692cc1e081c6f75d8afcd709920a756666b57691
3821 Author: Victoria Yang <victoria.j.yang@gmail.com>
3822 Date: Sat Nov 10 16:55:40 2018 -0500
3823
3824 Added exprs
3825
3826 commit c73b47d29c17a4a7d550bd6e9d439ab61d3dabf8
3827 Author: Victoria Yang <victoria.j.yang@gmail.com>
3828 Date: Sat Nov 10 16:50:46 2018 -0500
3829
3830 Update binops
3831
3832 commit 99a30cea1789b9b9f34d63a16f84f78ee8dba3da
3833 Author: Victoria Yang <victoria.j.yang@gmail.com>
3834 Date: Sat Nov 10 16:23:46 2018 -0500
3835
3836 Added our unary operators
3837
3838 commit 8c97e454de28e2d2817d2cf2ca07e06a2f148065
3839 Merge: 3d37bfa 92529fa
3840 Author: Ashley An <ashley.an@columbia.edu>
3841 Date: Sat Nov 10 16:21:16 2018 -0500
3842
3843 Merge branch 'master' of https://github.com/soybean/PLT-f18
3844
3845 commit 3d37bfaab8a2091e759e79b374b9ac2786791283
3846 Author: Ashley An <ashley.an@columbia.edu>
3847 Date: Sat Nov 10 16:21:12 2018 -0500
3848
3849 got rid of ability to make a scope
3850
3851 commit 92529fa2f7850a48a1404a285239cb94aa90b624
3852 Author: Christine Hsu <christine.hhsu@gmail.com>
3853 Date: Sat Nov 10 16:15:52 2018 -0500
3854
3855 updated codegen, does not work still
3856
```

```
3857 | commit 94c895ac0b9bdc9354e502958926ff639ac823c0
3858 | Merge: 35b1420 a67b6b6
3859 | Author: Ashley An <ashley.an@columbia.edu>
3860 | Date: Sat Nov 10 16:12:42 2018 -0500
3861 |
3862 | Merge branch 'master' of https://github.com/soybean/PLT-f18
3863 |
3864 | commit 35b14209de1ee703d643464234ea2c639ff27db2
3865 | Author: Ashley An <ashley.an@columbia.edu>
3866 | Date: Sat Nov 10 16:12:38 2018 -0500
3867 |
3868 | array get element and array assign element should not be unary operators
3869 |
3870 | commit a67b6b69e53605fed8baf285ceaf8b755b699d33
3871 | Author: Christine Hsu <christine.hhsu@gmail.com>
3872 | Date: Sat Nov 10 14:20:53 2018 -0500
3873 |
3874 | preliminary codegen for hello world
3875 |
3876 | commit c60b174d9636cf28c89cf6fb27a95338e65db1f3
3877 | Merge: 3345110 07b3f87
3878 | Author: Ashley An <ashley.an@columbia.edu>
3879 | Date: Fri Nov 9 21:13:20 2018 -0500
3880 |
3881 | Merge branch 'master' of https://github.com/soybean/PLT-f18
3882 |
3883 | commit 334511083497d28dea61bc06d90bc7676de5e6f7
3884 | Author: Ashley An <ashley.an@columbia.edu>
3885 | Date: Fri Nov 9 21:13:14 2018 -0500
3886 |
3887 | created hello world test
3888 |
3889 | commit 07b3f87e993f7a21dc7d253fe0e42f78c1d99afa
3890 | Author: Christine Hsu <christine.hhsu@gmail.com>
3891 | Date: Fri Nov 9 21:12:11 2018 -0500
3892 |
3893 | change HI to PASS in sast
3894 |
3895 | commit 6efca2f506316217601214d88f9cb7b1aae3d327
3896 | Author: Christine Hsu <christine.hhsu@gmail.com>
3897 | Date: Fri Nov 9 20:33:06 2018 -0500
3898 |
3899 | update SAST to match current AST
3900 |
3901 | commit 567055834f26082d4397404a8d9b7381c5259c5f
3902 | Author: Christine Hsu <christine.hhsu@gmail.com>
3903 | Date: Fri Nov 9 20:25:42 2018 -0500
3904 |
3905 | update bawk.ml
3906 |
3907 | commit 920b29f317197776e821bad080dcc9b62df184ae
3908 | Author: Christine Hsu <christine.hhsu@gmail.com>
3909 | Date: Fri Nov 9 20:23:57 2018 -0500
3910 |
3911 | update semant.ml
3912 |
```

3913 commit e4f6316e6df55fc2ac7f5bb4bd6f484f83f2cf9a  
3914 Author: Ashley An <ashley.an@columbia.edu>  
3915 Date: Fri Nov 9 20:16:04 2018 -0500  
3916  
3917 changed HI to PASS  
3918  
3919 commit 6b065c770cf80974628437b98c1a2e2c1f9667a1  
3920 Author: Ashley An <ashley.an@columbia.edu>  
3921 Date: Fri Nov 9 20:14:09 2018 -0500  
3922  
3923 added new line for readability for tests  
3924  
3925 commit 5609e9ccc6fb1b284a3a33c959ee9cb822c7248d  
3926 Author: Ashley An <ashley.an@columbia.edu>  
3927 Date: Fri Nov 9 20:12:54 2018 -0500  
3928  
3929 fixed regex reading  
3930  
3931 commit 75f468177f87093b44b59a8b5815ac3de22535b3  
3932 Author: Ashley An <ashley.an@columbia.edu>  
3933 Date: Fri Nov 9 20:08:20 2018 -0500  
3934  
3935 got rid of duplicates  
3936  
3937 commit ba00d24d31c2b24aebde39347dc27a067ae48712  
3938 Author: Ashley An <ashley.an@columbia.edu>  
3939 Date: Fri Nov 9 19:53:02 2018 -0500  
3940  
3941 added more map tests  
3942  
3943 commit af35a6f841f7b5bf14e7fd23c0828461b707453c  
3944 Author: Ashley An <ashley.an@columbia.edu>  
3945 Date: Fri Nov 9 19:45:54 2018 -0500  
3946  
3947 fixed config and enhanced for loop  
3948  
3949 commit f5687785d5308ee7ffc031e67d186043f0611600  
3950 Merge: 6f1796c ddc593d  
3951 Author: Ashley An <ashley.an@columbia.edu>  
3952 Date: Fri Nov 9 19:23:23 2018 -0500  
3953  
3954 merged  
3955  
3956 commit 6f1796c6c372fbcf2975ea2eab9a7cc3b7e9ab7f  
3957 Author: Ashley An <ashley.an@columbia.edu>  
3958 Date: Fri Nov 9 19:22:20 2018 -0500  
3959  
3960 fixed arrays  
3961  
3962 commit ddc593dded8f5c836d4ada928cab8ae7106a20ed  
3963 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3964 Date: Fri Nov 9 19:17:09 2018 -0500  
3965  
3966 fix empty block problem  
3967  
3968 commit 14e93db6d2d084260a882544516cac6adefc76d5



3969 Merge: 2f3b529 71fd9ce  
3970 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3971 Date: Fri Nov 9 19:09:22 2018 -0500  
3972  
3973 Merge branch 'master' of https://github.com/soybean/PLT-f18  
3974  
3975 commit 2f3b5290034646b25f7b943eb3d2690da01fbfa7  
3976 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3977 Date: Fri Nov 9 19:09:13 2018 -0500  
3978  
3979 get rid of reduce reduce errors  
3980  
3981 commit 71fd9ce8a8ef08fac6bf42932bba1c89a87e7189  
3982 Author: Victoria Yang <victoria.j.yang@gmail.com>  
3983 Date: Fri Nov 9 18:59:52 2018 -0500  
3984  
3985 Add rgx  
3986  
3987 commit c7f58d544242dbefe206151b4448826c648269aa  
3988 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3989 Date: Fri Nov 9 17:50:05 2018 -0500  
3990  
3991 fix empty config block  
3992  
3993 commit 645bed233204dc74cc4b459a0b57363c36fd08c5  
3994 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
3995 Date: Fri Nov 9 17:38:01 2018 -0500  
3996  
3997 add empty map  
3998  
3999 commit 5ea2f5f56d82d3a98da0360d7910270a8f62426a  
4000 Merge: a9d61cb e4b9685  
4001 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4002 Date: Fri Nov 9 16:18:59 2018 -0500  
4003  
4004 fix merge conflicts  
4005  
4006 commit a9d61cbe8cbd6e8e97350658c963931ef6438259  
4007 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4008 Date: Fri Nov 9 16:15:02 2018 -0500  
4009  
4010 empty map  
4011  
4012 commit e4b96851cc4fbd4cbf4ec03e22f249f0e913fb3d  
4013 Author: Ashley An <ashley.an@columbia.edu>  
4014 Date: Fri Nov 9 15:54:29 2018 -0500  
4015  
4016 edited map test  
4017  
4018 commit 848823b3a1a3fd17e2d9bec46f1923df43d63d6a  
4019 Author: Ashley An <ashley.an@columbia.edu>  
4020 Date: Fri Nov 9 15:21:19 2018 -0500  
4021  
4022 modified map1 test  
4023  
4024 commit 411d64c472613072d911a12713165c7ec4f52a94

4025 Author: Ashley An <ashley.an@columbia.edu>  
4026 Date: Fri Nov 9 14:52:15 2018 -0500  
4027  
4028 Move test-script  
4029  
4030 commit 087dc405dc8131cabd00b3220cec9f26381a317b  
4031 Author: Ashley An <ashley.an@columbia.edu>  
4032 Date: Fri Nov 9 14:51:05 2018 -0500  
4033  
4034 fixed script for running all the tests  
4035  
4036 commit 21eed276a2d9221a8bf58fdc343e0b6098985f07  
4037 Merge: 24edf97 4cbdebd  
4038 Author: Ashley An <ashley.an@columbia.edu>  
4039 Date: Fri Nov 9 14:43:07 2018 -0500  
4040  
4041 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
4042  
4043 commit 24edf974eccb06af5880e0cfef560d31c9deb450  
4044 Author: Ashley An <ashley.an@columbia.edu>  
4045 Date: Fri Nov 9 14:43:00 2018 -0500  
4046  
4047 changed names for array get and assign to be clearer  
4048  
4049 commit 4cbdebd927e44bba378210316d82714e560f678a  
4050 Author: Victoria Yang <victoria.j.yang@gmail.com>  
4051 Date: Fri Nov 9 14:42:46 2018 -0500  
4052  
4053 Changed builtin functions for new array definition  
4054  
4055 commit bae6a54ad6e9f0a96fa253f6d3f273da35bdb5d1  
4056 Author: Ashley An <ashley.an@columbia.edu>  
4057 Date: Fri Nov 9 14:40:29 2018 -0500  
4058  
4059 fixed arrays and added array tests  
4060  
4061 commit 90006336b26182f5c9e2046ad6618b158b7ac3d9  
4062 Author: Victoria Yang <victoria.j.yang@gmail.com>  
4063 Date: Fri Nov 9 14:30:29 2018 -0500  
4064  
4065 Create test-script.sh  
4066  
4067 commit 8ff0abe533e7cd8ebd37428b0f3c4c1c7c210acd  
4068 Author: Victoria Yang <victoria.j.yang@gmail.com>  
4069 Date: Fri Nov 9 14:18:51 2018 -0500  
4070  
4071 Added builtin keyword list  
4072  
4073 commit ffb82d418f3c27f80b3863771492f0b0f467a29d  
4074 Author: Ashley An <ashley.an@columbia.edu>  
4075 Date: Fri Nov 9 12:52:18 2018 -0500  
4076  
4077 fixed config  
4078  
4079 commit 37e875cbfd788fa31ffb792150c8102ac0738036  
4080 Author: Victoria Yang <victoria.j.yang@gmail.com>

```
4081 Date: Fri Nov 9 12:41:01 2018 -0500
4082
4083 Removed extra in
4084
4085 commit 90de74fdb341acaa87c5ff0b9670742557660ee4
4086 Author: Victoria Yang <victoria.j.yang@gmail.com>
4087 Date: Fri Nov 9 12:39:45 2018 -0500
4088
4089 Returned stuff that I deleted
4090
4091 commit 927995ff903de01635f68b6e215df26288c533b5
4092 Author: Victoria Yang <victoria.j.yang@gmail.com>
4093 Date: Fri Nov 9 12:27:07 2018 -0500
4094
4095 Fixed int to string declaration
4096
4097 commit 189cf159dd4973b746af8c43aab3ae1b9972f630
4098 Merge: 326d74b 45e5ad3
4099 Author: Ashley An <ashley.an@columbia.edu>
4100 Date: Fri Nov 9 12:21:05 2018 -0500
4101
4102 Merge branch 'master' of https://github.com/soybean/PLT-f18
4103
4104 commit 326d74becadf38c8738e9de7990fefa6fdb7b6f2
4105 Author: Ashley An <ashley.an@columbia.edu>
4106 Date: Fri Nov 9 12:20:56 2018 -0500
4107
4108 modified regex tests to have rgx_to_string
4109
4110 commit 45e5ad3b5da0e8fe383c20b8e2c3af230768f6ba
4111 Author: Victoria Yang <victoria.j.yang@gmail.com>
4112 Date: Fri Nov 9 12:19:07 2018 -0500
4113
4114 Added to string functions
4115
4116 commit 3729c4f765e41df6ee523de31657e94deac81c56
4117 Author: Ashley An <ashley.an@columbia.edu>
4118 Date: Fri Nov 9 12:16:05 2018 -0500
4119
4120 modified empty map in test
4121
4122 commit c4cca1ebf4321c6dc5b713c768204b3f25b07dc3
4123 Merge: 0f80933 f8f42c5
4124 Author: Ashley An <ashley.an@columbia.edu>
4125 Date: Fri Nov 9 12:09:06 2018 -0500
4126
4127 Merge branch 'master' of https://github.com/soybean/PLT-f18
4128
4129 commit 0f80933c83115cf3abe0bfe8c45681ba33447812
4130 Author: Ashley An <ashley.an@columbia.edu>
4131 Date: Fri Nov 9 12:08:52 2018 -0500
4132
4133 modified files and added more tests
4134
4135 commit f8f42c533245b8b467f55afdc437dd40dd9f7c79
4136 Author: Victoria Yang <victoria.j.yang@gmail.com>
```

4137 Date: Fri Nov 9 11:10:12 2018 -0500  
4138  
4139 Remove floats from semant  
4140  
4141 commit a214584393a6fe70110dbfe1dcf72545d23c1db4  
4142 Author: Victoria Yang <victoria.j.yang@gmail.com>  
4143 Date: Fri Nov 9 10:56:37 2018 -0500  
4144  
4145 Update built-in functions  
4146  
4147 commit 3acb96f4067101aebc59b3f6608cc70088ac5404  
4148 Author: Victoria Yang <victoria.j.yang@gmail.com>  
4149 Date: Fri Nov 9 10:46:10 2018 -0500  
4150  
4151 Changed symbols back  
4152  
4153 commit 95ca0091c052f67557b2aac3f07631bab6b155e2  
4154 Author: Victoria Yang <victoria.j.yang@gmail.com>  
4155 Date: Fri Nov 9 10:44:40 2018 -0500  
4156  
4157 Updated typing  
4158  
4159 commit a1952cb36a6e7a19d4cf251b076792eab26a5d25  
4160 Author: Victoria Yang <victoria.j.yang@gmail.com>  
4161 Date: Fri Nov 9 10:37:59 2018 -0500  
4162  
4163 Added println  
4164  
4165 commit 7448f211c6c371dd75bdaf4a16aa90af64741b8c  
4166 Author: Victoria Yang <victoria.j.yang@gmail.com>  
4167 Date: Fri Nov 9 10:33:28 2018 -0500  
4168  
4169 Added built-in function declarations  
4170  
4171 commit f50196261d337487d108916625a44fa631250fba  
4172 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>  
4173 Date: Fri Nov 9 08:31:25 2018 -0500  
4174  
4175 add RgxLiteral  
4176  
4177 commit 638f2cae05f9b192ddae2a08407bd40cf8879189  
4178 Author: Ashley An <ashley.an@columbia.edu>  
4179 Date: Thu Nov 8 23:02:07 2018 -0800  
4180  
4181 Update ast.ml  
4182  
4183 commit 029a8e121a48b07c9d1e279cf79e32d51bd46247  
4184 Author: Ashley An <ashley.an@columbia.edu>  
4185 Date: Thu Nov 8 23:01:41 2018 -0800  
4186  
4187 Update ast.ml  
4188  
4189 commit 056066f6b7510e608ed5f31a75c51b6d67692a07  
4190 Author: Ashley An <ashley.an@columbia.edu>  
4191 Date: Fri Nov 9 01:55:46 2018 -0500  
4192

```
4193     fixed decrement and increment tests
4194
4195 commit 3fa2a4a8dc64b68ff93784c4e931782e26d77f1a
4196 Author: Christine Hsu <christine.hhsu@gmail.com>
4197 Date:   Fri Nov 9 01:39:21 2018 -0500
4198
4199     Update scanner.mll
4200
4201 commit aa62f06e60713e1259f13425f4438cb17a4f102b
4202 Author: Christine Hsu <christine.hhsu@gmail.com>
4203 Date:   Fri Nov 9 01:36:23 2018 -0500
4204
4205     add stringliteral to ast
4206
4207 commit 9e5735806791baffealf5879a544803ebfd1cdf5
4208 Author: Christine Hsu <christine.hhsu@gmail.com>
4209 Date:   Fri Nov 9 01:35:14 2018 -0500
4210
4211     added StringLiteral to scanner, pass_string3 fails
4212
4213 commit 02cd342ed76880da62a83d8c8daf4b7d56d82503
4214 Author: Christine Hsu <christine.hhsu@gmail.com>
4215 Date:   Fri Nov 9 01:34:43 2018 -0500
4216
4217     added StringLiteral to parser
4218
4219 commit 1480906f5eeee35d4d4754c7282149cc6ec1fbf8
4220 Author: Ashley An <ashley.an@columbia.edu>
4221 Date:   Fri Nov 9 01:23:02 2018 -0500
4222
4223     changed map to be defined with LT and GT tokens instead
4224
4225 commit 89a5d2bdb74bc134b389fbe30b899e1d88a1eb83
4226 Author: Christine Hsu <christinehsu@Sam-Smith5s-MacBook-Air.local>
4227 Date:   Fri Nov 9 00:30:27 2018 -0500
4228
4229     fixed parser
4230
4231 commit 25adeb93a6f27c7e040aac36ab6224f6bcf699f9
4232 Author: Ashley An <ashley.an@columbia.edu>
4233 Date:   Thu Nov 8 22:35:56 2018 -0500
4234
4235     added typ for function declarations
4236
4237 commit 12d6d50f5e07e5cb9c56d690674d74161060e37c
4238 Merge: a8f1e6c 63c4015
4239 Author: Ashley An <ashley.an@columbia.edu>
4240 Date:   Thu Nov 8 22:12:16 2018 -0500
4241
4242     Merge branch 'master' of https://github.com/soybean/PLT-f18
4243
4244 commit a8f1e6c36c2c67dfcd1ed7ebaa7ba8cd24e8fc5c
4245 Author: Ashley An <ashley.an@columbia.edu>
4246 Date:   Thu Nov 8 22:12:09 2018 -0500
4247
4248     fixed print function in tests
```

4249  
4250 commit 63c4015c522346ab80e5814c41399f3196c4c40e  
4251 Author: Ashley An <ashley.an@columbia.edu>  
4252 Date: Thu Nov 8 14:25:49 2018 -0800  
4253  
4254 Update pass-add1.bawk  
4255  
4256 commit a7279228d66144979556e69f42f75a3a48f00a50  
4257 Author: Ashley An <ashley.an@columbia.edu>  
4258 Date: Thu Nov 8 17:25:08 2018 -0500  
4259  
4260 fixed config to be at bottom  
4261  
4262 commit ae3e5bc35e1b212ad1102c1a9b7747715d0a1f86  
4263 Author: Ashley An <ashley.an@columbia.edu>  
4264 Date: Thu Nov 8 17:18:37 2018 -0500  
4265  
4266 added some tests  
4267  
4268 commit e904645fdf1f5c7f4eb73f83a1ac41c2371b038e  
4269 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4270 Date: Thu Nov 8 16:49:46 2018 -0500  
4271  
4272 update empty map syntax  
4273  
4274 commit f7953cff7db70ad73d0777613bb6ea2ade68d802  
4275 Author: Victoria Yang <victoria.j.yang@gmail.com>  
4276 Date: Wed Nov 7 22:12:30 2018 -0500  
4277  
4278 Make sure rgx can't be in list  
4279  
4280 commit 051add2399c792e0344c4ccd2e975f6630e28570  
4281 Author: Victoria Yang <victoria.j.yang@gmail.com>  
4282 Date: Wed Nov 7 19:50:54 2018 -0500  
4283  
4284 Removed built-in functions we don't have  
4285  
4286 commit e35ad95412e5c73270b57ff11d1da34b3ac66077  
4287 Merge: 72bc22b 6686706  
4288 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4289 Date: Wed Nov 7 19:44:40 2018 -0500  
4290  
4291 Merge branch 'master' of <https://github.com/soybean/PLT-f18>  
4292  
4293 commit 72bc22b64be9d1e99a2c2bda2b3a3a9e48918d30  
4294 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4295 Date: Wed Nov 7 19:44:28 2018 -0500  
4296  
4297 add semant  
4298  
4299 commit 668670621f1ac9051063cc9ccd53b14c915fd329  
4300 Merge: 7e3c052 ea3518f  
4301 Author: Victoria Yang <victoria.j.yang@gmail.com>  
4302 Date: Wed Nov 7 19:43:08 2018 -0500  
4303  
4304 Merge pull request #5 from soybean/victorias-branch

4305  
4306       Victorias branch  
4307  
4308 commit ea3518fe2e9f185791a925f5aa40efdfb222bf10  
4309 Author: Victoria Yang <victoria.j.yang@gmail.com>  
4310 Date:   Wed Nov 7 19:42:33 2018 -0500  
4311  
4312       Changed sast.ml to ast.ml  
4313  
4314 commit 7e3c0521047ca81c71a472f5192832dbc6747731  
4315 Merge: d221afe defe120  
4316 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4317 Date:   Wed Nov 7 17:41:49 2018 -0700  
4318  
4319       Merge pull request #4 from soybean/dev  
4320  
4321       fix config block  
4322  
4323 commit defe1206605170f5a8fd2e1c3d1f0c11e7c5d293  
4324 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4325 Date:   Wed Nov 7 19:41:19 2018 -0500  
4326  
4327       fix config block  
4328  
4329 commit a9f6fcc00885569ba0329a9c23cd954822609b9e  
4330 Author: Victoria Yang <victoria.j.yang@gmail.com>  
4331 Date:   Wed Nov 7 19:34:39 2018 -0500  
4332  
4333       Made copy of ast.ml in sast.ml  
4334  
4335 commit 318c188474e2f2975eb216db5803234e6ad78760  
4336 Author: Victoria Yang <victoria.j.yang@gmail.com>  
4337 Date:   Wed Nov 7 19:34:03 2018 -0500  
4338  
4339       Create sast.ml  
4340  
4341 commit d221afe84de6be9eb4624997be3617aab1d64c4c  
4342 Merge: e9a7549 d31a8e5  
4343 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4344 Date:   Wed Nov 7 17:19:04 2018 -0700  
4345  
4346       Merge pull request #3 from soybean/broken-stuff  
4347  
4348       Broken stuff  
4349  
4350 commit d31a8e5b79bdca1fe96d49464b7faa8cfaeb9710  
4351 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4352 Date:   Wed Nov 7 19:17:35 2018 -0500  
4353  
4354       fix config  
4355  
4356 commit fffa080436930448dae3f2879b60ad2589d86f3b  
4357 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4358 Date:   Wed Nov 7 19:15:58 2018 -0500  
4359  
4360       add config

4361  
4362 commit e9a7549fec7219d1693175c53633043f8215ab70  
4363 Merge: 1428eb7 c42d710  
4364 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4365 Date: Wed Nov 7 16:53:42 2018 -0700  
4366  
4367 Merge pull request #2 from soybean/broken-stuff  
4368  
4369 Broken stuff  
4370  
4371 commit c42d7100ef175d74060e9063eb434955de35e7ed  
4372 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4373 Date: Wed Nov 7 18:52:52 2018 -0500  
4374  
4375 fix regex  
4376  
4377 commit a6db87815d2e36165ce09e1bbfc23f58d18c82c4  
4378 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4379 Date: Wed Nov 7 18:49:20 2018 -0500  
4380  
4381 add for to scanner  
4382  
4383 commit 7aefdd502f21fa840a83d4a9c44570b96820b4e3  
4384 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4385 Date: Wed Nov 7 18:48:26 2018 -0500  
4386  
4387 fix parser  
4388  
4389 commit f0d15ea60c0eb958ba684716741c1c7a7ab04103  
4390 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4391 Date: Wed Nov 7 16:56:37 2018 -0500  
4392  
4393 fix increment  
4394  
4395 commit b7e93c8fe2a543b232c91e561c186223f3f459b8  
4396 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4397 Date: Wed Nov 7 16:37:39 2018 -0500  
4398  
4399 add updated ast  
4400  
4401 commit 1428eb77a092025b1a0827684316a53a577661fd  
4402 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4403 Date: Mon Oct 29 23:31:32 2018 -0400  
4404  
4405 fix syntax error in scanner  
4406  
4407 commit 419d90cbbed96ea9cd5503e54b2b83099307e696  
4408 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4409 Date: Mon Oct 29 23:28:59 2018 -0400  
4410  
4411 fix syntax error  
4412  
4413 commit 56d7bf6773534b63daa5f11670151ae2faccc9a9  
4414 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4415 Date: Mon Oct 29 22:50:10 2018 -0400  
4416



4417 broken stuff  
4418  
4419 commit 6c910adf0dd7e572c8c826568b7d0b58b966cc55  
4420 Merge: e8ce506 ad12dce  
4421 Author: Ashley An <ashley.an@columbia.edu>  
4422 Date: Sun Oct 28 21:22:12 2018 -0400  
4423  
4424 Merge branch 'master' of https://github.com/soybean/PLT-f18  
4425  
4426 commit e8ce5060f582d2f4dff230cfc8394b78b1fba37f  
4427 Author: Ashley An <ashley.an@columbia.edu>  
4428 Date: Sun Oct 28 21:21:48 2018 -0400  
4429  
4430 added bawk.ml and made few changes to ast.ml to fix it  
4431  
4432 commit ad12dce46a66dd4402b8b7909cb2cd36b176b278  
4433 Author: Christine Hsu <christine.hhsu@gmail.com>  
4434 Date: Sat Oct 27 23:52:54 2018 -0400  
4435  
4436 Create semant.ml  
4437  
4438 commit 4ee56f480c55b82eff8da18abfe12460ab0e10e7  
4439 Author: Ashley An <ashley.an@columbia.edu>  
4440 Date: Tue Oct 23 00:28:44 2018 -0400  
4441  
4442 alphabetized names in README.md to test committing from local machine  
4443  
4444 commit 6e5ba1d48288fa478b819eb91820a4eaa5c5a107  
4445 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4446 Date: Fri Oct 12 14:08:55 2018 -0400  
4447  
4448 add all necessary updates  
4449  
4450 commit c4e72613803f364405091e9666aa04ce2fb15645  
4451 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4452 Date: Fri Oct 12 13:59:34 2018 -0400  
4453  
4454 move statements to expressions  
4455  
4456 commit c956e7353c099dc4e60a10976ad0986f567e405e  
4457 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4458 Date: Fri Oct 12 13:36:14 2018 -0400  
4459  
4460 add no else  
4461  
4462 commit 7deb0ee95fedd33b9d28328d6d6a08c9df0fe940  
4463 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4464 Date: Fri Oct 12 13:21:40 2018 -0400  
4465  
4466 fix expression list  
4467  
4468 commit 5474c104216caf56dc22d32ce95a3219c0da9f4a  
4469 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4470 Date: Fri Oct 12 02:42:55 2018 -0400  
4471  
4472 final updates

4473  
4474 commit f560251bd80edf7bc786afeb2bbb54db9e4ce763  
4475 Author: Ashley An <ashley.an@columbia.edu>  
4476 Date: Fri Oct 12 01:23:07 2018 -0400  
4477  
4478 Update scanner.mll  
4479  
4480 commit bc458f55c22b707f9993852342fb27e395f53013  
4481 Author: Ashley An <ashley.an@columbia.edu>  
4482 Date: Fri Oct 12 01:22:54 2018 -0400  
4483  
4484 Update parser.mly  
4485  
4486 commit 14acae7e2c60f23c32c2b500f11d2c990488b46c  
4487 Author: Ashley An <ashley.an@columbia.edu>  
4488 Date: Fri Oct 12 01:22:39 2018 -0400  
4489  
4490 Update ast.ml  
4491  
4492 commit 9dcfe0a7eeb6f54305f493e9f14b7ac7205a1d31  
4493 Author: Ashley An <ashley.an@columbia.edu>  
4494 Date: Fri Oct 12 01:09:57 2018 -0400  
4495  
4496 Update scanner.mll  
4497  
4498 commit dd6e13d8ff721b902b0b812192d71b0b0aee2d2e  
4499 Author: Ashley An <ashley.an@columbia.edu>  
4500 Date: Fri Oct 12 01:09:42 2018 -0400  
4501  
4502 Update parser.mly  
4503  
4504 commit b3f02a42588ddbe2325bea57e7645de2e84ab277  
4505 Author: Ashley An <ashley.an@columbia.edu>  
4506 Date: Fri Oct 12 01:09:23 2018 -0400  
4507  
4508 Update ast.ml  
4509  
4510 commit 6227021988cf2bf0043fbd04231a9567f23eb83f  
4511 Author: Ashley An <ashley.an@columbia.edu>  
4512 Date: Thu Oct 11 23:53:22 2018 -0400  
4513  
4514 Update scanner.mll  
4515  
4516 commit 8aa80ae37cf932e47f67fc8e15f2e04ae56c2045  
4517 Author: Ashley An <ashley.an@columbia.edu>  
4518 Date: Thu Oct 11 23:53:06 2018 -0400  
4519  
4520 Update parser.mly  
4521  
4522 commit d232fc90731e6cdec993a6f3e25946125f77101  
4523 Author: Ashley An <ashley.an@columbia.edu>  
4524 Date: Thu Oct 11 23:52:51 2018 -0400  
4525  
4526 Update ast.ml  
4527  
4528 commit ef9eabab57a5e26cdf585fc65709613373d4ccf9

4529 Author: Ashley An <ashley.an@columbia.edu>  
4530 Date: Thu Oct 11 23:34:28 2018 -0400  
4531  
4532 Update ast.ml  
4533  
4534 commit 6fcc764bd4c098b71f0f3d79dd4e371d0d6282d8  
4535 Author: Ashley An <ashley.an@columbia.edu>  
4536 Date: Thu Oct 11 23:33:20 2018 -0400  
4537  
4538 Update scanner.mll  
4539  
4540 commit a2c48c787c095f77d398fa0f5b78286091a0d510  
4541 Author: Ashley An <ashley.an@columbia.edu>  
4542 Date: Thu Oct 11 23:32:55 2018 -0400  
4543  
4544 Update parser.mly  
4545  
4546 commit ec37941a07868847fbb69239e5460816a90f936b  
4547 Author: Ashley An <ashley.an@columbia.edu>  
4548 Date: Thu Oct 11 23:32:35 2018 -0400  
4549  
4550 Update ast.ml  
4551  
4552 commit 405379de1662bbcc7cbfe98a93263e8d45143bb4  
4553 Author: Ashley An <ashley.an@columbia.edu>  
4554 Date: Thu Oct 11 23:21:29 2018 -0400  
4555  
4556 Update ast.ml  
4557  
4558 commit 3b4699aa53619c5c50fe7954333f34289ab45305  
4559 Author: Ashley An <ashley.an@columbia.edu>  
4560 Date: Thu Oct 11 23:20:58 2018 -0400  
4561  
4562 Update parser.mly  
4563  
4564 commit 1efdda81904c619551f772c82b3235efddbda2ce  
4565 Author: Ashley An <ashley.an@columbia.edu>  
4566 Date: Thu Oct 11 23:20:26 2018 -0400  
4567  
4568 Update scanner.mll  
4569  
4570 commit 5a0d628139c3659c950325ec995bc2b7884b459e  
4571 Author: Ashley An <ashley.an@columbia.edu>  
4572 Date: Thu Oct 11 23:05:58 2018 -0400  
4573  
4574 Update parser.mly  
4575  
4576 commit 6dc5e3de6fb35c9f71af596a4c9579847a558f17  
4577 Author: Ashley An <ashley.an@columbia.edu>  
4578 Date: Thu Oct 11 22:34:22 2018 -0400  
4579  
4580 Update parser.mly  
4581  
4582 commit 778f19e9695ac2593bcd8770fc04e67dcf9d7353  
4583 Author: Ashley An <ashley.an@columbia.edu>  
4584 Date: Thu Oct 11 22:33:16 2018 -0400

```
4585
4586     Update parser.mly
4587
4588 commit b6ff7dd4a7f151664da50cf7595b833ad92c4c67
4589 Author: Ashley An <ashley.an@columbia.edu>
4590 Date:   Thu Oct 11 22:32:33 2018 -0400
4591
4592     Update scanner.mll
4593
4594 commit ac06d5803e0f7d7dde4c216631b563b728fe7280
4595 Author: Ashley An <ashley.an@columbia.edu>
4596 Date:   Thu Oct 11 22:32:13 2018 -0400
4597
4598     Update parser.mly
4599
4600 commit ea6db582f04c5d6f5f3303f57e09bbfa56c15496
4601 Author: Ashley An <ashley.an@columbia.edu>
4602 Date:   Thu Oct 11 22:31:49 2018 -0400
4603
4604     Update ast.ml
4605
4606 commit b6ffb1301951ee0d6af3b305b0c81b812853edb4
4607 Author: Christine Hsu <christine.hhsu@gmail.com>
4608 Date:   Thu Oct 11 21:59:45 2018 -0400
4609
4610     Update parser.mly
4611
4612 commit 41839f00950cd652639a453bbe599ac57ca76b19
4613 Author: Christine Hsu <christine.hhsu@gmail.com>
4614 Date:   Thu Oct 11 21:52:33 2018 -0400
4615
4616     Update parser.mly
4617
4618 commit 01096c9692865c865c5cd47aa3bacc187c19c9e5
4619 Author: Christine Hsu <christine.hhsu@gmail.com>
4620 Date:   Thu Oct 11 21:34:56 2018 -0400
4621
4622     added declarations to parser
4623
4624     CONFIG block is unfinished
4625
4626 commit 25c469e85917f22157358f1b8b6a4ff610bba72a
4627 Author: Melanie Sawyer <melaniensawyer@gmail.com>
4628 Date:   Thu Oct 11 00:31:08 2018 -0400
4629
4630     Update parser.mly
4631
4632 commit 033a3e04ab694dc643ac5e2eddd6075e9cf85339
4633 Author: Melanie Sawyer <melaniensawyer@gmail.com>
4634 Date:   Wed Oct 10 23:49:06 2018 -0400
4635
4636     Update scanner.mll
4637
4638 commit e0597158056e51f8f44eef1c04619b9aad2a8e3a
4639 Author: Melanie Sawyer <melaniensawyer@gmail.com>
4640 Date:   Wed Oct 10 23:07:45 2018 -0400
```

4641  
4642 Update parser.mly  
4643  
4644 commit 49445099dd53a452d6ed13c020f3271928567f5d  
4645 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4646 Date: Wed Oct 10 23:07:22 2018 -0400  
4647  
4648 Update scanner.mll  
4649  
4650 commit dfe5e4d36b8cdb21270cae34283b747cfd2c879  
4651 Author: Ashley An <ashley.an@columbia.edu>  
4652 Date: Wed Oct 10 22:36:42 2018 -0400  
4653  
4654 Update parser.mly  
4655  
4656 commit 2ff7381a0773c35c7f0916caed6dab72964594dc  
4657 Author: Ashley An <ashley.an@columbia.edu>  
4658 Date: Wed Oct 10 20:30:11 2018 -0400  
4659  
4660 Update ast.ml  
4661  
4662 commit 1f1cf3218f5ddcd135aca061c8176fe22a3fee0f  
4663 Author: Ashley An <ashley.an@columbia.edu>  
4664 Date: Wed Oct 10 20:28:24 2018 -0400  
4665  
4666 Update scanner.mll  
4667  
4668 commit f8de63a86a8180a45a8320a7d65137f5063ba0ea  
4669 Author: Ashley An <ashley.an@columbia.edu>  
4670 Date: Wed Oct 10 20:26:41 2018 -0400  
4671  
4672 Update scanner.mll  
4673  
4674 commit 12940233aa2decac34e4ecbb6d42c7b4fbf66c3b  
4675 Author: Ashley An <ashley.an@columbia.edu>  
4676 Date: Tue Oct 9 00:00:53 2018 -0400  
4677  
4678 Create parser.mly  
4679  
4680 commit 637b93adb12cf2ef4e253c6166d4497da4ee25b7  
4681 Author: Victoria Yang <victoria.j.yang@gmail.com>  
4682 Date: Mon Oct 8 23:56:21 2018 -0400  
4683  
4684 Create ast.mli  
4685  
4686 commit a3297512472614f648925a9b9845f566bd462fc7  
4687 Author: Ashley An <ashley.an@columbia.edu>  
4688 Date: Mon Oct 8 23:55:18 2018 -0400  
4689  
4690 Update scanner.mll  
4691  
4692 commit 272e7ddfd0601db23cd7127660ca884ebddd42c6  
4693 Author: Christine Hsu <christine.hhsu@gmail.com>  
4694 Date: Mon Oct 8 23:54:47 2018 -0400  
4695  
4696 Update README.md

4697  
4698 commit e7e68c5d8d8b8c01097dbac0843b4ab629991bdd  
4699 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4700 Date: Mon Oct 8 23:51:31 2018 -0400  
4701  
4702 Update scanner.mll  
4703  
4704 commit 4b49aa26fae72a802ceb48153066de31973b106d  
4705 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4706 Date: Mon Oct 8 23:41:25 2018 -0400  
4707  
4708 Update scanner.mll  
4709  
4710 commit 17b6f846178d655ed2e90d13bde54fe05c9baec2  
4711 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4712 Date: Mon Oct 8 23:39:30 2018 -0400  
4713  
4714 Update scanner.mll  
4715  
4716 commit 4c738cd59c36bafc02abaef235cfd6ed0fd507f  
4717 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4718 Date: Mon Oct 8 23:28:10 2018 -0400  
4719  
4720 Update scanner.mll  
4721  
4722 commit 7c2e939c377d1549e4f9a751e0cb3fbee8ec0348  
4723 Author: Ashley An <ashley.an@columbia.edu>  
4724 Date: Mon Oct 8 23:18:59 2018 -0400  
4725  
4726 Update scanner.mll  
4727  
4728 commit ed86b16ef8a864bcc2ace0ed66c0d12617ee166f  
4729 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4730 Date: Mon Oct 8 23:12:42 2018 -0400  
4731  
4732 Update scanner.mll  
4733  
4734 commit e1f203b5a4e55602e0d65da45bc8be66692bac9d  
4735 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4736 Date: Mon Oct 8 23:12:31 2018 -0400  
4737  
4738 Update scanner.mll  
4739  
4740 commit ea77a459ca113570f414227cb3fd92f8df437c68  
4741 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4742 Date: Mon Oct 8 23:00:52 2018 -0400  
4743  
4744 initial lexer  
4745  
4746 commit 5e80a92fae1b7ac108808b401bd7df764f3bd082  
4747 Author: Melanie Sawyer <melaniensawyer@gmail.com>  
4748 Date: Sun Oct 7 04:20:01 2018 -0400  
4749  
4750 Update README.md  
4751  
4752 commit 53d24f473d55634f05b1d4e49e121fd7021de4fa

```
4753 Author: Melanie Sawyer <melaniensawyer@gmail.com>
4754 Date: Thu Sep 6 16:57:08 2018 -0400
4755
4756 Initial commit
```

### 4.3 Development Process

Our development progress traced through the stages of compiler architecture. We started with the scanner, then the parser, and worked on the semantic checker and code generation in parallel. In an effort to start code generation early, we initially used the AST rather than the semantically checked SAST, and then switched back to the SAST once the semantic checker was complete.

During the beginning half of our project over the front-end stage of the compiler, our team met individually once a week and then on Fridays with our TA John. Our meetings with John were flexible and usually consisted of us asking him various questions. Each member of the team contributed evenly, and we maintained open and constant communication throughout the project.

### 4.4 Software Development Tools

Each member of our group used vim as a text editor. To build and run our project, we used the `num1` virtual machine provided by John. As a version control system, we used Github. We mostly stayed on the master branch, but often created branches to test volatile features.

### 4.5 Programming Style Guide

The following summarizes the style guides our group tried to abide to.

- All variables are snake case, all C function declarations are camel case.
- Tabs are equivalent to four spaces.

- Keep lines 80 characters long.
- All variable declarations should be at the top of the file. Global variable declarations in the C files should be kept to a minimum.
- The C files are split up by the category of their function, and additional functions should be kept in the proper files. (`convert.c` maintains all functions used for converting or altering data, `rgx.c` maintains our regex library, `structure.c` contains as little possible code and should only be code that is directly executed when the program is run.)
- Code should be commented as thoroughly as possible. Large chunks of OCaml should be preceded by a block explaining the function.

#### 4.6 Timeline

| Task         | Deadline                            |
|--------------|-------------------------------------|
| September 19 | Project Proposal Submitted          |
| October 15   | Language Reference Manual submitted |
| October 15   | Finished scanner, parser, and AST   |
| November 14  | Hello World program compiled        |
| December 2   | Semantic checker finished           |
| December 13  | Code Generation finished            |
| December 19  | Project presentation and submission |

#### 4.7 Language Evolution

Overall the structure of `bawk` did not evolve much over the course of the project. We removed a couple minor features we had planned on implementing, namely enhanced for loops and maps (i.e. associative arrays). We chose to remove these features largely due to the fact that `bawk` contains normal for loops and nested arrays already, which achieve the same functionalities as enhanced for loops and maps would have. Aside from that we were able to implement every feature we set out to do.

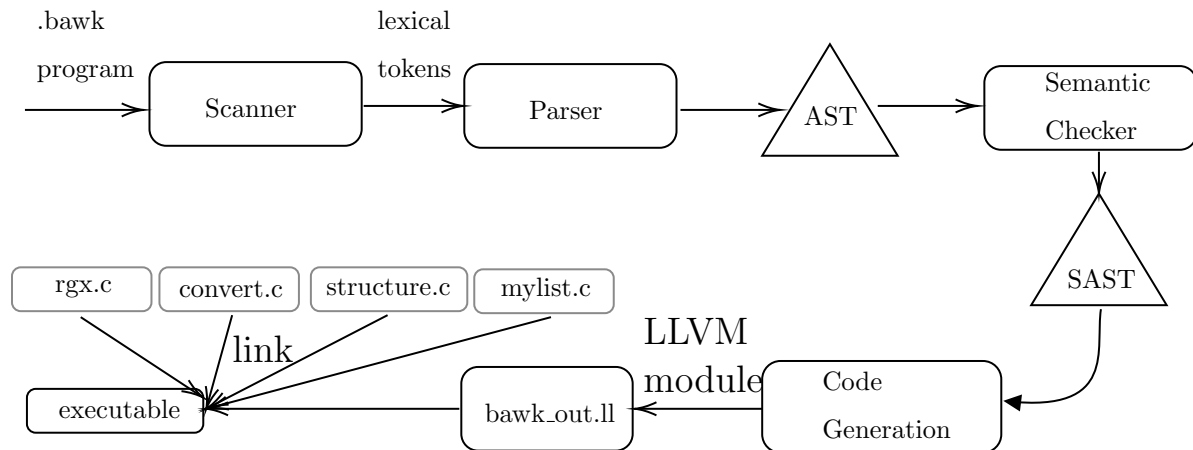


One major change that happened during the implementation of our language arose during the semantic checker to SAST phase. This required changing the structure of a program by hoisting blocks into functions in the semantic checker. This made it easier to implement the LOOP and END blocks in IR code generation and allowed us to avoid writing duplicate code.

## 4.8 Testing Process

Throughout the development of bawk, we tested each new stage with unit tests. Then, we ran tests that were written for previous stages to make sure that they were still behaving as expected. More tests were added when it was still unclear as to what exactly was causing problems. Each of these tests targeted a specific part of our language, and having these tests that covered a wide range of our language features helped us pinpoint what exactly was working or needed to be fixed. More details about testing are included in the Test Plan section.

## 5 System Architecture



## 5.1 Scanner

The scanner performs lexical analysis on the input file, reading in the raw text and converting it to lexical tokens, which are in turn passed to the parser. The scanner also removes all whitespace (spaces, tabs, newlines) from the program and converts escape characters to single characters. Finally, the scanner removes quotation marks from string and regex literals, which prevents codegen from having to manually strip quotation marks from strings. The scanner will terminate execution of the program if unrecognized tokens are passed through, but syntax and grammar issues could still be present.

## 5.2 Parser

The parser reads in lexical tokens from the scanner and generates an abstract syntax tree (AST) based on the grammatical rules of our language. If an input .bawk file successfully passes through the parser, it is guaranteed to be syntactically correct. There could be semantic errors however, which are identified during the semantic checking and code generation stages.

## 5.3 Semantic Checker

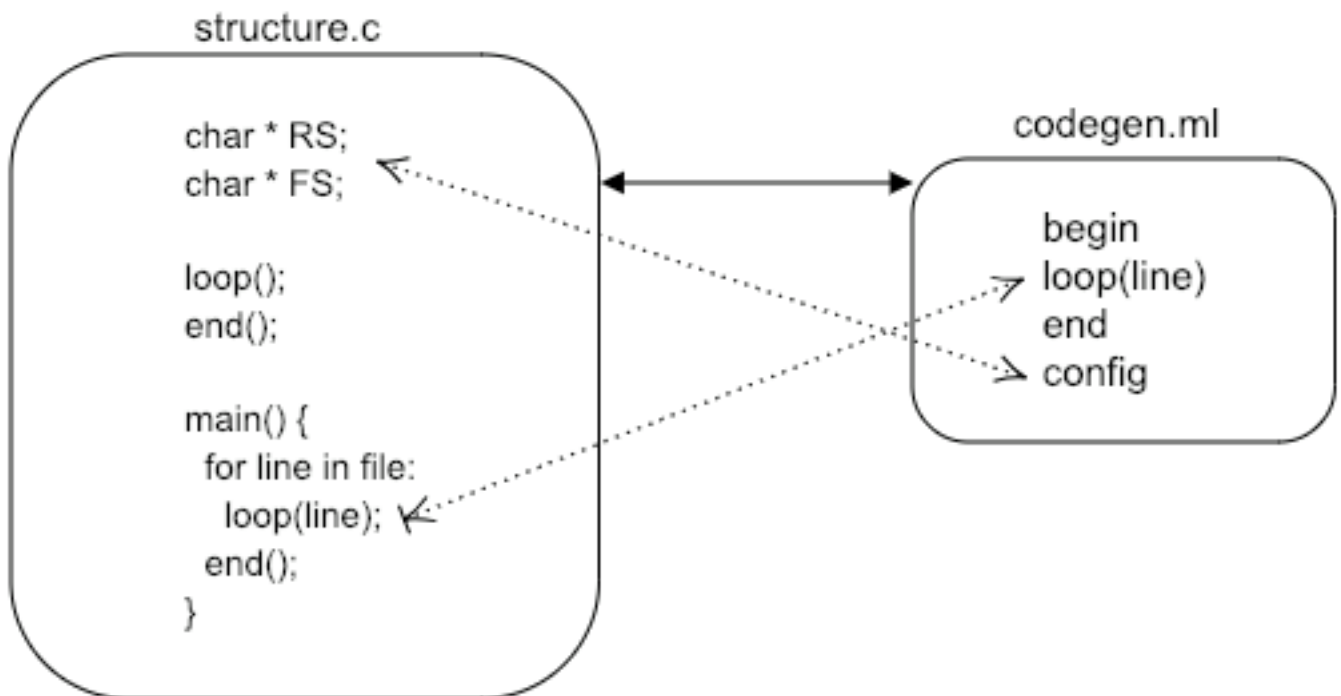
The semantic checker recursively traverses the AST and converts it into a semantically-checked abstract syntax tree (SAST). The semantic checker makes sure that types in the bawk program are correct. For example, arrays cannot contain elements of different types; a program that attempts to insert incorrect types into an array will be halted in the semantic checking phase. Each block in the program is checked separately, but global variables in the begin block can be used in loop and end.

## 5.4 Code Generation

The code generator traverses the semantically-checked AST (generated by `semant`) and translates each node into LLVM code. The code generator also links the necessary C libraries and makes calls to the proper C functions when necessary.

## 5.5 C Libraries

Because of the complexity of some aspects of our language, we outsourced some functionality to a number of C programs and libraries which allowed us to implement higher-level processes without having to rely on complicated LLVM constructs. The figure below shows the coordination between the `structure.c` file and `codegen.ml`



`regex.c`

Since the nature of our language is to process and filter text files, one core feature we needed was regular expression matching. Due to the vast number of regex patterns, it would be an immense time-suck to implement our own regex system, and the complexity would only be compounded by the fact that we'd have to write it in LLVM. Since C already has a relatively easy-to-use regex library, we wrote a small wrapper for it and directed all regex expressions through it.

### **POSIX regex library**

The POSIX regex library is the most commonly-used regex matching library in C. In order to parse a regex expression, the POSIX library requires that the expression first be compiled into a `regex_t` data structure. This type is then passed to the `regexexec` function to be executed. After some experimentation, we used the `REG_EXTENDED` (adds extended syntax) and the `REG_NOSUB` (only reports success/fail) flags to provide the most intuitive usage.

### **structure.c**

In contrast to the structure of a micro-C program, bawk programs loop through files in the `LOOP` block, and execute the `BEGIN`, `END`, and `CONFIG` blocks only once. Generating this structure (and simply reading through a file) is tremendously difficult using only OCaml LLVM bindings, so we elected to write a C file that would effectively moderate the execution of the program. This file, `structure.c`, declared the three functions `begin()`, `loop()`, and `end()`, without actually defining them. Codegen injects the actual code into these three functions, as it is translated from the AST. `structure.c` also defines the main function, which is executed upon runtime. The main function reads in the filename from the command line, loops through the file, and calls the `loop()` function on each line. Finally, it calls the `end()` function. `structure.c` also maintains the values for `RS` and `FS`, which are declared as external global variables. Codegen reads in the values from the config block and stores them in these globals, which are then used in the actual splitting of the file, and in functions such as the line access function (`$`) which splits a line into its fields.

## **mylist.c**

The bulk of array implementation is done in C using a linked list implementation we built in `mylist.c`. These linked lists are designed to be generic, so that they can hold data of any type, including other lists. This simplified our array implementation because we were able to use the same linked list implementation for all array types (int, bool, string, regex) and for nested arrays.

Each built-in function for arrays is written in C and called in the code generator. Each list keeps track of the size of the underlying type (int, string, rgx, bool), and the depth of the array (number of dimensions). The underlying type of the array is used to decide which comparator to pass into functions. The built-in array functions for C are as follows:

`initList`, which creates a list.

`addFront`, which takes in a long that represents the data and adds a node to the front of the list.

`length`, which traverses the list to return the length of the list.

`insert`, which inserts a specified element at a specified index.

`delete`, which deletes an element at a specified index.

`contains`, which checks if an element is in the array. It takes in a comparator function pointer, which is passed from the code generator based on the type of the second element.

`index_of`, which finds the first index of a specified element. It takes in a comparator function pointer, which is passed from the code generator based on the type of the second element.

## 6 Test Plan

### 6.1 Test Suite

Testing is arguably the most important part of building a compiler. Without testing your code, there is no guarantee that anything you wrote actually works. Thus, throughout development we made sure to write tests for each stage (lexer, parser, semantic checking, code generation) that pass or fail. These tests cover all the features of our language. Our tests include small tests that pinpoint specific features such as operators, loops, and built-in functions. They also include larger tests such as fibonacci and gcd to confirm that the correct output is being produced.

A listing of the tests and their respective outputs is included in the Code Listing section.

### 6.2 Automation

With such a large test suite, it is necessary to eventually automate the testing process. As a result, we wrote a testing script, `testall.sh`, which runs every test and reports OK if the test behaved as expected or FAILED if the test didn't behave as expected.

The testing script is included in the Code Listing section. The script can be executed within the root bawk directory by running

```
$ ./bawk.sh <program.bawk> <input.txt>
```

### 6.3 Examples and .ll output

Test 1: tests/pass-dynamicarr2.bawk

```
1 | BEGIN {}  
2 |
```

```

3 LOOP {}
4
5 END {
6   bool[] a;
7   a = [true, false, false];
8   delete(a, 2);
9   print(int_to_string(length(a)));
10 }

```

## Output 1: dynamicarr2.ll

```

1 ; ModuleID = 'Bawk'
2 source_filename = "Bawk"
3
4 %Array = type { %Node*, i32, i32 }
5 %Node = type { i64, %Node* }
6
7 @_RS = private unnamed_addr constant [2 x i8] c"\0A\00", align 1
8 @RS = global i8* @getelementptr inbounds ([2 x i8], [2 x i8]* @_RS, i32 0, i32 0)
9 @_FS = private unnamed_addr constant [2 x i8] c" \00", align 1
10 @FS = global i8* @getelementptr inbounds ([2 x i8], [2 x i8]* @_FS, i32 0, i32 0)
11 @fmt = private unnamed_addr constant [4 x i8] c"%s\0A\00"
12
13 declare i32 @printf(i8*, ...)
14
15 declare i8* @int_to_string(i32)
16
17 declare i8* @bool_to_string(i1)
18
19 declare i32 @string_to_int(i8*)
20
21 declare i8* @concat(i8*, i8*)
22
23 declare i1 @comp(i8*, i8*)
24
25 declare i1 @ncomp(i8*, i8*)
26
27 declare i1 @equals(i8*, i8*)
28
29 declare i1 @nequals(i8*, i8*)
30
31 declare i8* @rgx_to_string(i8*)
32
33 declare i8* @access(i8*, i32)
34
35 declare i1 @streq(i8*, i8*)
36
37 declare i1 @strneq(i8*, i8*)
38
39 declare i1 @strless(i8*, i8*)
40
41 declare i1 @strgreater(i8*, i8*)
42
43 declare i1 @strgeq(i8*, i8*)
44

```

```

45 declare i1 @strleq(i8*, i8*)
46
47 declare i32 @numfields(i8*)
48
49 declare %Array* @initList(i64, i32)
50
51 declare %Node* @addFront(%Array*, i64)
52
53 declare i64 @getElement(%Array*, i32)
54
55 declare i32 @assignElement(%Array*, i32, i64)
56
57 declare i8 @reverseList(%Array*)
58
59 declare i32 @length(%Array*)
60
61 declare i64 @removeNode(%Array*, i32)
62
63 declare i8 @insertElement(%Array*, i32, i64)
64
65 declare i32 @compareInts(i64, i64)
66
67 declare i32 @compareBools(i64, i64)
68
69 declare i32 @compareStrs(i64, i64)
70
71 declare i1 @contains(%Array*, i64, i32 (i64, i64)*)
72
73 declare i32 @findIndexOfNode(%Array*, i64, i32 (i64, i64)*)
74
75 define void @loop(i8* %line) {
76 entry:
77   %line1 = alloca i8*
78   store i8* %line, i8** %line1
79   ret void
80 }
81
82 define void @end() {
83 entry:
84   %a = alloca %Array*
85   %initList = call %Array* @initList(i64 ptrtoint (i1* getelementptr (i1, i1* null
      , i32 1) to i64), i32 1)
86   %addFront = call %Node* @addFront(%Array* %initList, i64 1)
87   %addFront1 = call %Node* @addFront(%Array* %initList, i64 0)
88   %addFront2 = call %Node* @addFront(%Array* %initList, i64 0)
89   %reverseList = call i8 @reverseList(%Array* %initList)
90   store %Array* %initList, %Array** %a
91   %a3 = load %Array*, %Array** %a
92   %removeNode = call i64 @removeNode(%Array* %a3, i32 2)
93   %a4 = load %Array*, %Array** %a
94   %length = call i32 @length(%Array* %a4)
95   %int_to_string = call i8* @int_to_string(i32 %length)
96   %printf = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4 x
      i8]* @fmt, i32 0, i32 0), i8* %int_to_string)
97   ret void
98 }

```



## Test 2: tests/pass-ops4.bawk

```
1 BEGIN {  
2  
3 LOOP {  
4  
5 END {  
6   print(bool_to_string('[0-9]*' % '[0]*'));  
7   print(bool_to_string('[0-9]*' % '[0-9]*'));  
8   print(int_to_string(99));  
9   print(bool_to_string('[0-9]*' !% '[0]*'));  
10  print(bool_to_string('[0-9]*' !% '[0-9]*'));  
11  print(int_to_string(99));  
12  print(bool_to_string("1234" ~ '[0]'));  
13  print(bool_to_string("1234" ~ '[0-9]*'));  
14  print(int_to_string(99));  
15  print(bool_to_string("1234" !~ '[0]'));  
16  print(bool_to_string("1234" !~ '[0-9]*'));  
17 }
```

## Output 2: ops2.11

```
1 ; ModuleID = 'Bawk'  
2 source_filename = "Bawk"  
3  
4 %Array = type { %Node*, i32, i32 }  
5 %Node = type { i64, %Node* }  
6  
7 @_RS = private unnamed_addr constant [2 x i8] c"\0A\00", align 1  
8 @RS = global i8* @getelementptr inbounds ([2 x i8], [2 x i8]* @_RS, i32 0, i32 0)  
9 @_FS = private unnamed_addr constant [2 x i8] c" \00", align 1  
10 @FS = global i8* @getelementptr inbounds ([2 x i8], [2 x i8]* @_FS, i32 0, i32 0)  
11 @0 = global [7 x i8] c"'[0]*'\00"  
12 @1 = global [9 x i8] c"'[0-9]*'\00"  
13 @fmt = private unnamed_addr constant [4 x i8] c"%s\0A\00"  
14 @2 = global [9 x i8] c"'[0-9]*'\00"  
15 @3 = global [9 x i8] c"'[0-9]*'\00"  
16 @fmt.1 = private unnamed_addr constant [4 x i8] c"%s\0A\00"  
17 @fmt.2 = private unnamed_addr constant [4 x i8] c"%s\0A\00"  
18 @4 = global [7 x i8] c"'[0]*'\00"  
19 @5 = global [9 x i8] c"'[0-9]*'\00"  
20 @fmt.3 = private unnamed_addr constant [4 x i8] c"%s\0A\00"  
21 @6 = global [9 x i8] c"'[0-9]*'\00"  
22 @7 = global [9 x i8] c"'[0-9]*'\00"  
23 @fmt.4 = private unnamed_addr constant [4 x i8] c"%s\0A\00"  
24 @fmt.5 = private unnamed_addr constant [4 x i8] c"%s\0A\00"  
25 @8 = global [6 x i8] c"'[0]'\00"  
26 @9 = global [5 x i8] c"1234\00"  
27 @fmt.6 = private unnamed_addr constant [4 x i8] c"%s\0A\00"  
28 @10 = global [9 x i8] c"'[0-9]*'\00"  
29 @11 = global [5 x i8] c"1234\00"  
30 @fmt.7 = private unnamed_addr constant [4 x i8] c"%s\0A\00"  
31 @fmt.8 = private unnamed_addr constant [4 x i8] c"%s\0A\00"  
32 @12 = global [6 x i8] c"'[0]'\00"  
33 @13 = global [5 x i8] c"1234\00"
```

```

34 @fmt.9 = private unnamed_addr constant [4 x i8] c"%s\0A\00"
35 @14 = global [9 x i8] c"'[0-9]*'\00"
36 @15 = global [5 x i8] c"1234\00"
37 @fmt.10 = private unnamed_addr constant [4 x i8] c"%s\0A\00"
38
39 declare i32 @printf(i8*, ...)
40
41 declare i8* @int_to_string(i32)
42
43 declare i8* @bool_to_string(i1)
44
45 declare i32 @string_to_int(i8*)
46
47 declare i8* @concat(i8*, i8*)
48
49 declare i1 @comp(i8*, i8*)
50
51 declare i1 @ncomp(i8*, i8*)
52
53 declare i1 @equals(i8*, i8*)
54
55 declare i1 @nequals(i8*, i8*)
56
57 declare i8* @rgx_to_string(i8*)
58
59 declare i8* @access(i8*, i32)
60
61 declare i1 @streq(i8*, i8*)
62
63 declare i1 @strneq(i8*, i8*)
64
65 declare i1 @strless(i8*, i8*)
66
67 declare i1 @strgreater(i8*, i8*)
68
69 declare i1 @strgeq(i8*, i8*)
70
71 declare i1 @strleq(i8*, i8*)
72
73 declare i32 @numfields(i8*)
74
75 declare %Array* @initList(i64, i32)
76
77 declare %Node* @addFront(%Array*, i64)
78
79 declare i64 @getElement(%Array*, i32)
80
81 declare i32 @assignElement(%Array*, i32, i64)
82
83 declare i8 @reverseList(%Array*)
84
85 declare i32 @length(%Array*)
86
87 declare i64 @removeNode(%Array*, i32)
88
89 declare i8 @insertElement(%Array*, i32, i64)

```

```

90
91 declare i32 @compareInts(i64, i64)
92
93 declare i32 @compareBools(i64, i64)
94
95 declare i32 @compareStrs(i64, i64)
96
97 declare i1 @contains(%Array*, i64, i32 (i64, i64)*)
98
99 declare i32 @findIndexOfNode(%Array*, i64, i32 (i64, i64)*)
100
101 define void @loop(i8* %line) {
102 entry:
103   %line1 = alloca i8*
104   store i8* %line, i8** %line1
105   ret void
106 }
107
108 define void @end() {
109 entry:
110   %equals = call i1 @equals(i8* getelementptr inbounds ([9 x i8], [9 x i8]* @1,
111     i32 0, i32 0), i8* getelementptr inbounds ([7 x i8], [7 x i8]* @0, i32 0, i32
112     0))
113   %bool_to_string = call i8* @bool_to_string(i1 %equals)
114   %printf = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4 x
115     i8]* @fmt, i32 0, i32 0), i8* %bool_to_string)
116   %equals1 = call i1 @equals(i8* getelementptr inbounds ([9 x i8], [9 x i8]* @3,
117     i32 0, i32 0), i8* getelementptr inbounds ([9 x i8], [9 x i8]* @2, i32 0, i32
118     0))
119   %bool_to_string2 = call i8* @bool_to_string(i1 %equals1)
120   %printf3 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4
121     x i8]* @fmt.1, i32 0, i32 0), i8* %bool_to_string2)
122   %int_to_string = call i8* @int_to_string(i32 99)
123   %printf4 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4
124     x i8]* @fmt.2, i32 0, i32 0), i8* %int_to_string)
125   %nequals = call i1 @nequals(i8* getelementptr inbounds ([9 x i8], [9 x i8]* @5,
126     i32 0, i32 0), i8* getelementptr inbounds ([7 x i8], [7 x i8]* @4, i32 0, i32
127     0))
128   %bool_to_string5 = call i8* @bool_to_string(i1 %nequals)
129   %printf6 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4
130     x i8]* @fmt.3, i32 0, i32 0), i8* %bool_to_string5)
131   %nequals7 = call i1 @nequals(i8* getelementptr inbounds ([9 x i8], [9 x i8]* @7,
132     i32 0, i32 0), i8* getelementptr inbounds ([9 x i8], [9 x i8]* @6, i32 0,
133     i32 0))
134   %bool_to_string8 = call i8* @bool_to_string(i1 %nequals7)
135   %printf9 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4
136     x i8]* @fmt.4, i32 0, i32 0), i8* %bool_to_string8)
137   %int_to_string10 = call i8* @int_to_string(i32 99)
138   %printf11 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4
139     x i8]* @fmt.5, i32 0, i32 0), i8* %int_to_string10)
140   %comp = call i1 @comp(i8* getelementptr inbounds ([5 x i8], [5 x i8]* @9, i32 0,
141     i32 0), i8* getelementptr inbounds ([6 x i8], [6 x i8]* @8, i32 0, i32 0))
142   %bool_to_string12 = call i8* @bool_to_string(i1 %comp)
143   %printf13 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4
144     x i8]* @fmt.6, i32 0, i32 0), i8* %bool_to_string12)
145   %comp14 = call i1 @comp(i8* getelementptr inbounds ([5 x i8], [5 x i8]* @11, i32

```

```

    0, i32 0), i8* getelementptr inbounds ([9 x i8], [9 x i8]* @10, i32 0, i32
    0))
130 %bool_to_string15 = call i8* @bool_to_string(i1 %comp14)
131 %printf16 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4
    x i8]* @fmt.7, i32 0, i32 0), i8* %bool_to_string15)
132 %int_to_string17 = call i8* @int_to_string(i32 99)
133 %printf18 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4
    x i8]* @fmt.8, i32 0, i32 0), i8* %int_to_string17)
134 %ncomp = call i1 @ncomp(i8* getelementptr inbounds ([5 x i8], [5 x i8]* @13, i32
    0, i32 0), i8* getelementptr inbounds ([6 x i8], [6 x i8]* @12, i32 0, i32
    0))
135 %bool_to_string19 = call i8* @bool_to_string(i1 %ncomp)
136 %printf20 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4
    x i8]* @fmt.9, i32 0, i32 0), i8* %bool_to_string19)
137 %ncomp21 = call i1 @ncomp(i8* getelementptr inbounds ([5 x i8], [5 x i8]* @15,
    i32 0, i32 0), i8* getelementptr inbounds ([9 x i8], [9 x i8]* @14, i32 0,
    i32 0))
138 %bool_to_string22 = call i8* @bool_to_string(i1 %ncomp21)
139 %printf23 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4
    x i8]* @fmt.10, i32 0, i32 0), i8* %bool_to_string22)
140 ret void
141 }
142 a

```

## 7 Lessons Learned

### 7.1 Ashley

I learned a lot this semester about compilers and functional programming. Above all, I learned about working in a team. I realized the importance of effectively delegating tasks and ensuring that they're actually getting completed on time. It is equally important to efficiently manage your time with your group and keep open communication throughout the entire semester. It is also extremely important to have weekly group meetings and weekly check-ins with your TA.

I'd advise future groups to make sure to really understand what needs to get done and plan for how it will get done before jumping into coding. Jumping straight into coding is never the answer. Be sure to break down everything piece by piece so you can build up your project incrementally, and set deadlines for when you want to complete each part. Make sure to constantly test

along the way as well, until you literally do not want to write another test ever again. Finally, take advantage of the TAs that will be happy to help you along the way!

## 7.2 Christine

This was my first time really using functional programming, so there was a steep learning curve involved in learning OCaml. I also learned a lot about working in a group, which requires dividing tasks but still knowing the entire architecture and necessitates constant and open communication. Find a team that will contribute even work (which my team was awesome at)!

I would advise future groups to think carefully about designing the parser and the overall structure of the program, since you don't want to be changing things at the last minute. Everyone says this, but start tackling codegen early because it takes a lot of time to adjust to and learn OCaml and the LLVM module (especially because the documentation is quite sparse). Outsource the hard things in C if you can.

## 7.3 Melanie

I learned a lot about working in a team this semester. It's important for everyone to be on the same page all the time – our group found it useful to write down the discrete tasks required of everyone each week, and to continuously check in with each other to see who needed help. One of our more challenging roadblocks was when we realized that some code needed to be changed in parser during the last fourth of the project – as a result, we had to modify a large chunk of code. It's best to make sure that your parser (and even scanner) are exactly how you want them before implementing large blocks of codegen.

It's also okay to scale back some parts of your language, but try to think about what the core functionality is before starting to implement features.

Implement the most important features first— if you don't get around to the less important ones, it's ok. One of the core functions of our language is file reading and looping, (and this was probably one of the more challenging aspects of the program), so we implemented this first. Some of the features we implemented towards the end of the semester were less important, and had we not gotten them to work, it wouldn't have been a big deal.

Finally, please actually learn how pattern matching works. The number of frivolous if statements I wrote in OCaml before realizing how simple and robust pattern matching is was pretty astounding.

## 7.4 Victoria

This project involved a lot of group collaboration, which meant making sure that all members of the team had at least a vague idea of what everyone else was working on, even if they weren't actually involved. It's important that every member of the group has a good grasp on the language and the frontend of the language, no matter who actually implements them, since this will provide a good foundation for everything that follows. It is also important for the group to know what is expected of them, which we achieved by setting concrete goals to be accomplished before every time we met.

We were very ambitious in constructing our language, which wasn't necessarily a bad thing, but it meant that we had to think critically about what was feasible and most important to our language in later parts. It would have been better to hash this out when writing our LRM instead of later in the process.

## 8 Code Listing

scanner.mll

Authors: Mel, Ashley, Christine, Victoria

```
1 { open Parser }  
2
```

```

3 rule token = parse
4   [' ' '\t' '\r' '\n'] { token lexbuf } (* Whitespace *)
5 | "#"           { comment lexbuf } (* Comments *)
6 | "("           { LPAREN }
7 | ")"           { RPAREN }
8 | "{"           { LCURLY }
9 | "}"           { RCURLY }
10 | "["          { LSQUARE }
11 | "]"          { RSQUARE }
12 | ";"          { SEMI }
13 | '''          { read_string (Buffer.create 17) lexbuf }
14 | '''          { read_rgx (Buffer.create 17) lexbuf }
15 | "&"          { STRCAT }
16 | "$"          { DOLLAR }
17 | ","          { COMMA }
18 | "+"          { PLUS }
19 | "-"          { MINUS }
20 | "/"          { DIVIDE }
21 | "*"          { MULTIPLY }
22 | "="          { ASSIGN }
23 | "=="         { EQUALS }
24 | "!="         { NEQ }
25 | "<"          { LT }
26 | "<="        { LEQ }
27 | ">"          { GT }
28 | ">="        { GEQ }
29 | "&&"         { AND }
30 | "||"         { OR }
31 | "+="         { PLUSEQ }
32 | "-="         { MINUSEQ }
33 | "++"         { INCREMENT }
34 | "--"         { DECREMENT }
35 | "if"         { IF }
36 | "else"       { ELSE }
37 | "while"      { WHILE }
38 | "for"        { FOR }
39 | "!"          { NOT }
40 | "return"     { RETURN }
41 | "function"   { FUNCTION }
42 | "int"        { INT }
43 | "bool"       { BOOL }
44 | "void"       { VOID }
45 | "string"     { STRING }
46 | "rgx"        { RGX }
47 | "%"          { RGXEQ }
48 | "!%"         { RGXNEQ }
49 | "~"          { RGXSTRCMP }
50 | "!~"         { RGXSTRNOT }
51 | "CONFIG"     { CONFIG }
52 | "RS"         { RS }
53 | "FS"         { FS }
54 | "NF"         { NF }
55 | "BEGIN"     { BEGIN }
56 | "LOOP"      { LOOP }
57 | "END"       { END }
58 | "true"      { TRUE }

```

```

59 | "false"      { FALSE }
60 | "in"         { IN }
61 | ['0'-'9']+ as lxm { LITERAL(int_of_string lxm) }
62 | "\" [^'\\']* \" as lxm { RGX_LITERAL(lxm) }
63 | ['a'-'z' 'A'-'Z'] ['a'-'z' 'A'-'Z' '0'-'9' '_' ]* as lxm { ID(lxm) }
64 | eof { EOF }
65 | _ as char { raise (Failure("illegal character " ^ Char.escaped char)) }
66
67 and comment = parse
68   '\n' { token lexbuf }
69 | _    { comment lexbuf }
70
71 and read_string buf =
72   parse
73   | '"'      { STRING_LITERAL (Buffer.contents buf) }
74   | '\\' '/' { Buffer.add_char buf '/'; read_string buf lexbuf }
75   | '\\' '\\' { Buffer.add_char buf '\\'; read_string buf lexbuf }
76   | '\\' 'b' { Buffer.add_char buf '\b'; read_string buf lexbuf }
77   | '\\' 'f' { Buffer.add_char buf '\012'; read_string buf lexbuf }
78   | '\\' 'n' { Buffer.add_char buf '\n'; read_string buf lexbuf }
79   | '\\' 'r' { Buffer.add_char buf '\r'; read_string buf lexbuf }
80   | '\\' 't' { Buffer.add_char buf '\t'; read_string buf lexbuf }
81   | [^ '"' '\\' ]+
82   { Buffer.add_string buf (Lexing.lexeme lexbuf);
83     read_string buf lexbuf
84   }
85 | _ { raise (Failure ("Illegal string character: " ^ Lexing.lexeme lexbuf)) }
86 | eof { raise (Failure ("String is not terminated")) }
87
88 and read_rgx buf =
89   parse
90   | '''      { RGX_LITERAL (Buffer.contents buf) }
91   | / [^ ''' ]+
92   { Buffer.add_string buf (Lexing.lexeme lexbuf);
93     read_string buf lexbuf
94   }
95 | _ { raise (Failure ("Illegal string character: " ^ Lexing.lexeme lexbuf)) }
96 | eof { raise (Failure ("String is not terminated")) }

```

## parser.mly

Authors: Mel, Ashley, Christine

```

1 %{ open Ast %}
2
3 %token LPAREN RPAREN LCURLY RCURLY LSQUARE RSQUARE SEMI COMMA
4 %token PLUS MINUS DIVIDE MULTIPLY ASSIGN STRCAT
5 %token EQUALS NEQ LT LEQ GT GEQ AND OR NOT
6 %token PLUSEQ MINUSEQ INCREMENT DECREMENT
7 %token RGXEQ RGXNEQ RGXSTRCMP RGXSTRNOT
8 %token RETURN FUNCTION
9 %token CONFIG BEGIN LOOP END
10 %token INT BOOL VOID STRING RGX TRUE FALSE
11 %token RS FS NF
12 %token IF ELSE WHILE FOR IN

```



```

13 %token DOLLAR
14 %token <int> LITERAL
15 %token <string> STRING_LITERAL
16 %token <string> RGX_LITERAL
17 %token <string> ID
18 %token EOF
19
20 %start program
21 %type <Ast.program> program
22
23 %nonassoc NOELSE
24 %nonassoc ELSE
25 %right ASSIGN PLUSEQ MINUSEQ
26 %left OR
27 %left AND
28 %left EQUALS NEQ RGXEQ RGXNEQ RGXSTRCMP RGXSTRNOT
29 %left LT LEQ GT GEQ
30 %left PLUS MINUS STRCAT
31 %left MULTIPLY DIVIDE
32 %right NOT NEG
33 %right INCREMENT DECREMENT
34 %left LSQUARE
35 %right DOLLAR
36
37 %%
38
39 program:
40     begin_block loop_block end_block config_block EOF { ($1, $2, $3, $4) }
41
42 begin_block:
43     BEGIN LCURLY global_vars_list func_list RCURLY { (List.rev $3, List.rev $4) }
44
45 loop_block:
46     LOOP LCURLY local_vars_list stmt_list RCURLY { (List.rev $3, List.rev $4) }
47
48 end_block:
49     END LCURLY local_vars_list stmt_list RCURLY { (List.rev $3, List.rev $4) }
50
51 config_block:
52     /* nothing */ { [] }
53     | CONFIG LCURLY config_expr_list RCURLY { (List.rev $3) }
54
55 primitive:
56     STRING { String }
57     | INT { Int }
58     | BOOL { Bool }
59     | RGX { Rgx }
60
61 typ:
62     VOID { Void }
63     | array_type { $1 }
64
65 array_type:
66     array_type LSQUARE RSQUARE { ArrayType($1) }
67     | primitive { $1 }
68

```

```

69 func_list:
70     /* nothing */ { [] }
71     | func_list func { $2 :: $1 }
72
73 global_vars_list:
74     /* nothing */ { [] }
75     | global_vars_list var_decl { $2 :: $1 }
76
77 local_vars_list:
78     /* nothing */ { [] }
79     | local_vars_list var_decl { $2 :: $1 }
80
81 func:
82     FUNCTION typ ID LPAREN formals_opt RPAREN LCURLY local_vars_list stmt_list
83         RCURLY
84     { { fname = $3;
85         formals = $5;
86         ret_type = $2;
87         locals = List.rev $8;
88         body = List.rev $9 } }
89
90 formals_opt:
91     /* nothing */ { [] }
92     | formals_list { List.rev $1 }
93
94 formals_list:
95     typ ID { [($1, $2)] }
96     | formals_list COMMA typ ID { ($3, $4) :: $1 }
97
98 var_decl:
99     typ ID SEMI { ($1, $2) }
100
101 config_expr_list:
102     /* nothing */ { [] }
103     | config_expr_list config_expr { $2 :: $1 }
104
105 config_expr:
106     RS ASSIGN expr SEMI { RSAssign($3) }
107     | FS ASSIGN expr SEMI { FSAssign($3) }
108
109 stmt_list:
110     /* nothing */ { [] }
111     | stmt_list stmt { $2 :: $1 }
112
113 stmt:
114     expr SEMI { Expr $1 }
115     | RETURN expr_opt SEMI { Return $2 }
116     | WHILE LPAREN expr RPAREN code_block { While($3, $5) }
117     | FOR LPAREN expr SEMI expr SEMI expr RPAREN code_block { For($3, $5, $7, $9) }
118     | FOR LPAREN ID IN ID RPAREN code_block { EnhancedFor($3, $5, $7
119         ) }
120     | IF LPAREN expr RPAREN code_block %prec NOELSE { If($3, $5, Block([]))
121         }
122     | IF LPAREN expr RPAREN code_block ELSE code_block { If($3, $5, $7) }
123
124 code_block: LCURLY stmt_list RCURLY { Block(List.rev $2) }

```

```

122
123 expr_opt:
124     /* nothing */ { Noexpr }
125     | expr         { $1 }
126
127 expr:
128     LITERAL           { Literal($1) }
129     | STRING_LITERAL  { StringLiteral($1) }
130     | RGX_LITERAL     { RgxLiteral($1) }
131     | TRUE            { BoolLit(true) }
132     | FALSE          { BoolLit(false) }
133     | ID              { Id($1) }
134     | expr PLUS expr  { Binop($1, Add, $3) }
135     | expr MINUS expr { Binop($1, Sub, $3) }
136     | expr MULTIPLY expr { Binop($1, Mult, $3) }
137     | expr DIVIDE expr { Binop($1, Div, $3) }
138     | expr EQUALS expr { Binop($1, Equal, $3) }
139     | expr NEQ expr   { Binop($1, Neq, $3) }
140     | expr LT expr    { Binop($1, Less, $3) }
141     | expr LEQ expr   { Binop($1, Leq, $3) }
142     | expr GT expr    { Binop($1, Greater, $3) }
143     | expr GEQ expr   { Binop($1, Geq, $3) }
144     | expr AND expr   { Binop($1, And, $3) }
145     | expr OR expr    { Binop($1, Or, $3) }
146     | expr PLUSEQ expr { Pluseq($1, $3) }
147     | expr MINUSEQ expr { Minuseq($1, $3) }
148     | expr INCREMENT  { Increment($1) }
149     | expr DECREMENT  { Decrement($1) }
150     | expr STRCAT expr { Strcat($1,$3) }
151     | expr RGXEQ expr  { Rgxeq($1, $3) }
152     | expr RGXNEQ expr { Rgxneq($1, $3) }
153     | expr RGXSTRCMP expr { Rgxcomp($1,$3) }
154     | expr RGXSTRNOT expr { Rgxnot($1, $3) }
155     | LSQUARE actuals_opt RSQUARE { ArrayLit($2) }
156     | expr LSQUARE expr RSQUARE { ArrayDeref($1, $3) }
157     | NOT expr           { Unop(Not, $2) }
158     | LPAREN expr RPAREN { $2 }
159     | ID LPAREN actuals_opt RPAREN { Call($1, $3) }
160     | NF                  { NumFields }
161     | DOLLAR expr        { Access($2) }
162     | MINUS expr %prec NEG { Unop(Neg, $2) }
163     | expr ASSIGN expr   { Assign($1, $3) }
164
165 actuals_opt:
166     /* nothing */ { [] }
167     | actuals_list { List.rev $1 }
168
169 actuals_list:
170     expr           { [$1] }
171     | actuals_list COMMA expr { $3 :: $1 }

```

ast.ml

Authors: Victoria, Ashley

```

1 (* Loosely based on MicroC *)
2
3 type op = Add | Sub | Mult | Div | Equal | Neq | Less | Leq | Greater |
4         Geq | And | Or
5
6 type uop = Not | Neg
7
8 type typ = Int | Bool | Void | String | Rgx | ArrayType of typ
9
10 type bind = typ * string
11
12 type expr =
13     Binop of expr * op * expr
14     | BoolLit of bool
15     | Literal of int
16     | StringLiteral of string
17     | RgxLiteral of string
18     | Id of string
19     | Assign of expr * expr
20     | Call of string * expr list
21     | Unop of uop * expr
22     | ArrayLit of expr list
23     | ArrayDeref of expr * expr
24     | Access of expr
25     | Pluseq of expr * expr
26     | Minuseq of expr * expr
27     | Increment of expr
28     | Decrement of expr
29     | Strcat of expr * expr
30     | Rgxeq of expr * expr
31     | Rgxneq of expr * expr
32     | Rgxcomp of expr * expr
33     | Rgxnot of expr * expr
34     | NumFields
35     | Noexpr
36
37 type config_expr =
38     RSAssign of expr
39     | FSAssign of expr
40
41 type stmt =
42     Return of expr
43     | Expr of expr
44     | Block of stmt list
45     | While of expr * stmt
46     | If of expr * stmt * stmt
47     | For of expr * expr * expr * stmt
48     | EnhancedFor of string * string * stmt
49
50 type func_decl = {
51     ret_type : typ;
52     fname    : string;
53     formals  : bind list;
54     locals   : bind list;
55     body     : stmt list;
56 }

```

```

57
58 type begin_list = bind list * func_decl list
59 type loop_list = bind list * stmt list
60 type end_list = bind list * stmt list
61 type config_list = config_expr list
62
63 type program = begin_list * loop_list * end_list * config_list
64
65 (* Pretty-printing functions *)
66
67 let string_of_op = function
68   Add -> "+"
69   | Sub -> "-"
70   | Mult -> "*"
71   | Div -> "/"
72   | Equal -> "=="
73   | Neq -> "!="
74   | Less -> "<"
75   | Leq -> "<="
76   | Greater -> ">"
77   | Geq -> ">="
78   | And -> "&&"
79   | Or -> "||"
80
81
82 let string_of_uop = function
83   Neg -> "-"
84   | Not -> "!"
85
86 let rec string_of_expr = function
87   Binop(e1, o, e2) ->
88     string_of_expr e1 ^ " " ^ string_of_op o ^ " " ^ string_of_expr e2
89   | BoolLit(true) -> "true"
90   | BoolLit(false) -> "false"
91   | Literal(l) -> string_of_int l
92   | StringLiteral(s) -> s
93   | RgxLiteral(r) -> r
94   | Id(s) -> s
95   | Assign(v, e) -> string_of_expr v ^ " = " ^ string_of_expr e
96   | Call(f, el) ->
97     f ^ "(" ^ String.concat ", " (List.map string_of_expr el) ^ ")"
98   (*| Rgx(r) -> r*)
99   | Unop(o, e) -> string_of_uop o ^ string_of_expr e
100  | ArrayLit(el) -> "[" ^ String.concat ", " (List.map string_of_expr el) ^ "]"
101  | ArrayDeref(v, e) -> string_of_expr v ^ "[" ^ string_of_expr e ^ "]"
102  | Access(e) -> "$" ^ string_of_expr e
103  | Strcat(e, g) -> string_of_expr e ^ "&" ^ string_of_expr g
104  | Rgxreq(e, e1) -> string_of_expr e ^ "%" ^ string_of_expr e1
105  | Rgxneq(e, e1) -> string_of_expr e ^ "!%" ^ string_of_expr e1
106  | Rgxcomp(e, e1) -> string_of_expr e ^ "~" ^ string_of_expr e1
107  | Rgxnot(e, e1) -> string_of_expr e ^ "!~" ^ string_of_expr e1
108  | Increment(e) -> string_of_expr e ^ "++"
109  | Decrement(e) -> string_of_expr e ^ "--"
110  | Plusseq(e, e1) -> string_of_expr e ^ "+=" ^ string_of_expr e1
111  | Minuseq(e, e1) -> string_of_expr e ^ "-=" ^ string_of_expr e1
112  | NumFields -> "NF"

```

```

113 | Noexpr -> ""
114
115 let string_of_config_expr = function
116   RSAssign(e) -> "RS = " ^ string_of_expr e ^ "\n"
117   | FSAssign(e) -> "FS = " ^ string_of_expr e ^ "\n"
118
119 let rec string_of_stmt = function
120   Return(expr) -> "return " ^ string_of_expr expr ^ ";\n";
121   | Expr(expr) -> string_of_expr expr ^ ";\n";
122   | Block(stmts) ->
123     "{\n" ^ String.concat "" (List.map string_of_stmt stmts) ^ "}\n"
124   | While(e, s) -> "while (" ^ string_of_expr e ^ ") " ^ string_of_stmt s
125   | If(e, s, Block([])) -> "if (" ^ string_of_expr e ^ ")\n" ^ string_of_stmt s
126   | If(e, s1, s2) -> "if (" ^ string_of_expr e ^ ")\n" ^
127     string_of_stmt s1 ^ "else\n" ^ string_of_stmt s2
128   | For(e1, e2, e3, s) ->
129     "for (" ^ string_of_expr e1 ^ " ; " ^ string_of_expr e2 ^ " ; " ^
130     string_of_expr e3 ^ ") " ^ string_of_stmt s
131   | EnhancedFor(str1, str2, s) -> "for (" ^ str1 ^ " in " ^ str2 ^ ") " ^
132     string_of_stmt s
133
134 let rec string_of_ttyp = function
135   Int -> "int"
136   | Bool -> "bool"
137   | Void -> "void"
138   | String -> "string"
139   | Rgx -> "rgx"
140   | ArrayType(t) -> string_of_ttyp t ^ "[]"
141
142 let string_of_vdecl (t, id) = string_of_ttyp t ^ " " ^ id ^ ";\n"
143
144 let string_of_fdecl fdecl =
145   "function " ^ string_of_ttyp fdecl.ret_type ^ " " ^
146   fdecl.fname ^ "(" ^ String.concat ", " (List.map snd fdecl.formals) ^
147   ")\n{\n" ^
148   String.concat "" (List.map string_of_vdecl fdecl.locals) ^
149   String.concat "" (List.map string_of_stmt fdecl.body) ^
150   "}\n"
151
152 let string_of_beginBlock (globals, funcs) =
153   "BEGIN {\n" ^ String.concat "" (List.map string_of_vdecl globals) ^ "\n" ^
154   String.concat "" (List.map string_of_fdecl funcs) ^ "\n}\n"
155
156 let string_of_loopBlock (locals, stmts) =
157   "LOOP {\n" ^ String.concat "" (List.map string_of_vdecl locals) ^ "\n" ^
158   String.concat "" (List.map string_of_stmt stmts) ^ "\n}\n"
159
160 let string_of_endBlock (locals, stmts) =
161   "END {\n" ^ String.concat "" (List.map string_of_vdecl locals) ^ "\n" ^
162   String.concat "" (List.map string_of_stmt stmts) ^ "\n}\n"
163
164 let string_of_configBlock (configs) =
165   "CONFIG {\n" ^ String.concat "" (List.map string_of_config_expr configs) ^ "\n}\n"
166
167 let string_of_program(beginBlock, loopBlock, endBlock, configBlock) =

```

```

167  "" ^ string_of_beginBlock beginBlock ^ "\n" ^
168  "" ^ string_of_loopBlock loopBlock ^ "\n" ^
169  "" ^ string_of_endBlock endBlock ^ "\n" ^
170  "" ^ string_of_configBlock configBlock

```

## sast.ml

Authors: Ashley, Victoria

```

1  open Ast
2
3  type sexpr = typ * sx
4  and sx =
5      SBinop of sexpr * op * sexpr
6      | SBoolLit of bool
7      | SLiteral of int
8      | SStringLiteral of string
9      | SRgxLiteral of string
10     | SId of string
11     | SAssign of sexpr * sexpr
12     | SCall of string * sexpr list
13     | SRgx of string
14     | SUnop of uop * sexpr
15     | SArrayLit of sexpr list
16     | SArrayDeref of sexpr * sexpr
17     | SAccess of sexpr
18     | SPluseq of sexpr * sexpr
19     | SMinuseq of sexpr * sexpr
20     | SIncrement of sexpr
21     | SDecrement of sexpr
22     | SStrcat of sexpr * sexpr
23     | SRgxseq of sexpr * sexpr
24     | SRgxneq of sexpr * sexpr
25     | SRgxcomp of sexpr * sexpr
26     | SRgxnot of sexpr * sexpr
27     | SNumFields
28     | SNoexpr
29
30  type sstmt =
31     SReturn of sexpr
32     | SExpr of sexpr
33     | SBlock of sstmt list
34     | SWhile of sexpr * sstmt
35     | SIf of sexpr * sstmt * sstmt
36     | SFor of sexpr * sexpr * sexpr * sstmt
37     | SEnhancedFor of string * string * sstmt
38
39  type sfunc_decl = {
40      sret_type : typ;
41      sfname    : string;
42      sformals  : bind list;
43      slocals   : bind list;
44      sbody     : sstmt list;
45  }
46

```

```

47 type sbegin_list = bind list * sfunc_decl list
48 type sloop_list = sfunc_decl
49 type send_list = sfunc_decl
50 type sconfig_pair = string * string (* RS * FS *)
51
52 type sprogram = sbegin_list * sloop_list * send_list * sconfig_pair
53
54 (* Pretty-printing functions *)
55
56 let rec string_of_sexpr (t, e) =
57   "(" ^ string_of_ttyp t ^ " : " ^ (match e with
58     SBinop(e1, o, e2) ->
59       string_of_sexpr e1 ^ " " ^ string_of_op o ^ " " ^ string_of_sexpr e2
60   | SBoolLit(true) -> "true"
61   | SBoolLit(false) -> "false"
62   | SLiteral(l) -> string_of_int l
63   | SStringLiteral(s) -> s
64   | SRgxLiteral(r) -> r
65   | SId(s) -> s
66   | SAssign(v, e) -> string_of_sexpr v ^ " = " ^ string_of_sexpr e
67   | SCall(f, el) ->
68     f ^ "(" ^ String.concat ", " (List.map string_of_sexpr el) ^ ")"
69   | SRgx(r) -> r
70   | SStrcat(a, b) -> string_of_sexpr a ^ string_of_sexpr b
71   | SRgxcomp(a, b) -> string_of_sexpr a ^ string_of_sexpr b
72   | SRgxnot(a, b) -> string_of_sexpr a ^ string_of_sexpr b
73   | SRgxreq(a, b) -> string_of_sexpr a ^ string_of_sexpr b
74   | SRgxneq(a, b) -> string_of_sexpr a ^ string_of_sexpr b
75   | SPluseq(a, b) -> string_of_sexpr a ^ string_of_sexpr b
76   | SMinuseq(a, b) -> string_of_sexpr a ^ string_of_sexpr b
77   | SIncrement(a) -> string_of_sexpr a
78   | SDecrement(a) -> string_of_sexpr a
79   | SUNop(o, e) -> string_of_uop o ^ string_of_sexpr e
80   | SArrayLit(el) -> "[" ^ String.concat ", " (List.map string_of_sexpr el) ^ "]"
81   | SArrayDeref(v, e) -> string_of_sexpr v ^ "[" ^ string_of_sexpr e ^ "]"
82   | SAccess(e) -> "$" ^ string_of_sexpr e
83   | SNumFields -> "NF"
84   | SNoexpr -> ""
85     ) ^ ")"
86
87 let string_of_config_pair (rs, fs) =
88   "RS = " ^ rs ^ "\nFS = " ^ fs ^ "\n"
89
90 let rec string_of_sstmt = function
91   SReturn(expr) -> "return " ^ string_of_sexpr expr ^ ";\n";
92   SExpr(expr) -> string_of_sexpr expr ^ ";\n";
93   SBlock(stmts) ->
94     "{\n" ^ String.concat "" (List.map string_of_sstmt stmts) ^ "}\n"
95   | SWhile(e, s) -> "while (" ^ string_of_sexpr e ^ ") " ^ string_of_sstmt s
96   | SIf(e, s, SBlock([])) -> "if (" ^ string_of_sexpr e ^ ")\n" ^ string_of_sstmt
97     s
98   | SIf(e, s1, s2) -> "if (" ^ string_of_sexpr e ^ ")\n" ^
99     string_of_sstmt s1 ^ "else\n" ^ string_of_sstmt s2
100  | SFor(e1, e2, e3, s) ->
101    "for (" ^ string_of_sexpr e1 ^ " ; " ^ string_of_sexpr e2 ^ " ; " ^
    string_of_sexpr e3 ^ ") " ^ string_of_sstmt s

```



```

102 | SEnhancedFor(str1, str2, s) -> "for (" ^ str1 ^ " in " ^ str2 ^ ") " ^
    | string_of_sstmt s
103
104 let string_of_sfdecl fdecl =
105   "function " ^ string_of_typ fdecl.sret_type ^ " " ^
106   fdecl.sfname ^ "(" ^ String.concat ", " (List.map snd fdecl.sformals) ^
107   ")\\n{\\n" ^
108   String.concat "" (List.map string_of_vdecl fdecl.slocals) ^
109   String.concat "" (List.map string_of_sstmt fdecl.sbody) ^
110   "\\n"
111
112 let string_of_sbeginBlock (globals, funcs) =
113   "BEGIN {\\n" ^ String.concat "" (List.map string_of_vdecl globals) ^ "\\n" ^
114   String.concat "\\n" (List.map string_of_sfdecl funcs) ^ "\\n}\\n"
115
116 let string_of_sloopBlock (loop_func) =
117   "LOOP {\\n" ^ String.concat "" (List.map string_of_vdecl loop_func.slocals) ^ "\\n"
118   ^
119   String.concat "\\n" (List.map string_of_sstmt loop_func.sbody) ^ "\\n}\\n"
120
121 let string_of_sendBlock (end_func) =
122   "END {\\n" ^ String.concat "" (List.map string_of_vdecl end_func.slocals) ^ "\\n"
123   ^
124   String.concat "\\n" (List.map string_of_sstmt end_func.sbody) ^ "\\n}\\n"
125
126 let string_of_sconfigBlock (configs) =
127   "CONFIG {\\n" ^ (string_of_config_pair configs) ^ "\\n}\\n"
128
129 let string_of_sprogram(beginBlock, loopBlock, endBlock, configBlock) =
130   "" ^ string_of_sbeginBlock beginBlock ^ "\\n" ^
131   "" ^ string_of_sloopBlock loopBlock ^ "\\n" ^
132   "" ^ string_of_sendBlock endBlock ^ "\\n" ^
133   "" ^ string_of_sconfigBlock configBlock

```

## semant.ml

Authors: Victoria

```

1 (* Loosely based on MicroC *)
2
3 open Ast
4 open Sast
5
6 module StringMap = Map.Make(String)
7
8 (* Semantic checking of the AST. Returns an SAST if successful,
9   throws an exception if something is wrong.
10   Check each global variable, then check each function *)
11
12 let check (begin_list, loop_list, end_list, config_list) =
13
14   (* Verify a list of bindings has no void types or duplicate names *)
15   let check_binds (kind : string) (binds : bind list) =
16     List.iter (function
17       (Void, b) -> raise (Failure ("illegal void " ^ kind ^ " " ^ b))

```

```

18     | _ -> () binds;
19     let rec dups = function
20         [] -> ()
21         | ((_,n1) :: (_,n2) :: _) when n1 = n2 ->
22         raise (Failure ("duplicate " ^ kind ^ " " ^ n1))
23         | _ :: t -> dups t
24     in dups (List.sort (fun (_,a) (_,b) -> compare a b) binds)
25 in
26
27 (**** Check global variables ****)
28
29 let (globals, _) = begin_list in check_binds "global" globals;
30
31
32 (**** Check functions ****)
33 (* Collect function declarations for built-in functions: no bodies *)
34 let built_in_decls =
35 let add_bind map (ftype, name, flist) = StringMap.add name {
36     ret_type = ftype;
37     fname = name;
38     formals = flist;
39     locals = []; body = [] } map
40 in List.fold_left add_bind StringMap.empty [ (Int, "string_to_int", [(String,
41     "a")]);
42     (String, "int_to_string", [(Int, "a")]);
43     (String, "bool_to_string", [(Bool, "a")]);
44     (String, "rgx_to_string", [(Rgx, "a")]);
45     (Void, "length", []);
46     (Void, "print", [(String, "a")]);
47     (Void, "nprint", [(String, "a")]);
48     (Void, "contains", []);
49     (Int, "index_of", []);
50     (Void, "insert", []);
51     (Void, "delete", []);
52     (Void, "LOOP", []);
53     (Void, "END", []);
54     (Void, "loop", []);
55     (Void, "end", [])]
56 in
57
58 (* Add function name to symbol table *)
59 let add_func map fd =
60     let built_in_err = "function " ^ fd.fname ^ " may not be defined"
61     and dup_err = "duplicate function " ^ fd.fname
62     and make_err er = raise (Failure er)
63     and n = fd.fname (* Name of the function *)
64     in match fd with (* No duplicate functions or redefinitions of built-ins *)
65         _ when StringMap.mem n built_in_decls -> make_err built_in_err
66         | _ when StringMap.mem n map -> make_err dup_err
67         | _ -> StringMap.add n fd map
68 in
69 (* Collect all function names into one symbol table *)
70 let (_, functions) = begin_list in let function_decls = List.fold_left add_func
71     built_in_decls functions

```

```

72 in
73
74 (* Return a function from our symbol table *)
75 let find_func s =
76   try StringMap.find s function_decls
77   with Not_found -> raise (Failure ("unrecognized function " ^ s))
78 in
79
80 let (loop_locals,loop_stmts) = loop_list in check_binds "local" loop_locals;
81
82 let (end_locals,end_stmts) = end_list in check_binds "local" end_locals;
83
84 let check_function func =
85   (* Make sure no formals or locals are void or duplicates *)
86   check_binds "formal" func.formals;
87   check_binds "local" func.locals;
88
89   (* Raise an exception if the given rvalue type cannot be assigned to
90    the given lvalue type *)
91   let check_assign lvaluet rvaluet err =
92     if (lvaluet = rvaluet) then lvaluet else raise (Failure err)
93   in
94
95   (* Build local symbol table of variables for this function *)
96   let symbols = List.fold_left (fun m (ty, name) -> StringMap.add name ty m) (*
97     this might need to be changed later*)
98     StringMap.empty (globals @ func.formals @ func.locals )
99   in
100
101   (* Return a variable from our local symbol table *)
102   let type_of_identifier s =
103     try StringMap.find s symbols
104     with Not_found -> raise (Failure ("undeclared identifier " ^ s))
105   in
106   (* Return a semantically-checked expression, i.e., with a type *)
107   let rec expr = function
108     Literal l -> (Int, SLiteral l)
109     | StringLiteral l -> (String, SStringLiteral l)
110     | BoolLit l -> (Bool, SBoolLit l)
111     | RgxLiteral l -> (Rgx, SRgxLiteral l)
112     | Noexpr -> (Void, SNoexpr)
113     | Id s -> (type_of_identifier s, SId s)
114     | Access(a) as acc ->
115       let (a, a') = expr a in
116       if a <> Int then
117         raise (Failure ("incorrect access in " ^ string_of_expr acc))
118       else (String, SAccess(a, a'))
119     | ArrayLit(l) ->
120       if List.length l > 0 then
121         let typ = expr(List.nth l 0) in
122         let (arraytype, _) = typ in
123         let check_array e =
124           let (et, e') = expr e in (et, e')
125           in let l' = List.map check_array l in
126             let types e =
127               let (a, _) = expr e in

```

```

127         if a <> arraytype then
128             raise(Failure("array of different types,
129                 expected " ^
130                 string_of_typ arraytype ^ " found " ^
131                 string_of_typ a))
132         in let _ = List.map types l in (ArrayType(
133             arraytype), SArrayLit(l'))
134     else (Void, SArrayLit([]))
135 | ArrayDeref (e1, e2) as e ->
136     let (arr, e1') = expr e1 and (num, e2') = expr e2 in
137     if num <> Int then raise(Failure("Int expression expected in " ^
138         string_of_expr e))
139     else let find_typ arr =
140         match arr with
141         | Bool | String | Rgx | Void | Int ->
142             raise(Failure("array deref should be called on
143                 array, not " ^ string_of_typ arr))
144         | _ -> raise(Failure("array deref should be called on
145             array, not " ^ string_of_typ arr))
146     in (find_typ arr, SArrayDeref((arr, e1'), (num, e2')))
147 | NumFields -> (Int, SNumFields)
148 | Assign(NumFields, _) -> raise (Failure ("illegal assignment of NF"))
149 | Assign(e1, e2) as ex ->
150     let check_expr typ e =
151         let (t, et') =
152             match e with
153             | ArrayLit(l) ->
154                 if List.length l > 0 then expr e
155                 else (typ, SArrayLit([]))
156             | _ -> expr e
157         in (t, et') in
158     let (lt, e1') = expr e1
159     in let (rt, e2') = check_expr lt e2 in
160     let err = "illegal assignment " ^ string_of_typ lt ^ " = " ^
161         string_of_typ rt ^ " in " ^ string_of_expr ex
162     in (check_assign lt rt err, SAssign((lt, e1'), (rt, e2')))
163 | Unop(op, e) as ex ->
164     let (t, e') = expr e in
165     let ty =
166         match op with
167         | Neg when t = Int -> Int
168         | Not when t = Bool -> Bool
169         | _ -> raise (Failure ("illegal unary operator " ^ string_of_uop op
170             ^ string_of_typ t ^ " in " ^ string_of_expr ex))
171     in (ty, SUnop(op, (t, e')))
172 | Binop(e1, op, e2) as e ->
173     let (t1, e1') = expr e1 and (t2, e2') = expr e2 in
174     let same = t1 = t2 in
175     (* Determine expression type based on operator and operand types *)
176     let ty =
177         match op with
178         | Add | Sub | Mult | Div when same && t1 = Int -> Int
179         | Equal | Neq when same -> Bool
180         | Less | Leq | Greater | Geq when same && (t1 = Int || t1 = String)
181         -> Bool
182         | And | Or when same && t1 = Bool -> Bool
183         | _ -> raise(Failure ("illegal binary operator " ^ string_of_typ

```

```

175         t1 ^ " " ^ string_of_op op ^ " " ^
176             string_of_typ t2 ^ " in " ^ string_of_expr e))
177 | Increment(a) as e ->
178     let (et, e') = expr a in
179     if (et <> Int) then raise (Failure("Int expected for " ^ string_of_expr e
180         ))
181     else (Int, SIncrement(et, e'))
182 | Decrement(a) as e ->
183     let (et, e') = expr a in
184     if (et <> Int) then raise (Failure("Int expected for " ^ string_of_expr e
185         ))
186     else (Int, SDecrement(et, e'))
187 | Strcat(e1, e2) as e ->
188     let (t1, e1') = expr e1 and (t2, e2') = expr e2 in
189     if (t1 <> String || t2 <> String) then raise(Failure("Strings expected
190         for " ^ string_of_expr e))
191     else (String, SStrcat((t1, e1'), (t2, e2')))
192 | Pluseq(e1, e2) as e ->
193     let (t1, e1') = expr e1 and (t2, e2') = expr e2 in
194     if (t1 <> Int || t2 <> Int) then raise (Failure("Ints are expected for "
195         ^ string_of_expr e))
196     else (Int, SPluseq((t1, e1'), (t2, e2')))
197 | Minuseq(e1, e2) as e ->
198     let (t1, e1') = expr e1 and (t2, e2') = expr e2 in
199     if (t1 <> Int || t2 <> Int) then raise (Failure("Ints are expected for "
200         ^ string_of_expr e))
201     else (Int, SMinuseq((t1, e1'), (t2, e2')))
202 | Rgxeq(e1, e2) as e ->
203     let (t1, e1') = expr e1 and (t2, e2') = expr e2 in
204     if (t1 <> Rgx || t2 != Rgx) then raise(Failure("Rgx expected for " ^
205         string_of_expr e))
206     else (Bool, SRgxeq((t1, e1'), (t2, e2')))
207 | Rgxneq(e1, e2) as e ->
208     let (t1, e1') = expr e1 and (t2, e2') = expr e2 in
209     if (t1 <> Rgx || t2 <> Rgx) then raise(Failure("Rgx expected for " ^
210         string_of_expr e))
211     else (Bool, SRgxneq((t1, e1'), (t2, e2')))
212 | Rgxcomp(e1, e2) as e ->
213     let (t1, e1') = expr e1 and (t2, e2') = expr e2 in
214     if ((t1 = Rgx && t2 = String) || (t1 = String && t2 = Rgx)) then
215         (Bool, SRgxcomp((t1, e1'), (t2, e2')))
216     else raise(Failure("Different types expected for " ^ string_of_expr e))
217 | Rgxnot(e1, e2) as e ->
218     let (t1, e1') = expr e1 and (t2, e2') = expr e2 in
219     if ((t1 = Rgx && t2 = String) || (t1 = String && t2 = Rgx)) then
220         (Bool, SRgxnot((t1, e1'), (t2, e2')))
221 else raise(Failure("Different types expected for " ^ string_of_expr e))
222 | Call("length", args) as length ->
223     if List.length args != 1 then raise (Failure("expecting one argument for
224         " ^ string_of_expr length))
225     else let (et, e') = expr (List.nth args 0) in
226     if (et = String || et = Bool || et = Void || et = Rgx || et = Int) then
227         raise (Failure("illegal argument found " ^
228             string_of_typ et ^ " arraytype expected in " ^ string_of_expr
229                 length))

```

```

221     else (Int, SCall("length", [(et, e')]))
222 | Call ("insert", args) as insert ->
223     if List.length args !=3 then raise (Failure("expecting three arguments
224     for " ^ string_of_expr insert))
225     else let (t1, e1') = expr (List.nth args 0) and (t2, e2') = expr(List.
226     nth args 1) and (t3, e3') = expr(List.nth args 2) in
227     if t2 <> Int then raise (Failure("expecting index argument for insert
228     but had " ^ string_of_ttyp t2))
229     else
230         (match t1 with
231         String | Bool | Void | Rgx | Int -> raise (Failure("illegal
232         argument found " ^ string_of_ttyp t1 ^ " expected arraytype")
233         )
234         | ArrayType(t) ->
235             if (string_of_ttyp t = string_of_ttyp(t3) && t3 !=
236             Void)
237             then (Void, SCall("insert", [(t1, e1');(t2, e2');
238             (t3, e3')]))
239             else raise(Failure("cannot perform insert on " ^
240             string_of_ttyp t1 ^ " and " ^
241             string_of_ttyp t3)))
242 | Call("delete", args) as delete ->
243     if List.length args != 2 then raise (Failure("expecting two arguments
244     for " ^ string_of_expr delete))
245     else let (t1, e1') = expr (List.nth args 0) and (t2, e2') = expr (List.
246     nth args 1) in
247     if t2 <> Int then raise (Failure("expecting index argument for delete
248     but had " ^ string_of_ttyp t2))
249     else (t1, SCall("delete", [(t1, e1');(t2, e2')]))
250 | Call("contains", args) as contains ->
251     if List.length args != 2 then raise (Failure("expecting two arguments
252     for " ^ string_of_expr contains))
253 else let (t1, e1') = expr (List.nth args 0) and (t2, e2') = expr (List.nth
254     args 1) in
255     (match t1 with
256     String | Bool | Void | Rgx | Int ->
257         raise (Failure("illegal argument found " ^ string_of_ttyp t1 ^ "
258         arraytype expected"))
259     | ArrayType(t) ->
260         if (string_of_ttyp(t) = string_of_ttyp(t2) && t2 != Void)
261         then (Bool, SCall("contains", [(t1, e1');(t2, e2')]))
262         else raise(Failure("cannot perform contains on " ^ string_of_ttyp
263         (t1) ^ " and " ^ string_of_ttyp(t2)))
264 | Call("index_of", args) as index_of ->
265     if List.length args != 2 then raise (Failure("expecting two arguments
266     for " ^ string_of_expr index_of))
267 else let (t1, e1') = expr (List.nth args 0) and (t2, e2') = expr (List.nth
268     args 1) in
269     (match t1 with
270     String | Bool | Void | Rgx | Int ->
271         raise (Failure("illegal argument found " ^ string_of_ttyp t1 ^ "
272         arraytype expected"))
273     | ArrayType(t) ->
274         if (string_of_ttyp(t) = string_of_ttyp(t2) && t2 != Void)
275         then (Int, SCall("index_of", [(t1, e1');(t2, e2')]))
276         else raise(Failure("cannot perform index_of on " ^ string_of_ttyp(t1) ^ "

```

```

259         and " ^ string_of_typ(t2))))
260 | Call(fname, args) as call ->
261     let fd = find_func fname in
262     let param_length = List.length fd.formals in
263     if List.length args != param_length then
264         raise (Failure ("expecting " ^ string_of_int param_length ^
265             " arguments in " ^ string_of_expr call))
266     else let check_call (ft, _) e =
267         let (et, e') = expr e in
268         let err = "illegal argument found " ^ string_of_typ et ^
269             " expected " ^ string_of_typ ft ^ " in " ^ string_of_expr e
270         in (check_assign ft et err, e')
271         in
272         let args' = List.map2 check_call fd.formals args
273         in (fd.ret_type, SCall(fname, args'))
274
275 in
276
277 let check_bool_expr e =
278     let (t', e') = expr e
279     and err = "expected Boolean expression in " ^ string_of_expr e
280     in if t' != Bool then raise (Failure err) else (t', e')
281
282 in
283
284 (* Return a semantically-checked statement i.e. containing sexprs *)
285 let rec check_stmt = function
286     Expr e -> SExpr (expr e)
287     | If(p, b1, b2) -> SIf(check_bool_expr p, check_stmt b1, check_stmt b2)
288     | For(e1, e2, e3, st) ->
289     SFor(expr e1, check_bool_expr e2, expr e3, check_stmt st)
290     | EnhancedFor(s1, s2, st) ->
291         let s2_type = type_of_identifier s2 in
292         (match s2_type with
293         Bool | Rgx | String | Int | Void ->
294             raise (Failure("cannot iterate over type " ^ string_of_typ
295                 s2_type))
296         | ArrayType(t) ->
297             if (string_of_typ t = string_of_typ (type_of_identifier s1))
298             then SEnhancedFor(s1, s2, check_stmt st)
299             else raise(Failure("mismatch in " ^ string_of_typ (
300                 type_of_identifier s1) ^ " and " ^ string_of_typ s2_type)))
301     | While(p, s) -> SWhile(check_bool_expr p, check_stmt s)
302     | Return e -> let (t, e') = expr e in
303         if t = func.ret_type then SReturn (t, e')
304         else raise (Failure ("return gives " ^ string_of_typ t ^ " expected " ^
305             string_of_typ func.ret_type ^ " in " ^ string_of_expr e))
306
307 (* A block is correct if each statement is correct and nothing
308     follows any Return statement.  Nested blocks are flattened. *)
309 | Block s1 ->
310     let rec check_stmt_list = function
311         [Return _ as s] -> [check_stmt s]
312         | Return _ :: _ -> raise (Failure "nothing may follow a return")
313         | Block s1 :: ss -> check_stmt_list (s1 @ ss) (* Flatten blocks *)
314         | s :: ss -> check_stmt s :: check_stmt_list ss
315         | [] -> []
316     in SBlock(check_stmt_list s1)

```

```

311
312   in (* body of check_function *)
313   { sret_type = func.ret_type;
314     sfname = func.fname;
315     sformals = func.formals;
316     slocals = func.locals;
317     sbody = match check_stmt (Block func.body) with
318 SBlock(s1) -> s1
319   | _ -> raise (Failure ("internal error: block didn't become a block?"))
320   }
321 in
322 let check_config config_list =
323   (*
324   let expr = function
325     StringLiteral l -> (String, SStringLiteral l)
326   | _ -> raise(Failure("shouldn't have this expr in config"))
327   in
328   *)
329   let obtain_config (default_rs, default_fs) config_list =
330     let obtain_config_folder (rs, fs) = function
331       | RSAssign e -> (match rs with
332         | Some _ -> raise (Failure "RS set twice")
333         | None -> (Some (string_of_expr e), fs)
334         )
335       | FSAssign e -> (match fs with
336         | Some _ -> raise (Failure "fS set twice")
337         | None -> (fs, Some (string_of_expr e))
338         )
339     in
340     match List.fold_left
341       obtain_config_folder (None, None) config_list
342     with
343     | (Some rs, Some fs) -> rs, fs
344     | (Some rs, None) -> rs, default_fs
345     | (None, Some fs) -> default_rs, fs
346     | (None, None) -> default_rs, default_fs
347   in
348   obtain_config ("\n", " ") config_list
349 in
350 let gen_fn name fs binds stmts =
351 check_function ({
352   ret_type = Void;
353   fname = name;
354   formals = fs;
355   locals = binds;
356   body = stmts
357 })
358 in
359 ((globals, List.map check_function functions),
360 gen_fn "loop" [(String, "line")] loop_locals loop_stmts,
361 gen_fn "end" [] end_locals end_stmts,
362 check_config config_list)

```

codegen.ml



Authors: Christine Mel

```
1 (* Loosely based on MicroC *)
2
3 module L = Llvm
4 module A = Ast
5 open Sast
6
7 module StringMap = Map.Make(String)
8
9 let translate (begin_block, loop_block, end_block, config_block) =
10
11   let context = L.global_context () in
12
13   (* Create the LLVM compilation module into which
14    we will generate code *)
15   let the_module = L.create_module context "Bawk" in
16
17   (* Get types from the context *)
18   let i32_t = L.i32_type context
19       and i64_t = L.i64_type context in
20     let i8_t = L.i8_type context
21         and i1_t = L.i1_type context
22     and str_t = L.pointer_type ( L.i8_type context ) in
23     let node_t = let node_t = L.named_struct_type context "Node" in
24                 L.struct_set_body node_t [| i64_t ; L.pointer_type node_t |]
25                 false;
26                 node_t in
27     let node_p_t = L.pointer_type node_t in
28     let arr_t = let arr_t = L.named_struct_type context "Array" in
29                L.struct_set_body arr_t [| node_p_t ; i32_t ; i32_t |] false;
30                arr_t in
31     let arr_p_t = L.pointer_type arr_t in
32     let void_t = L.void_type context in
33
34   (* Return the LLVM type for a bawk type *)
35   let ltype_of_typ = function
36     A.Int    -> i32_t
37   | A.Bool   -> i1_t
38   | A.Void   -> void_t
39   | A.String -> str_t
40     | A.Rgx  -> str_t
41     | A.ArrayType _ -> arr_p_t
42   in
43
44   (* Array helper functions *)
45   let arr_elem_type = function
46     A.ArrayType t -> t
47   | _ -> raise (Failure "not an array type")
48   in
49
50   let rec arr_depth = function
51     A.ArrayType t -> (arr_depth t) + 1
52   | _ -> 0
53   in
54
```

```

55 (* Declare built-in functions *)
56 let printf_t : L.lltype =
57   L.var_arg_function_type i32_t [| L.pointer_type i8_t |] in
58 let printf_func : L.llvalue =
59   L.declare_function "printf" printf_t the_module in
60
61 let int_to_string_t : L.lltype =
62   L.function_type str_t [| i32_t |] in
63 let int_to_string_func : L.llvalue =
64   L.declare_function "int_to_string" int_to_string_t the_module in
65
66 let bool_to_string_t : L.lltype =
67   L.function_type str_t [| i1_t |] in
68 let bool_to_string_func : L.llvalue =
69   L.declare_function "bool_to_string" bool_to_string_t the_module in
70
71 let string_to_int_t : L.lltype =
72   L.function_type i32_t [| str_t |] in
73 let string_to_int_func : L.llvalue =
74   L.declare_function "string_to_int" string_to_int_t the_module in
75
76 let concat_t : L.lltype =
77   L.function_type str_t [| str_t; str_t|] in
78 let concat_func : L.llvalue =
79   L.declare_function "concat" concat_t the_module in
80
81 let rgxcomp_t : L.lltype =
82   L.function_type i1_t [| str_t; str_t |] in
83 let rgxcomp_func : L.llvalue =
84   L.declare_function "comp" rgxcomp_t the_module in
85 let rgxnot_func : L.llvalue =
86   L.declare_function "ncomp" rgxcomp_t the_module in
87
88 let rgxeq_func : L.llvalue =
89   L.declare_function "equals" rgxcomp_t the_module in
90 let rgxneq_func : L.llvalue =
91   L.declare_function "nequals" rgxcomp_t the_module in
92
93 let rgx_to_str_t : L.lltype =
94   L.function_type str_t [| str_t |] in
95 let rgx_to_str_func : L.llvalue =
96   L.declare_function "rgx_to_string" rgx_to_str_t the_module in
97
98 let access_t : L.lltype =
99   L.function_type str_t [| str_t; i32_t|] in
100 let access_func : L.llvalue =
101   L.declare_function "access" access_t the_module in
102
103 let strequals_t : L.lltype =
104   L.function_type i1_t [| str_t; str_t |] in
105 let strequals_func : L.llvalue =
106   L.declare_function "streq" strequals_t the_module in
107
108 let strnequals_t : L.lltype =
109   L.function_type i1_t [| str_t; str_t |] in
110 let strnequals_func : L.llvalue =

```

```

111     L.declare_function "strneq" strnequals_t the_module in
112
113 let strless_func : L.llvalue =
114     L.declare_function "strless" strequals_t the_module in
115
116 let strgreater_func : L.llvalue =
117     L.declare_function "strgreater" strequals_t the_module in
118
119 let strgeq_func : L.llvalue =
120     L.declare_function "strgeq" strequals_t the_module in
121
122 let strleq_func : L.llvalue =
123     L.declare_function "strleq" strequals_t the_module in
124
125 let numfields_t : L.lltype =
126     L.function_type i32_t [| str_t |] in
127 let numfields_func : L.llvalue =
128     L.declare_function "numfields" numfields_t the_module in
129
130 (* array functions *)
131 let initlist_t : L.lltype =
132     L.function_type arr_p_t [| i64_t; i32_t |] in
133 let initlist_func : L.llvalue =
134     L.declare_function "initList" initlist_t the_module in
135
136 let addfront_t : L.lltype =
137     L.function_type node_p_t [| arr_p_t; i64_t|] in
138 let addfront_func : L.llvalue =
139     L.declare_function "addFront" addfront_t the_module in
140
141 let arrayderef_t : L.lltype =
142     L.function_type i64_t [| arr_p_t; i32_t |] in
143 let arrayderef_func : L.llvalue =
144     L.declare_function "getElement" arrayderef_t the_module in
145
146 let assign_t : L.lltype =
147     L.function_type i32_t [| arr_p_t; i32_t; i64_t |] in
148 let assign_func : L.llvalue =
149     L.declare_function "assignElement" assign_t the_module in
150
151 let reverse_t : L.lltype =
152     L.function_type i8_t [| arr_p_t |] in
153 let reverse_func : L.llvalue =
154     L.declare_function "reverseList" reverse_t the_module in
155
156 let length_t : L.lltype =
157     L.function_type i32_t [| arr_p_t |] in
158 let length_func : L.llvalue =
159     L.declare_function "length" length_t the_module in
160
161 let delete_t : L.lltype =
162     L.function_type i64_t [| arr_p_t; i32_t |] in
163 let delete_func : L.llvalue =
164     L.declare_function "removeNode" delete_t the_module in
165
166 let insert_t : L.lltype =

```

```

167     L.function_type i8_t [| arr_p_t; i32_t; i64_t |] in
168 let insert_func : L.llvalue =
169     L.declare_function "insertElement" insert_t the_module in
170
171 let compare_t : L.lltype =
172     L.function_type i32_t [| i64_t ; i64_t |] in
173
174 let compare_p_t = L.pointer_type compare_t in
175
176 let compareint_func : L.llvalue =
177     L.declare_function "compareInts" compare_t the_module in
178 let comparebools_func : L.llvalue =
179     L.declare_function "compareBools" compare_t the_module in
180 let comparestr_func : L.llvalue =
181     L.declare_function "compareStrs" compare_t the_module in
182
183 let contains_t : L.lltype =
184     L.function_type i1_t [| arr_p_t ; i64_t ; compare_p_t |] in
185 let contains_func : L.llvalue =
186     L.declare_function "contains" contains_t the_module in
187
188 let indexof_t : L.lltype =
189     L.function_type i32_t [| arr_p_t ; i64_t ; compare_p_t |] in
190 let indexof_func : L.llvalue =
191     L.declare_function "findIndexOfNode" indexof_t the_module in
192
193 (*--- Build begin block: globals ---*)
194 (* Create a map of global variables after creating each *)
195 let global_vars : L.llvalue StringMap.t =
196     let global_var m (t, n) =
197         let int_init = L.const_int (ltype_of_ttyp t) 0
198         in StringMap.add n (L.define_global n int_init the_module) m
199     in
200     List.fold_left global_var StringMap.empty (fst begin_block)
201
202 in
203
204 let add_terminal_builder instr =
205     match L.block_terminator (L.insertion_block builder) with
206     Some _ -> ()
207     | None -> ignore (instr builder) in
208
209 (*---Build config block ---*)
210 let build_config_block (rs, fs) =
211     let build_global_str str name =
212         let const_str = L.define_global ("_" ^ name) (L.const_stringz context str)
213         the_module in
214     let _ = L.set_unnamed_addr true const_str;
215         L.set_global_constant true const_str;
216         L.set_linkage L.Linkage.Private const_str;
217         L.set_alignment 1 const_str;
218     in
219     let get_ptr = L.const_in_bounds_gep const_str [| L.const_int i32_t 0; L.
220         const_int i32_t 0 |] in
221     (*let _ = raise (Failure "shid") in *)

```

```

221     L.define_global name get_ptr the_module
222   in
223   ignore(build_global_str rs "RS");
224   ignore(build_global_str fs "FS");
225
226   in
227
228   (*--- Build begin block: function declarations ---*)
229   let function_decls_no_loop_end : (L.llvalue * sfunc_decl) StringMap.t =
230     let function_decl m fdecl =
231       let formal_types =
232         Array.of_list (List.map (fun (t,_) -> ltype_of_typ t) fdecl.sformals)
233       and name = fdecl.sfname
234       in let ftype = L.function_type (ltype_of_typ fdecl.sret_type) formal_types
235         in
236         StringMap.add name (L.define_function name ftype the_module, fdecl) m in
237     List.fold_left function_decl StringMap.empty (snd begin_block)
238
239   in
240
241   let function_decls_loop : (L.llvalue * sfunc_decl) StringMap.t =
242     let function_decl m fdecl =
243       let formal_types =
244         Array.of_list (List.map (fun (t,_) -> ltype_of_typ t) fdecl.sformals)
245       and name = fdecl.sfname
246       in let ftype = L.function_type (ltype_of_typ fdecl.sret_type) formal_types
247         in
248         StringMap.add name (L.define_function name ftype the_module, fdecl) m in
249     function_decl function_decls_no_loop_end (loop_block) in
250
251   let function_decls : (L.llvalue * sfunc_decl) StringMap.t =
252     let function_decl m fdecl =
253       let formal_types =
254         Array.of_list (List.map (fun (t,_) -> ltype_of_typ t) fdecl.sformals)
255       and name = fdecl.sfname
256       in let ftype = L.function_type (ltype_of_typ fdecl.sret_type) formal_types
257         in
258         StringMap.add name (L.define_function name ftype the_module, fdecl) m in
259     function_decl function_decls_loop (end_block)
260
261   in
262   (*--- Build function bodies defined in BEGIN block ---*)
263   let build_function_body fdecl =
264     let (the_function, _) = StringMap.find fdecl.sfname function_decls in
265     let func_builder = L.builder_at_end context (L.entry_block the_function) in
266     let string_format_str builder = L.build_global_stringptr "%s\n" "fmt" builder
267     in
268     let nstring_format_str builder = L.build_global_stringptr "%s" "fmt" builder
269     in
270     let local_vars =
271       let add_formal m (t, n) p =
272         L.set_value_name n p; (* p = LLVM value of param from function declaration
273           we created earlier, n = name from fdecl *)
274       in
275       let local = L.build_alloca (ltype_of_typ t) n func_builder in

```

```

271     ignore (L.build_store p local func_builder);
272     StringMap.add n local m
273
274     and add_local m (t, n) =
275         let local_var = L.build_alloca (ltype_of_typ t) n func_builder
276         in StringMap.add n local_var m
277
278     in
279
280     let formals = List.fold_left2 add_formal StringMap.empty fdecl.sformals
281         (Array.to_list (L.params the_function)) in
282
283     List.fold_left add_local formals fdecl.slocals
284
285     in
286
287     let lookup n = try StringMap.find n local_vars
288         with Not_found -> StringMap.find n global_vars
289
290     in
291
292     let rec expr builder((ty,e): sexpr) = match e with
293     SStringLiteral s -> let l = L.define_global "" (L.const_stringz context s)
294         the_module in
295         L.const_bitcast (L.const_gep l [|L.const_int i32_t 0|]) str_t
296     | SBoolLit b -> L.const_int i1_t (if b then 1 else 0)
297     | SLiteral i -> L.const_int i32_t i
298     | SNoexpr -> L.const_int i32_t 0
299     | SArrayLit a -> array_gen builder ty a
300     | SId i -> L.build_load (lookup i) i builder
301     | SAssign (e1, e2) ->
302         let (_, e) = e1 in
303         let rhs = expr builder e2 in
304         (match e with
305         SId i -> ignore (L.build_store rhs (lookup i) builder); rhs
306         | SArrayDeref (ar, idx) ->
307             let (ty, _) = ar in
308             let arr_type =
309                 match ty with
310                 A.ArrayType t -> t
311                 | _ -> raise(Failure "ArrayDeref assign is not an array") in
312             let long =
313                 match arr_type with
314                 A.Int | A.Bool -> L.build_zext rhs i64_t "arrayDerefAssign"
315                 builder
316                 | A.String | A.Rgx -> L.build_pointercast rhs i64_t "
317                 arrayDerefAssign" builder
318                 | A.ArrayType _ -> L.build_pointercast rhs i64_t "
319                 arrayDerefAssign" builder
320                 | _ -> raise(Failure "unmatched type")
321             in ignore (L.build_call assign_func [| expr builder ar; expr
322                 builder idx; long |] "assignElement" builder); rhs
323         | _ -> raise (Failure "No match on left"))
324     | SArrayDeref (ar, idx) ->
325         let (ty, _) = ar in
326         let arr_type =

```

```

322         match ty with
323         | A.ArrayType t -> t
324         | _ -> raise (Failure "ArrayDeref is not an array") in
325     let v = L.build_call arrayderef_func [| expr builder ar; expr builder idx
326         |] "getElement" builder in
327     (match arr_type with
328     | A.String | A.Rgx -> L.build_inttoptr v str_t "arrayDeref" builder
329     | A.Int -> L.build_trunc v i32_t "arrayDeref" builder
330     | A.Bool -> L.build_trunc v i1_t "arrayDeref" builder
331     | A.ArrayType _ -> L.build_inttoptr v arr_p_t "arrayDeref" builder
332     | _ -> raise (Failure "Cannot find array type"))
333 | SBinop(e1, op, e2) ->
334     let e1' = expr builder e1
335     and e2' = expr builder e2 in
336     let int_binop op =
337         (match op with
338         | A.Add -> L.build_add
339         | A.Sub -> L.build_sub
340         | A.Mult -> L.build_mul
341         | A.Div -> L.build_sdiv
342         | A.And -> L.build_and
343         | A.Or -> L.build_or
344         | A.Equal -> L.build_icmp L.Icmp.Eq
345         | A.Neq -> L.build_icmp L.Icmp.Ne
346         | A.Less -> L.build_icmp L.Icmp.Slt
347         | A.Leq -> L.build_icmp L.Icmp.Sle
348         | A.Greater -> L.build_icmp L.Icmp.Sgt
349         | A.Geq -> L.build_icmp L.Icmp.Sge
350         ) e1' e2' "tmp" builder in
351     let str_binop op =
352         (match op with
353         | A.Equal -> L.build_call strequals_func [| e1'; e2' |] "streq"
354         | A.Neq -> L.build_call strnequals_func [| e1'; e2' |] "strneq"
355         | A.Greater -> L.build_call strgreater_func [| e1'; e2' |] "strgreater"
356         | A.Less -> L.build_call strless_func [| e1'; e2' |] "strless" builder
357         | A.Leq -> L.build_call strleq_func [| e1'; e2' |] "strleq" builder
358         | A.Geq -> L.build_call strgeq_func [| e1'; e2' |] "strgeq" builder
359         | _ -> raise (Failure "Invalid operation on string"))
360     ) in
361     let bool_binop op =
362         (match op with
363         | A.Equal -> L.build_icmp L.Icmp.Eq
364         | A.Neq -> L.build_icmp L.Icmp.Ne
365         | A.And -> L.build_and
366         | A.Or -> L.build_or
367         | _ -> raise (Failure "Boolean operator not supported")) e1' e2' "
368         tmp" builder in
369     if (L.type_of e1' = i32_t) then int_binop op
370     else if (L.type_of e1' = i1_t) then bool_binop op
371     else str_binop op
372 | SUnop(uop, e) ->
373     let e' = expr builder e in
374     (match uop with

```

```

373     A.Neg -> L.build_neg
374     | A.Not -> L.build_not
375     ) e' "tmp" builder;
376 | SStrcat(e1, e2) -> L.build_call concat_func [| expr builder e1; expr
    builder e2 |] "concat" builder
377 | SRgxcomp (e1, e2) -> L.build_call rgxcomp_func [| expr builder e1; expr
    builder e2 |] "comp" builder
378 | SRgxnot (e1, e2) -> L.build_call rgxnot_func [| expr builder e1; expr
    builder e2 |] "ncomp" builder
379 | SRgxeq (e1, e2) -> L.build_call rgxeq_func [| expr builder e1; expr
    builder e2 |] "equals" builder
380 | SRgxneq (e1, e2) -> L.build_call rgxneq_func [| expr builder e1; expr
    builder e2 |] "nequals" builder
381 | SCall ("rgx_to_string", [e]) -> L.build_call rgx_to_str_func [| expr
    builder e |] "rgx_to_string" builder
382 | SCall ("int_to_string", [e]) -> L.build_call int_to_string_func [| expr
    builder e |] "int_to_string" builder
383 | SCall ("string_to_int", [e]) -> L.build_call string_to_int_func [| expr
    builder e |] "string_to_int" builder
384 | SCall ("bool_to_string", [e]) -> L.build_call bool_to_string_func [| expr
    builder e |] "bool_to_string" builder
385 | SCall ("print", [e]) ->
386     L.build_call printf_func [| string_format_str builder; (expr builder e) |]
    "printf" builder
387 | SCall ("nprint", [e]) ->
388     L.build_call printf_func [| nstring_format_str builder; (expr builder e)
    |] "printf" builder
389 | SCall ("length", [e]) ->
390     L.build_call length_func [| expr builder e |] "length" builder
391 | SCall ("delete", [e1; e2]) ->
392     L.build_call delete_func [| expr builder e1 ; expr builder e2 |] "
    removeNode" builder
393 | SCall ("insert", [e1; e2; e3]) ->
394     L.build_call insert_func [| expr builder e1 ; expr builder e2 ;
    cast_unsigned builder e3|] "insertElement" builder
395 | SCall ("contains", [e1; e2]) ->
396     L.build_call contains_func [| expr builder e1 ; cast_unsigned builder e2 ;
    choose_compar builder e2 |] "contains" builder
397 | SCall ("index_of", [e1 ; e2]) ->
398     L.build_call indexof_func [| expr builder e1 ; cast_unsigned builder e2
    ; choose_compar builder e2 |] "findIndexOfNode" builder
399 | SCall (f, args) ->
400     let (fdef, fdecl) = StringMap.find f function_decls in
401     let llargs = List.rev (List.map (expr builder) (List.rev args)) in
402     let result = (match fdecl.sret_type with
403         A.Void -> ""
404         | _ -> f ^ "_result") in
405     L.build_call fdef (Array.of_list llargs) result builder
406 | SAccess (a) ->
407     let (loop_func, _) = StringMap.find "loop" function_decls in
408     L.build_call access_func [| L.param loop_func 0; expr builder a|] "access
    " builder
409
410 | SNumFields ->
411     let (loop_func, _) = StringMap.find "loop" function_decls in
412     L.build_call numfields_func [| L.param loop_func 0 |] "numfields"

```



```

        builder
413 | SIncrement(e) -> let e2 = (A.Int, SAssign(e, (A.Int, SBinop(e, A.Add, (A.
      Int, SLiteral(1)))))) in expr builder e2
414 | SDecrement(e) -> let e2 = (A.Int, SAssign(e, (A.Int, SBinop(e, A.Sub, (A.
      Int, SLiteral(1)))))) in expr builder e2
415 | SPluseq(e1, e2) -> let e = (A.Int, SAssign(e1, (A.Int, SBinop(e1, A.Add,
      e2)))) in expr builder e
416 | SMinuseq(e1, e2) -> let e = (A.Int, SAssign(e1, (A.Int, SBinop(e1, A.Sub,
      e2)))) in expr builder e
417 | SRgxLiteral s -> let l = L.define_global "" (L.const_stringz context s)
      the_module in
418 | L.const_bitcast (L.const_gep l [|L.const_int i32_t 0|]) str_t
419 | _ -> raise (Failure "Cannot find expression")
420 | and
421
422 (* array gen functions *)
423 find_arr_type ty =
424   let ast_typ = arr_elem_type ty in
425   L.size_of (ltype_of_typ ast_typ)
426
427 and get_depth ty =
428   let depth_int = arr_depth ty in L.const_int i32_t depth_int
429
430 and array_gen builder ty arr =
431   let arr_type = find_arr_type ty in (* type of 1 depth down *)
432   let depth = get_depth ty in
433
434   let lst =
435     L.build_call initlist_func [| arr_type; depth |] "initList" builder
436   in
437
438   let add_front e =
439     let red_expr = expr builder e in
440     let ty = L.type_of red_expr in
441     let data = match L.classify_type ty with
442       | L.TypeKind.Pointer -> L.build_pointercast red_expr i64_t "addFrontCast"
443         " builder
444       | L.TypeKind.Integer -> L.build_zext red_expr i64_t "addFrontCast"
445         builder
446       | _ -> raise (Failure "Cannot find type of array")
447     in
448     ignore(L.build_call addfront_func [| lst; data |] "addFront" builder)
449   in
450   List.iter add_front arr; ignore(L.build_call reverse_func [| lst |] "
      reverseList" builder); lst
451
452 and cast_unsigned builder e3 =
453   let red_expr = expr builder e3 in
454   let ty = L.type_of red_expr in match (L.classify_type ty) with
455     | L.TypeKind.Pointer -> L.build_pointercast red_expr i64_t "castToUnsigned"
456       " builder
457     | L.TypeKind.Integer -> L.build_sext_or_bitcast red_expr i64_t "
      castToUnsigned" builder
458     | _ -> raise (Failure "Cannot cast data to i64")

```

```

458 and choose_compar builder e2 =
459   let (e2_ty, _) = e2 in
460   let rec compar_from_ttyp ty = match ty with
461     A.Int -> L.build_zext_or_bitcast compareint_func compare_p_t "
         compareCast" builder
462     | A.Bool -> L.build_zext_or_bitcast comparebools_func compare_p_t "
         compareCast" builder
463     | A.String -> L.build_zext_or_bitcast comparestr_func compare_p_t "
         compareCast" builder
464     | A.Rgx -> L.build_zext_or_bitcast comparestr_func compare_p_t "
         compareCast" builder
465     | A.ArrayType t -> compar_from_ttyp t
466     | _ -> raise (Failure "Unable to find comparator for list")
467   in compar_from_ttyp e2_ty
468
469 in
470
471
472 let rec stmt builder = function
473   SExpr ex -> ignore(expr builder ex); builder
474 | SBlock s1 -> List.fold_left stmt builder s1
475 | SReturn e -> ignore (match fdecl.sret_type with
476   A.Void -> L.build_ret_void builder
477   | _ -> L.build_ret (expr builder e) builder); builder
478 | SIf (predicate, then_stmt, else_stmt) ->
479   let bool_val = expr builder predicate in
480   let merge_bb = L.append_block context "merge" the_function in
481     let build_br_merge = L.build_br merge_bb in (* partial function *)
482
483     let then_bb = L.append_block context "then" the_function in
484     add_terminal (stmt (L.builder_at_end context then_bb) then_stmt)
485     build_br_merge;
486
487     let else_bb = L.append_block context "else" the_function in
488     add_terminal (stmt (L.builder_at_end context else_bb) else_stmt)
489     build_br_merge;
490
491     ignore(L.build_cond_br bool_val then_bb else_bb builder);
492     L.builder_at_end context merge_bb
493 | SWhile (predicate, body) ->
494   let pred_bb = L.append_block context "while" the_function in
495   ignore(L.build_br pred_bb builder);
496
497   let body_bb = L.append_block context "while_body" the_function in
498   add_terminal (stmt (L.builder_at_end context body_bb) body)
499   (L.build_br pred_bb);
500
501   let pred_builder = L.builder_at_end context pred_bb in
502   let bool_val = expr pred_builder predicate in
503
504   let merge_bb = L.append_block context "merge" the_function in
505   ignore(L.build_cond_br bool_val body_bb merge_bb pred_builder);
506   L.builder_at_end context merge_bb
507 | SFor (e1, e2, e3, body) -> stmt builder
508   ( SBlock [SExpr e1 ; SWhile (e2, SBlock [body ; SExpr e3]) ] )
509 | _ -> raise (Failure "Cannot pattern match statement")

```

```

510     in
511
512     let func_builder = stmt func_builder (SBlock fdecl.sbody) in
513
514     (* return types for functions *)
515     add_terminal func_builder (match fdecl.sret_type with
516     | A.Void -> L.build_ret_void
517     | A.Int  -> L.build_ret (L.const_int i32_t 0)
518     | A.Bool -> L.build_ret (L.const_int i1_t 0)
519     | A.String -> L.build_ret (L.const_pointer_null str_t)
520     | t -> L.build_ret (L.const_int (ltype_of_typ t) 0))
521
522     (*----- END OF build_func_body ----- *)
523
524     in
525
526     (* Call the things that happen in main *)
527     build_config_block config_block;
528     List.iter build_function_body (snd begin_block);
529     build_function_body loop_block;
530     build_function_block end_block;
531
532     the_module

```

## structure.c

Author: Mel

```

1  /* Allows LOOP block to loop through a file */
2
3  #include <stdio.h>
4  #include <string.h>
5
6  extern char *RS;
7  extern char *FS;
8
9  void setRS(char *newRS) {
10     RS = newRS;
11 }
12
13 void setFS(char *newFS) {
14     FS = newFS;
15 }
16
17 void loop(char *line);
18
19 void end();
20
21 int main(int argc, char **argv) {
22
23     /* Error handling for if filename is not specified */
24     if (argc < 2) {
25         fprintf(stderr, "usage: ./bawk.sh [bawk file] [input file]\n");
26     }
27     else {

```

```

28     char *filename = argv[1];
29     FILE *fp = fopen(filename, "rw");
30     char buffer[256];
31     size_t n = sizeof(buffer);
32     char *buf = buffer;
33
34     while(getdelim(&buf, &n, *RS, fp) > 0){
35         buffer[strcspn(buffer, RS)] = '\0';
36         loop(buffer);
37     }
38
39     end();
40 }
41 }

```

## mylist.c

Author: Ashley

```

1  /*
2  * Arrays
3  * Loosely based on Lab 3 from COMS 3157
4  */
5
6  #include <stdio.h>
7  #include <stdbool.h>
8  #include <stdlib.h>
9  #include <string.h>
10
11 /* A node in a linked list */
12 struct Node {
13     unsigned long data;
14     struct Node *next;
15 };
16
17 /* A linked list. 'head' points to the first node in the list */
18 struct List {
19     struct Node *head;
20     size_t size_of_type;
21     int depth;
22 };
23
24 /* Traverse list to find length of list */
25 int length(struct List *list) {
26     struct Node *node = list->head;
27     int count = 0;
28     while(node) {
29         node = node->next;
30         count++;
31     }
32     return count;
33 }
34
35 /* Compare two bools. Return -1 if a < b, return 1 if a > b, and return 0 if a ==
    b */

```

```

36 int compareBools(unsigned long a, unsigned long b)
37 {
38     if ((bool)a < (bool)b) return -1;
39     if ((bool)a > (bool)b) return 1;
40     return 0;
41 }
42
43 /* Compare two ints. Return -1 if a < b, return 1 if a > b, and return 0 if a == b
    */
44 int compareInts(unsigned long a, unsigned long b)
45 {
46     if ((int)a < (int)b) return -1;
47     if ((int)a > (int)b) return 1;
48     return 0;
49 }
50
51 /* Compare two strings. Return -1 if a < b, return 1 if a > b, and return 0 if a
    == b */
52 int compareStrs(unsigned long a, unsigned long b)
53 {
54     return strcmp((const char *)a, (const char *)b);
55 }
56
57 /* Compare two lists. Return -1 if a < b, return 1 if a > b, and return 0 if a ==
    b */
58 int compareLists(unsigned long a, unsigned long b, int (*compar)(unsigned long,
    unsigned long))
59 {
60     struct List *lista = (struct List *)a;
61     struct List *listb = (struct List *)b;
62     if ( length(lista) != length(listb) )
63         return -1;
64     if ( lista->depth > 1 ) {
65         struct Node *nodea = lista->head;
66         struct Node *nodeb = listb->head;
67         int total = 0;
68         while( nodea ) {
69             total += abs(compareLists(nodea->data, nodeb->data, compar));
70             nodea = nodea->next;
71             nodeb = nodeb->next;
72         }
73         if ( total == 0 )
74             return 0;
75         else
76             return -1;
77     }
78     if ( lista->depth == 1 ) {
79         struct Node *nodea = lista->head;
80         struct Node *nodeb = listb->head;
81         int total = 0;
82         while( nodea ) {
83             total += abs(compar(nodea->data, nodeb->data));
84             nodea = nodea->next;
85             nodeb = nodeb->next;
86         }
87         if ( total == 0 )

```

```

88     return 0;
89     else
90         return -1;
91 }
92 return -1;
93 }
94
95 /* Initialize and return an empty list */
96 struct List *initList(size_t size_of_type, int depth)
97 {
98     struct List *list = malloc( sizeof(struct List) );
99     if(list == NULL){
100         perror("malloc returned NULL");
101         exit(1);
102     }
103     list->head = 0;
104     list->size_of_type = size_of_type;
105     list->depth = depth;
106     return list;
107 }
108
109 /* Traverse the list until node at index found. NULL if not found */
110 struct Node *findByIndex(struct List *list, int indexSought)
111 {
112     int arr_len = length(list);
113
114     struct Node *node = list->head;
115     int indexAt = 0;
116     while(node) {
117         if( indexSought == indexAt )
118             return node;
119         node = node->next;
120         indexAt++;
121     }
122     return NULL;
123 }
124
125 /*
126 * Traverse the list, comparing each data item with 'dataSought' using
127 * 'compar' function. ('compar' returns 0 if the data pointed to by
128 * the two parameters are equal, non-zero value otherwise.)
129 *
130 * Returns the first node containing the matching data,
131 * NULL if not found.
132 */
133 struct Node *findNode(struct List *list, unsigned long dataSought, int (*compar)(
134     unsigned long, unsigned long))
135 {
136     struct Node *node = list->head;
137     if(list->depth == 1) {
138         while(node) {
139             if( compar(dataSought, node->data) == 0 )
140                 return node;
141             node = node->next;
142         }
143     }

```

```

143     if(list->depth > 1) {
144         while(node) {
145             if( compareLists(dataSought, node->data, compar) == 0 )
146                 return node;
147             node = node->next;
148         }
149     }
150     return NULL;
151 }
152
153 /* Returns 1 if dataSought is found in list, and 0 otherwise */
154 int contains(struct List *list, unsigned long dataSought, int (*compar)(unsigned
    long, unsigned long)) {
155     struct Node *found = findNode(list, dataSought, compar);
156     if (found)
157         return 1;
158     return 0;
159 }
160
161 /* Traverse list to find index of node */
162 int findIndexOfNode(struct List *list, unsigned long dataSought, int (*compar)(
    unsigned long, unsigned long))
163 {
164     struct Node *node = list->head;
165     int count = 0;
166     if(list->depth == 1) {
167         while(node) {
168             if( compar(dataSought, node->data) == 0 ) {
169                 return count;
170             }
171             node = node->next;
172             count++;
173         }
174     }
175     if(list->depth > 1) {
176         while(node) {
177             if( compareLists(dataSought, node->data, compar) == 0 ) {
178                 return count;
179             }
180             node = node->next;
181             count++;
182         }
183     }
184     return count;
185 }
186
187 /*
188  * Remove node at specific index and return the 'data' pointer that was stored in
    the node.
189  * Returns NULL if the list is empty
190  */
191 unsigned long removeNode(struct List *list, int indexSought)
192 {
193     struct Node *node = list->head;
194     int indexAt = 0;
195

```

```

196 int arr_len = length(list);
197 if (indexSought >= arr_len || indexSought < 0) {
198     fprintf(stderr, "%s\n", "Index Out of Bounds Error.");
199     exit(0);
200 }
201
202 // check that list is not empty
203 if ( list->head ) {
204     // if remove first index, it's just like popFront()
205     if ( indexSought == 0 ) {
206         unsigned long data = list->head->data;
207         struct Node *node = list->head;
208         list->head = list->head->next;
209         free(node);
210         return data;
211     }
212
213     // remove index other than first index
214     while(node) {
215         if( indexSought - 1 == indexAt ) {
216             unsigned long data = node->next->data;
217             struct Node *deletedNode = node->next;
218             node->next = node->next->next;
219             free(deletedNode);
220             return data;
221         }
222         node = node->next;
223         indexAt++;
224     }
225 }
226 return 0;
227 }
228
229 /*
230  * Remove the first node from the list, and return the 'data' pointer that was
231   * stored in the node.
232  * Returns NULL is the list is empty.
233  */
234 unsigned long popFront(struct List *list)
235 {
236     if(list->head) {
237         unsigned long data = list->head->data;
238         struct Node *node = list->head;
239         list->head = list->head->next;
240         free(node);
241         return data;
242     }
243     return 0;
244 }
245
246 /* Remove all nodes from the list */
247 void removeAllNodes(struct List *list)
248 {
249     while(list->head) {
250         popFront(list);
251     }

```



```

251 }
252
253 /*
254 * Create a node that holds the given data pointer,
255 * and add the node to the front of the list.
256 * Returns the newly created node on success and NULL on failure.
257 */
258 struct Node *addFront(struct List *list, unsigned long data)
259 {
260     struct Node *node = malloc( sizeof(struct Node) );
261     if(node == NULL){
262         perror("malloc returned NULL");
263         exit(1);
264     }
265     node->next = list->head;
266     node->data = data;
267     list->head = node;
268     return node;
269 }
270
271 /*
272 * Create a node that holds the given data pointer,
273 * and add the node right after the node passed in as the 'prevNode'
274 * parameter. If 'prevNode' is NULL, this function is equivalent to
275 * addFront().
276 * Returns the newly created node on success and NULL on failure.
277 */
278 struct Node *addAfter(struct List *list, struct Node *prevNode, unsigned long data
279 )
280 {
281     struct Node *node = malloc( sizeof(struct Node) );
282     if( prevNode ) {
283         node->next = prevNode->next;
284         node->data = data;
285         prevNode->next = node;
286     }
287     else {
288         node->next = list->head;
289         node->data = data;
290         list->head = node;
291     }
292     return node;
293 }
294 /* Reverse the list. */
295 void reverseList(struct List *list)
296 {
297     struct Node *prv = NULL;
298     struct Node *cur = list->head;
299     struct Node *nxt;
300
301     while (cur) {
302         nxt = cur->next;
303         cur->next = prv;
304         prv = cur;
305         cur = nxt;

```

```

306 }
307
308 list->head = prv;
309 }
310
311 /* Get element at specified index from a list */
312 unsigned long getElement(struct List *list, int index)
313 {
314     struct Node *node_by_index = findByIndex(list, index);
315     int arr_len = length(list);
316     if (index >= arr_len || index < 0) {
317         fprintf(stderr, "%s\n", "Index Out of Bounds Error.");
318         exit(0);
319     }
320     return node_by_index->data;
321 }
322
323 /* Insert element in list at specified index */
324 void insertElement(struct List *list, int index, unsigned long insert)
325 {
326     int arr_len = length(list);
327     if (index > arr_len || index < 0) {
328         fprintf(stderr, "%s\n", "Index Out of Bounds Error.");
329         exit(0);
330     }
331     if (index == 0)
332         addFront(list, insert);
333     else {
334         int x = index - 1;
335         struct Node *node = findByIndex(list, x);
336         addAfter(list, node, insert);
337     }
338 }
339
340 /* Assign element at specified index to a new value */
341 void assignElement(struct List *list, int index, unsigned long insert)
342 {
343     struct Node *node = findByIndex(list, index);
344     node->data = insert;
345 }

```

## rgx.c

Author: Mel

```

1 /* Regex operators */
2
3 #include <regex.h>
4 #include <stdlib.h>
5 #include <stdio.h>
6 #include <assert.h>
7 #include <string.h>
8
9 /* Returns 1 if string a matches rgx b, 0 otherwise */
10 int comp(char *a, char *b) {

```

```

11 | if (b[0] == '\\') {
12 |     b++;
13 | }
14 | if (b[strlen(b) - 1] == '\\') {
15 |     b[strlen(b) - 1] = 0;
16 | }
17 | regex_t regex;
18 | int comp = regcomp(&regex, b, REG_EXTENDED|REG_NOSUB);
19 | if (comp) {
20 |     printf("Regex expression could not be compiled");
21 |     return 0;
22 | }
23 | comp = regexec(&regex, a, 0, NULL, 0);
24 | if (!comp) {
25 |     return 1;
26 | }
27 | return 0;
28 | }
29 |
30 | /* Returns 1 if string a doesn't match rgx b, 0 otherwise */
31 | int ncomp(char *a, char *b) {
32 |     int opp = comp(a, b);
33 |     return !opp;
34 | }
35 |
36 | /* rgx a == rgx b */
37 | int equals(char *a, char *b) {
38 |     if (strcmp(a, b) == 0) {
39 |         return 1;
40 |     }
41 |     return 0;
42 | }
43 |
44 | /* rgx a != rgx b */
45 | int nequals(char *a, char *b) {
46 |     return !equals(a, b);
47 | }

```

## convert.c

Author: Mel

```

1 | /* Built-in conversion functions, string operations, NF, $ */
2 |
3 | #include <stdio.h>
4 | #include <stdlib.h>
5 | #include <string.h>
6 |
7 | extern char *FS;
8 |
9 | /* Convert a string to an int */
10 | int string_to_int(char *a) {
11 |     char *cleaned = malloc(strlen(a) + 1);
12 |     strcpy(cleaned, a);
13 |     return atoi(cleaned);

```

```

14 }
15
16 /* string a == string b */
17 int streq(char *a, char *b) {
18     return (strcmp(a, b) == 0);
19 }
20
21 /* string a != string b */
22 int strneq(char *a, char *b) {
23     return (strcmp(a, b) != 0);
24 }
25
26 /* string a > string b */
27 int strgreater(char *a, char *b) {
28     return (strcmp(a, b) > 0);
29 }
30
31 /* string a < string b */
32 int strless(char *a, char *b) {
33     return (strcmp(a, b) < 0);
34 }
35
36 /* string a >= string b */
37 int strgeq(char *a, char *b) {
38     return (strcmp(a, b) >= 0);
39 }
40
41 /* string a <= string b */
42 int strleq(char *a, char *b) {
43     return (strcmp(a, b) <= 0);
44 }
45
46 /* Concatenate string b to string a and then return the new string */
47 char *concat(char *a, char *b) {
48     char *str = malloc(strlen(a) + strlen(b));
49     size_t len1 = strlen(a), len2 = strlen(b);
50     char *concat = (char*) malloc(len1 + len2 + 1);
51
52     memcpy(concat, a, len1);
53     memcpy(concat+len1, b, len2+1);
54     return concat;
55 }
56
57 /* Convert a bool to a string */
58 char *bool_to_string(int a) {
59     if(a) {
60         return "true";
61     }
62     else {
63         return "false";
64     }
65     return "";
66 }
67
68 /* Convert an int to a string */
69 char *int_to_string(int a) {

```

```

70  int length = sprintf( NULL, 0, "%d", a);
71  char* str = malloc( length + 1 );
72  sprintf( str, length + 1, "%d", a );
73  return str;
74 }
75
76 /* Convert a regex to a string */
77 char *rgx_to_string(char *a) {
78     if (a[0] == '\\') {
79         a++;
80     }
81     if (a[strlen(a)-1] == '\\') {
82         a[strlen(a)-1] = '\0';
83     }
84     return a;
85 }
86
87 /* Return the number of fields in a line */
88 int numfields(char *line) {
89     if (strlen(line)==1 && line[0] == '\n') {
90         return 0;
91     }
92     if (strlen(line)==0) {
93         return 0;
94     }
95     int numSpaces = 1;
96
97     for(int i = 0; i < strlen(line); i++) {
98         if(line[i] == ' '){
99             numSpaces++;
100        }
101    }
102    return numSpaces;
103 }
104
105 /* Return whatever is at the specified field */
106 char *access(char *line, int field) {
107     if (field == 0) {
108         return line;
109     }
110     char *copy = malloc(strlen(line) + 1);
111     strcpy(copy, line);
112     char *token;
113     int count = 1;
114
115     token = strtok(copy, FS);
116
117     /* walk through other tokens */
118     while( token != NULL ) {
119         if (count == field) {
120             return token;
121         }
122         token = strtok(NULL, FS);
123         count++;
124     }
125     return "";

```

## testall.sh

Author: Ashley

```
1 LLI="lli"
2 LLC="llc"
3 CC="cc"
4 BAWK="./bawk.native"
5
6 ulimit -t 30
7
8 globallog=testall.log
9 rm -f $globallog
10 error=0
11 globalerror=0
12
13 keep=0
14
15 Usage() {
16     echo "Usage: testall.sh [options] [.bawk files]"
17     echo "-k    Keep intermediate files"
18     echo "-h    Print this help"
19     exit 1
20 }
21
22 SignalError() {
23     if [ $error -eq 0 ] ; then
24         echo "FAILED"
25         error=1
26     fi
27     echo " $1"
28 }
29
30 Compare() {
31     generatedfiles="$generatedfiles $3"
32     echo diff -b $1 $2 ">" $3 1>&2
33     diff -b "$1" "$2" > "$3" 2>&1 || {
34         SignalError "$1 differs"
35         echo "FAILED $1 differs from $2" 1>&2
36     }
37 }
38
39 Run() {
40     echo $* 1>&2
41     eval $* || {
42         SignalError "$1 failed on $*"
43         return 1
44     }
45 }
46
47 RunFail() {
48     echo $* 1>&2
49     eval $* && {
```

```

50 SignalError "failed: $* did not report an error"
51 return 1
52 }
53 return 0
54 }
55
56 Check() {
57     error=0
58     basename='echo $1 | sed 's/.*\\\///
59                 s/.bawk//''
60     reffile='echo $1 | sed 's/.bawk$//''
61     basedir="'echo $1 | sed 's/\[^\/\]*$//''/'
62
63     echo -n "$basename..."
64
65     echo 1>&2
66     echo "##### Testing $basename" 1>&2
67
68     generatedfiles=""
69
70     generatedfiles="$generatedfiles ${basename}.ll ${basename}.s ${basename}.exe $
71     {basename}.out" &&
72     Run "$BAWK" "-c" "$1" ">" "${basename}.ll" &&
73     Run "$LLC" "-relocation-model=pic" "${basename}.ll" ">" "${basename}.s" &&
74     Run "$CC" "-o" "${basename}.exe" "${basename}.s" "convert.o" "structure.o" "
75     mylist.o" "rgx.o" &&
76     Run "./${basename}.exe" "input.txt" ">" "${basename}.out" &&
77     Compare ${basename}.out ${reffile}.out ${basename}.diff
78
79     # Report the status and clean up the generated files
80
81     if [ $error -eq 0 ] ; then
82     if [ $keep -eq 0 ] ; then
83         rm -f $generatedfiles
84     fi
85     echo "OK"
86     echo "##### SUCCESS" 1>&2
87     else
88     echo "##### FAILED" 1>&2
89     globalerror=$error
90     fi
91 }
92
93 CheckFail() {
94     error=0
95     basename='echo $1 | sed 's/.*\\\///
96                 s/.bawk//''
97     reffile='echo $1 | sed 's/.bawk$//''
98     basedir="'echo $1 | sed 's/\[^\/\]*$//''/'
99
100     echo -n "$basename..."
101
102     echo 1>&2
103     echo "##### Testing $basename" 1>&2
104
105     generatedfiles=""

```

```

104
105     generatedfiles="$generatedfiles ${basename}.err ${basename}.diff" &&
106     RunFail "$BAWK" "-c" "<" $1 "2>" "${basename}.err" ">>" $globallog &&
107     Compare ${basename}.err ${reffile}.err ${basename}.diff
108
109     # Report the status and clean up the generated files
110
111     if [ $error -eq 0 ] ; then
112 if [ $keep -eq 0 ] ; then
113     rm -f $generatedfiles
114 fi
115     echo "OK"
116     echo "##### SUCCESS" 1>&2
117     else
118     echo "##### FAILED" 1>&2
119     globalerror=$error
120     fi
121 }
122
123 while getopts kdpsh c; do
124     case $c in
125     k) # Keep intermediate files
126         keep=1
127         ;;
128     h) # Help
129         Usage
130         ;;
131     esac
132 done
133
134 shift `expr $OPTIND - 1`
135
136 LLIFail() {
137     echo "Could not find the LLVM interpreter \"$LLI\"."
138     echo "Check your LLVM installation and/or modify the LLI variable in testall.sh"
139     exit 1
140 }
141
142 which "$LLI" >> $globallog || LLIFail
143
144 if [ $# -ge 1 ]
145 then
146     files=$@
147 else
148     files="tests/pass-*.bawk tests/fail-*.bawk"
149 fi
150
151 for file in $files
152 do
153     case $file in
154     *pass-*)
155         Check $file 2>> $globallog
156         ;;
157     *fail-*)
158         CheckFail $file 2>> $globallog
159         ;;

```



```
160 *)
161     echo "unknown file type $file"
162     globalerror=1
163     ;;
164     esac
165 done
166
167 exit $globalerror
```

## bawk.sh

Author: Mel

```
1 set -e
2 ./bawk.native -c $1 > bawk_out.ll
3 llc -relocation-model=pic bawk_out.ll > bawk_out.s
4 cc -o bawk_out.exe bawk_out.s convert.o structure.o mylist.o rgx.o
5 ./bawk_out.exe $2
```

## All tests

Author: Ashley

### fail-FS.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     FS = " "; # Fail: reassigning FS
7 }
8
9 CONFIG {
10     FS = " ";
11 }
```

### fail-RS.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     RS = "\n\n"; # Fail: reassigning RS
7 }
8
9 CONFIG {
10     RS = "\n";
11 }
```

### fail-array1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     void[] a; # Fail: Creating an array of voids
7     a = [];
8 }
```

### fail-array2.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int[] a;
7     bool b;
8     a = [1,2,3];
9     b = contains(a, true); # Fail: Checking for bool inside int array
10 }
```

### fail-array3.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int[] a;
7     int b;
8     a = [1,2,3];
9     b = index_of(a, true); # Fail: Finding index of bool in an int array
10 }
```

### fail-arrayassign1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int[] a;
7     a = [[1,2], [3,4]]; # Fail: assigning a 2D array to a 1D array
8 }
```

### fail-arrayassign2.bawk

```
1 BEGIN {}
2
```

```
3 LOOP {}
4
5 END {
6   int[] a;
7   a = [1, "hi"]; # Fail: assigning a string as an element in an int array
8 }
```

### fail-arrayassign3.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int[][] a;
7   a = [[1,2], ["hi", "bye"]]; # Fail: Assigning 2D array of int type and string
   type to 2D array of int type
8 }
```

### fail-assign1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int i;
7   bool b;
8
9   i = 42;
10  i = 10;
11  b = true;
12  b = false;
13  i = false; # Fail: assigning a bool to an integer
14 }
```

### fail-assign2.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int i;
7   bool b;
8
9   b = 48; # Fail: assigning an integer to a bool
10 }
```

### fail-assign3.bawk

```
1 BEGIN {
2   function void myvoid()
3   {
4     return;
5   }
6 }
7
8 LOOP {}
9
10 END {
11   int i;
12
13   i = myvoid(); # Fail: assigning a void to an integer
14 }
```

### fail-assign4.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   void x; # Fail: Declaring a variable with void type
7 }
```

### fail-assign5.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   string s;
7   s = '[0]*'; # Fail: assigning regex to string
8 }
```

### fail-assign6.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   NF = 3; # Fail: assigning a value to NF
7 }
```

### fail-assign7.bawk

```
1 BEGIN {}
2
```

```
3 LOOP {}
4
5 END {
6     rgx r;
7     r = "hello"; # Fail: assigning string to regex
8 }
```

### fail-blocks1.bawk

```
1 BEGIN {}
2
3 BEGIN {} # Fail: two BEGIN blocks
4
5 LOOP {}
6
7 END {}
```

### fail-conversion1.bawk

```
1 BEGIN {}
2
3 LOOP {
4     print(int_to_string(string_to_int(8))); # Fail: Passing an int instead of string
5     to string_to_int
6 }
7 END {}
```

### fail-conversion2.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     print(int_to_string("hi")); # Fail: Passing a string instead of int
7 }
```

### fail-conversion3.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     print(rgx_to_string(2)); # Fail: Passing a int instead of rgx
7 }
```

### fail-conversion4.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   print(bool_to_string('[0-9]*')); # Fail: Passing a rgx instead of bool
7 }
```

### fail-conversion5.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   print(bool_to_string('[0-9]*')); # Fail: Passing a rgx instead of bool
7 }
```

### fail-dead1.bawk

```
1 BEGIN {
2   function int foo(){
3     int i;
4
5     i = 15;
6     return i;
7     i = 32; # Error: code after a return
8   }
9 }
10
11 LOOP {}
12
13 END {}
```

### fail-decl1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int NF; # Fail: declaring a variable called NF
7 }
```

### fail-decl2.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
```

```
5 END {
6   int RS; # Fail: declaring RS
7 }
```

### fail-decl3.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int FS; # Fail: declaring FS
7 }
```

### fail-dollarbegin.bawk

```
1 BEGIN {
2   string hello;
3   hello = $0;
4 }
5
6 LOOP {}
7
8 END {}
```

### fail-dynamicarr1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int[] a;
7   a = [1,2,3];
8   insert(a, 1, true); # Fail: Inserting bool into int array
9 }
```

### fail-expr1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int i;
7   bool b;
8
9   i = 42;
10  i = 10;
11  b = true;
12  b = false;
```

```
13 i = false; # Fail: assigning a bool to an integer
14 }
```

### fail-expr2.bawk

```
1 BEGIN {
2   int a;
3   bool b;
4
5   function void foo(int c, bool d)
6   {
7     int dd;
8     bool e;
9     a + c;
10    c - a;
11    a * 3;
12    c / 2;
13    d + a; # Error: bool + int
14  }
15 }
16
17 LOOP {}
18
19 END {}
```

### fail-expr3.bawk

```
1 BEGIN {
2   int a;
3   bool b;
4
5   function void foo(int c, bool d)
6   {
7     int d;
8     bool e;
9     b + a; # Error: bool + int
10  }
11 }
12
13 LOOP {}
14
15 END {}
```

### fail-expr4.bawk

```
1 BEGIN {
2   int a;
3   string b;
4
5   function void foo(int c, string d)
6   {
7     int d;
```



```
8     string e;
9     b + a; # Error: string + int
10  }
11 }
12
13 LOOP {}
14
15 END {}
```

### fail-expr5.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     '[0]*' ~ true; # Fail: Comparing rgx and bool
7 }
```

### fail-for1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int i;
7
8     for (i = 0 ; i < 10 ; i = i + 1) {
9         if (i == 3) {
10            print(int_to_string(42));
11        }
12    }
13
14    for (j = 0; i < 10 ; i = i + 1) {} # j undefined
15 }
```

### fail-for2.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int i;
7
8     for (i = 0; j < 10 ; i = i + 1) {} # j undefined
9 }
```

### fail-for3.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int i;
7
8     for (i = 0; i ; i = i + 1) {} # i is an integer, not Boolean
9 }
```

#### fail-for4.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int i;
7
8     for (i = 0; i < 10 ; i = j + 1) {} # j undefined
9 }
```

#### fail-for5.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int i;
7
8     for (i = 0; i < 10 ; i = i + 1) {
9         foo(); # Error: no function foo
10    }
11 }
```

#### fail-for6.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int i;
7     int x;
8
9     x = 10;
10    for (i in x) {} # Fail: enhanced for loop on an integer
11 }
```

### fail-for7.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int x;
7     string[] a;
8     for(x in a) {} # Fail: Enhanced for loop on a string array using an int
9 }
```

### fail-func1.bawk

```
1 BEGIN {
2     function int foo() {}
3
4     function int bar() {}
5
6     function int baz() {}
7
8     function void bar() {} # Error: duplicate function bar
9 }
10
11 LOOP {}
12
13 END {}
```

### fail-func2.bawk

```
1 BEGIN {
2     function int foo(int a, bool b, int c) { }
3
4     function void bar(int a, bool b, int a) {} # Error: duplicate formal a in bar
5 }
6
7 LOOP {}
8
9 END {}
```

### fail-func3.bawk

```
1 BEGIN {
2     function int foo(int a, bool b, int c) { }
3
4     function void bar(int a, void b, int c) {} # Error: illegal void formal b
5 }
6
7 LOOP {}
8
9 END {}
```

### fail-func4.bawk

```
1 BEGIN {
2   function int foo() {}
3
4   function void bar() {}
5
6   function int print() {} # Should not be able to define print
7
8   function void baz() {}
9 }
10
11 LOOP {}
12
13 END {}
```

### fail-func5.bawk

```
1 BEGIN {
2   function int foo() {}
3
4   function int bar() {
5     int a;
6     void b; # Error: illegal void local b
7     bool c;
8
9     return 0;
10  }
11 }
12
13 LOOP {}
14
15 END {}
```

### fail-func6.bawk

```
1 BEGIN {
2   function void foo(int a, bool b)
3   {
4   }
5 }
6
7 LOOP {}
8
9 END {
10  foo(42, true);
11  foo(42); # Wrong number of arguments
12 }
```

### fail-func7.bawk

```
1 BEGIN {
2   function void foo(int a, bool b)
```

```

3  {
4  }
5 }
6
7 LOOP {}
8
9 END {
10  foo(42, true);
11  foo(42, true, false); # Wrong number of arguments
12 }

```

### fail-func8.bawk

```

1 BEGIN {
2   function void foo(int a, bool b)
3   {
4   }
5
6   function void bar()
7   {
8   }
9 }
10
11 LOOP {}
12
13 END {
14   foo(42, true);
15   foo(42, bar()); # int and void, not int and bool
16 }

```

### fail-func9.bawk

```

1 BEGIN {
2   function void foo(int a, bool b)
3   {
4   }
5 }
6
7 LOOP {}
8
9 END {
10  foo(42, true);
11  foo(42, 42); # Fail: int, not bool
12 }

```

### fail-global1.bawk

```

1 BEGIN {
2   int c;
3   bool b;
4   void a; # global variables should not be void
5 }

```

```
6
7 LOOP {}
8
9 END {}
```

### fail-global2.bawk

```
1 BEGIN {
2   int b;
3   bool c;
4   int a;
5   int b; # Duplicate global variable
6 }
7
8 LOOP {}
9
10 END {}
```

### fail-helloworldbegin.bawk

```
1 BEGIN {
2   print("Hello World!");
3 }
4
5 LOOP {}
6
7 END {
8 }
9
10 CONFIG {}
```

### fail-if1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   if (true) {}
7   if (false) {} else {}
8   if (42) {} # Error: non-bool predicate
9 }
```

### fail-if2.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   if (true) {
```

```
7     foo; # Error: undeclared variable
8   }
9 }
```

### fail-if3.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   if (true) {
7     42;
8   } else {
9     bar; # Error: undeclared variable
10  }
11 }
```

### fail-length.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int x;
7   x = 0;
8   length(x); # Fail: Finding length of int
9 }
```

### fail-print1.bawk

```
1 BEGIN {
2   # Should be illegal to redefine
3   function void print() {}
4 }
5
6 LOOP {}
7
8 END {}
```

### fail-print2.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   print(5); # Fail: Printing an int instead of a string
7 }
```

### fail-return1.bawk

```
1 BEGIN {
2   function void foo()
3   {
4     if (true) {
5       return 42; # Should return void
6     }
7     else {
8       return;
9     }
10  }
11 }
12
13 LOOP {}
14
15 END {}
```

### fail-scope1.bawk

```
1 BEGIN{}
2 LOOP{
3   int i;
4
5   {
6     i = 15;
7   }
8
9   i = 32;
10 }
11 END{}
```

### fail-structure1.bawk

```
1 BEGIN {
2   function void loop() {}
3 }
4
5 LOOP {}
6
7 END {}
```

### fail-structure2.bawk

```
1 BEGIN {
2   function void end() {}
3 }
4
5 LOOP {}
6
7 END {}
```



### fail-while1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int i;
7
8     while (true) {
9         i = i + 1;
10    }
11
12    while (42) { # Should be boolean
13        i = i + 1;
14    }
15 }
```

### fail-while2.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int i;
7
8     while (true) {
9         i = i + 1;
10    }
11
12    while (true) {
13        foo(); # foo undefined
14    }
15 }
```

### pass-add1.bawk

```
1 BEGIN {
2     function int add(int x, int y) {
3         return x + y;
4     }
5 }
6
7 LOOP {}
8
9 END {
10    print(int_to_string(add(1, 2)));
11 }
12
13 CONFIG {}
```

### pass-add2.bawk

```
1 BEGIN {
2   function void add(int x, int y) {
3     print(int_to_string(x + y));
4   }
5 }
6
7 LOOP {}
8
9 END {
10  add(1, 2);
11 }
```

### pass-arith1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int x;
7   x = 0;
8   x = x + 2;
9   print(int_to_string(x));
10 }
```

### pass-arith2.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   print(int_to_string(1 + 2 * 3 + 4));
7 }
```

### pass-arith3.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   print(int_to_string(1 - 2 / 3 - 4));
7 }
```

### pass-arith4.bawk

```
1 BEGIN {}
2
3 LOOP {}
```

```
4
5 END {
6   int x;
7   x = 2;
8   x += 2;
9   print(int_to_string(x));
10 }
```

### pass-arith5.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int x;
7   x = 2;
8   x -= 2;
9   print(int_to_string(x));
10 }
```

### pass-arrayliterals.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int[] a;
7   a = [1+1, 2+2, 3+2];
8   print(int_to_string(a[0]));
9 }
```

### pass-bool1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   bool b;
7   b = true;
8   if (b) {
9     print(int_to_string(10));
10  }
11 }
```

### pass-bool2.bawk

```
1 BEGIN {
2   function bool cond() {
```

```
3     bool b;
4     b = true;
5     return b;
6 }
7 }
8
9 LOOP {}
10
11 END {
12     if (cond()) {
13         print(int_to_string(10));
14     }
15 }
```

### pass-boolarr1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     bool[] a;
7     a = [];
8     print(int_to_string(length(a)));
9 }
```

### pass-boolarr2.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     bool[] a;
7     a = [true, false, false];
8     print(int_to_string(length(a)));
9 }
```

### pass-boolarr3.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     bool[] a;
7     a = [true, false, false];
8     print(bool_to_string(a[0]));
9 }
```

### pass-boolarr4.bawk

```

1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     bool[] a;
7     a = [true, false, false];
8     a[0] = false;
9     print(bool_to_string(a[0]));
10 }

```

### pass-boolarr5.bawk

```

1 BEGIN {
2     function bool[] foo() {
3         bool[] a;
4         a = [true, false, false];
5         return a;
6     }
7 }
8
9 LOOP {}
10
11 END {
12     bool[] a;
13     a = foo();
14     print(bool_to_string(a[0]));
15 }

```

### pass-boolarr6.bawk

```

1 BEGIN {
2     function bool[] foo (bool[] a) {
3         return a;
4     }
5 }
6
7 LOOP {}
8
9 END {
10     bool[] a;
11     a = [true, false, false];
12     a = foo(a);
13     print(bool_to_string(a[0]));
14 }

```

### pass-boolarr7.bawk

```

1 BEGIN {}
2
3 LOOP {}

```

```

4
5 END {
6   bool[] [] m;
7   bool[] [] [] a;
8   bool[] [] [] [] b;
9
10  m = [[false, false], [false, false]];
11  m[0][0] = true;
12  print(bool_to_string(m[0][0]));
13
14  a =
15  [
16    [ [true, false], [false, false] ], [ [false, false], [false, false] ]
17  ];
18  print(bool_to_string(a[0][0][0]));
19
20  b =
21  [
22    [ [ [true, false], [false, false] ], [ [false, false], [false, false] ] ],
23    [ [ [false, false], [false, false] ], [ [false, false], [false, false] ] ]
24  ];
25  print(bool_to_string(b[0][0][0][0]));
26 }

```

### pass-boolarr8.bawk

```

1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   bool[] a;
7   a = [false, false, false];
8   print(bool_to_string(contains(a, false)));
9   print(bool_to_string(contains(a, true)));
10 }

```

### pass-boolarr9.bawk

```

1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   bool[] a;
7   a = [true, false, false];
8   print(int_to_string(index_of(a, true)));
9 }

```

### pass-comment.bawk

```

1 BEGIN {

```

```
2  function int add(int x, int y) {
3      return x + y;
4  }
5 }
6
7 LOOP {}
8
9 END {
10  # this should not print
11  # print(int_to_string(add(1, 2)));
12  print(int_to_string(add(1, 2)));
13 }
14
15 CONFIG {}
```

### pass-config1.bawk

```
1 BEGIN {}
2
3 LOOP {
4     print($1);
5 }
6
7 END {}
8
9 CONFIG {
10     RS = "\n";
11 }
```

### pass-config2.bawk

```
1 BEGIN {}
2
3 LOOP {
4     print($1);
5 }
6
7 END {}
8
9 CONFIG {
10     FS = " ";
11 }
```

### pass-config3.bawk

```
1 BEGIN {}
2
3 LOOP {
4     print($1);
5 }
6
7 END {}
```

```
8
9 CONFIG {
10     RS = "\n";
11     FS = " ";
12 }
```

### pass-dollar1.bawk

```
1 BEGIN {}
2
3 LOOP {
4     print($0);
5 }
6
7 END {}
```

### pass-dollar2.bawk

```
1 BEGIN {}
2
3 LOOP {
4     string hello;
5     hello = $0;
6     print(hello);
7 }
8
9 END {}
```

### pass-dynamicarr1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     bool[] a;
7     a = [true, false, false];
8     insert(a, 3, true);
9     print(int_to_string(length(a)));
10 }
```

### pass-dynamicarr2.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     bool[] a;
7     a = [true, false, false];
8     delete(a, 2);
```



```
9 | print(int_to_string(length(a)));
10| }
```

### pass-dynamicarr3.bawk

```
1 | BEGIN {}
2 |
3 | LOOP {}
4 |
5 | END {
6 |   int[] a;
7 |   int b;
8 |   a = [1,2,3];
9 |   b = a[3]; # Fail: Accessing past size of array
10| }
```

### pass-dynamicarr4.bawk

```
1 | BEGIN {}
2 |
3 | LOOP {}
4 |
5 | END {
6 |   int[] a;
7 |   a = [1,2,3];
8 |   insert(a, 6, 4); # Fail: Inserting past (size of array + 1)
9 | }
```

### pass-dynamicarr5.bawk

```
1 | BEGIN {}
2 |
3 | LOOP {}
4 |
5 | END {
6 |   int[] a;
7 |   a = [1,2,3];
8 |   insert(a, -1, 4); # Fail: Inserting at illegal index
9 | }
```

### pass-dynamicarr6.bawk

```
1 | BEGIN {}
2 |
3 | LOOP {}
4 |
5 | END {
6 |   int[] a;
7 |   a = [1,2,3];
8 |   delete(a, 4); # Fail: Deleting element from invalid index
9 | }
```

### pass-dynamicarr7.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int[] a;
7     a = [1,2,3];
8     delete(a, -1); # Fail: Deleting from an invalid index
9 }
```

### pass-fib.bawk

```
1 BEGIN {
2     function int fib(int x)
3     {
4         if (x < 2) {
5             return 1;
6         }
7         return fib(x-1) + fib(x-2);
8     }
9 }
10
11 LOOP {}
12
13 END {
14     print(int_to_string(fib(0)));
15     print(int_to_string(fib(1)));
16     print(int_to_string(fib(2)));
17     print(int_to_string(fib(3)));
18     print(int_to_string(fib(4)));
19     print(int_to_string(fib(5)));
20 }
```

### pass-for1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int i;
7     for (i = 0 ; i < 5 ; i = i + 1) {
8         print(int_to_string(i));
9     }
10     print(int_to_string(42));
11 }
```

### pass-for2.bawk

```
1 BEGIN {}
2
```

```

3 LOOP {}
4
5 END {
6   int i;
7   int[] arr;
8   arr = [1, 2, 3];
9
10  for (i = 0; i < length(arr); i++) {
11    print(int_to_string(arr[i]));
12  }
13  print(int_to_string(42));
14 }

```

### pass-func1.bawk

```

1 BEGIN {
2   function int fun(int x, int y)
3   {
4     return 0;
5   }
6 }
7
8 LOOP {}
9
10 END {
11   int i;
12   i = 1;
13
14   fun(i = 2, i = i+1);
15
16   print(int_to_string(i));
17 }

```

### pass-func2.bawk

```

1 BEGIN {
2   function void printem(int a, int b, int c, int d)
3   {
4     print(int_to_string(a));
5     print(int_to_string(b));
6     print(int_to_string(c));
7     print(int_to_string(d));
8   }
9 }
10
11 LOOP {}
12
13 END {
14   printem(42,17,192,8);
15 }

```

### pass-func3.bawk

```

1 BEGIN {
2   function int add(int a, int b)
3   {
4     int c;
5     c = a + b;
6     return c;
7   }
8 }
9
10 LOOP {}
11
12 END {
13   int d;
14   d = add(52, 10);
15   print(int_to_string(d));
16 }

```

### pass-func4.bawk

```

1 BEGIN {
2   function int foo(int a)
3   {
4     return a;
5   }
6 }
7
8 LOOP {}
9
10 END {
11   print( int_to_string(foo(2)) );
12 }

```

### pass-func5.bawk

```

1 BEGIN {
2   function void foo() {}
3
4   function int bar(int a, bool b, int c) { return a + c; }
5 }
6
7 LOOP {}
8
9 END {
10   print(int_to_string(bar(17, false, 25)));
11 }

```

### pass-func6.bawk

```

1 BEGIN {
2   int a;
3

```

```
4  function void foo(int c)
5  {
6      a = c + 42;
7  }
8 }
9
10 LOOP {}
11
12 END {
13     foo(73);
14     print(int_to_string(a));
15 }
```

### pass-func7.bawk

```
1 BEGIN {
2     function void foo(int a)
3     {
4         print(int_to_string(a + 3));
5     }
6 }
7
8 LOOP {}
9
10 END {
11     foo(40);
12 }
```

### pass-func8.bawk

```
1 BEGIN {
2     function void foo(int a)
3     {
4         print(int_to_string(a + 3));
5         return;
6     }
7 }
8
9 LOOP {}
10
11 END {
12     foo(40);
13 }
```

### pass-func9.bawk

```
1 BEGIN {
2     function int x() {
3         return 2;
4     }
5
6     function int y() {
```

```

7     return x();
8   }
9 }
10
11 LOOP {}
12
13 END {
14   print( int_to_string(y()) );
15 }

```

### pass-gcd.bawk

```

1 BEGIN {
2   function int gcd(int a, int b) {
3     while (a != b) {
4       if (a > b) {
5         a = a - b;
6       }
7       else {
8         b = b - a;
9       }
10    }
11    return a;
12  }
13 }
14
15 LOOP {}
16
17 END {
18   print(int_to_string(gcd(2,14)));
19   print(int_to_string(gcd(3,15)));
20   print(int_to_string(gcd(99,121)));
21 }

```

### pass-global1.bawk

```

1 BEGIN {
2   int a;
3   int b;
4
5   function int add(int x, int y) {
6     return x + y;
7   }
8 }
9
10 LOOP {}
11
12 END {
13   a = 1;
14   b = 2;
15   print(int_to_string(add(a, b)));
16 }

```

## pass-global2.bawk

```
1 BEGIN {
2   int a;
3   int b;
4
5   function void printa()
6   {
7     print(int_to_string(a));
8   }
9
10  function void printbb()
11  {
12    print(int_to_string(b));
13  }
14
15  function void incab()
16  {
17    a = a + 1;
18    b = b + 1;
19  }
20 }
21
22 LOOP {}
23
24 END {
25   a = 42;
26   b = 21;
27   printa();
28   printbb();
29   incab();
30   printa();
31   printbb();
32 }
```

## pass-global3.bawk

```
1 BEGIN {
2   bool i;
3 }
4
5 LOOP {}
6
7 END {
8   int i; # Should hide the global i
9
10  i = 42;
11  print(int_to_string(i + i));
12 }
```

## pass-helloworld.bawk

```
1 BEGIN {}
2
```

```
3 LOOP {}
4
5 END {
6   print("Hello World!");
7 }
8
9 CONFIG {}
```

### pass-helloworldloop.bawk

```
1 BEGIN {}
2
3 LOOP {
4   print("Hello World");
5 }
6
7 END {}
8
9 CONFIG {}
```

### pass-helloworldloopend.bawk

```
1 BEGIN {}
2
3 LOOP {
4   print("Hello loop!");
5 }
6
7 END {
8   print("Hello end!");
9 }
```

### pass-if1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   if (true) {
7     print(int_to_string(10));
8   }
9   print(int_to_string(2));
10 }
```

### pass-if2.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
```



```

5 END {
6   if(true) {
7     print(int_to_string(10));
8   }
9   else {
10    print(int_to_string(5));
11  }
12  print(int_to_string(2));
13 }

```

### pass-if3.bawk

```

1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   if(false) {
7     print(int_to_string(10));
8   }
9   print(int_to_string(2));
10 }

```

### pass-if4.bawk

```

1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   if(false) {
7     print(int_to_string(10));
8   }
9   else {
10    print(int_to_string(5));
11  }
12  print(int_to_string(2));
13 }

```

### pass-if5.bawk

```

1 BEGIN {
2   function int cond(bool b) {
3     int x;
4     if (b) {
5       x = 10;
6     }
7     else {
8       x = 5;
9     }
10    return x;
11  }

```

```

12 }
13
14 LOOP {}
15
16 END {
17     print(int_to_string(cond(true)));
18     print(int_to_string(cond(false)));
19 }

```

### pass-if6.bawk

```

1 BEGIN {
2     function int cond(bool b) {
3         int x;
4         if (!b) {
5             x = 10;
6         }
7         else {
8             x = 5;
9         }
10        return x;
11    }
12 }
13
14 LOOP {}
15
16 END {
17     print(int_to_string(cond(true)));
18     print(int_to_string(cond(false)));
19 }

```

### pass-if7.bawk

```

1 BEGIN {
2     function int cond(bool b) {
3         int x;
4         x = 10;
5         if (b) {
6             if (x == 10) {
7                 x = 10;
8             }
9         }
10        else {
11            x = 5;
12        }
13        return x;
14    }
15 }
16
17 LOOP {}
18
19 END {
20     print(int_to_string(cond(true)));
21     print(int_to_string(cond(false)));

```

```
22 }
```

### pass-intarr1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int[] a;
7   a = [];
8   print(int_to_string(length(a)));
9 }
```

### pass-intarr2.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int[] a;
7   a = [1, 2, 3];
8   print(int_to_string(length(a)));
9 }
```

### pass-intarr3.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int[] a;
7   a = [1, 2, 3];
8   print(int_to_string(a[0]));
9 }
```

### pass-intarr4.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int[] a;
7   a = [1, 2, 3];
8   a[0] = 0;
9   print(int_to_string(a[0]));
10 }
```

### pass-intarr5.bawk

```
1 BEGIN {
2   function int[] foo() {
3     int[] a;
4     a = [1, 2, 3];
5     return a;
6   }
7 }
8
9 LOOP {}
10
11 END {
12   int[] a;
13   a = foo();
14   print(int_to_string(a[0]));
15 }
```

### pass-intarr6.bawk

```
1 BEGIN {
2   function int[] foo (int[] a) {
3     return a;
4   }
5 }
6
7 LOOP {}
8
9 END {
10  int[] a;
11  a = [1, 2, 3];
12  a = foo(a);
13  print(int_to_string(a[0]));
14 }
```

### pass-intarr7.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int[] [] m;
7   int[] [] [] a;
8   int[] [] [] [] b;
9
10  m = [[1, 2], [3, 4]];
11  m[0][0] = 20;
12  print(int_to_string(m[0][0]));
13
14  a =
15  [
16    [ [1, 2], [3, 4] ], [ [5, 6], [7, 8] ]
17  ];
```

```

18 print(int_to_string(a[0][0][0]));
19
20 b =
21 [
22   [ [ [1, 2], [3, 4] ], [ [5, 6], [7, 8] ] ],
23   [ [ [9, 10], [11, 12] ], [ [13, 14], [15, 16] ] ]
24 ];
25 print(int_to_string(b[0][0][0][0]));
26 }

```

### pass-intarr8.bawk

```

1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int[] a;
7   a = [1, 2, 3];
8   print(bool_to_string(contains(a, 1)));
9   print(bool_to_string(contains(a, 4)));
10 }

```

### pass-intarr9.bawk

```

1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int[] a;
7   a = [1, 2, 3];
8   print(int_to_string(index_of(a, 1)));
9 }

```

### pass-local1.bawk

```

1 BEGIN {
2   function void foo(bool i)
3   {
4     int i; # Should hide the formal i
5
6     i = 42;
7     print(int_to_string(i + i));
8   }
9 }
10
11 LOOP {}
12
13 END {
14   foo(true);
15 }

```

## pass-local2.bawk

```
1 BEGIN {
2   function int foo(int a, bool b)
3   {
4     int c;
5     bool d;
6
7     c = a;
8
9     return c + 10;
10  }
11 }
12
13 LOOP {}
14
15 END {
16   print(int_to_string(foo(37, false)));
17 }
```

## pass-nf1.bawk

```
1 BEGIN {}
2
3 LOOP {
4   print(int_to_string(NF));
5 }
6
7 END {}
```

## pass-ops1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   print(int_to_string(1 + 2));
7   print(int_to_string(1 - 2));
8   print(int_to_string(-1 + 2));
9   print(int_to_string(1 * 2));
10  print(int_to_string(100 / 2));
11  print(int_to_string(99));
12  print(bool_to_string(1 == 2));
13  print(bool_to_string(1 == 1));
14  print(int_to_string(99));
15  print(bool_to_string(1 != 2));
16  print(bool_to_string(1 != 1));
17  print(int_to_string(99));
18  print(bool_to_string(1 < 2));
19  print(bool_to_string(2 < 1));
20  print(int_to_string(99));
21  print(bool_to_string(1 <= 2));
22  print(bool_to_string(1 <= 1));
```

```

23 print(bool_to_string(2 <= 1));
24 print(int_to_string(99));
25 print(bool_to_string(1 > 2));
26 print(bool_to_string(2 > 1));
27 print(int_to_string(99));
28 print(bool_to_string(1 >= 2));
29 print(bool_to_string(1 >= 1));
30 print(bool_to_string(2 >= 1));
31 }

```

### pass-ops2.bawk

```

1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   int x;
7   x = 42;
8   print(bool_to_string(true));
9   print(bool_to_string(false));
10  print(bool_to_string(true && true));
11  print(bool_to_string(true && false));
12  print(bool_to_string(false && true));
13  print(bool_to_string(false && false));
14  print(bool_to_string(true || true));
15  print(bool_to_string(true || false));
16  print(bool_to_string(false || true));
17  print(bool_to_string(false || false));
18  print(bool_to_string(!false));
19  print(bool_to_string(!true));
20  print(int_to_string(-10));
21  print(int_to_string(x--));
22  print(int_to_string(x++));
23 }

```

### pass-ops3.bawk

```

1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   print("hello " & "world");
7   print(int_to_string(99));
8   print(bool_to_string("hello" == "world"));
9   print(bool_to_string("hello" == "hello"));
10  print(int_to_string(99));
11  print(bool_to_string("hello" != "world"));
12  print(bool_to_string("hello" != "hello"));
13  print(int_to_string(99));
14  print(bool_to_string("hello" < "world"));
15  print(bool_to_string("world" < "hello"));
16  print(int_to_string(99));

```

```

17 print(bool_to_string("hello" <= "world"));
18 print(bool_to_string("hello" <= "hello"));
19 print(bool_to_string("world" <= "hello"));
20 print(int_to_string(99));
21 print(bool_to_string("hello" > "world"));
22 print(bool_to_string("world" > "hello"));
23 print(int_to_string(99));
24 print(bool_to_string("hello" >= "world"));
25 print(bool_to_string("hello" >= "hello"));
26 print(bool_to_string("world" >= "hello"));
27 }

```

### pass-ops4.bawk

```

1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   print(bool_to_string('[0-9]*' % '[0]*'));
7   print(bool_to_string('[0-9]*' % '[0-9]*'));
8   print(int_to_string(99));
9   print(bool_to_string('[0-9]*' !% '[0]*'));
10  print(bool_to_string('[0-9]*' !% '[0-9]*'));
11  print(int_to_string(99));
12  print(bool_to_string("1234" ~ '[0]'));
13  print(bool_to_string("1234" ~ '[0-9]*'));
14  print(int_to_string(99));
15  print(bool_to_string("1234" !~ '[0]'));
16  print(bool_to_string("1234" !~ '[0-9]*'));
17 }

```

### pass-print.bawk

```

1 BEGIN {}
2
3 LOOP {
4   print(int_to_string(string_to_int("8")));
5 }
6
7 END {}

```

### pass-printbegin.bawk

```

1 BEGIN {
2   function void print_begin () {
3     print("Hello world");
4   }
5 }
6 LOOP {}
7 END {
8   print_begin();

```



```
9 }
```

### pass-rgx1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   rgx r;
7   r = '[0-9]*';
8   print(rgx_to_string(r));
9 }
```

### pass-rgx2.bawk

```
1 BEGIN {
2   function rgx foo() {
3     return '[0-9]*';
4   }
5 }
6
7 LOOP {}
8
9 END {
10  print(rgx_to_string(foo()));
11 }
12
13 CONFIG {}
```

### pass-rgx3.bawk

```
1 BEGIN {
2   function bool foo(rgx a, rgx b) {
3     return a % b;
4   }
5 }
6
7 LOOP {}
8
9 END {
10  print(bool_to_string(foo('[0-9]*', '[0-9]*')));
11 }
12
13 CONFIG {}
```

### pass-rgx4.bawk

```
1 BEGIN {
2   function bool foo(rgx a, rgx b) {
3     return a !% b;
```

```

4 | }
5 | }
6 |
7 | LOOP {}
8 |
9 | END {
10 |   print(bool_to_string(foo('[0-9]*', '[0-9]*')));
11 | }
12 |
13 | CONFIG {}

```

### pass-rgx5.bawk

```

1 | BEGIN {
2 |   function bool foo(string a, rgx b) {
3 |     return a ~ b;
4 |   }
5 | }
6 |
7 | LOOP {}
8 |
9 | END {
10 |   print(bool_to_string(foo("1234", '[0-9]*')));
11 | }
12 |
13 | CONFIG {}

```

### pass-rgx6.bawk

```

1 | BEGIN {
2 |   function bool foo(string a, rgx b) {
3 |     return a !~ b;
4 |   }
5 | }
6 |
7 | LOOP {}
8 |
9 | END {
10 |   print(bool_to_string(foo("1234", '[0]')));
11 | }
12 |
13 | CONFIG {}

```

### pass-rgxarr1.bawk

```

1 | BEGIN {}
2 |
3 | LOOP {}
4 |
5 | END {
6 |   rgx[] a;
7 |   a = [];

```

```
8 print(int_to_string(length(a)));
9 }
```

### pass-rgxarr2.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   rgx[] a;
7   a = ['[0]*', '[1]*', '[1]*'];
8   print(int_to_string(length(a)));
9 }
```

### pass-rgxarr3.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   rgx[] a;
7   a = ['[0]*', '[1]*', '[1]*'];
8   print(rgx_to_string(a[0]));
9 }
```

### pass-rgxarr4.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   rgx[] a;
7   a = ['[0]*', '[1]*', '[1]*'];
8   a[0] = '[2]*';
9   print(rgx_to_string(a[0]));
10 }
```

### pass-rgxarr5.bawk

```
1 BEGIN {
2   function rgx[] foo() {
3     rgx[] a;
4     a = ['[0]*', '[1]*', '[1]*'];
5     return a;
6   }
7 }
8
9 LOOP {}
```

```

10|
11| END {
12|   rgx[] a;
13|   a = foo();
14|   print(rgx_to_string(a[0]));
15| }

```

### pass-rgxarr6.bawk

```

1| BEGIN {
2|   function rgx[] foo (rgx[] a) {
3|     return a;
4|   }
5| }
6|
7| LOOP {}
8|
9| END {
10|  rgx[] a;
11|  a = ['[0]*', '[1]*', '[1]*'];
12|  a = foo(a);
13|  print(rgx_to_string(a[0]));
14| }

```

### pass-rgxarr7.bawk

```

1| BEGIN {}
2|
3| LOOP {}
4|
5| END {
6|   rgx[] [] m;
7|   rgx[] [] [] a;
8|   rgx[] [] [] [] b;
9|
10|  m = [['[0]*', '[1]*'], ['[1]*', '[1]*']];
11|  m[0][0] = '[6*]';
12|  print(rgx_to_string(m[0][0]));
13|
14|  a =
15|  [
16|    ['[0*]', '[1*]', '[1*]', '[1*]'], ['[1*]', '[1*]', '[1*]', '[1*]']
17|  ];
18|  print(rgx_to_string(a[0][0][0]));
19|
20|  b =
21|  [
22|    [['[0*]', '[1*]', '[1*]', '[1*]'], ['[1*]', '[1*]', '[1*]', '[1*]']
23|    ],
24|    [['[1*]', '[1*]', '[1*]', '[1*]'], ['[1*]', '[1*]', '[1*]', '[1*]']
25|    ]
26|  ];
27|  print(rgx_to_string(b[0][0][0][0]));
28| }

```

---

### pass-rgxarr8.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   rgx[] a;
7   a = ['[0]*', '[1]*', '[1]*'];
8   print(bool_to_string(contains(a, '[0]*')));
9   print(bool_to_string(contains(a, '[2]*')));
10 }
```

### pass-rgxarr9.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   rgx[] a;
7   a = ['[0]*', '[1]*', '[1]*'];
8   print(int_to_string(index_of(a, '[0]*')));
9 }
```

### pass-strarr1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   string[] a;
7   a = [];
8   print(int_to_string(length(a)));
9 }
```

### pass-strarr2.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6   string[] a;
7   a = ["a", "b", "c"];
8   print(int_to_string(length(a)));
9 }
```

### pass-strarr3.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     string[] a;
7     a = ["a", "b", "c"];
8     print(a[0]);
9 }
```

### pass-strarr4.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     string[] a;
7     a = ["a", "b", "c"];
8     a[0] = "p";
9     print(a[0]);
10 }
```

### pass-strarr5.bawk

```
1 BEGIN {
2     function string[] foo() {
3         string[] a;
4         a = ["a", "b", "c"];
5         return a;
6     }
7 }
8
9 LOOP {}
10
11 END {
12     string[] a;
13     a = foo();
14     print(a[0]);
15 }
```

### pass-strarr6.bawk

```
1 BEGIN {
2     function string[] foo (string[] a) {
3         return a;
4     }
5 }
6
7 LOOP {}
8
```

```

9| END {
10|     string[] a;
11|     a = ["a", "b", "c"];
12|     a = foo(a);
13|     print(a[0]);
14| }

```

### pass-strarr7.bawk

```

1| BEGIN {}
2|
3| LOOP {}
4|
5| END {
6|     string[][] m;
7|     string[][][] a;
8|     string[][][][] b;
9|
10|    m = [["a", "b"], ["c", "d"]];
11|    m[0][0] = "hi";
12|    print(m[0][0]);
13|
14|    a =
15|    [
16|        [ ["a", "b"], ["c", "d"] ], [ ["e", "f"], ["g", "h"] ]
17|    ];
18|    print(a[0][0][0]);
19|
20|    b =
21|    [
22|        [ [ ["a", "b"], ["c", "d"] ], [ ["e", "f"], ["g", "h"] ] ],
23|        [ [ ["i", "j"], ["k", "l"] ], [ ["m", "n"], ["o", "p"] ] ]
24|    ];
25|    print(b[0][0][0][0]);
26| }

```

### pass-strarr8.bawk

```

1| BEGIN {}
2|
3| LOOP {}
4|
5| END {
6|     string[] a;
7|     a = ["a", "b", "c"];
8|     print(bool_to_string(contains(a, "a")));
9|     print(bool_to_string(contains(a, "e")));
10| }

```

### pass-strarr9.bawk

```

1| BEGIN {}

```

```
2|
3| LOOP {}
4|
5| END {
6|   string[] a;
7|   a = ["a", "b", "c"];
8|   print(int_to_string(index_of(a, "a")));
9| }
```

### pass-string1.bawk

```
1| BEGIN {}
2|
3| LOOP {}
4|
5| END {
6|   string s;
7|   s = "hello";
8|   print(s);
9| }
```

### pass-string2.bawk

```
1| BEGIN {
2|   function string foo() {
3|     return "hello";
4|   }
5| }
6|
7| LOOP {}
8|
9| END {
10|  print(foo());
11| }
12|
13| CONFIG {}
```

### pass-string3.bawk

```
1| BEGIN {
2|   function string concat(string a, string b) {
3|     return a & b;
4|   }
5| }
6|
7| LOOP {}
8|
9| END {
10|  print(concat("hello ", "world"));
11| }
12|
13| CONFIG {}
```



### pass-string4.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int x;
7     string s;
8
9     s = "3";
10    x = string_to_int(s);
11    print(int_to_string(x));
12 }
```

### pass-while1.bawk

```
1 BEGIN {}
2
3 LOOP {}
4
5 END {
6     int i;
7     i = 5;
8     while (i > 0) {
9         print(int_to_string(i));
10        i = i - 1;
11    }
12    print(int_to_string(42));
13 }
```

### pass-while2.bawk

```
1 BEGIN {
2     function int foo(int a)
3     {
4         int j;
5         j = 0;
6         while (a > 0) {
7             j = j + 2;
8             a = a - 1;
9         }
10        return j;
11    }
12 }
13
14 LOOP {}
15
16 END {
17     print(int_to_string(foo(7)));
18 }
```

Failed tests output

### fail-FS.err

```
1 Fatal error: exception Parsing.Parse_error
```

### fail-RS.err

```
1 Fatal error: exception Parsing.Parse_error
```

### fail-array1.err

```
1 Fatal error: exception Parsing.Parse_error
```

### fail-array2.err

```
1 Fatal error: exception Failure("cannot perform contains on int[] and bool")
```

### fail-array3.err

```
1 Fatal error: exception Failure("cannot perform index_of on int[] and bool")
```

### fail-arrayassign1.err

```
1 Fatal error: exception Failure("illegal assignment int[] = int[][] in a = [[1, 2],  
  [3, 4]]")
```

### fail-arrayassign2.err

```
1 Fatal error: exception Failure("array of different types, expected int found  
  string")
```

### fail-arrayassign3.err

```
1 Fatal error: exception Failure("array of different types, expected int[] found  
  string[]")
```

### fail-assign1.err

```
1 Fatal error: exception Failure("illegal assignment int = bool in i = false")
```

### fail-assign2.err

```
1 Fatal error: exception Failure("illegal assignment bool = int in b = 48")
```

### fail-assign3.err

```
1 Fatal error: exception Failure("illegal assignment int = void in i = myvoid()")
```

### fail-assign4.err

```
1 Fatal error: exception Failure("illegal void local x")
```

### fail-assign5.err

```
1 Fatal error: exception Failure("illegal assignment string = rgx in s = '[0]*'")
```

### fail-assign6.err

```
1 Fatal error: exception Failure("illegal assignment of NF")
```

### fail-assign7.err

```
1 Fatal error: exception Failure("illegal assignment rgx = string in r = hello")
```

### fail-blocks1.err

```
1 Fatal error: exception Parsing.Parse_error
```

### fail-conversion1.err

```
1 Fatal error: exception Failure("illegal argument found int expected string in 8")
```

### fail-conversion2.err

```
1 Fatal error: exception Failure("illegal argument found string expected int in hi")
```

### fail-conversion3.err

```
1 Fatal error: exception Failure("illegal argument found int expected rgx in 2")
```

### fail-conversion4.err

```
1 Fatal error: exception Failure("illegal argument found rgx expected bool in  
  '[0-9]*'")
```

### fail-conversion5.err

```
1 Fatal error: exception Failure("illegal argument found rgx expected bool in  
  '[0-9]*'")
```

### fail-dead1.err

```
1 Fatal error: exception Failure("nothing may follow a return")
```

### fail-decl1.err

```
1 Fatal error: exception Parsing.Parse_error
```

### fail-decl2.err

```
1 Fatal error: exception Parsing.Parse_error
```

### fail-decl3.err

```
1 Fatal error: exception Parsing.Parse_error
```

### fail-dollarbegin.err

```
1 Fatal error: exception Parsing.Parse_error
```

### fail-dynamicarr1.err

```
1 Fatal error: exception Failure("cannot perform insert on int[] and bool")
```

### fail-expr1.err

```
1 Fatal error: exception Failure("illegal assignment int = bool in i = false")
```

### fail-expr2.err

```
1 Fatal error: exception Failure("illegal binary operator bool + int in d + a")
```

### fail-expr3.err

```
1 Fatal error: exception Failure("illegal binary operator bool + int in b + a")
```

### fail-expr4.err

```
1 Fatal error: exception Failure("illegal binary operator string + int in b + a")
```

### fail-expr5.err

```
1 Fatal error: exception Failure("Different types expected for '[0]*'~true")
```

### fail-for1.err

```
1 Fatal error: exception Failure("undeclared identifier j")
```

### fail-for2.err

```
1 Fatal error: exception Failure("undeclared identifier j")
```

### fail-for3.err

```
1 Fatal error: exception Failure("expected Boolean expression in i")
```

### fail-for4.err

```
1 Fatal error: exception Failure("undeclared identifier j")
```

### fail-for5.err

```
1 Fatal error: exception Failure("unrecognized function foo")
```

### fail-for6.err

```
1 Fatal error: exception Failure("cannot iterate over type int")
```

### fail-for7.err

```
1 Fatal error: exception Failure("mismatch in int and string[]")
```

### fail-func1.err

```
1 Fatal error: exception Failure("duplicate function bar")
```

### fail-func2.err

```
1 Fatal error: exception Failure("duplicate formal a")
```

### fail-func3.err

```
1 Fatal error: exception Failure("illegal void formal b")
```

### fail-func4.err

```
1 Fatal error: exception Failure("function print may not be defined")
```

### fail-func5.err

```
1 Fatal error: exception Failure("illegal void local b")
```

### fail-func6.err

```
1 Fatal error: exception Failure("expecting 2 arguments in foo(42)")
```

### fail-func7.err

```
1 Fatal error: exception Failure("expecting 2 arguments in foo(42, true, false)")
```

### fail-func8.err

```
1 Fatal error: exception Failure("illegal argument found void expected bool in bar()  
")
```

### fail-func9.err

```
1 Fatal error: exception Failure("illegal argument found int expected bool in 42")
```

### fail-global1.err

```
1 Fatal error: exception Failure("illegal void global a")
```

### fail-global2.err

```
1 Fatal error: exception Failure("duplicate global b")
```

### fail-helloworldbegin.err

```
1 Fatal error: exception Parsing.Parse_error
```

### fail-if1.err

```
1 Fatal error: exception Failure("expected Boolean expression in 42")
```

### fail-if2.err

```
1 Fatal error: exception Failure("undeclared identifier foo")
```

### fail-if3.err

```
1 Fatal error: exception Failure("undeclared identifier bar")
```

### fail-length.err

```
1 Fatal error: exception Failure("illegal argument found int arraytype expected in  
length(x)")
```

### fail-print1.err

```
1 Fatal error: exception Failure("function print may not be defined")
```

### fail-print2.err

```
1 Fatal error: exception Failure("illegal argument found int expected string in 5")
```

### fail-return1.err

```
1 Fatal error: exception Failure("return gives int expected void in 42")
```

### fail-scope1.err

```
1 Fatal error: exception Parsing.Parse_error
```

### fail-structure1.err

```
1 Fatal error: exception Failure("function loop may not be defined")
```

### fail-structure2.err

```
1 Fatal error: exception Failure("function end may not be defined")
```

### fail-while1.err

```
1 Fatal error: exception Failure("expected Boolean expression in 42")
```

### fail-while2.err

```
1 Fatal error: exception Failure("unrecognized function foo")
```

Passed tests output pass-add1.out

1 3

pass-add2.out

1 3

pass-arith1.out

1 2

pass-arith2.out

1 11

pass-arith3.out

1 -3

pass-arith4.out

1 4

pass-arith5.out

1 0

pass-arrayliterals.out

1 2

pass-bool1.out

1 10

pass-bool2.out

1 10

pass-boolarr1.out

1 0



pass-boolarr2.out

```
1 3
```

pass-boolarr3.out

```
1 true
```

pass-boolarr4.out

```
1 false
```

pass-boolarr5.out

```
1 true
```

pass-boolarr6.out

```
1 true
```

pass-boolarr7.out

```
1 true
2 true
3 true
```

pass-boolarr8.out

```
1 true
2 false
```

pass-boolarr9.out

```
1 0
```

pass-comment.out

```
1 3
```

pass-config1.out

```
1 Hello
2 Line
3 This
4 Wow
5 Here
```

### pass-config2.out

```
1 Hello
2 Line
3 This
4 Wow
5 Here
```

### pass-config3.out

```
1 Hello
2 Line
3 This
4 Wow
5 Here
```

### pass-dollar1.out

```
1 Hello
2 Line 2
3 This is the best line
4 Wow
5 Here we go
```

### pass-dollar2.out

```
1 Hello
2 Line 2
3 This is the best line
4 Wow
5 Here we go
```

### pass-dynamicarr1.out

```
1 4
```

### pass-dynamicarr2.out

```
1 2
```

### pass-dynamicarr3.out

```
1 Index Out of Bounds Error.
```

### pass-dynamicarr4.out

```
1 Index Out of Bounds Error.
```

pass-dynamicarr5.out

```
1 Index Out of Bounds Error.
```

pass-dynamicarr6.out

```
1 Index Out of Bounds Error.
```

pass-dynamicarr7.out

```
1 Index Out of Bounds Error.
```

pass-fib.out

```
1 1
2 1
3 2
4 3
5 5
6 8
```

pass-for1.out

```
1 0
2 1
3 2
4 3
5 4
6 42
```

pass-for2.out

```
1 1
2 2
3 3
4 42
```

pass-func1.out

```
1 2
```

pass-func2.out

```
1 42
2 17
3 192
4 8
```

pass-func3.out

```
1 62
```

pass-func4.out

```
1 2
```

pass-func5.out

```
1 42
```

pass-func6.out

```
1 115
```

pass-func7.out

```
1 43
```

pass-func8.out

```
1 43
```

pass-func9.out

```
1 2
```

pass-gcd.out

```
1 2
```

```
2 3
```

```
3 11
```

pass-global1.out

```
1 3
```

pass-global2.out

```
1 42
```

```
2 21
```

```
3 43
```

```
4 22
```

pass-global3.out

```
1 84
```

pass-helloworld.out

```
1 Hello World!
```

pass-helloworldloop.out

```
1 Hello World
2 Hello World
3 Hello World
4 Hello World
5 Hello World
6 Hello World
```

pass-helloworldloopend.out

```
1 Hello loop!
2 Hello loop!
3 Hello loop!
4 Hello loop!
5 Hello loop!
6 Hello loop!
7 Hello end!
```

pass-if1.out

```
1 10
2 2
```

pass-if2.out

```
1 10
2 2
```

pass-if3.out

```
1 2
```

pass-if4.out

```
1 5
2 2
```

pass-if5.out

```
1 10  
2 5
```

pass-if6.out

```
1 5  
2 10
```

pass-if7.out

```
1 10  
2 5
```

pass-intarr1.out

```
1 0
```

pass-intarr2.out

```
1 3
```

pass-intarr3.out

```
1 1
```

pass-intarr4.out

```
1 0
```

pass-intarr5.out

```
1 1
```

pass-intarr6.out

```
1 1
```

pass-intarr7.out

```
1 20  
2 1  
3 1
```

pass-intarr8.out

```
1 true
2 false
```

pass-intarr9.out

```
1 0
```

pass-local1.out

```
1 84
```

pass-local2.out

```
1 47
```

pass-nf1.out

```
1 1
2 2
3 5
4 1
5 3
6 0
```

pass-ops1.out

```
1 3
2 -1
3 1
4 2
5 50
6 99
7 false
8 true
9 99
10 true
11 false
12 99
13 true
14 false
15 99
16 true
17 true
18 false
19 99
20 false
21 true
22 99
```

```
23 false
24 true
25 true
```

### pass-ops2.out

```
1 true
2 false
3 true
4 false
5 false
6 false
7 true
8 true
9 true
10 false
11 true
12 false
13 -10
14 41
15 42
```

### pass-ops3.out

```
1 hello world
2 99
3 false
4 true
5 99
6 true
7 false
8 99
9 true
10 false
11 99
12 true
13 true
14 false
15 99
16 false
17 true
18 99
19 false
20 true
21 true
```

### pass-ops4.out

```
1 false
2 true
3 99
4 true
```



```
5 false
6 99
7 false
8 true
9 99
10 true
11 false
```

pass-print.out

```
1 8
2 8
3 8
4 8
5 8
6 8
```

pass-printbegin.out

```
1 Hello world
```

pass-rgx1.out

```
1 [0-9]*
```

pass-rgx2.out

```
1 [0-9]*
```

pass-rgx3.out

```
1 true
```

pass-rgx4.out

```
1 false
```

pass-rgx5.out

```
1 true
```

pass-rgx6.out

```
1 true
```

pass-rgxarr1.out

1 0

pass-rgxarr2.out

1 3

pass-rgxarr3.out

1 [0]\*

pass-rgxarr4.out

1 [2]\*

pass-rgxarr5.out

1 [0]\*

pass-rgxarr6.out

1 [0]\*

pass-rgxarr7.out

1 [6\*]

2 [0\*]

3 [0\*]

pass-rgxarr8.out

1 true

2 false

pass-rgxarr9.out

1 0

pass-strarr1.out

1 0

pass-strarr2.out

1 3

pass-strarr3.out

1 a

pass-strarr4.out

1 p

pass-strarr5.out

1 a

pass-strarr6.out

1 a

pass-strarr7.out

1 hi  
2 a  
3 a

pass-strarr8.out

1 true  
2 false

pass-strarr9.out

1 0

pass-string1.out

1 hello

pass-string2.out

1 hello

pass-string3.out

```
1 hello world
```

pass-string4.out

```
1 3
```

pass-while1.out

```
1 5
2 4
3 3
4 2
5 1
6 42
```

pass-while2.out

```
1 14
```

**Demo program** For our demo, we put an ASCII art image in a file, split it by columns delimited by "—", and shuffled the columns. Then, we took an excerpt of text (from the Constitution) and injected it randomly into the shuffled columns. The bawk program that shuffled the original image is listed below:

shuffled.bawk

```
1 BEGIN {}
2 LOOP {
3   nprint("|");
4   nprint($5);
5   nprint("|");
6   nprint($3);
7   nprint("|");
8   nprint($1);
9   nprint("|");
10  nprint($2);
11  nprint("|");
12  nprint($4);
13  nprint("|");
14  nprint($6);
15  nprint("|");
16  nprint("\n");
17 }
18 END {}
19 CONFIG {
```

```
20 | FS = "|";
21 | }
```

The goal of our demo was to unshuffle the file and use regex matching to remove and isolate the Constitution words. We also wanted to demonstrate the functionality of our multidimensional arrays, so we wrote a function to perform matrix multiplication and ran it on some inputs in the END block. The full text of the demo program is listed below.

### demo.bawk

```
1 BEGIN {
2   function int[][] matrix_mult(int[][] product, int[][] a, int[][] b) {
3     int i;
4     int j;
5     int k;
6
7     for (i = 0; i < length(a); i++) {
8       for (j = 0; j < length(b[0]); j++) {
9         for (k = 0; k < length(a[0]); k++) {
10          product[i][j] += a[i][k] * b[k][j];
11        }
12      }
13    }
14
15    return product;
16  }
17 }
18
19 LOOP {
20   int i;
21   int j;
22   int col;
23   int[] reshuffle_arr;
24   string constitution_word;
25   rgx rgx_match;
26
27   reshuffle_arr = [3, 4, 2, 5, 1, 6];
28   rgx_match = '\*[a-zA-Z]*\*';
29   constitution_word = "";
30
31   # Reshuffle columns
32   for (i = 0; i < length(reshuffle_arr); i++) {
33     col = reshuffle_arr[i];
34     if ($col !~ rgx_match) {
35       nprint($col);
36     }
37     else {
38       constitution_word = $col;
39     }
40   }
```

```

41 | nprint("\t");
42 | nprint(constitution_word);
43 | nprint("\n");
44 | }
45 |
46 | END {
47 |   int i;
48 |   int j;
49 |   int k;
50 |   int l;
51 |   int m;
52 |   int n;
53 |   int[][] a;
54 |   int[][] b;
55 |   int[][] product;
56 |
57 |   a =      [[1,2,3,1,2,3],
58 |            [1,2,3,1,2,3],
59 |            [1,2,3,1,2,3],
60 |            [1,2,3,1,2,3],
61 |            [1,2,3,1,2,3],
62 |            [1,2,3,1,2,3]];
63 |   b =      [[1,2,3,1,2,3],
64 |            [1,2,3,1,2,3],
65 |            [1,2,3,1,2,3],
66 |            [1,2,3,1,2,3],
67 |            [1,2,3,1,2,3],
68 |            [1,2,3,1,2,3]];
69 |   product = [[0,0,0,0,0,0],
70 |             [0,0,0,0,0,0],
71 |             [0,0,0,0,0,0],
72 |             [0,0,0,0,0,0],
73 |             [0,0,0,0,0,0],
74 |             [0,0,0,0,0,0]];
75 |   product = matrix_mult(product, a, b);
76 |
77 |   # Print A
78 |   print("A: ");
79 |   for (i = 0; i < length(a); i++) {
80 |     for (j = 0; j < length(a[0]); j++) {
81 |       nprint(int_to_string(a[i][j]));
82 |       nprint(" ");
83 |     }
84 |     nprint("\n");
85 |   }
86 |   nprint("\n");
87 |
88 |   # Print B
89 |   print("B: ");
90 |   for (k = 0; k < length(b); k++) {
91 |     for (l = 0; l < length(b[0]); l++) {
92 |       nprint(int_to_string(b[k][l]));
93 |       nprint(" ");
94 |     }
95 |     nprint("\n");
96 |   }

```

```
97  nprint("\n");
98
99  # Print A x B
100 print("A x B: ");
101 for (m = 0; m < length(product); m++) {
102     for (n = 0; n < length(product[0]); n++) {
103         nprint(int_to_string(product[m][n]));
104         nprint(" ");
105     }
106     nprint("\n");
107 }
108 }
109
110 CONFIG {
111     FS = "|";
112 }
```

The output of the demo is too verbose to put in the report, but can be produced by running

```
$ ./bawk.sh demo/demo.bawk demo/shuffled.txt
```