FUNC-y Java

Programming Languages and Translators: Spring 2021

Liseidy Bueno (lb3347)
Kenya Plenty (kgp2111)
Pazit Schrecker (prs2143)
Lindsey Wales (lbw2149)
Katrina Zhao (kz2335)

Sections

Contents

1 Introduction

FUNC-y JAVA is a statically typed procedural language that helps new programmers warm up to coding in Java without overwhelming them with classes, objects, complex built-in data structures, etc.

Func-y Java combines powerful functionality with clear syntax. FUNC-y Java is a stepping-stone language with the goal of helping new programmers build their skills and making it easier to learn other languages in the future. For many, learning to code for the first time is difficult. At Columbia, most students begin with Java, an Object-oriented programming language. This can be difficult because it requires users to learn about the construction of algorithms, control flow, bracket-ing and syntax, and types at the same time as classes and objects.

1.1 Tutorial

1.2 Setting up FUNC-y Java

1.2.1 Environment Setup

To begin, download the code for FUNC-y Java's compiler and tests from http://www.cs.columbia.edu/ sedwards/classes/2021/4115-spring/ Then you will have to install LLVM 10.0.0 as well as OCaml 4.11.1 and OPAM.

1.2.2 Compilation

Build the compiler using the make command. If needed, use the make clean command to get rid of any old object or executable files:

```
make
make clean
```

To run all of the tests in our automated testing suite, use the make test command.

```
make
make test
```

If a passing test produces the expected output or if a failing test produces the expected error, then you should see "AMAZING" in your terminal. If this is not the case, check that you have set up up your environment correctly in the previous step. To make your own FUNC-y Java program, first check that your source code file ends with the .fj extension. To write your program, follow the rules laid out in the language reference manual in Section 2. After saving your file, you can compile it into an executable using the following commands in the terminal.

```
./funcyjava.native  <filename> > <filename>.ll
llc -relocation-model=pic <filename>.ll > <filename>.s
cc -o <filename>.exe <filename>.s
./<filename>.exe
```

1.2.3 Language Tutorial

Let's create our first FUNC-y Java program! Make a file called hello-world.fj. Add the following code to the file:

```
func int main() {
     str x;
     x = ("Hello World!");
     print(x);
}
```

Using the instructions mentioned above create run your hello-world file and go crazy man.

2 Language Reference Manual

2.1 Lexical Elements

2.1.1 Whitespace

Whitespace is a term used to describe several different characters: tab, space, and newline. In FUNC-y Java, whitespace is ignored for the most part. For example, if you put spaces around operands in a FUNC-y Java program, this whitespace will be ignored and not taken into consideration when compiling the program. The only instance in which whitespace is taken into account is when it appears in a string.

2.1.2 Comments

FUNC-y Java supports both multi-line and single-line comments. It is important to note that during compilation, the compiler will ignore anything that appears in the body of the comment. A comment starts with the characters /* and ends with the characters */.

```
/* "I am a comment." */
```

2.1.3 Identifiers

In FUNC-y Java identifiers are a sequence of at least one character. An identifier must start with at least one lowercase ASCII character. After the first character, the remainder of the identifier can be filled with any combination of underscores and lower and uppercase ASCII characters only. Finally, identifiers cannot have the same name as the reserved words listed below.

2.1.4 Keywords

char	**int**	**float**
func	**array**	**for**
while	**if**	**else**
not	**true**	**false**
return	**cry**	**and**
or	**bool**	**main**
str		

2.2 Literals

Each literal has a specific data type that relates to one of the primitive data types defined below. FUNC-y Java does not support automatic promotion or combining of different literals.

2.2.1 Integer

An optionally signed sequence of at least one digit (0-9). These literals will be treated as whole numbers. In order to make an integer negative, you must prepend the '-' character in front of it (see examples below).

```
9 -10 11
```

2.2.2 Floating Point

An optionally signed sequence of at least one digit (0-9) followed by a '.' and another sequence of at least one digit. You can declare a negative float literal by prepending the '-' in front of it.

```
9.0 0.9 0.999
```

2.2.3 Character

A singular ASCII character enclosed in single quotation marks. It is important to note that chars in FUNC-y Java cannot be used in combination with any of our operators.

```
'c' '!' '#' '\'
```

2.2.4 String

A sequence of at least one ASCII character enclosed in double quotation marks. It is important to note that in the backend, all strings are stored as an array of individual characters and they cannot be used with any operators.

```
"Hi" "lovely" "c"
```

2.2.5 Boolean

In FUNC-y Java, **true** and **false** represent our boolean literals. In the backend, they are represented as 1 and 0 respectively.

2.3 Data Type Specifiers

char	An 8-bit ASCII character.
int	A 32-bit signed integer.
float	A 64-bit signed decimal number.
bool	A 1-bit variable with an underlying value of 1 or 0.
str	A sequence of ASCII characters.
cry	Represents void in FUNC-y Java.

2.4 Arrays

In FUNC-y Java, an array is an immutable and ordered collection of primitives of the same type. The elements in an array can be of type **int**, **str**, or **float**. Additionally, arrays can be initialized with negative numbers and decimals, but they will simply be converted to 0 or 0.0 at run time.

2.4.1 Declaring and Initializing Arrays

An array has a non-mutable length that must be defined at declaration, and declaration and initialization of the array must happen together. An array declaration should be of the following form:

array<*data-type-specifier, int-size-of-array*> *identifier = [primitive, primitive,...];*

2.4.2 Accessing Array Elements

An array's elements may be accessed using an index that is either an integer literal, a variable that is of type **int**, an expression of any kind. It is important to note that no matter the index passed, it will be treated as valid during compile time and will not be checked in any way. Furthermore, you can only access an array's elements in a variable declaration of assignment. The following are examples of valid and invalid ways to access an array's elements in FUNC-y Java.

2.4.3 Array Mutation

It is important to note that in FUNC-y Java, the elements of an array cannot be changed in any way after declaration. In other words, you cannot change an individual element in the array using its index and you cannot reassign an array to a whole new set of values at any time. (See examples below)

```
array<str,3> greetingsArray = ["hi", "hey", "howdy"];
/* The following lines of code are not allowed.*/
```

```
greetingsArray[0] = "yo";
greetingsArray = ["sup", "hiii", "heya"];
```

2.5 Expressions

2.5.1 Primary Expressions

identifiers
In FUNC-y Java, identifiers are used to name functions and variables. It is important to note that for the remainder of this LRM, the phrase *identifier-expression* will be used to denote occasions in which variable names have to be one of the operators.

identifier-expression(expression-list)
A function call is an *identifier-expression* followed by parentheses containing a possibly empty comma-separated sequence of expressions, which constitute the actual arguments that will be passed to the function. It is also important to note that functions can be recursive in FUNC-y Java. As a result, an identifier-expression (expression-listopt) can be the return value of any function.

literals
Please see the sections above for more information about literals in FUNC-y JAVA.

(expression)
A parenthesized expression has the same value and type as an unadorned expression. The presence of the parentheses does not change associativity of the expression.

2.5.2 Unary Operators

not *expression*
Placing the **not** operator in front of any logical expression will cause its final boolean value to be inverted. For example, if the expression would evaluate to **true** normally, the not would cause it to evaluate to **false**.

- expression
The result is the negative expression with exactly the same type. The type of this type of expression must be int or float.

identifier-expression++
The overall result of this expression is that the value referred to by the identifier-expression gets incremented by one. If ++ operator were to be placed in front of the *identifier-expression*, this would cause an error to be thrown This operator can only be used with **int**.

identifier-expression−
The overall result of this expression is that the value referred to by the *identifier-expression* gets decremented by one. The same operator restrictions apply as with ++.

2.5.3 Multiplicative Operators

The multiplicative operators *, /, % and group left-to-right. For almost all of the operators listed below, the operands must both be of type **int** or **float**. Attempting to mix operands of type **int** and **float** will result in an error. It is not allowed.

expression * expression
The binary * operator indications multiplication. When evaluated, the expression above will multiply the values of both expressions.

expression / expression
The binary / operator indicates division. If both expressions are of type **int**, then integer division will be performed and any remainder will be thrown away.

expression % expression
The binary % operator yields the remainder from the division of the first expression by the second expression. Both operands must be int and the sign of the remainder will match the sign of the numerator or the first expression.

2.5.4 Additive Operators

The additive operators are + and -. They will group left-to-right. Just like in the case of the multiplicative operators both operands must be of type **int** or **float**.

expression + expression
The result is the sum of the values that each operand evaluates to.

expression - expression
The result is the subtraction of the value of the second expression from the first expression.

2.5.5 Relational Expressions

The relational operators group left-to-right. They can be used with the following types: **int** and **float**. It is important to note that boolean literals in FUNC-y Java are converted to the integer values 1 and 0. Finally, types of both operands must be the same.

expression* < *expression
expression* > *expression
expression* <= *expression
expression* >= *expression

The binary operators < (less than), > (greater than), <= (less than or equal to), and >= (greater than or equal to) all yield false if the specified relation is **false** and **true** if the specified relation is true. They can only be applied on **int** and **float**.

2.5.6 Equality Operators

It is important to note that both operands must be of the same type for proper evaluation to take place. Additionally, both operators below can be used with all of our primitive data types except for **cry** (really just void).

expression != expression
expression == expression
The == (equal to) checks that the value of the expressions are equal and != (not equal to) checks that they are not the same. Additionally, when printing the result of an equality operator, true and false will appear in the terminal.

2.5.7 Logical Operators

expression and expression
The and operator will return **true** if both of its operands evaluate to **true**. If the previous condition is not met, then **false** will be returned. The and operator guarantees that evaluation will take place from **true** to right. As a result, if the left operand evaluates to **false**, then the right operand will not be evaluated at all.

expression or expression
The or operator will return **true** if either of the operands are **true**. If both operands evaluate to false, then false will be returned. Just like the and operator, evaluation will take place from left to right. If the first operand evaluates to true, then the right operand will not be evaluated at all.

2.5.8 Assignment Operators

FUNC-y Java has a number of assignment operators. They all group right-to-left. It is also important to note that the left side of an assignment operation must be an identifier for a variable and that the type of the variable and expression must be the same at all times.

identifier-expression* = *expression

The value of the expression on the right replaces that of the variable referenced by *identifier-expression*. An assignment may be given to an already initialized identifier or it may be given on the same line. Both of the following assignments are valid:

identifier-expression* += *expression
identifier-expression* −= *expression
identifier-expression* ∗= *expression
identifier-expression* /= *expression
identifier-expression* %= *expression

The behavior of an expression in the form *identifier-expression* = *expression*, will operate as if they were written as *identifier-expression* = *identifier-expression operator expression*. In

other words, the value of the *identifier-expression* will be mutated by the *operator* and the *expression*, and then that value will be assigned back to the original *identifier-expression*. These assignments can only be used with initialized variables only.

2.6 Operator Precedence and Associativity Summary

Operators	Associativity
− (unary minus), not, ++, −−	right-to-left
∗, /, %	left-to-right
+ , −	left-to-right
<, >, <=, >=	left-to-right
! =, ==, and, or	left-to-right (only guaranteed for and, or)
=, + =, − =, ∗ =, / =, % =	right-to-left

2.7 Statements

Unless stated otherwise all statements are executed in sequence

2.7.1 Expression Statements

The majority of statements in FUNC-y Java are usually expression statements that usually indicate assignment or function calls. They must always be followed by semicolons. They have the form:

expression;

2.7.2 Conditional Statements

There are two types of conditional statements in FUNC-y Java. They have the following forms:

if (*expression*) {*statement*}
if (*expression*) { *statement* } **else** { *statement* }

In both cases, if the expression is resolved to **true**, the first statement will execute. If it resolves to **false**, then the second statement will execute. If there is just a singular if statement and the expression resolves to **false**, then regular execution continues on from outside the scope of the first statement. If there happens to be a "dangling" else, it will be assigned to the last elseless if that has been encountered.

2.7.3 While Statements

In FUNC-y Java, while statements have the following form:

while (*expression*) { *statement* }

At the beginning of each round of execution, the *expression* is evaluated. If the result is **true**, then execution of the statement occurs. If the expression evaluates to **false**, then the statements will be skipped over and normal execution will continue from outside the scope of the *statement*.

2.7.4 For Statement

Borrowing from its sibling Java, FUNC-y Java's for statements have the following form:

for($expression_1$; $expression_2$; $expression_3$) { *statement* }

The first expression sets the starting value of the counter variable. The second expression specifies a test, made before each iteration, such that the loop is exited if it evaluates to **false**. The third expression usually specifies an incrementation of the loop variable. All three expressions are required in order to use this **for** loop. It is important to note that the counter variable must be declared outside of the **for** loop statement and set to its starting value inside the for statement.

2.7.5 Return Statement

return *expression*;

In FUNC-y Java, a **return** statement is used to exit out of the scope of a particular function and **return** back to the caller function with the specified expression. It is important to note that return statements are required for all functions in FUNC-y Java that are not **main** (see more about **main** below).

2.7.6 Statement Lists

statement

In FUNC-y Java statements that appear one right after each other in between curly brackets are considered a singular unit and are run in sequence. These typically show up in the body of a function, while statements, etc.

2.8 Functions

There are several factors one has to take into account when using user-defined functions in FUNC-y JAVA. From this point on, the term function refers to user-defined functions. First, all functions must have a return type (see data type specifier section). However, parameters to functions cannot have the type cry (because this is void). Additionally, it is not necessary for functions to take in parameters or to have statements in its body.

2.8.1 Declaring Functions

All user-defined functions in FUNC-y Java must follow the format given below.

function-definition:
    func *type-specifier function-declaration*

function-declaration:
    identifier (*parameter-list*$_{opt}$)

parameter-list:
    data-type-specifier identifier,
    data-type-specifier identifier, *parameter-list*,

function-body:
    function-statement

function-statement:
    statement-list$_{opt}$

Below are examples of a valid function declaration in FUNC-y Java.

2.8.2 Calling Functions

As stated above, functions in FUNC-y Java can only be defined outside of **main** and may only be run inside of **main**. As a result, one must use the following format to correctly call a function within **main**. It is important to note when calling a function, you only need to pass expressions that evaluate to the types associated with each parameter when the function was declared.

identifier expression-list$_{opt}$

2.8.3 main

In FUNC-y Java, **main** is a special function that must appear in every valid FUNC-y Java file. Any statements, expressions, functions, etc that you want to run should appear or be called in main. Additionally, it is important to note that you cannot declare functions within **main** and you may not pass any parameters to **main** either. Additionally, **main** can only have a return type of int. However, it is not necessary to put a return statement at the end of **main**. A valid return statement will be automatically appended to the end of the body of every file's **main** function.

2.8.4 print(x)

This is a built-in function that the user does not need to declare in order to use it. This function takes the parameter x and outputs it directly to the terminal. This function can only be passed expressions that evaluate to the following types: **str**, **char**, **int**, **bool**, and **float**. Attempting to pass any other type will result in an error being thrown. It is important to note that string and character literals will be printed with quotation marks and float literals will be printed without truncating the decimal values.

3 Project Plan

3.1 Project Roles

Liseidy Bueno
Role: System Architect
While I was one of the systems architects, I think the way that the project developed has us all working on compiler implementation. At first, we worked by file (the Parser and Scanner, then the AST, SAST, and Semantic Checker), but then found it simpler to work by feature and make sure that everything was consistent throughout all our project files. I personally worked mostly on conditionals and arrays after the "Hello World" milestone.

Kenya Plenty
Role: Manager
As stated by my other teammates, in the beginning we all kind of swarmed or worked on files together. However, after we changed our language to FUNC-y Java, I was primarily responsible for creating and ensuring that all parts of the compiler, Makefile, and testing script were working properly to execute Hello World. After this point, we started splitting up features for each person to work on. While I would be available to help with debugging and GitHub issues, I was primarily focused on creating strings, figuring out our in place operators (ie ++, –, *=, etc) in our language as well as creating the arrays that exist in FUNC-y Java. It is important to note that while I was the designated manager of the group, towards the end of the project, Pazit really took on the role of organizing meetings and other logistical tasks for the team.

Pazit Schrecker
Role: Tester
As the tester for this project, I reworked the automated microc testall.sh testing script to work for FUNC-y Java so that we could run many tests at once and verify that their outputs were correct. In this role I also wrote the majority of our 130 tests (almost every test except those concerning arrays, which were handled by Kenya and Liseidy). For each test I also created the corresponding expected .err and .out files. While writing unit and integration tests I also found places in which semantic errors were not handled and added more error checking to semant.ml as well as adding more pretty-printing functionality to ast.ml for more descriptive error messages.

Along with my role as tester, I also contributed to the source code for this project. I added the functionality to allow local variables to be declared throughout a function rather than all at the top (as is the case in microc). Xijiao (our TA) was instrumental in helping me understand how to do it. Kenya also helped me debug this and we then worked together to implement variable declaration and assignment in the same line, as well as in separate lines. I also worked with Lindsey to implement the built-in print function that prints all primitive types.

Katrina Zhao
Role: System Architect
Before the "Hello World" milestone, everyone worked together on all of the compiler files, but I was primarily focused on the Parser and Scanner. Afterwards, when we started working by features, I worked on control flow statements and return statements as well as helping with array implementation in the semantic checker and codegen and cleaning up our warnings.

Lindsey Wales
Role: Language Guru In the beginning of the project we all worked on all of the files/features, although I probably spent most of my time on the AST. After the "Hello World" milestone I focused on binary operators, unary operators, boolean functionality (printing booleans as "true" and "false" instead of 0 and 1), a universal print function, and control flow. I also spent time helping with other sections like arrays and string functionality.

3.2 Project Timeline

Date	Achievement
Jan 25	First team meeting where different language ideas were proposed
Jan 28	Decided to move forward with CardY
Feb 1	Completed CaRdY Project Proposal
Feb 24	Completed CaRdY Language Reference Manual and Parser
Mar 8	Decided to change our language to FUNC-y Java
Mar 16	Completed LRM for FUNC-y Java
Apr 2	Completed Hello World Milestone
Apr 23	Finished Implementing FUNC-y Java Features
Apr 24	Finished Automated Testing Suite and Code Review
Apr 25	Finished Final Presentation and Report

3.3 Language Evolution

As seen in the above timeline, we pivoted pretty significantly shortly after beginning work on our project. We had originally planned a language called CaRdY, but as we learned more about the structure of compilers we realized that it made sense to refine our vision. After quite a bit of discussion, we decided on FUNC-y Java, a language that combines Java-like syntax with aspects of Python's more functional paradigm. Once we redefined our language we got to work on a new Language Reference Manual and began working on the "Hello World" milestone. Once the basic arithmetic functionality and primitive types

were implemented we shifted our attention to more advanced features such as arrays. As we implemented features we wrote unit tests to accompany them. Our tester also built up a more complete test suite featuring over one hundred tests and automated testing. Although we weren't ultimately able to implement every feature we had planned on, we ended up with what we think is a pretty cool little language for beginners and anyone scared of objects.

3.4 Future Work

Although our language currently provides the basics for beginner programmers who want an easy transition into Java, there are some additional features that might be useful for programmers who want to implement slightly harder code. For example, although our language lets programmers make their own arrays and access them, they currently cannot do more complicated operations like changing the values of an existing array or combining elements from multiple arrays together. Thus, we would like to implement more array operations such as element reassignment, array concatenation, checking to see if an element is in an array, finding the index of an element, reversing an array, and sorting an array. Our array type is currently of a fixed length as well. We could implement another collection type like lists that function similarly to arrays but have a mutable length.

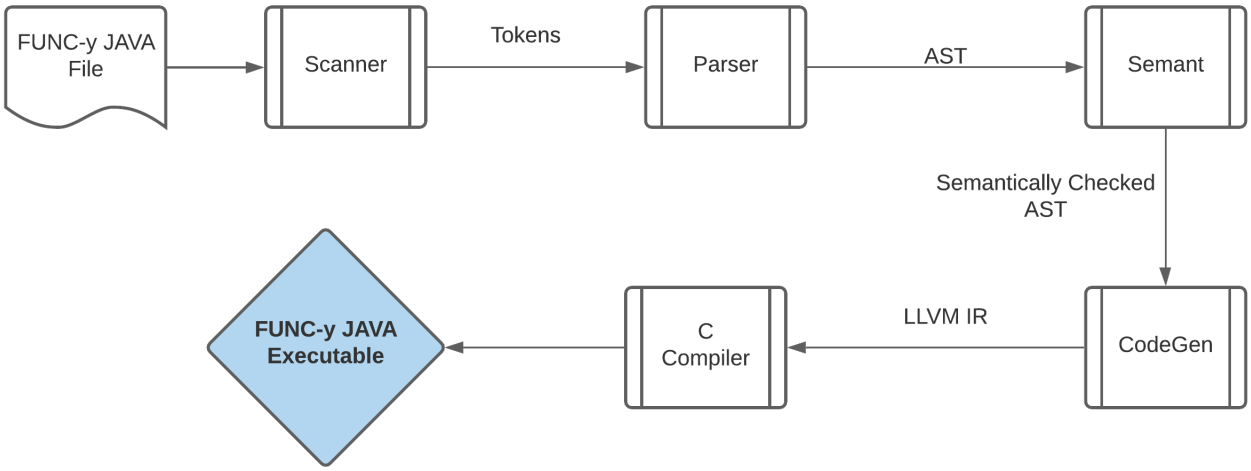
Additionally, we could also add more operations on strings by linking in a library of functions in C with our current architecture in codegen. We currently can perform basic operations like checking for equality between two strings and printing out a string, but with an external library of C functions, we could implement much more complex functions cleanly. These string functions could include operations like getting the length of a string, taking a substring of a string, concatenating strings, finding the character at an index of a string, seeing if a string contains a certain substring, etc.

Finally, our language does not allow for type conversion. We would like to implement more built in functions that could convert an int, float, or char into a string, a float into an int, and an int into a float. Being able to convert from one type of an array to another (such as an int array to a string array) would be another interesting extension of our current features. Having these conversions as well as other built in string and array functions could adapt FUNC-y Java to give intermediate programmers more flexibility with their programs.

3.5 Project Tool

Each member of our team worked on either Visual Studio Code or Atom as a text editor; we each worked locally on our own computers and used GitHub for versioning. We created branches for each of our features and merged them to a main branch once it was working and tested.

4 Translator Architecture



4.1 Scanner

File: scanner.ml
The scanner takes in a FUNC-y Java program file and breaks it up into valid tokens within our language. An example of a valid token would be the keyword **array**. It is important to note that while the program detects whitespace characters in the tokenizing process, these are mostly thrown out in the compilation process unless they appear within a string a character element.

4.2 Parser

File: parser.mly
The parser attempts to convert the incoming stream of tokens into an Abstract Syntax Tree (AST) based on FUNC-y Java's context free grammar rules as described in the Language Reference Manual section as well as in the parser.mly file itself.

4.3 Semantic Checking or Semant

File: semant.ml
The semantic checker recursively traverses the AST and converts it to a Semantically Checked Abstract Syntax Tree (SAST). At this point, scoping or typing errors will be detected and properly reported to the user. For example, if variables are assigned to a type that they were not declared with, then an error will be produced.

4.4 Codegen

File: codegen.ml
Within the codegen file, the SAST produced through the semantic checking processing is

recursively traversed in order to generate LLVM IR with the help of LLVM modules.

4.5 C Compiler

Although this stage is simply referred to as 'C Compiler' there are several intermediary steps that take place here. At this stage, the LLVM IR is run through the llc compiler to produce assembly code. At this point, our assembly code is passed through a cc compiler to create the final FUNC-y Java executable. Yeah! It is important to note that although FUNC-y Java does not incorporate any outside libraries at this point, we intend to expand our language to include string operations and other features. These methods can only be incorporated into our language by bringing them in as outside compiled object files that will be linked into our final executable. Thus, while it may seem unnecessary to push our files through the cc compiler, it was a necessary step to unsure that future code could be incorporated from outside.

5 Testing Process

5.1 Automated Testing Suite

We tested FUNC-y Java syntax and features using a shell testing script, adapted from the microc testing script. The automated testing suite runs two types of tests: failing tests and passing tests. Failing tests were named in the form 'fail-testname.fj' and were tested by comparing the produced output to the expected output error messages, stored in a file of the name but ending with .err instead of .fj. Passing tests were named in the form 'test-testname.fj' and were tested by comparing the produced output to the expected output messages, stored in a file of the name but ending with .out instead of .fj. The automated testing suite contains a total of 130 tests.

5.2 Unit Tests

Unit tests were written by each member of the group when a new feature was added, before pushing the feature to the main branch. This was done to ensure that features worked properly before being incorporated into the official codebase. After the features were pushed to main, the tester wrote more unit tests for the feature to check for edge cases. Failing tests were also added after integrating the code.

5.3 Integration Tests

Passing integration tests were written by the tester after many large features had been added, such as the entirety of control flow or all binary operators. These combined multiple features to make sure that all features integrated correctly and to check for any unexpected behavior.

5.4 Error Messages

When adding unit and integration tests to the automated test suite, we came across a number of bugs and error messages that were not properly implemented. Appropriate error messages were added throughout the testing process in order to provide FUNC-y Java programmers with a clear understanding of where they went wrong.

6 Lessons Learned and Advice

6.1 Liseidy Bueno (System Architect)

Some lessons I learned while working on this project is to dream big but also be realistic with time constraints. I think in general it's easy to get carried away with ideas and ultimately need to scale back. I also learned that working by feature is a lot more effective than working by file. Communication proved to be key, and pair programming was much more helpful than all five of us trying to work on one thing together.

My advice to future groups: Having clear expectations of your project and concrete deadlines will definitely help move the project along. Start early, and focus on one feature at a time. Make use of office hours and your TA as much as possible as well.

6.2 Kenya Plenty (Manager)

The most significant lesson I learned working on this project came from my experience as the team manager. As someone who feels more comfortable overworking myself than asking for help, I initially struggled with delegating tasks. As the project progressed, I realized that the more well-defined our roles and responsibilities were, the more quickly and efficiently things got done.

I would advise future teams (especially future managers) to keep track of the tasks that remain to be completed and, when necessary, ask team members to take on specific responsibilities. Clearly communicated expectations and regular check-ins are two of the most important tools for keeping a team on track.

6.3 Pazit Schrecker (Tester)

The primary lesson I learned in working on this project is the importance of clear and regular communication when working in a group. We struggled with this at times and lack of or miscommunication cost us time. It is especially important in order to make sure that every team member has clearly assigned work so that no two people are wasting time implementing the same feature while another feature remains unfinished. This project experience also underscored the importance of planning ahead because there are some aspects of the project that built on other components and these needed to be implemented in a specified order.

I would advise future teams to start the project as early as possible. I found that compiler structure and OCaml have a steep learning curve and I didn't fully understand how all the pieces fit together until I began implementing features. Lastly, I would highly recommend being the group tester. Watching over 100 tests in a test suite all compile is extremely satisfying.

6.4 Lindsey Wales (Language Guru)

Similar to what my teammates expressed, my biggest takeaway from this project was the importance of clear scheduling and communication guidelines. This was by far the longest I've ever worked on a single project, and I don't think I realized until fairly late in the semester how seriously you need to treat longterm time management with a project this big. I definitely have a habit of treating long-term goals as sort of hazy ideas to be dealt with later, whereas short-term goals are more tangible, definite commitments. After working on this project I realized that in order to keep a three month long process under control, you need to have as clear of a plan for accomplishing far-away deadlines as you do for short term ones.

My advice to future teams is to start everything earlier than you think you need to! The worst case is things go sideway and you end up needing all of the extra time (or more than) you allotted. Best case, you finish early and spend the extra time implementing more features or eating snacks and hanging out.

6.5 Katrina Zhao (Systems Architect)

One of the main lessons I learned from this project is to have a very clear idea of what features you want to implement early but be mentally prepared to cut back on them. We were very ambitious with the number of things we wanted to implement in our language after our first meeting and in our proposal, but as we tackled each task, we found that features that sounded very simple at first were actually very difficult to make. It's very important to both start early and to give yourself extra time to finish in case unexpected complications come up at the last minute.

I also learned to try to work on one small aspect of your language at a time. If you try to knock out every aspect of your language in one file before working on the next file, you will inevitably run into a lot of bugs and issues with running them together at the end. Additionally, making use of your resources will be super helpful to you. Our group met up with our TA quite frequently, and she gave us a lot of good advice on how to improve our language conceptually as well as how to actually implement the trickier parts of our code.

7 Appendix

7.1 Project Log

```
commit 8458e6d3d84c66257b5fda26985be0a31e4e1027
Author: Liseidy Bueno <liseidybueno@gmail.com>
Date:   Mon Apr 26 01:34:12 2021 -0400

    add tests for float arrays

commit cf6f01a1b10549bf042d2934a9e34e873c316d6d
Merge: 56a34b1 e2cadf3
Author: Katrina <kz2335@columbia.edu>
Date:   Mon Apr 26 00:41:39 2021 -0400

    Merge branch 'arrayt' of github.com:kenyaplenty/FUNC-y-Java into arrayt

commit 56a34b1e95f51db5a5116581f4a5c1a1d3b6a7d7
Author: Katrina <kz2335@columbia.edu>
Date:   Mon Apr 26 00:41:31 2021 -0400

    more string tests

commit e2cadf36e8e9af4172a2e3a87e00fa9a19a980b9
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Mon Apr 26 00:39:59 2021 -0400

    addes lisiedy tests back

commit e114d22c03747c4dc55468af04b697b780d8ef32
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Mon Apr 26 00:06:08 2021 -0400

    added more passing int array tests and fixed typo in parser

commit e2efc12f1ced00b23716794cbe11181da8de0ec3
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Apr 25 22:54:36 2021 -0400

    updated syntax for arrays and some new tests

commit e9a97c982a7bc12223cd08bd167c5248f949e9f4
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Apr 25 21:56:42 2021 -0400
```

```
    changed the array suntax to line up

commit dd46ca3143f3e7576507ee833ffd99896a45df41
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Apr 25 21:16:04 2021 -0400

    removed warnings

commit 94019ee79ede5a2c9019cce19a1d328fac0c1278
Author: Pazit Schrecker <pazitrose1999@gmail.com>
Date:   Sun Apr 25 10:54:56 2021 -0400

    Updated readme with instructions for running code

commit c4180af003f8d25f99a89c691ebb46dcdce5f59a
Merge: 9909c2e 0e9d1d6
Author: Pazit Schrecker <pazitrose1999@gmail.com>
Date:   Sun Apr 25 10:49:54 2021 -0400

    Merge pull request #16 from kenyaplenty/testing

    Testing

commit 0e9d1d67a3d2c9fe800d0ef5fb4f3a6995378b17
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sun Apr 25 10:48:51 2021 -0400

    fixed all non-array related warnings

commit 9909c2e26716a7a10daa5cea83393ab8b18bc231
Author: Lindsey Wales <43915800+lindseywales5@users.noreply.github.com>
Date:   Sun Apr 25 10:31:43 2021 -0400

    Adding "not" to the scanner

commit 6c6f42a91d5518326169fadd1f54c8eeb4005079
Author: Lindsey Wales <43915800+lindseywales5@users.noreply.github.com>
Date:   Sun Apr 25 10:30:14 2021 -0400

    Adding " not" functionality

commit 08a763f7212780d342120f88da14e1bd389b931f
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sun Apr 25 10:06:59 2021 -0400
```

```
    Added string_of_op for better error message
    printing, updated tests to comply with this

commit 26e1a99005299e01b27121b6bc8708fa90099749
Merge: fd8d811 bbebc00
Author: Pazit Schrecker <pazitrose1999@gmail.com>
Date:   Sun Apr 25 09:44:33 2021 -0400

    Merge pull request #15 from kenyaplenty/testing

    Testing branch: now contains failing tests for error message comparison

commit bbebc00dba9bf598a0677168a677f5463519f7b8
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sun Apr 25 09:41:36 2021 -0400

    Added more failing tests

commit fd8d8113953b3a91b73b95cf4db147aff6cedaa2
Merge: ff7e1bd 8d205a8
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Apr 25 01:58:35 2021 -0400

    Merge pull request #14 from kenyaplenty/new_list

    Added list stuff and created test

commit 8d205a8022c4d57bffab08afec80249d4f904c1f
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Apr 25 01:57:27 2021 -0400

    finished the last merge

commit 1dffc64e5ef10758f2368750454b7d782708ff5d
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Apr 25 01:55:18 2021 -0400

    removed the build files

commit 8002a536aee521d4b6eea42fb28cc6a5565714cf
Merge: 99b3cd4 f00cc2e
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Apr 25 01:54:32 2021 -0400
```

```
    merged

commit 99b3cd40fecf6218397b741898ac080657db290b
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Apr 25 01:50:33 2021 -0400

    Did some clean up and created relevant test file

commit 3c775cc9df1a831c53a4a187e6a63faf1550f264
Author: Liseidy Bueno <liseidybueno@gmail.com>
Date:   Sun Apr 25 00:39:03 2021 -0400

    added lists of ints, strings, and floats

commit f00cc2ef3352cecd7de330845f53a81403efe030
Author: Liseidy Bueno <liseidybueno@gmail.com>
Date:   Sun Apr 25 00:39:03 2021 -0400

    added lists of ints, strings, and floats

commit 93f1f7086bca23978ad2b4a10d2b49acd2f76f67
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sat Apr 24 22:08:05 2021 -0400

    Added failing tests for assignment operators,
    updated ast to include stirng representation of char for
    testing

commit ff7e1bdcddb34cbf0f396b2aa90f2ed2e9200780
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sat Apr 24 21:48:32 2021 -0400

    Added assignment and binop failing tests with working error messages

commit a2123beeab715278d51f3e06641b890b4b370ab2
Merge: 46f6462 5976bc1
Author: Pazit Schrecker <pazitrose1999@gmail.com>
Date:   Sat Apr 24 20:27:04 2021 -0400

    Merge pull request #13 from kenyaplenty/testing

    Testing

commit 46f64627829c013aa8b2dadc785d9f9530dc5e0e
Author: Pazit Schrecker <pazitrose1999@gmail.com>
```

```
Date:   Sat Apr 24 20:26:46 2021 -0400

    Delete old printbool

commit 5976bc1669e196c7e8f5fafa1897cd8c10ae6d7c
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sat Apr 24 19:05:27 2021 -0400

    added more tests for function returns, fixed codegen to work with returning multiple

commit 0a47034fa95c4761f64ba3b0401aef2d7799fe3b
Author: Lindsey Wales <43915800+lindseywales5@users.noreply.github.com>
Date:   Sat Apr 24 18:01:33 2021 -0400

    "and" and "or" are working (added tests for each)

commit 1b0ffaa5bd7d51c24a4cd4cf088673fa949c0258
Author: Lindsey Wales <43915800+lindseywales5@users.noreply.github.com>
Date:   Sat Apr 24 17:20:24 2021 -0400

    Booleans are now printable as true/false

    Also had a bug where multiple successive boolean statements
    would all return the same result as the first one. That is now fixed.

commit cf34abe5b2875fc0945b02ff8600c114bf62bd25
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sat Apr 24 15:51:02 2021 -0400

    Changed variable names to accept uppercase

commit 9bc214dbcb66f8a10ea02c188331b02f13337bb0
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sat Apr 24 15:17:36 2021 -0400

    Added more loop tests

commit 0e35913f2a216585b3f96bfff813f76e7ed148e7
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sat Apr 24 12:22:49 2021 -0400

    Added some control flow tests

commit 669955b2d935ae3e7581bdcd41ed3e6eb150da19
Author: Pazit <pazitrose1999@gmail.com>
```

```
Date:   Sat Apr 24 10:47:08 2021 -0400

    Added local variable tests for assignment,
    printing, reassignment, and binops

commit d2e478ff94489ac78e6a59f637c9da48abce8c0e
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sat Apr 24 01:42:05 2021 -0400

    Updated type of main function to int, adde dmore automated testing

commit 4f7c8d5fde36c05480e5fb74e85d941d246a8172
Author: Pazit <pazitrose1999@gmail.com>
Date:   Fri Apr 23 20:22:35 2021 -0400

    Updated some testing

commit d248e3feec4579f4da330022e7f395731b717f45
Merge: 8887e0f e94b4fa
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Thu Apr 22 13:58:13 2021 -0400

    Merge pull request #12 from kenyaplenty/mod

    Added code to incorporate the mod function

commit e94b4fa28aa8f3acb9ab373692246a6287ad21af
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Thu Apr 22 13:57:51 2021 -0400

    Removed extraneous assignment

commit 0cdeae6084f3a1f482bbd8fcc80d4ea3a75472e1
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Thu Apr 22 13:56:36 2021 -0400

    Added code to incorporate the mod function

commit 8887e0ffcc6fccbd32592c353a93f104d827f5d8
Merge: 2e02952 e60dcaa
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Wed Apr 21 15:50:31 2021 -0400

    Merge pull request #11 from kenyaplenty/assignops
```

```
    Changed the for loop test

commit e60dcaa2c444dfb75a69dce571f9baa393a3f09f
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Wed Apr 21 15:50:05 2021 -0400

    Changed the for loop test

commit 2e029527efa00f72da7fd34bd9cc092b6aa7118d
Merge: 3cfcdd2 1559f54
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Wed Apr 21 15:46:40 2021 -0400

    Merge pull request #10 from kenyaplenty/assignops

    Added code for ++ and --

commit 1559f5425e776ff6410c94be8c898d4f1e272e9c
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Wed Apr 21 15:46:04 2021 -0400

    TAdded code for ++ and --

commit 3cfcdd2702ffd8ea9970f89d57f667123454da6c
Merge: 33de083 9e1c62f
Author: Pazit Schrecker <prs2143@barnard.edu>
Date:   Wed Apr 21 14:37:16 2021 -0400

    Merge pull request #9 from kenyaplenty/style-fix

    Removed use of fst and snd for better style in semant

commit 9e1c62f5a574cba33a9e12c048f56bff5cda21aa
Author: Pazit <pazitrose1999@gmail.com>
Date:   Wed Apr 21 14:35:06 2021 -0400

    Removed use of fst and snd for better style

commit 33de0834f0b96d8969b27b753f7e5810fef19f3c
Merge: 25e6a3f f07752e
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Wed Apr 21 11:17:44 2021 -0400

    Merge pull request #8 from kenyaplenty/assignops
```

```
    Added the code to make assignment operators work

commit f07752e1e395055b34bcf1ebf8629a40ef00087d
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Wed Apr 21 11:16:59 2021 -0400

    Added the code neccessary to make assignment operators to work

commit 25e6a3f7351bc629c2910bb3bfdea477c10bf74d
Merge: 1b7b400 af6c7b4
Author: Pazit Schrecker <prs2143@barnard.edu>
Date:   Wed Apr 21 09:31:04 2021 -0400

    Merge pull request #7 from kenyaplenty/new-return

    Control Flow branch with if, while, for, return

commit af6c7b4c0e557dd26967a5159ea12b2442a05ec7
Author: Katrina <kz2335@columbia.edu>
Date:   Tue Apr 20 22:19:48 2021 -0400

    working return stmt with control flow

commit 01ca9610bbd3508b484d339f5559c4284b4b6a2c
Author: Katrina <kz2335@columbia.edu>
Date:   Tue Apr 20 21:31:08 2021 -0400

    added while and for loops

commit a23e919e1630c530f448252129aec136f19d3eac
Author: Lindsey Wales <43915800+lindseywales5@users.noreply.github.com>
Date:   Tue Apr 20 20:27:18 2021 -0400

    "If" is working

commit 1b7b4009b45831b1e183934a1271e3489696715f
Merge: 0c88134 ae0a545
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Tue Apr 20 17:43:30 2021 -0400

    Merge pull request #6 from kenyaplenty/locals

    Locals

commit ae0a545567d579a9b3f5aef8e42380351461f182
```

```
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Tue Apr 20 17:42:58 2021 -0400

    restored codegen

commit 1b0aa77367cd3774c20c37b785284fd4d21f4881
Merge: 90aea82 0c88134
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Tue Apr 20 17:32:51 2021 -0400

    Merge branch 'main' into locals

commit 90aea8295d06d7d3092dc5222d182e893a622c60
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Tue Apr 20 17:18:13 2021 -0400

    Fixed everything so local variable declaration and initialization
    are allowed throughout the function body

commit 0c8813490310aef5515396b899b147eedd16b97f
Merge: 9ec0ee6 9442d6b
Author: Pazit Schrecker <prs2143@barnard.edu>
Date:   Mon Apr 19 22:34:20 2021 -0400

    Merge pull request #5 from kenyaplenty/vars

    fixed unmatched case and parser shadowing errors

commit 9442d6b2df9cbc4a8fcebf94b5ca62bacb3af315
Author: Pazit <pazitrose1999@gmail.com>
Date:   Mon Apr 19 22:33:45 2021 -0400

    fixed unmatched case and parser shadowing errors

commit f0f0cd6d019cef7ab4b5f49f46cebe2adf20f6a8
Author: Pazit <pazitrose1999@gmail.com>
Date:   Mon Apr 19 21:12:51 2021 -0400

    Fixed parser shadow errors

commit f46ddc6089ac2e46d12e16fbb329a49ded98f0ce
Author: Katrina Zhao <48104668+kzhao18@users.noreply.github.com>
Date:   Mon Apr 19 20:44:26 2021 -0400

    got rid of pattern matching warnings
```

```
commit 16ea438c3b1247e87a75591b9ca59db5fe1a6d10
Author: Katrina Zhao <48104668+kzhao18@users.noreply.github.com>
Date:   Mon Apr 19 20:43:11 2021 -0400

    got rid of pattern matching warnings

commit f07e663f07c870f751125e17b68dbc79cabfa3e9
Author: Pazit <pazitrose1999@gmail.com>
Date:   Mon Apr 19 19:11:28 2021 -0400

    Updated m_locals

commit 96c8f07ba57c534433af81fa6fa279e0bd2337f1
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sun Apr 18 18:34:02 2021 -0400

    added local variable test

commit 07a0563934056aeeb6600610b735f74ad135b1c3
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sun Apr 18 18:33:47 2021 -0400

    Updated codegen (map still not updating correctly

commit b381d825d407d58098ed60f9eb360641a10c1d9e
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sun Apr 18 17:20:08 2021 -0400

    Added recursive map to store variables in codegen

commit 38e935cdfb8a8fe47e98e8bc62db87abe6a2716f
Author: Pazit <pazitrose1999@gmail.com>
Date:   Tue Apr 13 18:10:36 2021 -0400

    Added map as an argument in semant

commit 4e47ec0210685bf72ec8e573d714cb9fdc945a08
Merge: fcc696b 6bfd56b
Author: Pazit Schrecker <prs2143@barnard.edu>
Date:   Tue Apr 13 11:34:54 2021 -0400

    Merge pull request #4 from kenyaplenty/xl/locals

    fixed locals (Xijiao)
```

```
commit 6bfd56b7e3d9649bb63e24be63d48b15d56afefc
Author: Xijiao Li <xl2950@columbia.edu>
Date:   Tue Apr 13 03:33:01 2021 -0400

    fixed locals

commit fcc696bd02c5b160c57f146a9d52265bc73d260c
Author: Pazit <pazitrose1999@gmail.com>
Date:   Mon Apr 12 19:10:04 2021 -0400

    Fixed add_local function call

commit 4afb4ac41ddd4c11808ce04346f748198e269502
Author: Pazit <pazitrose1999@gmail.com>
Date:   Mon Apr 12 19:04:28 2021 -0400

    Fixed add_local function call

commit a823f579ab0be57fc46ccd058d4d598c5bb6ca6d
Author: Pazit <pazitrose1999@gmail.com>
Date:   Mon Apr 12 15:39:45 2021 -0400

    Updated ways of adding local variables

commit 4c547cac7b54e0fa7e5978e528fa9cee0dec5bc3
Author: Pazit <pazitrose1999@gmail.com>
Date:   Mon Apr 12 11:36:46 2021 -0400

    Added assign, added basis for declare + assign

commit 9ec0ee6cb15cc859619bda04d18b2671a83ce5f3
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Apr 11 15:04:18 2021 -0400

    Added the print char test

commit 1a7756cc01e7bb401879d653de4021f1d886539d
Merge: 942133d 0f13c3f
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Apr 11 15:03:03 2021 -0400

    Merge pull request #3 from kenyaplenty/chars

    Added code to support characters in our language
```

```
commit 0f13c3fdffc3ae1bc7067d427685598bcc533843
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Apr 11 15:02:20 2021 -0400

    Added code to support characters in our language

commit 942133dc74d4d975d9862ca03d50384dcaf8a346
Author: Lindsey Wales <43915800+lindseywales5@users.noreply.github.com>
Date:   Sun Apr 11 13:55:57 2021 -0400

    Added unops, boolean functionality, and the ability to evaluate
    boolean expressions in print statements

commit b546f19647e672e4cb48acf18eb0b5557b4fce2c
Author: Lindsey Wales <43915800+lindseywales5@users.noreply.github.com>
Date:   Sun Apr 11 13:17:51 2021 -0400

    Added Binops (with both ints and floats)

commit e0201aa2a1d15bcb9f4bacade85c3f9d5b1b6ba8
Merge: 9dd8de3 f82add9
Author: Pazit Schrecker <prs2143@barnard.edu>
Date:   Fri Apr 9 18:21:58 2021 -0400

    Merge pull request #2 from kenyaplenty/primitives

    Merging print function that works for ints, floats, and strings

commit f82add9f28367348862b0fc3e7fdc1acb95c7d32
Author: Pazit <pazitrose1999@gmail.com>
Date:   Fri Apr 9 18:18:51 2021 -0400

    Added printint, printfloat tests

commit 0d3efdb3af92e652abd91bdb4e06c4785f0dd9f5
Author: Pazit <pazitrose1999@gmail.com>
Date:   Fri Apr 9 18:18:14 2021 -0400

    Added print function that works for ints, floats, and strings

commit e6f82d3d613102566c82731a8b8bb391088f7fa2
Author: Pazit <pazitrose1999@gmail.com>
Date:   Thu Apr 8 18:59:43 2021 -0400
```

```
    Added separate print function for ints (not fully working0
    )

commit 0605f020e37b30e37379937803f84f3d2a2d1e80
Author: Pazit <pazitrose1999@gmail.com>
Date:   Thu Apr 8 17:38:52 2021 -0400

    updated semant

commit 254cf240448ffc02543ac6bfeeccd47493955078
Author: Pazit <pazitrose1999@gmail.com>
Date:   Tue Apr 6 12:54:03 2021 -0400

    Trying to fix printing

commit 379e0247bbe4ff5e507e11092bb0993e46f63762
Author: Pazit <pazitrose1999@gmail.com>
Date:   Mon Apr 5 16:38:20 2021 -0400

    Added functionality to print ints

commit a25f7e32416cb2a83f6eba6a8876d6c2fe29044c
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sat Apr 3 12:18:39 2021 -0400

    trying to add printing ints directly

commit 2c9d065cd7a17c45764a774e98ebbc7f81257cc8
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sat Apr 3 11:07:08 2021 -0400

    Added float primitive type

commit 2945ddd2c4e02bca6d1dec3ba5ab7b06d7f48c09
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sat Apr 3 11:03:50 2021 -0400

    Added boolean primitive type

commit 4490451aba1566b352a7c1d39e285e1b539c2fec
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sat Apr 3 11:00:36 2021 -0400

    Added integer primitive type
```

```
commit 9dd8de3d49044fa38ae42053bdc989f73337d754
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Fri Apr 2 16:30:45 2021 -0400

    Updated hello world output file to have the proper output

commit b2df74d1fd661ba152344a4467c20bd9c4c3a2e9
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Fri Apr 2 16:15:37 2021 -0400

    Updated the _tags file to have newlines at the end

commit 1b8f973c2015109768b5bd5740f76b76d6997b1c
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Fri Apr 2 16:14:07 2021 -0400

    Removed old helloworld file

commit dbaf170c498b8ccb2943400a7839f4a7d27859fb
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Fri Apr 2 16:13:14 2021 -0400

    Updated the README to have the correct instructions
    for running hello world

commit 7acdd1df67ce2b344ef803225ad09f172e6ced1a
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Fri Apr 2 16:09:17 2021 -0400

    Created preliminary testing script

commit 45277a6d20ad2d2d9d4b486d892f2a9499556935
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Fri Apr 2 16:08:44 2021 -0400

    Created test folder with hello world  test and expected output

commit dc2cef6b27053e5d2b278022a4d8a95fd70167ca
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Fri Apr 2 16:07:44 2021 -0400

    Updated the Makefile to automatically run the testing script

commit 4faac5569bdc386a61a37e3a59808d66360858d3
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
```

```
Date:   Fri Apr 2 16:07:03 2021 -0400

    Updated funcyjava.ml to get it to take in input from files

commit d297f1309d07101ac6aa01e58fad1a3ab28e0e24
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Fri Apr 2 14:04:14 2021 -0400

    Created correct Makefile with tabs

commit b1a7a800afd632fc2ee124ed683bd3ef30cdab72
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Fri Apr 2 14:03:05 2021 -0400

    Removed misnamed MakeFile

commit 704d5efa579e647743cbdfae4caa19477ae659d2
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Fri Apr 2 14:00:26 2021 -0400

    Updated README to contain Makefile instructions

commit e3b40a303dbb0a77f5491c6b9e9b19d5827441dd
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Wed Mar 31 14:35:42 2021 -0400

    Added the tags file for code gen to work

commit 96053bf98f74e5fa2776b40501202e0a6d669486
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Wed Mar 31 14:30:43 2021 -0400

    Finished complete codegen for hello world and updated README
    with instructions on how to run hello world

commit 4e79a03d4dfc790c193d46cd612dad970ebac56f
Merge: 347a2c8 7e489c5
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Wed Mar 24 22:51:14 2021 -0400

    Merge branch 'main' of https://github.com/kenyaplenty/FUNC-y-Java into main

commit 347a2c81121b724f8884d46f555d936bbf2305e6
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Wed Mar 24 22:50:26 2021 -0400
```

```
    Created semantic checker for hello world

commit 7e489c5bced43d7057d9db4da9fc7612e7a9550d
Author: Lindsey Wales <43915800+lindseywales5@users.noreply.github.com>
Date:   Wed Mar 24 22:05:19 2021 -0400

    Veryyy beginnings of code gen

    This is 90\% MicroC but I think the strings could work

commit 459670c7a626a5a6e9b9bd02e86c4cece063e7c6
Author: Katrina Zhao <48104668+kzhao18@users.noreply.github.com>
Date:   Wed Mar 24 21:00:42 2021 -0400

    updated with semantic checking

commit bcfe4cca3782fa8cdba3551c00d91b3018cafef2
Author: Katrina Zhao <48104668+kzhao18@users.noreply.github.com>
Date:   Wed Mar 24 20:59:47 2021 -0400

    updated sast with pretty print

commit dc5acda9970b7c4b71cd370f73e4d557eee0c003
Author: Katrina Zhao <48104668+kzhao18@users.noreply.github.com>
Date:   Wed Mar 24 20:59:05 2021 -0400

    semantic checker first draft

commit fddb8670a4f9745dc9f6fe141bbc72bc9060659d
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Wed Mar 24 14:09:06 2021 -0400

    Corrected the function parameter order in parser

commit 32dda5336b9d53aac8f2ab79b969f2d89f7d41f6
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Wed Mar 24 14:07:19 2021 -0400

    Removed the extra double quotation around strings

commit 5787588fbd29616cd7715e571a44b7543675a14d
Author: Pazit <pazitrose1999@gmail.com>
Date:   Wed Mar 24 13:38:34 2021 -0400
```

```
    Fixed parser to allow for different function declarations
    with varying parameters

commit e7f7289b87e8a1e821cb1a042b7a5218115dff73
Author: Pazit <pazitrose1999@gmail.com>
Date:   Wed Mar 24 13:38:05 2021 -0400

    Fixed variable names to allow for single characters.

commit 9388088c3261100c4306337c18b434b348e04308
Merge: 3f2ef53 d6465e8
Author: Pazit Schrecker <prs2143@barnard.edu>
Date:   Wed Mar 24 11:46:36 2021 -0400

    Merge pull request #1 from kenyaplenty/xl/parsing-fix

    Parsing error fixed

commit d6465e8e193d67bb6013719079be5ac99f47ca34
Author: Xijiao Li <xl2950@columbia.edu>
Date:   Wed Mar 24 11:08:43 2021 -0400

    fix parsing error

commit 3f2ef539a34dc8b74cc5ef5005b5d20c1a2a6d0f
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Tue Mar 23 18:06:21 2021 -0400

    Added files to start putting together our compiler

commit 6830c8bf2a1e10872fa3a98d405518fb7be05368
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Tue Mar 23 13:32:53 2021 -0400

    changed wording in README.md

commit 74beb1f2bf5215e91644f5a79447bae1e6577268
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Tue Mar 23 13:32:31 2021 -0400

    Added instructions for compiling the ast, scanner, parser together

commit e501afeaff3f9c9778128de585d9b0944b3a7e96
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Tue Mar 23 12:28:51 2021 -0400
```

```
    Changed the hello_world file to mimic what is in the Makefile

commit 59c8998dfacaeb59f6a7cda01fb08e521ca61156
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Tue Mar 23 11:51:48 2021 -0400

    Filled out hello world

commit badec660e40c70edf2e8b9faaafb1af7aca46378
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Tue Mar 23 11:46:38 2021 -0400

    Created the helloworld file

commit be288b392cb07c77df424889619db19593913f69
Author: Pazit <pazitrose1999@gmail.com>
Date:   Mon Mar 22 10:49:49 2021 -0400

    Added Makefile -- WIP, waiting on other files to add more

commit db98919eac37d5b55854e0d791965d8a64053502
Author: Pazit <pazitrose1999@gmail.com>
Date:   Mon Mar 22 10:46:31 2021 -0400

    Removed elif from sast to comply with updated ast

commit 8a02dc20af8e43a5bfcf77865c76d98aeaa6e590
Merge: 580b81b bd37c84
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Mon Mar 22 10:41:05 2021 -0400

    Merge branch 'main' of https://github.com/kenyaplenty/FUNC-y-Java into main

commit 580b81bdae22d96cdf4b1a4933b6623f17bb5f84
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Mon Mar 22 10:40:56 2021 -0400

    Changed the way that we do elif statements

commit bd37c84c8bbe3f1a5dd0f898effd7b6e3bc22569
Author: Pazit <pazitrose1999@gmail.com>
Date:   Mon Mar 22 10:40:29 2021 -0400

    Added sast.ml based on updated AST)
```

```
commit 89a323bd36bfbd85c3589fa7b92294ac0ae983d1
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Mon Mar 22 10:22:08 2021 -0400

    Added the elif token to the scanner

commit 50e0e819a7558ae8ec7887900db4acd9e92acb0a
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Mon Mar 22 10:21:19 2021 -0400

    Changed the variable name associated with the character pattern

commit 3bb760b453428ad34c8103b9e24ec08fb5ec6bad
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Mon Mar 22 10:20:31 2021 -0400

    Added code to set up elif statements and defined
    the char in the scanner

commit 8265810c5051156a0f572a193798d6428c6c373a
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Mar 21 19:12:34 2021 -0400

    Renamed to parser

commit 5ac674fd7e9ce2482744bfd0690180013ddf0564
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Mar 21 19:12:13 2021 -0400

    Renamed to ast.ml

commit 561dd6b14bb4b1f21c0bbeda3525243fbda01669
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Mar 21 19:11:53 2021 -0400

    Delete ast

commit 4c439c609ca8f059a5d46b74a1888122078e969c
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Mar 21 19:11:28 2021 -0400

    removed all parser

commit cf24501c2101bd51ef22032381ceb24eb6f5e9c9
```

```
Merge: 08729cf e8bc46b
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Mar 21 19:09:41 2021 -0400

    Merge branch 'main' of https://github.com/kenyaplenty/FUNC-y-Java into main

commit e8bc46b14c67502eab68e1a9f49ee3ddd6d24b19
Author: Pazit Schrecker <pazitrose1999@gmail.com>
Date:   Sun Mar 21 19:09:14 2021 -0400

    Delete sast.ml

commit 08729cf86622a5b49e105da40ce94689ca7c37d6
Merge: b5fac19 e12c2cb
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Mar 21 19:09:07 2021 -0400

    Merge branch 'main' of https://github.com/kenyaplenty/FUNC-y-Java into main

commit b5fac19be72ee5ada235b4c88b1f7b621e2de490
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Mar 21 19:08:45 2021 -0400

    Latest changes

commit e12c2cbe17f54ef83ead9129f4b27179f9ab42a5
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sun Mar 21 18:59:36 2021 -0400

    Added sast first draft

commit 50863f939990cf78634ed17c4251fce1e8e69972
Author: Pazit Schrecker <pazitrose1999@gmail.com>
Date:   Sun Mar 21 18:47:19 2021 -0400

    Deleted  sast.ml

commit 4f6cb9983499de28e658df48d11b13a37f7ed1c5
Author: Pazit <pazitrose1999@gmail.com>
Date:   Sun Mar 21 18:23:50 2021 -0400

    Added sast first draft

commit 33af0f96489bc45e7723fefc202e310f2a2afd76
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
```

```
Date:   Sun Mar 21 15:17:19 2021 -0400

    Changed parser to reflect  how floats are represented

commit 458ccbb1b1c3db9ed6b9da92269c757f9dedad60
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Mar 21 14:55:17 2021 -0400

    New ast with almost everything

commit 3b8b4bb6f5d0d0313093b48561745bf0ed4d5f8d
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Mar 21 14:54:01 2021 -0400

    new parser without elif and foreach

commit a84cd85144ca68962118f7355b7508bcc665fba1
Merge: bf00214 c52477e
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Mar 21 14:43:09 2021 -0400

    Merge branch 'main' of https://github.com/kenyaplenty/FUNC-y-Java into main

commit c52477e1b947d6a0f8c937bbbcaa28e41389d960
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Mar 21 14:42:41 2021 -0400

    Created parser2.mly

commit bf00214015d1eb8eafac736c7cec3312a68834a7
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sun Mar 21 14:41:29 2021 -0400

    Most of the new parser. Trying to add in foreach and the if statements

commit e11100e3c087ccb73b32f1318aab68643314c06f
Author: lindseywales5 <43915800+lindseywales5@users.noreply.github.com>
Date:   Fri Mar 19 10:00:50 2021 -0400

    Update ast.ml

commit 4ff08a3cbee097b71fef30552edcf70101dbcdb2
Author: Katrina Zhao <48104668+kzhao18@users.noreply.github.com>
Date:   Thu Mar 18 17:44:57 2021 -0400
```

```
    made lt, gt, leq, geq nonassoc

commit 05a7b7d8ffe3fd17f7c08120e78b28cb06eeaaf7
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Thu Mar 18 17:14:53 2021 -0400

    Updated some of the formatting for the parser

commit 70d342016b486cd577acfd785a16d1499b92525c
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Thu Mar 18 17:00:50 2021 -0400

    Added cry to the ast

commit d21cf22eabffec669c6782576805c932ea2702cd
Author: Liseidy Bueno <58915720+liseidybueno@users.noreply.github.com>
Date:   Thu Mar 18 08:56:08 2021 -0400

    Update parser.mly

    deleted: objects, var.var; added: foreach,
    var in expr, comments/questions;
    edited: declarations of collections

commit 5ac3c7f25b27b54946fcc9662b90aa08e3aea89d
Author: Liseidy Bueno <58915720+liseidybueno@users.noreply.github.com>
Date:   Thu Mar 18 08:52:32 2021 -0400

    Update ast.ml

    added comment about floats and added foreach

commit a87fada11ebdd2750d163a536e4fe57855d3402c
Author: Liseidy Bueno <58915720+liseidybueno@users.noreply.github.com>
Date:   Thu Mar 18 08:15:24 2021 -0400

    Update scanner.mll

    replaced COLON with SEMI for ; and added : and COLON

commit de5933bf60d415d3b322c6a6ae90e4c560e6bbdf
Author: lindseywales5 <43915800+lindseywales5@users.noreply.github.com>
Date:   Thu Mar 18 07:46:34 2021 -0400

    Update ast.ml
```

```
commit beefb73621ca03ee7511d602b2c3d7331186ee91
Author: lindseywales5 <43915800+lindseywales5@users.noreply.github.com>
Date:   Wed Mar 17 21:43:20 2021 -0400

    Update parser.mly

commit cd5d53e67a30066fd6a223cf56ff33d7fed53ac0
Author: lindseywales5 <43915800+lindseywales5@users.noreply.github.com>
Date:   Wed Mar 17 21:32:51 2021 -0400

    Update parser.mly

commit a59b3e63a7c7bf8a51f6e1cbb3ff1b05920f4173
Author: lindseywales5 <43915800+lindseywales5@users.noreply.github.com>
Date:   Wed Mar 17 21:32:23 2021 -0400

    Update parser.mly

commit 736dd65d4efb3b80810b638368dcd35c245366d7
Author: lindseywales5 <43915800+lindseywales5@users.noreply.github.com>
Date:   Wed Mar 17 20:39:54 2021 -0400

    Initial AST

commit 60996c728580e75c1907001bec5e5b5e84db323e
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Tue Mar 16 21:32:24 2021 -0400

    Started updating the parser and scanner

commit e8753285bcde10d21711e415266461cf68b155c3
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Tue Mar 16 20:24:07 2021 -0400

    Created Scanner.mll

commit 26a423f9075943a08c3756465597cdf2ed985fc8
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Tue Mar 16 20:23:25 2021 -0400

    created parser.mly

    Last full version of the parser
```

```
commit e71f2f503a05cedbfad3b3234c3618926a47b5e1
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Tue Mar 16 20:15:38 2021 -0400

    Updated Readme.md

commit 158c10d64980ddd90ba8a5636e8ee260dab9be87
Author: Kenya <51925141+kenyaplenty@users.noreply.github.com>
Date:   Sat Feb 20 13:51:42 2021 -0500

    Initial commit
```

7.2 Codebase

7.2.1 scanner.mll

```
{open Parser}

let digit = ['0'-'9']
let letter = ['a'-'z']
let uletter = ['A' - 'Z']
let ascii = ([' '-'!' '#'-'[' ']'-'~'] | digit)

rule token=
  parse [' ' '\t' '\r' '\n']+        { token lexbuf}
      | eof                          { EOF }
      | "char"                       { CHAR }
      | "int"                        { INT }
      | "str"                        { STRING }
      | "bool"                       { BOOL }
      | "float"                      { FLOAT }
      | "func"                       { FUNC }
      | "array"                      { ARRAY }
      | "for"                        { FOR }
      | "while"                      { WHILE }
      | "if"                         { IF }
      | "else"                       { ELSE }
      | "elif"                       { ELIF }
      | "true"                       { BLIT(true) }
      | "false"                      { BLIT(false) }
      | "return"                     { RETURN }
      | "cry"                        { CRY }
      | "foreach"                    { FOREACH }
      | "in"                         { IN }
      | "and"                        { AND }
      | "or"                         { OR }
      | "not"                        { NOT }
      | '+'                          { PLUS }
      | '-'                          { MINUS }
      | '*'                          { TIMES }
      | '/'                          { DIVIDE }
```

```
      | '='                           { ASSN }
      | "%"                           { MOD }
      | "+="                          { PASSN }
      | "-="                          { MASSN }
      | "/="                          { DASSN }
      | "*="                          { TASSN }
      | "%="                          { MOASSN }
      | "++"                          { PPLUS}
      | "--"                          { MMINUS}
      | digit+ as num                 { NUM(int_of_string num) }
      | digit+ '.' digit+ as frac     { DEC(float_of_string frac) }
      | letter(letter| uletter| '_')* as id    { VAR(id) }
      | ';'                           { SEMI }
      | ':'                           { COLON }
      | ','                           { COMMA }
      | '('                           { LPAREN }
      | ')'                           { RPAREN }
      | '{'                           { LBRACK }
      | '}'                           { RBRACK }
      | '['                           { LSQ }
      | ']'                           { RSQ }
      | '.'                           { DOT }
      | "=="                          { EQ }
      | "!="                          { NEQ }
      | "<"                           { LT }
      | ">"                           { GT }
      | "<="                          { LEQ }
      | ">="                          { GEQ }
      | '"'([^'"']*)'"' as lxm        { SLIT(lxm) }
      | '''(ascii)''' as char         { CHARA(String.get char 1)}
      |  "/*"                         { comment lexbuf }
      | _ as char                   { raise (Failure("illegal character " ^
    Char.escaped char))}
and comment =
    parse "*/"                        { token lexbuf }
        | _                           { comment lexbuf }


(* Add string and char into the scanner *)
```

7.2.2 parser.mly

```
%{open Ast %}

%token BOOL INT FLOAT CHAR STRING FUNC ARRAY CRY
%token LBRACK RBRACK LSQ RSQ LPAREN RPAREN COLON SEMI COMMA DOT
%token IF ELIF ELSE WHILE FOR FOREACH IN AND OR RETURN NOT
%token GT LT LEQ GEQ EQ NEQ
%token PLUS MINUS TIMES DIVIDE ASSN MOD PASSN MASSN DASSN TASSN MOASSN
   PPLUS MMINUS
%token <int> NUM
%token<float> DEC
%token<char> CHARA
%token<bool> BLIT
```

```
%token<string> VAR SLIT
%token EOF

%start program
%type<Ast.program> program

%nonassoc NOELSE
%nonassoc ELSE
%nonassoc ELIF
%right ASSN PASSN MASSN DASSN TASSN MOASSN
%left OR
%left AND
%left EQ NEQ
%left LT GT LEQ GEQ
%left PLUS MINUS
%left TIMES DIVIDE MOD
%right PPLUS MINNUS
%right NOT

%%
program:
    decls EOF { $1 }

decls:
   /*nothing*/ { []}
   | decls fdecl { $2 :: $1 }

fdecl:
    FUNC typ VAR LPAREN RPAREN LBRACK stmt_list RBRACK
        {{typ = $2; fname = $3; formals = []; body = List.rev $7;}}
    | FUNC typ VAR LPAREN RPAREN LBRACK RBRACK
        {{typ = $2; fname = $3; formals = []; body = [];}}
    | FUNC typ VAR LPAREN formal_list RPAREN LBRACK stmt_list RBRACK
        {{typ = $2;
        fname = $3;
        formals = List.rev $5;
        body = List.rev $8;}}

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

expr_list:
    /*nothing*/ { [] }
    | expr      { [$1] }
    | expr_list COMMA expr { $3::$1 }

typ:
      INT { Int }
    | FLOAT { Float }
    | CHAR { Char }
    | STRING { String }
    | BOOL { Bool }
    | CRY { Cry }
```

```
66
67
68 primitive_typ:
69     INT {Int}
70     | FLOAT {Float}
71     | CHAR { Char }
72     | STRING { String }
73     | BOOL { Bool }
74
75 stmt_list:
76     /*nothing {[]}*/
77     stmt {[$1]}
78     | stmt_list stmt {$2::$1}
79
80 stmt:
81       expr SEMI { Expr($1) }
82     | RETURN expr SEMI         {Return($2)}
83     | LBRACK stmt_list RBRACK {Block(List.rev $2)}
84     | FOR LPAREN expr SEMI expr SEMI expr RPAREN stmt {For($3, $5, $7, $9
       )}
85     | WHILE LPAREN expr RPAREN stmt {While($3, $5)}
86     | IF LPAREN expr RPAREN stmt ELSE stmt {If($3, $5, $7)}
87     | IF LPAREN expr RPAREN stmt %prec NOELSE { If($3, $5, Block([]))}
88     | IF LPAREN expr RPAREN stmt elif_else_stmt { If($3, $5, $6)}
89     | primitive_typ VAR SEMI { Vdecl($1, $2) }
90     | primitive_typ VAR ASSN expr SEMI { VdeclAssign($1, $2, $4) }
91     | ARRAY LT INT COMMA NUM GT VAR ASSN LSQ expr_list RSQ SEMI {
       VdeclAssign(IntList($5), $7, IntListLit($10))}
92     | ARRAY LT FLOAT COMMA NUM GT VAR ASSN LSQ expr_list RSQ SEMI {
       VdeclAssign(FloatList($5), $7, FloatListLit($10))}
93     | ARRAY LT STRING COMMA NUM GT VAR ASSN LSQ expr_list RSQ SEMI {
       VdeclAssign(StringList($5), $7, StringListLit($10))}
94
95
96 elif_else_stmt:
97     ELIF LPAREN expr RPAREN stmt elif_else_stmt   {If($3, $5, $6)}
98     | ELIF LPAREN expr RPAREN stmt ELSE stmt      {If($3, $5, $7)}
99
100
101
102 expr:
103       NUM   { Num($1) }
104     | VAR   { Var($1) }
105     | CHARA { Chara($1) }
106     | BLIT  { Blit($1) }
107     | DEC   { Dec($1) }
108     | SLIT  { Slit($1) }
109     | LPAREN expr RPAREN { $2 }
110     | VAR LSQ expr RSQ {ArrayAccess($1, $3)}
111     | expr PLUS expr { Binop($1, Add, $3) }
112     | expr MINUS expr { Binop($1, Sub, $3) }
113     | expr TIMES expr { Binop($1, Mult, $3) }
114     | expr DIVIDE expr { Binop($1, Div, $3) }
115     | expr MOD expr    { Binop($1, Mod, $3) }
```

```
    | expr EQ expr     { Binop($1, Equal, $3)}
    | expr NEQ expr    {Binop($1, Neq, $3) }
    | expr LT expr   { Binop($1, Less, $3) }
    | expr LEQ expr {Binop($1, Leq, $3) }
    | expr GT expr   {Binop($1, Greater, $3) }
    | expr GEQ expr  {Binop($1, Geq, $3) }
    | expr AND expr  {Binop($1, And, $3) }
    | expr OR expr  {Binop($1, Or, $3) }
    | MINUS expr %prec NOT { Unop(Neg, $2)}
    | NOT expr { Unop(Not, $2)}
    | VAR PPLUS {Assign($1, (Binop(Var($1), Add, Num(1))))}
    | VAR MMINUS {Assign($1, (Binop(Var($1), Sub, Num(1))))}
    | VAR ASSN expr { Assign($1, $3)}
    | VAR PASSN expr {Assign($1, (Binop(Var($1), Add, $3)))}
    | VAR MASSN expr { Assign($1, (Binop(Var($1), Sub, $3)))}
    | VAR TASSN expr { Assign($1, (Binop(Var($1), Mult, $3)))}
    | VAR DASSN expr  { Assign($1, (Binop(Var($1), Div, $3)))}
    | VAR MOASSN expr {Assign($1, (Binop(Var($1), Mod, $3)))}
    | VAR LPAREN args_opt RPAREN { Call($1, $3)}



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

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

7.2.3 ast.ml

```
type op = Add | Sub | Mult | Div | Mod | Equal | Neq | Less | Leq |
   Greater | Geq | And | Or
type uop = Neg | Not
type up = Plus | Minus
type typ = Int | Float | Bool | Cry | String | Char | IntList of int |
   FloatList of int | StringList of int
type bind = typ * string

type expr =
    Num of int
  | Blit of bool
  | Var of string
  | Chara of char
  | Dec of float
  | Slit of string
  | Binop of expr * op * expr
  | Unop of uop * expr
  | Unp of expr * up
  | Assign of string * expr
  | Call of string * expr list
  | IntListLit of expr list
  | FloatListLit of expr list
```

```
  | StringListLit of expr list
  | ArrayAccess of string * expr


type stmt =
   Block of stmt list
  | Expr of expr
  | Return of expr
  | If of expr * stmt * stmt
  | For of expr * expr * expr * stmt
  | While of expr * stmt
  | Vdecl of typ * string
  | VdeclAssign of typ * string * expr

type func_decl = {
  typ: typ;
  fname: string;
  formals: bind list;
  body: stmt list;
}

type program =  func_decl list
let string_of_op = function
  Add -> "+"
  | Sub -> "-"
  | Mult -> "*"
  | Div -> "/"
  | Equal -> "=="
  | Neq -> "!="
  | Less -> "<"
  | Leq -> "<="
  | Greater -> ">"
  | Geq -> ">="
  | And -> "and"
  | Or -> "or"
  | Mod -> "%"

  let string_of_uop = function
  Neg -> "-"
| Not -> "not"

let rec string_of_expr = function
  Call(f, el) -> f ^ "(" ^ String.concat ", " (List.map string_of_expr el)
    ^ ");\n"
  | Slit l -> l
  | Blit(true) -> "true"
  | Blit(false) ->  "false"
  | Num n -> string_of_int n
  | Dec d -> string_of_float d
  | Chara c -> String.make 1 c
  | Var v -> v
  | Assign(v, e) -> v ^ " = " ^ string_of_expr e
  | Unop(uop, e) -> string_of_uop uop ^ " " ^ string_of_expr e
  | Binop(v, op, e) -> string_of_expr v ^ " " ^ string_of_op op ^ " " ^
```

```
    string_of_expr e
  | IntListLit(elements) -> "[" ^ String.concat ", " (List.map
    string_of_expr elements) ^ "]"
  | FloatListLit(elements) -> "[" ^ String.concat ", " (List.map
    string_of_expr elements) ^ "]"
  | StringListLit(elements) -> "[" ^ String.concat ", " (List.map
    string_of_expr elements) ^ "]"
  | ArrayAccess(n, index) -> n ^ "[" ^ string_of_expr index ^ "]"
  | _ -> ""

let string_of_typ = function
  Cry -> "cry"
  | Int -> "int"
  | Float -> "float"
  | Bool -> "cry"
  | String -> "string"
  | Char -> "char"
  | IntList x -> "array<int, " ^ (string_of_int x) ^ ">"
  | FloatList x -> "array<float, " ^ (string_of_int x) ^ ">"
  | StringList x -> "array<str, " ^ (string_of_int x) ^ ">"

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

let string_of_func_decl func_decl =
    string_of_typ func_decl.typ ^ " " ^
    func_decl.fname ^ "(" ^ String.concat ", " (List.map snd func_decl.
   formals) ^
    ")\n{\n" ^
    String.concat "" (List.map string_of_stmt func_decl.body) ^
    "}\n"

let string_of_program (funcs) =
  String.concat "\n" (List.map string_of_func_decl funcs)
```

7.2.4 semant.ml

```

open Ast
open Sast
```

```
4
5  module StringMap = Map.Make(String)
6
7
8  let check (functions) =
9
10   let built_in_decls =
11     let add_bind map (name, ty) = StringMap.add name {
12       typ = Cry;
13       fname = name;
14       formals = [(ty, "x")];
15       body = [] } map
16     in List.fold_left add_bind StringMap.empty [(("print", String))]
17   in
18
19   (* Add function names to map and catch duplicates *)
20   let add_func map fd =
21     let built_in_err = "the function " ^ fd.fname ^ " is built-in and may
      not be redefined"
22     and dup_err = "duplicate function: a function with name " ^ fd.fname ^
      " has already been defined"
23     and make_err er = raise (Failure er)
24     and n = fd.fname (* Name of the function *)
25     in match fd with (* check for duplicate functions or redefinition of
     built-in functions *)
26         _ when StringMap.mem n built_in_decls -> make_err built_in_err
27       | _ when StringMap.mem n map -> make_err dup_err
28       | _ ->  StringMap.add n fd map
29   in
30
31   let function_decls = List.fold_left add_func built_in_decls functions
32   (* Add all built in functions to this table, including all 3 types of
      print*)
33   in
34
35   let find_func s =
36     if String.equal s "main" then
37       (try StringMap.find s function_decls
38       with Not_found -> raise (Failure ("missing main function")))
39     else
40       try StringMap.find s function_decls
41       with Not_found -> raise (Failure ("unrecognized function " ^ s))
42   in
43   let _= find_func "main" in
44
45   let check_function m_locals func =
46     let check_assign funcname lvaluet rvaluet err =
47       (* Special case for print function that takes any input *)
48       if funcname = "print" ||
49       lvaluet = rvaluet then lvaluet
50       else raise (Failure err)
51     in
52
53     (* check_assign for var types *)
```

```
    let check_assign_vars lvaluet rvaluet err = match (lvaluet, rvaluet)
  with
     (Int, IntList(_)) -> lvaluet
     | (Float, FloatList(_)) -> lvaluet
     | (String, StringList(_)) -> lvaluet
     | _ -> if lvaluet = rvaluet then lvaluet else raise (Failure err)
    in

    (* Build local vars table for this function *)
    let locals = List.fold_left (fun m (typ, name) ->
      StringMap.add name typ m)
        m_locals func.formals
    in

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

    let rec expr m_locals = function
     Slit l    -> (String, SSlit l)
     | Num i   -> (Int, SNum i)
     | Dec d   -> (Float, SDec d)
     | Chara c  -> (Char, SChara c)
     | Blit b  -> (Bool, SBlit b)
     | Var v   -> (type_of_identifier v m_locals, SVar v)
     | IntListLit values ->
      let length = List.length values in
      let l' = List.map (expr m_locals) values in
      (match l' with
        (_,_) :: _ -> (IntList(length), SIntListLit(l'))
        |[] -> (IntList(0), SIntListLit(l')))
     | FloatListLit values ->
      let length = List.length values in
      let l' = List.map (expr m_locals) values in
      (match l' with
        (_, _) :: _ -> (FloatList(length), SFloatListLit(l'))
        | [] -> (FloatList(0), SFloatListLit(l')))
     | StringListLit values ->
        let length = List.length values in
        let l' = List.map (expr m_locals) values in
        (match l' with
          (_, _) :: _ -> (StringList(length), SStringListLit(l'))
          | [] -> (StringList(0), SStringListLit(l')))
     | ArrayAccess(name, index) ->
     let leftside = (type_of_identifier name m_locals)
     in let v = expr m_locals index in (leftside, SArrayAccess(name, v))
     | Unp (_,_) -> raise (Failure ("error: this operator is unsupported")
    )
     | Assign(var, e) as ex ->
       let lt = type_of_identifier var m_locals
       and (rt, e') = expr m_locals e in
       let err = "illegal assignment " ^ string_of_typ lt ^ " = " ^
```

```
106           string_of_typ rt ^ " in " ^ string_of_expr ex
107         in (check_assign_vars lt rt err, SAssign(var, (rt, e')))
108     | Binop(e1, op, e2) as e ->
109      let (t1, e1') = expr m_locals e1
110      and (t2, e2') = expr m_locals e2 in
111      (* All binary operators require operands of the same type *)
112       let same = t1 = t2 in
113      (* Determine expression type based on operator and operand types *)
114       let ty = match op with
115         Add | Sub | Mult | Div | Mod when same && t1 = Int   -> Int
116       | Add | Sub | Mult | Div when same && t1 = Float -> Float
117       | Equal | Neq            when same               -> Bool
118       | Less | Leq | Greater | Geq
119              when same && (t1 = Int || t1 = Float) -> Bool
120       | And | Or when same && t1 = Bool -> Bool
121       | _ -> raise (Failure ("illegal binary operator " ^
122                 string_of_typ t1 ^ " " ^ string_of_op op ^ " " ^
123                 string_of_typ t2 ^ " in " ^ string_of_expr e))
124     in (ty, SBinop((t1, e1'), op, (t2, e2')))
125     | Call(fname, args) as call -> (* Now dealing with every function
    call *)
126       let fd = find_func fname in
127       let param_length = List.length fd.formals in
128       if List.length args != param_length then
129        raise (Failure ("expecting " ^ string_of_int param_length ^
130                        " arguments in " ^ string_of_expr call))
131       else let check_call (ft, _) e =
132         let (et, e') = expr m_locals e in
133         let err = " illegal argument found " ^ string_of_typ et ^
134           " expected " ^ string_of_typ ft ^ " in " ^ string_of_expr e
135         in (check_assign fname ft et err, e')
136       in
137       let args' = List.map2 check_call fd.formals args
138       in
139       (fd.typ, SCall(fname, args'))
140     | Unop(op, e) as ex ->
141       let (t, e') = expr m_locals e in
142       let ty = match op with
143         Neg when t = Int || t = Float -> t
144       | Not when t = Bool -> Bool
145       | _ -> raise (Failure ("illegal unary operator " ^
146                              string_of_uop op ^ " " ^ string_of_typ t ^
147                              " in " ^ string_of_expr ex))
148       in (ty, SUnop(op, (t, e')))
149    in
150    let check_bool_expr m_locals e =
151      let (t', e') = expr m_locals e
152      and err = "expected Boolean expression, got " ^ string_of_expr e ^ "
    instead"
153      in if t' != Bool then raise (Failure err) else (t', e')
154    in
155          (* Return type: tuple --> (StringMap, Sast.sstmt) *)
156    let rec check_stmt m_locals = function
157      Expr e -> (m_locals, SExpr (expr m_locals e))
```

```
158       | If(p, b1, b2) -> let (_, b1') = check_stmt m_locals b1 and
159                           (_, b2') = check_stmt m_locals b2 in
160         (m_locals, SIf(check_bool_expr m_locals p, b1', b2'))
161       | While (p, b1) -> let (_, b1') = check_stmt m_locals b1 in
162                           (m_locals, SWhile(check_bool_expr m_locals p, b1')
    )
163       | For (e1,e2,e3,b1) -> let (_, b1') = check_stmt m_locals b1 in
164                           (m_locals, SFor(expr m_locals e1, check_bool_expr
    m_locals e2,
165                           expr m_locals e3, b1'))
166       | Return e -> let (t, e') = expr m_locals e in
167         if t = func.typ then (m_locals, SReturn (t, e'))
168         else raise ( Failure ("return gives " ^ string_of_typ t ^ "
    expected " ^
169             string_of_typ func.typ ^ " in " ^ string_of_expr e))
170       | Vdecl(t, var) -> (StringMap.add var t m_locals, SVdecl (t, var))
171       | VdeclAssign (t, var, e) as st ->
172         let (rt, e') = expr m_locals e in
173         let err = "illegal assignment " ^ string_of_typ t ^ " = " ^
174             string_of_typ rt ^ " in " ^ string_of_stmt st in
175             let m_locals = StringMap.add var t m_locals in
176             (m_locals, SVdeclAssign((check_assign_vars t rt err), var, (rt
    , e')))
177       | Block sl ->
178         (* Return type: tuple --> (StringMap, Sast.sstmt list) *)
179         let rec check_stmt_list m_locals = function
180             [Return _ as s] -> let (m_locals', st) = check_stmt m_locals s
181                                in (m_locals', [st])
182           | Return _ :: _   -> raise (Failure "nothing may follow a return
    ")
183           | Block sl :: ss  -> check_stmt_list m_locals (sl @ ss) (*
    Flatten blocks *)
184           | s :: ss         -> let (ml, st) = check_stmt m_locals s in (*
    new m_locals, translated instruction, change format*)
185                                let (ml', st') = check_stmt_list ml ss in
186                                let new_list = st :: st' in
187                                (ml', new_list)
188           | []              -> (m_locals, [])
189         in let (m_locals', sl') = check_stmt_list m_locals sl
190         in (m_locals', SBlock(sl'))
191
192     in
193
194     (* body of check_function *)
195     { styp = func.typ;
196         sfname = func.fname;
197         sformals = func.formals;
198         sbody = let (_, st') = check_stmt locals (Block func.body) in
199         match st' with
200         SBlock(sl) -> sl
201         | _ -> raise (Failure ("internal error: block didn't become a
    block?"))
202     }
203   in (List.map (check_function StringMap.empty) functions);
```

7.2.5 sast.ml

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

open Ast

type sexpr = typ * sx
and sx =
    SNum of int
  | SBlit of bool
  | SVar of string
  | SChara of char
  | SDec of float
  | SSlit of string
  | SBinop of sexpr * op * sexpr
  | SUnop of uop * sexpr
  | SAssign of string * sexpr
  | SCall of string * sexpr list
  | SIntListLit of sexpr list
  | SFloatListLit of sexpr list
  | SStringListLit of sexpr list
  | SArrayAccess of string * sexpr

type sstmt =
    SBlock of sstmt list
  | SExpr of sexpr
  | SReturn of sexpr
  | SIf of sexpr * sstmt * sstmt
  | SFor of sexpr * sexpr * sexpr * sstmt
  | SWhile of sexpr * sstmt
  | SVdecl of typ * string
  | SVdeclAssign of typ * string * sexpr

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

type sprogram = sfunc_decl list

(* Pretty-printing functions, used when printing out error messages *)
 let rec string_of_sexpr (t, e) =
  "(" ^ string_of_typ t ^ " : " ^ (match e with
    SSlit(l) -> l
  | SNum(n) -> string_of_int n
  | SDec(d) -> string_of_float d
  | SVar(v) -> v
  | SAssign(v, e) -> v ^ " = " ^ string_of_sexpr e
  | SBinop(e1, op, e2) -> string_of_sexpr e1 ^ " " ^ string_of_op op ^ " "
    ^ string_of_sexpr e2

```

```
  | SCall(f, el) ->
      f ^ "(" ^ String.concat ", " (List.map string_of_sexpr el) ^ ")"
  | SIntListLit(el) -> "[" ^ String.concat ", " (List.map string_of_sexpr
    el) ^ "]"
  | SFloatListLit(el) -> "[" ^ String.concat ", " (List.map
    string_of_sexpr el) ^ "]"
  | SStringListLit(el) -> "[" ^ String.concat ", " (List.map
    string_of_sexpr el) ^ "]"
  | SArrayAccess(name,e) -> name ^ "[" ^ string_of_sexpr e^ "]"
  | _ -> "") ^ ")"

let rec string_of_sstmt = function
    SBlock(stmts) ->
      "{\n" ^ String.concat "" (List.map string_of_sstmt stmts) ^ "}\n"
  | SExpr(expr) -> string_of_sexpr expr ^ ";\n";
  | SVdeclAssign(t, s, e) -> string_of_typ t ^ " " ^ s ^ " = " ^
    string_of_sexpr e ^ ";\n"
  | SVdecl(t, s) -> string_of_typ t ^ " " ^ s ^ ";\n"
  | _ -> ""


let string_of_sfdecl fdecl =
  string_of_typ fdecl.styp ^ " " ^
  fdecl.sfname ^ "(" ^ String.concat ", " (List.map snd fdecl.sformals) ^
  ")\n{\n" ^
  String.concat "" (List.map string_of_sstmt fdecl.sbody) ^
  "}\n"

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

7.2.6 codegen.ml

```
module L = Llvm
module A = Ast
open Sast
module StringMap = Map.Make(String)
let translate(functions) =
  let context = L.global_context() in
  let the_module = L.create_module context "FUNC-y Java" in
    let i32_t      = L.i32_type    context
    and i8_t       = L.i8_type     context
    and i1_t       = L.i1_type     context
    and double_t    = L.double_type context in
    let str_t      = L.pointer_type i8_t
    and cry_t = L.void_type context in

  let ltype_of_typ = function
    A.Int    -> i32_t
    | A.Bool   -> i1_t
    | A.Float  -> double_t
    | A.Cry    -> cry_t
    | A.String -> str_t
    | A.Char -> i8_t
```

```
    | A.IntList (t) -> L.array_type i32_t t
    | A.FloatList (t) -> L.array_type double_t t
    | A.StringList (t) -> L.array_type str_t t

  in
  let type_of_lltype typ =
    let ltype_str = L.string_of_lltype typ in
    match ltype_str  with
     "i32"     -> A.Int
     | "i8"    -> A.Char
     | "i1"    -> A.Bool
     | "double" -> A.Float
     | "i8*"  -> A.String
     | "cry"   -> A.Cry
     | _ -> raise (Failure ("error: this type is unsupported"))
  in
  let type_of_lval lval =
    let lltype = L.type_of lval in
    type_of_lltype lltype
  in
let printf_t = L.var_arg_function_type i32_t [| L.pointer_type i8_t |] in
let printf_func = L.declare_function "printf" printf_t the_module in

  let function_decls : (L.llvalue * sfunc_decl) StringMap.t =
    let function_decl m fdecl =
      let name = fdecl.sfname
      and formal_types =
  Array.of_list (List.map (fun (t,_) -> ltype_of_typ t) fdecl.sformals)
      in let ftype = L.function_type (ltype_of_typ fdecl.styp)
   formal_types in
      StringMap.add name (L.define_function name ftype the_module, fdecl)
   m in
    List.fold_left function_decl StringMap.empty functions  in
  let build_function_body m_locals fdecl =
    let (the_function, _) = StringMap.find fdecl.sfname function_decls in
    let builder = L.builder_at_end context (L.entry_block the_function) in
    let str_format_str = L.build_global_stringptr "%s\n" "fmt" builder
    and int_format_str = L.build_global_stringptr "%d\n" "fmt" builder
    and float_format_str = L.build_global_stringptr "%f\n" "fmt" builder
    (* and bool_format_str = L.build_global_stringptr "%d\n" "fmt" builder
    *)
    and true_str = L.build_global_stringptr "true\n" "fmt" builder
    and false_str = L.build_global_stringptr "false\n" "fmt" builder
    and char_format_str = L.build_global_stringptr "%c\n" "fmt" builder in
    let add_formal m (t, n) p =
      L.set_value_name n p;
      let local = L.build_alloca (ltype_of_typ t) n builder in
      ignore (L.build_store p local builder);
      StringMap.add n local m
    in
    let locals = List.fold_left2 add_formal m_locals fdecl.sformals
        (Array.to_list (L.params the_function)) in
    (* Return the value for a variable or formal argument by
       checking local names *)
```

```
73    let lookup n m =
74      try StringMap.find n m
75        with Not_found -> raise(Failure ("Variable " ^ n ^ " not declared
      ") )
76    in
77    let extractValues n = match n with
78      SNum(v) -> v
79      | _ -> 0
80    in
81    let extractFValues n = match n with
82      SDec(v) -> v
83      | _ -> 0.0
84    in
85    let extractSValues n = match n with
86      SSlit(v) -> v
87      | _ -> ""
88    in
89    let rec expr builder m_locals ((_, e): sexpr) = match e with
90      SNum i -> L.const_int i32_t i
91      | SDec d -> L.const_float double_t d
92      | SBlit b -> L.const_int i1_t (if b then 1 else 0)
93      | SSlit s -> L.build_global_stringptr s "strpr" builder
94      | SChara c -> L.const_int i8_t (Char.code c)
95      | SArrayAccess(name, index) ->
96        let list_index = (expr builder m_locals index)
97        in let value = L.build_gep (lookup name m_locals) [| (L.const_int
      i32_t 0); list_index |] "tmp" builder
98        in L.build_load value "tmp" builder
99      | SVar v -> L.build_load (lookup v m_locals) v builder
100     | SAssign (s, e) -> let e' = expr builder m_locals e in
101                         ignore(L.build_store e' (lookup s m_locals)
      builder); e'
102           | SBinop ((A.Float,_ ) as e1, op, e2) ->
103       let e1' = expr builder m_locals e1
104       and e2' = expr builder m_locals e2 in
105       (match op with
106           A.Add     -> L.build_fadd
107         | A.Sub     -> L.build_fsub
108         | A.Mult    -> L.build_fmul
109         | A.Div     -> L.build_fdiv
110         | A.Equal   -> L.build_fcmp L.Fcmp.Oeq
111         | A.Neq     -> L.build_fcmp L.Fcmp.One
112         | A.Less    -> L.build_fcmp L.Fcmp.Olt
113         | A.Leq     -> L.build_fcmp L.Fcmp.Ole
114         | A.Greater -> L.build_fcmp L.Fcmp.Ogt
115         | A.Geq     -> L.build_fcmp L.Fcmp.Oge
116         | A.And | A.Or -> raise (Failure "internal error: semant should
      have rejected and/or on float")
117         | _ -> raise (Failure ("error: this operator is unsupported"))
118       ) e1' e2' "tmp" builder
119     | SBinop (e1, op, e2) ->
120       let e1' = expr builder m_locals e1
121       and e2' = expr builder m_locals e2 in
122       (match op with
```

```
123           A.Add     -> L.build_add
124         | A.Sub     -> L.build_sub
125         | A.Mult    -> L.build_mul
126         | A.Div     -> L.build_sdiv
127         | A.Mod     -> L.build_srem
128         | A.And     -> L.build_and
129         | A.Or      -> L.build_or
130         | A.Equal   -> L.build_icmp L.Icmp.Eq
131         | A.Neq     -> L.build_icmp L.Icmp.Ne
132         | A.Less    -> L.build_icmp L.Icmp.Slt
133         | A.Leq     -> L.build_icmp L.Icmp.Sle
134         | A.Greater -> L.build_icmp L.Icmp.Sgt
135         | A.Geq     -> L.build_icmp L.Icmp.Sge
136       ) e1' e2' "tmp" builder
137     | SUnop(op, ((t, _) as e)) ->
138       let e' = expr builder m_locals e in
139       (match op with
140           A.Neg when t = A.Float -> L.build_fneg
141         | A.Neg                 -> L.build_neg
142         | A.Not                 -> L.build_not) e' "tmp" builder
143     | SIntListLit(literals) ->
144       let arrayValues = List.map(fun el -> snd(el)) literals in
145       let v = List.map extractValues arrayValues in
146       let values = List.map(L.const_int i32_t) v
147       in let reversed_v = List.rev values in
148       L.const_array i32_t (Array.of_list reversed_v)
149     | SFloatListLit(literals) ->
150       let arrayValues = List.map(fun el -> snd(el)) literals in
151       let v = List.map extractFValues arrayValues in
152       let values = List.map(L.const_float double_t) v
153       in let reversed_v = List.rev values in
154       L.const_array double_t (Array.of_list reversed_v)
155     | SStringListLit(literals) ->
156       let arrayValues = List.map(fun el -> snd(el)) literals in
157       let v = List.map extractSValues arrayValues in
158        let sexprList = List.map (fun e -> (A.String, SSlit(e))) v in
159        let values = List.map (fun f -> expr builder m_locals f)
    sexprList
160       in let reversed_v = List.rev values in
161       L.const_array str_t (Array.of_list reversed_v)
162     | SCall("print", [e]) ->
163       let expr1 = expr builder m_locals e in
164         (match (type_of_lval expr1) with
165           A.String -> L.build_call printf_func [| str_format_str; (expr1
    ) |] "printf" builder
166           | A.Int -> L.build_call printf_func [| int_format_str; (expr1)
     |] "printf" builder
167           | A.Float -> L.build_call printf_func [| float_format_str; (
    expr1) |] "printf" builder
168           | A.Bool ->
169             let bstring = L.build_select expr1 true_str false_str "
    select" builder in
170              L.build_call printf_func [| bstring ; (expr1)|] "printf"
    builder
```

```
171           | A.Char -> L.build_call printf_func [| char_format_str; (
    expr1)|] "printf" builder
172           | A.Cry -> raise (Failure ("error: this type is unsupported"))
173           | _ -> raise (Failure ("error: this type is unsupported"))
174         )
175         | SCall(f, args) ->
176           let (fdef, fdecl) = StringMap.find f function_decls in
177           let llargs = List.rev( List.map (expr builder m_locals) (List.
    rev args)) in
178           let result = (match fdecl.styp with
179                         A.Cry -> ""
180                         | _ -> f ^ "_result") in
181           L.build_call fdef (Array.of_list llargs) result builder
182
183         in
184       let add_terminal (_, builder) instr =
185         match L.block_terminator (L.insertion_block builder) with
186           Some _ -> ()
187           | None -> ignore (instr builder) in
188
189       let rec stmt (m_locals, builder) = function
190         | SBlock sl -> List.fold_left stmt (m_locals, builder) sl
191         | SExpr e -> ignore(expr builder m_locals e); (m_locals, builder
    )
192         | SIf (predicate, then_stmt, else_stmt) ->
193           let bool_val = expr builder m_locals predicate in
194             let merge_bb = L.append_block context "merge" the_function
    in
195           let build_br_merge = L.build_br merge_bb in
196             let then_bb = L.append_block context "then" the_function in
197               add_terminal (stmt (m_locals, (L.builder_at_end context
    then_bb)) then_stmt)
198               build_br_merge;
199
200             let else_bb = L.append_block context "else" the_function in
201             add_terminal (stmt (m_locals, (L.builder_at_end context
    else_bb)) else_stmt)
202               build_br_merge;
203           ignore(L.build_cond_br bool_val then_bb else_bb builder);
204           m_locals, (L.builder_at_end context merge_bb)
205
206         | SWhile (predicate, body) ->
207           let pred_bb = L.append_block context "while" the_function in
208           ignore(L.build_br pred_bb builder);
209
210           let body_bb = L.append_block context "while_body" the_function
     in
211             add_terminal (stmt (m_locals, (L.builder_at_end context
    body_bb)) body)
212             (L.build_br pred_bb);
213
214           let pred_builder = L.builder_at_end context pred_bb in
215           let bool_val = expr pred_builder m_locals predicate in
216
```

```
            let merge_bb = L.append_block context "merge" the_function in
            ignore(L.build_cond_br bool_val body_bb merge_bb pred_builder)
   ;
            m_locals,  (L.builder_at_end context merge_bb)

        | SFor (e1, e2, e3, body) -> stmt (m_locals, builder)
        ( SBlock [SExpr e1 ; SWhile (e2, SBlock [body ; SExpr e3]) ] )
        | SReturn e -> ignore(match fdecl.styp with
                              (* Special "return nothing" instr *)
                              A.Cry -> L.build_ret_void builder
                              (* Build return statement *)
                            | _ -> L.build_ret (expr builder m_locals e)
   builder );
                    m_locals, builder
        | SVdecl (t, v) -> let local = L.build_alloca (ltype_of_typ t) v
    builder in
        let m_locals = StringMap.add v local m_locals in (m_locals,
   builder)
      |SVdeclAssign (typ, v, ((t, e): sexpr)) ->
        let e' =  expr builder m_locals (t, e) in
      let add_local m (t, n) =
        let local_var = L.build_alloca (ltype_of_typ t) n builder
        in StringMap.add n local_var m
      in
      let m_locals = add_local m_locals (typ, v)
      in ignore(L.build_store e' (lookup v m_locals) builder); (m_locals
   , builder)
     in
     let (_, builder) = stmt (locals, builder) (SBlock fdecl.sbody) in
     add_terminal (m_locals, builder) (match fdecl.styp with
       A.Cry -> L.build_ret_void
       | A.Int -> L.build_ret (L.const_int i32_t 0)
       | A.Float -> L.build_ret (L.const_float double_t 0.0)
       | A.Char -> L.build_ret (L.const_int i8_t (Char.code '0'))
       | A.Bool -> L.build_ret (L.const_int i1_t 0)
       | t -> L.build_ret (L.const_int (ltype_of_typ t) 0))

    in List.iter (build_function_body StringMap.empty) functions;
    the_module
```

7.2.7 funcyjava.ml

```

let () =
  let usg_msg = "usage: ./funcyjava.native [file.fj]" in
  let channel = ref stdin in
  Arg.parse [] (fun filename -> channel := open_in filename) usg_msg;

  let lexbuf = Lexing.from_channel !channel in
  let ast = Parser.program Scanner.token lexbuf
  in let sast = Semant.check ast in
  let m = Codegen.translate sast in
    Llvm_analysis.assert_valid_module m;
  print_string (Llvm.string_of_llmodule m)
```

7.2.8 Makefile

```
funcyjava.native:
  opam config exec -- \
  ocamlbuild -use-ocamlfind funcyjava.native

# "make test" Compiles everything and runs the regression tests
.PHONY: test
test: funcyjava.native testall.sh
  ./testall.sh

# "make clean" removes all generated files
.PHONY : clean
clean:
  ocamlbuild -clean
  rm -rf testall.log ocamlllvm *.diff
  rm -rf *.cmx *.cmi *.cmo *.cmx *.o *.s *.ll *.out *.exe *.mli

.PHONY: demo1
demo1: funcyjava.native demos/demo1.fj
  ./funcyjava.native demos/demo1.fj > demo1.ll
  llc -relocation-model=pic demo1.ll > demo1.s
  cc -o demo1.exe demo1.s
  ./demo1.exe
  rm -rf *.cmx *.cmi *.cmo *.cmx *.o *.s *.ll *.out *.exe *.mli


.PHONY: demo2
demo2: funcyjava.native demos/demo2.fj
  ./funcyjava.native demos/demo2.fj > demo2.ll
  llc -relocation-model=pic demo2.ll > demo2.s
  cc -o demo2.exe demo2.s
  ./demo2.exe
  rm -rf *.cmx *.cmi *.cmo *.cmx *.o *.s *.ll *.out *.exe *.mli

.PHONY: demo3
demo3: funcyjava.native demos/demo3.fj
  ./funcyjava.native demos/demo3.fj > demo3.ll
  llc -relocation-model=pic demo3.ll > demo3.s
  cc -o demo3.exe demo3.s
  ./demo3.exe
  rm -rf *.cmx *.cmi *.cmo *.cmx *.o *.s *.ll *.out *.exe *.mli

TESTS = \
  print1 print2 print3 var1 var2 var3 var4 var5 var6 \
  mod1 assignop1 assignop2 binop1 binop2 \
  if1 if2 if3 if4 if5 ifelse1 for1 for2 \

FAILS = \
  assign1

TESTFILES = $(TESTS:%=test-%.fj) $(TESTS:%=test-%.out) \
      $(FAILS:%=fail-%.fj) $(FAILS:%=fail-%.err)

```

```
TARFILES = ast.ml sast.ml codegen.ml Makefile _tags funcyjava.ml parser.mly \
  README.md scanner.mll semant.ml testall.sh \
  Dockerfile \
  $(TESTFILES:%=tests/%)

funcyjava.tar.gz : $(TARFILES)
  cd .. && tar czf funcyjava/funcyjava.tar.gz \
  $(TARFILES:%=funcyjava/%)
```

7.3 Test Suite

7.3.1 testall.sh

```
#!/bin/sh

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

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

# Path to the LLVM compiler
LLC="llc"

# Path to the C compiler
CC="cc"

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

# Set time limit for all operations
ulimit -t 30

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

keep=0

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

```

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

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

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

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

Check() {
    error=0
    basename=`echo $1 | sed 's/.*\\///
                              s/.fj//'`
    reffile=`echo $1 | sed 's/.fj$//'`
    basedir="`echo $1 | sed 's/\/[^\/]*$//'`/."

    echo -n "$basename..."

    echo 1>&2
    echo "###### Testing $basename" 1>&2

    generatedfiles=""
```

```
93
94      generatedfiles="$generatedfiles ${basename}.ll ${basename}.s ${
        basename}.exe ${basename}.out" &&
95      Run "$FUNCYJAVA" "$1" ">" "${basename}.ll" &&
96      Run "$LLC" "-relocation-model=pic" "${basename}.ll" ">" "${basename}.s
        " &&
97      Run "$CC" "-o" "${basename}.exe" "${basename}.s" &&
98      Run "./${basename}.exe" > "${basename}.out" &&
99      Compare ${basename}.out ${reffile}.out ${basename}.diff
100
101     # Report the status and clean up the generated files
102
103     if [ $error -eq 0 ] ; then
104   if [ $keep -eq 0 ] ; then
105       rm -f $generatedfiles
106   fi
107   echo "AMAZING"
108   echo "###### SUCCESS" 1>&2
109     else
110   echo "###### FAILED" 1>&2
111   globalerror=$error
112     fi
113 }
114
115 CheckFail() {
116     error=0
117     basename=`echo $1 | sed 's/.*\\///
118                               s/.fj//'`
119     reffile=`echo $1 | sed 's/.fj$//'`
120     basedir="`echo $1 | sed 's/\/[^\/]*$//'`/."
121
122     echo -n "$basename..."
123
124     echo 1>&2
125     echo "###### Testing $basename" 1>&2
126
127     generatedfiles=""
128
129     generatedfiles="$generatedfiles ${basename}.err ${basename}.diff" &&
130     RunFail "$FUNCYJAVA" "<" $1 "2>" "${basename}.err" ">>" $globallog &&
131     Compare ${basename}.err ${reffile}.err ${basename}.diff
132
133     # Report the status and clean up the generated files
134
135     if [ $error -eq 0 ] ; then
136   if [ $keep -eq 0 ] ; then
137       rm -f $generatedfiles
138   fi
139   echo "AMAZING"
140   echo "###### SUCCESS" 1>&2
141     else
142   echo "###### FAILED" 1>&2
143   globalerror=$error
144     fi
```

```
}

while getopts kdpsh c; do
    case $c in
  k) # Keep intermediate files
      keep=1
      ;;
  h) # Help
      Usage
      ;;
    esac
done

shift `expr $OPTIND - 1`

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

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

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

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

exit $globalerror
```

7.3.2 tests

fail-assign1.fj

```
func cry main() {
    n = 5;
```

```
}
```

fail-assign1.err

```
Fatal error: exception Failure("undeclared identifier n")
```

fail-assign2.fj

```
func int main() {
    int x = 8;
    x = 8.8;
}
```

fail-assign2.err

```
Fatal error: exception Failure("illegal assignment int = float in x =
   8.8")
```

fail-assign3.fj

```
func int main() {
    float f;
    f = "I'm a func-y float!";
}
```

fail-assign3.err

```
Fatal error: exception Failure("illegal assignment float = string in f = "
   I'm a func-y float!"")
```

fail-assign4.fj

```
func int main() {
    char c;
    c = 9+3;
}
```

fail-assign4.err

```
Fatal error: exception Failure("illegal assignment char = int in c = 9 +
   3")
```

fail-assignop1.fj

```
func int main() {
    float f = 9.3;
    float ff = f++;
    print(ff);
}
```

fail-assignop1.err

```
Fatal error: exception Failure("illegal binary operator float + int in f +
    1")
```

fail-assignop2.fj

```
func int main() {
    char c = '4';
    c += '8';
}
```

fail-assignop2.err

```
Fatal error: exception Failure("illegal binary operator char + char in c +
    8")
```

fail-assignop3.fj

```
func int main() {
    int x = 4;
    x *= 8.8;
}
```

fail-assignop3.err

```
Fatal error: exception Failure("illegal binary operator int * float in x *
    8.8")
```

fail-assignop4.fj

```
func int main() {
    float f = 5.6;
    f %= 2.2;
}
```

fail-assignop4.err

```
Fatal error: exception Failure("illegal binary operator float % float in f
    % 2.2")
```

fail-binop1.fj

```
func cry main() {
    int n = 9;
    float m = 7.3;
    int p = n + m;
}
```

fail-binop1.err

```
Fatal error: exception Failure("illegal binary operator int + float in n +
    m")
```

fail-binop2.fj

```
func int main() {
    str sa = "hello";
    str sb = "world";
    print(sa * sb);
}
```

fail-binop2.err

```
Fatal error: exception Failure("illegal binary operator string * string in
    sa * sb")
```

fail-binop3.fj

```
func int main() {
    char ca = 'a';
    char cb = 'p';
    print(ca + cb);
}
```

fail-binop3.err

```
Fatal error: exception Failure("illegal binary operator char + char in ca
   + cb")
```

fail-binop4.fj

```
func int main() {
    print("hello" * 5);
}
```

fail-binop4.err

```
Fatal error: exception Failure("illegal binary operator string * int in "
   hello" * 5")
```

fail-binop5.fj

```
func int main() {
    print("hello" * 5);
}
```

fail-binop5.err

```
Fatal error: exception Failure("illegal binary operator string * int in "
   hello" * 5")
```

fail-boolop1.fj

```
func int main() {
    if (5 and 8) {
        print(8);
    }
}
```

fail-boolop1.err

```
Fatal error: exception Failure("illegal binary operator int and int in 5
   and 8")
```

fail-boolop2.fj

```
func int main() {
    print("yes" or "no");
}
```

fail-boolop2.err

```
Fatal error: exception Failure("illegal binary operator string or string
   in "yes" or "no"")
```

fail-boolop3.fj

```
func int main() {
    char c = 'a';
    if (not c) {
        print("this should not be able to print");
    }
}
```

fail-boolop3.err

```
Fatal error: exception Failure("illegal unary operator not char in not c")
```

fail-equalop1.fj

```
func char operate(char a, char b) {
    if (a > b) {
        return a;
    }
    else {
        return b;
    }
}

func int main() {
    print(operate('a', 'p'));
    print(operate(operate('p', 'l'), 't'));
}
```

fail-equalop1.err

```
Fatal error: exception Failure("illegal binary operator char > char in a > b")
```

fail-equalop2.fj

```
func bool geq(int a, float b) {
    if (a >= b) {
        return true;
    }
    else {
        return false;
    }
}

func int main() {
    print(geq(4, 5.7));
}
```

fail-equalop2.err

```
Fatal error: exception Failure("illegal binary operator int >= float in a >= b")
```

fail-floatarray.fj

```
func int main(){
    array<float, 1> flList = [5.6, 4.5];
    return 1;
}
```

fail-floatarray.err

```
Fatal error: exception Failure("illegal assignment array<float, 1> = array<float, 2> in array<float, 1> flList = [4.5, 5.6];
")
```

fail-floatarray2.fj

```
func int main(){
    array<float, 2> tired = [3.4, 5.0];
    int x = tired[0];
    return 1;
}
```

fail-floatarray2.err

```
Fatal error: exception Failure("illegal assignment int = array<float, 2>
   in int x = tired[0];
")
```

fail-floatarray3.fj

```
func int main(){

    array<float, 5> fL = [3.4, 4.4, 0.8, 3.22, 4.3];

    int x = 1 + fL[0];

    return 0;
}
```

fail-floatarray3.err

```
Fatal error: exception Failure("illegal binary operator int + array<float,
    5> in 1 + fL[0]")
```

fail-floatarray4.fj

```
func int main(){

    array<float, 5> fL = [3.4, 4.4, 0.8, 3.22, 4.3];

    str x = fL[0];

    return 0;
}
```

fail-floatarray4.err

```
Fatal error: exception Failure("illegal assignment string = array<float,
   5> in string x = fL[0];
")
```

fail-floatarray5.fj

```
func int main(){

    array<float, 5> fL = [3.4, 4.4, 0.8, 3.22, 4.3];

    int x = 1 + fL[0];

    return 0;
}
```

fail-floatarray5.err

```
Fatal error: exception Failure("illegal binary operator int + array<float,
    5> in 1 + fL[0]")
```

fail-floatarray6.fj

```
func int main(){
    array<float, 5> fL;
    fL = [3.4, 4.5, 0.3, 5.2, 2.7];
```

```
4
5       return 0;
6
7   }
```

fail-floatarray6.err

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

fail-for1.fj

```
1   func int main() {
2       for(i = 0; i < 10; i++) {
3           print(i);
4       }
5   }
```

fail-for1.err

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

fail-for2.fj

```
1   func int main() {
2       int i;
3       for(i = 0; 10; i++) {
4           print(i);
5       }
6   }
```

fail-for2.err

```
1 Fatal error: exception Failure("expected Boolean expression, got 10
     instead")
```

fail-for3.fj

```
1   func int main() {
2       int i;
3       for(i = 0; i < 10; i*=j) {
4           print(i);
5       }
6   }
```

fail-for3.err

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

fail-for4.fj

```
1   func int main() {
2       int i;
3       for(i = 0; 10; i++) {
4           printfunc(i, i+1);
5       }
6   }
```

fail-for4.err

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

fail-func1.fj

```
func char funcystring(char c) {
    return c;
}
```

fail-func1.err

```
Fatal error: exception Failure("missing main function")
```

fail-func2.fj

```
func int main() {
    str s = "It is I, the main function.";
    print(s);
}

func int main() {
    print("No! I am the main function!");
}
```

fail-func2.err

```
Fatal error: exception Failure("duplicate function: a function with name
   main has already been defined")
```

fail-func3.fj

```
func int plt() {
    return 7;
}
func int main() {
    print(8);
}

func int plt() {
    return 8;
}
```

fail-func3.err

```
Fatal error: exception Failure("duplicate function: a function with name
   plt has already been defined")
```

fail-func4.fj

```
func int main(){
    print(7);
}

func int print(int x) {
    return x*x;
}
```

fail-func4.err

```
Fatal error: exception Failure("the function print is built-in and may not
    be redefined")
```

fail-func5.fj

```
func int main(){
    floaty(7);
}

func float floaty(float f) {
    if (f == 9.9) {
        return f;
    }
    else {
        return 2.3;
    }
}
```

fail-func5.err

```
Fatal error: exception Failure(" illegal argument found int expected float
    in 7")
```

fail-if1.fj

```
func int main() {
    int i = 2;
    if (7) {
        print(i);
    }
}
```

fail-if1.err

```
Fatal error: exception Failure("expected Boolean expression, got 7 instead
   ")
```

fail-if2.fj

```
func int main() {
    if ("hello") {
        print("world");
    }
}
```

fail-if2.err

```
Fatal error: exception Failure("expected Boolean expression, got "hello"
   instead")
```

fail-intarray.fj

```
func int main(){
    array<int, 3> llist = [2, 3, 4];
    print(llist);
}
```

fail-intarray.err

```
Fatal error: exception Failure("error: this type is unsupported")
```

fail-intarray2.fj

```
func int main(){
    array<int, 2> llist = [2, 3, 4];
    return 0;
}
```

fail-intarray2.err

```
Fatal error: exception Failure("illegal assignment array<int, 2> = array<
    int, 3> in array<int, 2> llist = [4, 3, 2];
")
```

fail-intarray3.fj

```
func int main(){
    array<int, 3> iL = [3, 4, 5];
    float x = iL[1];
    return 0;
}
```

fail-intarray3.err

```
Fatal error: exception Failure("illegal assignment float = array<int, 3>
    in float x = iL[1];
")
```

fail-intarray4.fj

```

func int main(){
    array<int, 3> ai = [9, 6, 4];
    int x = ai[0] + 4.0;
    print(x);
    return 1;
}
```

fail-intarray4.err

```
Fatal error: exception Failure("illegal binary operator array<int, 3> +
    float in ai[0] + 4.")
```

fail-mod1.fj

```
func int main() {
    float x = 8.0;
    print(x%3);
}
```

fail-mod1.err

```
Fatal error: exception Failure("illegal binary operator float % int in x %
    3")
```

fail-return1.fj

```
func int badreturn() {
    float x = 8.0;
    return x;
}

func int main() {
    print(badreturn());
}
```

fail-return1.err

```
Fatal error: exception Failure("return gives float expected int in x")
```

fail-return2.fj

```
func float foo() {
    float x = 8.0;
    return x;
    x = 9.9;
    print(x);
}

func int main() {
    print(foo());
}
```

fail-return2.err

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

fail-stringarray.fj

```
func int main(){
    array<str, 3> sl = ["hopeless", "crying"];
    return 0;
}
```

fail-stringarray.err

```
Fatal error: exception Failure("illegal assignment array<str, 3> = array<
    str, 2> in array<str, 3> sl = ["crying", "hopeless"];
")
```

fail-stringarray2.fj

```
func int main(){
    array<str, 2> sL = ["hello", "world"];
    print(sL);
}
```

fail-stringarray2.err

```
Fatal error: exception Failure("error: this type is unsupported")
```

fail-stringarray3.fj

```
func int main(){
    array<str, 1> sList = ["sorrow"];
    int x = sList[0];
    return 0;
}
```

fail-stringarray3.err

```
Fatal error: exception Failure("illegal assignment int = array<str, 1> in
    int x = sList[0];
")
```

fail-uop1.fj

```
func int main() {
    print(negate());
}

```

```
func str negate() {
    return -"positive";
}
```

fail-uop1.err

```
Fatal error: exception Failure("illegal unary operator - string in - "
   positive"")
```

fail-while1.fj

```
func int main(){
    while (i < 6){
        print(i);
        i++;
    }
}
```

fail-while1.err

```
Fatal error: exception Failure("undeclared identifier i")
```

fail-while2.fj

```
func int main(){
    int i = 80;
    while (i >= 70){
        foo(i);
        i--;
    }
}
```

fail-while2.err

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

fail-while3.fj

```
func int main(){
    while ("func-y"){
        print("hello");
    }
}
```

fail-while3.err

```
Fatal error: exception Failure("expected Boolean expression, got "func-y"
   instead")
```

test-assnop1.fj

```
func int main() {
    int a = 2;
    print(a);
    a++;
    print(a);

    int b = a;
    b--;
    print(b);
```

```
    print(a);

    a *= 8;
    print(a);

    b += 10;
    print(b);

    b /= 2;
    print(b);

    int c = a++;
    print(c);
    print(a);

    int d = c /= 5;
    print(d);
    print(c);

    a %= 7;
    print(a);
    print(c);

    int e = c *= 3;
    print(e);
}
```

test-assnop1.out

```
2
3
2
3
24
12
6
25
25
5
5
4
5
15
```

test-assnop2.fj

```
func int main() {
    float a = 2.2;
    print(a);
    a += 4.4;
    print(a);

    float b = a;
    b -= 6.8;
    print(b);
    print(a);
```

```
    a *= 3.14;
    print(a);

    b /= 10.0;
    print(b);

    float c = a += 0.006;
    print(c);
    print(a);

    float d = c /= 5.0;
    print(d);
    print(c);
}
```

test-assnop2.out

```
2.200000
6.600000
-0.200000
6.600000
20.724000
-0.020000
20.730000
20.730000
4.146000
4.146000
```

test-binop1.fj

```
func int main() {
    print(6+7 * 9 + 34 %2);
    print(25/5/4/3 % 2 * 17);
    print(8*7+27+9*1 - 18);
    print(2+8%3);
}
```

test-binop1.out

```
69
0
74
4
```

test-binop2.fj

```
func int main() {
    print(6.7+3.4 * 9.2 - 0.2);
    print(25.0/5.1/4.2/3.3 * 17.1);
    print(2.0 + 2.4 * 3.1 - 3.1);
    print(9.1 - 7.198 + 8.2 * 0.0);
}
```

test-binop2.out

```
37.780000
6.047874
```

```
6.340000
1.902000
```

test-boolop-predicate1.fj

```
func int main() {
    int y = 4;
    int z;
    if (y > 0 and y < 8) {
        z = y;
    }
    else {
        z = 0;
    }
    print(z);
}
```

test-boolop-predicate1.out

```
4
```

test-boolop-predicate2.fj

```
func int main() {
    int y = 4;
    int z;
    if (y > 10 or y <= 0) {
        z = y;
    }
    else {
        z = 0;
    }
    print(z);
}
```

test-boolop-predicate2.out

```
0
```

test-boolop-predicate3.fj

```
func cry istrue(bool b) {
    if (not b) {
        print("b is false");
    }
    else {
        print("b is true");
    }
}
func int main() {
    istrue(true);
    istrue(false);
}
```

test-boolop-predicate3.out

```
"b is true"
"b is false"
```

test-boolop-predicate4.fj

```
func bool funcybools(bool a, bool b) {
    if (a and b or not a and not b) { /* a and b have same value */
        return true;
    }
    else {
        return false;
    }
}
func int main() {
    bool w = funcybools(false, true);
    bool x = funcybools(true, false);
    bool y = funcybools(true, true);
    bool z = funcybools(false, false);
    print(w);
    print(x);
    print(y);
    print(z);
}
```

test-boolop-predicate4.out

```
false
false
true
true
```

test-comments1.fj

```
func int main() {
    /* This file has simple if/else control flow */
    if (7 > 9) { /* This should be false! */
        print("That's odd");
    }

    else { /* That's better */
        print("Sanity check passed");
    }
}
```

test-comments1.out

```
"Sanity check passed"
```

test-comments2.fj

```
func int main() {
    /* This file has simple if/else control flow:
    the program must make a decision and will execute
    one of two blocks based on the value of a predicate */
    if (7 > 9) {
        print("That's odd");
    }
    /* The preidcate is false
    so now we move onto the else block */

    else {
        print("Sanity check passed");
```

```
    }
}
```

test-comments2.out

```
"Sanity check passed"
```

test-demo1.fj

```
/* function to return a message based on input name (a string) */
func str message(str s) {
    str m = "";
    if (s == "pazit" or s == "Pazit") {
        m = " is our tester";
    }
    else {
        if (s == "lindsey" or s == "Lindsey"){
            m = " is our language guru";
        }
        else {
            if (s == "katrina" or s == "Katrina") {
                m = " is a system architect";
            }
            else {
                if (s == "liseidy" or s == "Liseidy") {
                    m = " is a system architect";
                }
                else {
                    if (s == "kenya" or s == "Kenya"){
                    m = " is our fearless leader";
                    }
                    else {
                        m = " is not in our group!";
                    }
                }
            }
        }
    }
    return m;
}

func int main() {
    /* print strings or print directly from function returns */
    str spacer = "..............................";
    print("kenya... ");
    print(kenya());
    print(spacer);

    print("lindsey... ");
    print(lindsey());
    print(spacer);

    print("liseidy... ");
    print(liseidy());
    print(spacer);

```

```
    print("katrina... ");
    print(katrina());
    print(spacer);

    print("pazit... ");
    print(pazit());
    print(spacer);

    str nonmember = "the capybara...";
    print(nonmember);
    print(message(nonmember));
    print(spacer);
}

/* helper function that call more helper functions */
func str kenya() {
    str name = "kenya";
    return message(name);
}

func str lindsey() {
    str name;
    name = "Lindsey";
    return message(name);
}

func str katrina() {
    return message("Katrina");
}

func str liseidy() {
    str name;
    name = "liseidy";
    return message(name);
}

func str pazit() {
    return message("pazit");
}
```

test-demo1.out

```
"kenya... "
" is our fearless leader"
"............................."
"lindsey... "
" is our language guru"
"............................."
"liseidy... "
" is a system architect"
"............................."
"katrina... "
" is a system architect"
"............................."
"pazit... "
```

```
" is our tester"
"............................."
"the capybara..."
" is not in our group!"
"............................."
```

test-demo2.fj

```
func char grade(int g) {
    char grade = 'Z';
    if (g >= 60) {
        grade = 'D';
        if (g >= 70) {
            grade = 'C';
            if (g >= 80 and g < 90) {
                grade = 'B';
            }
            else {
                if (g >= 90) {
                    grade = 'A';
                }
            }
        }
    }
    else {
        grade = 'F';
    }
    return grade;
}

func int main() {
    print(grade(94));
    print(grade(87));
    print(grade(79));
    print(grade(60));
    print(grade(8));
}
```

test-demo2.out

```
A
B
C
D
F
```

test-demo3.fj

```
func int main(){
    array<int, 12> listy= [3, 62, 18, 7, 4, 9, 1, 55, 10, 12, 145, 17];
    int tempmax = listy[0];
    int tempmin = listy[0];
    bool sorted = true;
    int i;
    for(i = 0; i<11; i++){
        int x = listy[i];
        if(x < tempmin){
```

```
            tempmin = x;
        }
        if (x>tempmax){
            tempmax = x;
        }
    }
    int j;
    for(j=0; j<11; j++){
        int first = listy[j];
        int next = listy[j+1];
        if (first > next){
            sorted = false;
        }
    }
    print(sorted);
    print(tempmax);
}
```

test-demo3.out

```
false
145
```

test-equalop1.fj

```
func bool eqInt(int a, int b) {
    return(a==b);
}

func bool eqFloat(float a, float b) {
    return(a==b);
}

func bool eqChar(char a, char b) {
    return(a==b);
}

func bool eqBool(bool a, bool b) {
    return(a==b);
}

func bool eqStr(str a, str b) {
    return(a==b);
}

func int main() {
    print(eqInt(7+0, 5+1));
    print(eqFloat(3.14-0.14, 3.00));
    print(eqChar('3','3'));
    print(eqChar('a', 'p'));
    print(eqBool(true,true));
    print(eqBool(1>2, false and true));
    print(eqStr("hi","HI"));
    print(eqStr("hello", "hello"));
}
```

test-equalop1.out

```
false
true
true
false
true
true
false
true
```

test-equalop2.fj

```
func bool eqInt(int a, int b) {
    return(a==b);
}

func bool eqFloat(float a, float b) {
    return(a==b);
}

func bool eqChar(char a, char b) {
    return(a==b);
}

func bool eqBool(bool a, bool b) {
    return(a==b);
}

func bool eqStr(str a, str b) {
    return(a==b);
}

func int main() {
    int a = 7+0;
    print(eqInt(a, 5+1));
    float f = 3.00;
    print(eqFloat(3.14-0.14, f));

    char c = '3';
    print(eqChar(c,'3'));
    print(eqChar('a', c));


    bool b = true;
    print(eqBool(b,true));
    print(eqBool(1>2, b and not b));


    str s = "hi";
    print(eqStr(s,"HI"));
    print(eqStr(s, "hi"));
}
```

test-equalop2.out

```
false
true
true
false
true
true
false
true
```

test-equalop3.fj

```
func bool eqInt(int a, int b) {
    return(a!=b);
}

func bool eqFloat(float a, float b) {
    return(a!=b);
}

func bool eqChar(char a, char b) {
    return(a!=b);
}

func bool eqBool(bool a, bool b) {
    return(a!=b);
}

func bool eqStr(str a, str b) {
    return(a!=b);
}

func int main() {
    int a = 7+0;
    print(eqInt(a, 5+1));
    float f = 3.00;
    print(eqFloat(3.14-0.14, f));

    char c = '3';
    print(eqChar(c,'3'));
    print(eqChar('a', c));


    bool b = true;
    print(eqBool(b,true));
    print(eqBool(1>2, b and not b));


    str s = "hi";
    print(eqStr(s,"HI"));
    print(eqStr(s, "hi"));
}
```

test-equalop3.out

```
true
false
```

```
false
true
false
false
true
false
```

test-fact1.fj

```
func int fact(int n) {
    if (n <= 1) {
        return 1;
    }
    return n * fact(n-1);
}

func int main() {
    print(fact(8));
    print(fact(0));
}
```

test-fact1.out

```
40320
1
```

test-fib1.fj

```
func int fib(int n) {
    if (n <= 1) {
        return n;
    }

    return fib(n-1)+fib(n-2);
}

func int main() {
    int i;
    for (i = 0; i < 7; i++) {
       print(fib(i));
    }
}
```

test-fib1.out

```
0
1
1
2
3
5
8
```

test-floatarray.fj

```
func int main(){
    array<float, 2> fL = [3.4, 4.4];
    float x = fL[0];
```

```
    print(x);
}
```

test-floatarray.out

```
3.400000
```

test-floatarray2.fj

```
func int main(){
    array<float, 2> fL = [3.4, 4.5];

    int x;

    for(x = 0; x < 2; x++){

      print(fL[x]);

    }

    return 0;

}
```

test-floatarray2.out

```
3.400000
4.500000
```

test-floatarray3.fj

```
func int main(){
    array<float, 5> fL = [3.4, 4.5, 0.3, 5.2, 2.7];

    int x;

    for(x = 0; x < 5; x++){

      float y = fL[x];

      if(y >= 2.7){
        print(y);
      } else {
        print("Less than 2.7");
      }

    }

    return 0;

}
```

test-floatarray3.out

```
3.400000
4.500000
"Less than 2.7"
5.200000
2.700000
```

test-floatarray4.fj

```
func int main(){
    array<float, 5> fL = [3.4, 4.5, 0.3, 5.2, 2.7];

    int x;

    for(x = 0; x < 5; x++){

      float y = fL[x];

      float z = 1.0;

      float sum = y + z;

      print(sum);
    }

    return 0;

}
```

test-floatarray4.out

```
4.400000
5.500000
1.300000
6.200000
3.700000
```

test-floatarray5.fj

```
func int main(){

    array<float, 5> fL = [3.4, 4.5, 0.3, 5.2, 2.7];

    float a;
    a = fL[0];

    print(a);

    return 0;

}
```

test-floatarray5.out

```
3.400000
```

test-for1.fj

```
func int main(){
    int i;
    for (i = 0; i < 6; i++){
        print(i);
    }

    for (i = 80; i >= 70; i--){
```

```
        print(i);
    }
}
```

test-for1.out

```
0
1
2
3
4
5
80
79
78
77
76
75
74
73
72
71
70
```

test-for2.fj

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

    result = 1;
    for (i = 6; i > 0; i--) {
        result = result * i;
    }
    print(result);

    result = 10;
    for (i = 1; i < 4; i++) {
        result -= i;
    }
    print(result);
}
```

test-for2.out

```
15
720
4
```

test-for3.fj

```
func int main(){
    int result = 0;
```

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

    result = 1;
    for (i = 6; i > 0; i=i-1) {
        result = result * i;
    }
    print(result);

    result = 1;
    for (i = 1; i < 10; i=i*2) {
        result = result * i;
    }
    print(result);

    result = 0;
    for (i = 90000; i > 0; i= i/10) {
        result = result + i;
    }
    print(result);
}
```

test-for3.out

```
15
720
64
99999
```

test-for4.fj

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

    result = 1;
    for (i = 6; i > 0; i-=1) {
        result = result * i;
    }
    print(result);

    result = 1;
    for (i = 1; i < 10; i*=2) {
        result = result * i;
    }
    print(result);

    result = 0;
    for (i = 90000; i > 0; i/=10) {
```

```
        result = result + i;
    }
    print(result);
}
```

test-for4.out

```
15
720
64
99999
```

test-for6.fj

```
func int main() {
    int i;
    for (i = 0; i < 4; i++) {
        int j;
        for (j = 4; j < 8; j++) {
            print(i+j);
        }
    }
}
```

test-for6.out

```
4
5
6
7
5
6
7
8
6
7
8
9
7
8
9
10
```

test-forwhile1.fj

```
func int main() {
    int myvar = 5;
    while (myvar > 0) {
        int i;
        int result = 0;
        for (i = 0; i < myvar; i++) {
            result += i;
        }
        print(result);
        myvar--;
    }
}
```

test-forwhile1.out

```
10
6
3
1
0
```

test-forwhile2.fj

```
func int main(){
    int i;
    for (i = 1; i < 5; i++) {
        int j = 10;
        while (j > 0) {
            print(i*j);
            j /= 2;
        }
    }
}
```

test-forwhile2.out

```
10
5
2
1
20
10
4
2
30
15
6
3
40
20
8
4
```

test-func1.fj

```
func int main() {
    int x = square(8);
    print(x);
}

func int square(int s) {
    return s*s;
}
```

test-func1.out

```
64
```

test-func2.fj

```
func int cube(int s) {
    return s*s*s;
```

```
}

func int main() {
    print("8 squared is ");
    print(square(8));
    print("8 cubed is ");
    print(cube(8));
}

func int square(int s) {
    return s*s;
}
```

test-func2.out

```
"8 squared is "
64
"8 cubed is "
512
```

test-func3.fj

```
func int cube(int s) {
    return square(s)*s;
}

func int main() {
    print(badmultiply(4,4));
    print(square(4));
    print(cube(4));
    print(cube(4) * square(2));
    print(badmultiply(0,9));
}

func int square(int s) {
    return badmultiply(s,s);
}

func int badmultiply(int a, int b) {
    if (a == 0 or b == 0) {
        return 0;
    }
    else {
        int result = 0;
        while (b > 0) {
            result += a;
            b--;
        }
        return result;
    }
}
```

test-func3.out

```
16
16
64
```

```
256
0
```

test-func4.fj

```
func int cube(int s) {
    return s*s*s;
}

func str cbc(str s) {
    return s;
}

func int main() {
    print(4);
}

func int crb(int s) {
    return s*s;
}
```

test-func4.out

```
4
```

test-func5.fj

```
func int main() {
    str s = "CaFe GrUmPy";
    printStr(s);
}

func cry printStr(str s) {
    print(s);
}
```

test-func5.out

```
"CaFe GrUmPy"
```

test-func6.fj

```
func int main() {
    bool b = retBool(4.5, 9.4);
    print(b);
    print(retBool(5.5, 5.44));
}

func bool retBool(float a, float b) {
    if (a > b) {
       return true;
    }
    return false;
}
```

test-func6.out

```
false
true
```

test-gcd.fj

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

func int main(){
    int x = gcd(14, 19);
    print(x);

    int y = gcd(12, 4);
    print(y);

    int z = gcd(24, 9);
    print(z);
}
```

test-gcd.out

```
1
4
3
```

test-if1.fj

```
func int main() {
    int x = 9;
    int y = 8;

    if (x > y) {
        print("x is larger");
        print(x);
    }

    if (x < y) {
        print("x is smaller");
        print(y);
    }

    if (x == y) {
        print("equal");
    }

    int a = 3;
    int b = 7;

    if (a > b) {
        print("a is larger");
```

```
        print(a);
    }

    if (a < b) {
        print("a is smaller");
        print(b);
    }

    if (a == b) {
        print("equal");
    }

    int c = 2;
    int d = 2;

    if (c > d) {
        print("c is larger");
        print(c);
    }

    if (c < d) {
        print("c is smaller");
        print(d);
    }

    if (c == d) {
        print("equal");
    }

}
```

test-if1.out

```
"x is larger"
9
"a is smaller"
7
"equal"
```

test-if2.fj

```
func int main() {
    float x = 9.0001;
    float y = 8.1;

    if (x > y) {
        print("x is larger");
        print(x);
    }

    if (x < y) {
        print("x is smaller");
        print(y);
    }

    if (x == y) {
```

```
        print("equal");
    }

    float a = -3.21;
    float b = 1.4;

    if (a > b) {
        print("a is larger");
        print(a);
    }

    if (a < b) {
        print("a is smaller");
        print(b);
    }

    if (a == b) {
        print("equal");
    }

    float c = 2.0;
    float d = 2.0;

    if (c > d) {
        print("c is larger");
        print(c);
    }

    if (c < d) {
        print("c is smaller");
        print(d);
    }

    if (c == d) {
        print("equal");
    }

}
```

test-if2.out

```
"x is larger"
9.000100
"a is smaller"
1.400000
"equal"
```

test-if3.fj

```
func int main() {
    int x = 9;
    int y = 8;

    if (x >= y) {
        print("x is larger or equal");
        print(x);
```

```
    }

    if (x <= y) {
        print("x is smaller or equal");
        print(y);
    }

    if (x != y) {
        print("not equal");
    }

    int a = 3;
    int b = 7;

    if (a >= b) {
        print("a is larger or equal");
        print(a);
    }

    if (a <= b) {
        print("a is smaller or equal");
        print(b);
    }

    if (a != b) {
        print("not equal");
    }

    int c = 2;
    int d = 2;

    if (c >= d) {
        print("c is larger or equal");
        print(c);
    }

    if (c <= d) {
        print("c is smaller or equal");
        print(d);
    }

    if (c != d) {
        print("not equal");
    }

}
```

test-if3.out

```
"x is larger or equal"
9
"not equal"
"a is smaller or equal"
7
"not equal"
```

```
"c is larger or equal"
2
"c is smaller or equal"
2
```

test-if4.fj

```
func int main() {
    float x = 9.0001;
    float y = 8.1;

    if (x >= y) {
        print("x is larger or equal");
        print(x);
    }

    if (x <= y) {
        print("x is smaller or equal");
        print(y);
    }

    if (x != y) {
        print("not equal");
    }

    float a = -3.21;
    float b = 1.4;

    if (a >= b) {
        print("a is larger or equal");
        print(a);
    }

    if (a <= b) {
        print("a is smaller or equal");
        print(b);
    }

    if (a != b) {
        print("not equal");
    }

    float c = 2.0;
    float d = 2.0;

    if (c >= d) {
        print("c is larger or equal");
        print(c);
    }

    if (c <= d) {
        print("c is smaller or equal");
        print(d);
    }

```

```
    if (c != d) {
        print("not equal");
    }

}
```

test-if4.out

```
"x is larger or equal"
9.000100
"not equal"
"a is smaller or equal"
1.400000
"not equal"
"c is larger or equal"
2.000000
"c is smaller or equal"
2.000000
```

test-if5.fj

```
func cry printints(int x){
    if (3 > 1) {
        if (x > 2) {
            print(x);
            if (x - 2 > 2) {
                print(x);
            }
        }
        if (x <= 3) {
            print(x+1);
        }
    }
}

func int main() {
    printints(4);
    printints(18);
    printints(2);
    printints(3);
}
```

test-if5.out

```
4
18
18
3
3
4
```

test-ifelse1.fj

```
func int main() {
    if(2>1){
        print("yes");
    }
```

```
    else {
        print("no");
    }

    int x = 4;
    int y = 19;
    if (x > y) {
        print(x);
    }
    else {
        print(y);
    }

    float a = 4.3;
    float b = 1.7;
    if (a > b) {
        print(a);
    }
    else {
        print(b);
    }
}
```

test-ifelse1.out

```
"yes"
19
4.300000
```

test-ifelse2.fj

```
func char grade(int g) {
    char grade = 'Z';
    if (g >= 60) {
        grade = 'D';
        if (g >= 70) {
            grade = 'C';
            if (g >= 80 and g < 90) {
                grade = 'B';
            }
            else {
                if (g >= 90) {
                    grade = 'A';
                }
            }
        }
    }
    else {
        grade = 'F';
    }
    return grade;
}

func int main() {
    print(grade(94));
    print(grade(34));
```

```
    print(grade(65));
    print(grade(70));
}
```

test-ifelse2.out

```
A
F
D
C
```

test-ifelse3.fj

```
func cry foo(float a, float b) {
    if (a > b) {
        if (b > 8.8) {
            print("b is big");
        }
        if (b > 9.8) {
            print("b is very big");
        }
        else {
            print("b may be big, but not very big");
        }
    }
    else {
        if (a == b) {
            print("equal!");
        }
        else {
            print("not equal");
        }
        if (b - a < 0.5) {
            print(a);
        }
        else {
            print(b);
        }
    }
}

func int main() {
    foo(9.8, 9.1);
    foo(8.8, 9.1);
}
```

test-ifelse3.out

```
"b is big"
"b may be big, but not very big"
"not equal"
8.800000
```

test-intarray.fj

```
func int main(){
    array<int, 3> intList = [3, 4, 5];
```

```
    int x = intList[0];
    print(x);
    return 0;
}
```

test-intarray.out

```
3
```

test-intarray2.fj

```
func int main(){
    array<int, 3> all = [5, 8, 9];
    int i;
    int x = 1;
    for(i=0; i<3; i=i+1){
        int change = all[i];
        x += change;
        print(x);
    }
}
```

test-intarray2.out

```
6
14
23
```

test-intarray3.fj

```
func int main() {
    array<int, 5> snd = [2, 4, 6, 8, 10];
    array<int, 5> fst = [1, 3, 5, 7, 9];
    int i;
    int x = 0;
    int y = 0;
    for(i=0; i<5; i++){
        x = snd[i];
        if (x %2 ==0 ){
            print("even");
        }
        y = fst[i];
        if(y % 3 == 0){
            print(y);
            print("odd");
        }
        else {
            print("strange");
        }

    }
    return 0;

}
```

test-intarray3.out

```
"even"
```

```
"strange"
"even"
3
"odd"
"even"
"strange"
"even"
"strange"
"even"
9
"odd"
```

test-intarray4.fj

```
func int stuff(int x){
    if (x %2 == 0){
        return 0;
    }
    return 2;
}

func int main(){
    array<int, 3> iA  = [0, 13, 789];
    int i = 0;
    while (i < 4){
        int index = stuff(5);
        print(iA[index]);
        i++;
    }
    return 0;
}
```

test-intarray4.out

```
789
789
789
789
```

test-intarray5.fj

```
func int oooo(){
    array<int, 5> intArray = [8, 13, 666, 999, 0];
    int i = 0;
    int sum = 0;
    while(i < 5){
        int val = intArray[i];
        sum += val;
        i++;
    }
    return sum;
}


func int main(){
    print(oooo());
}
```

test-intarray5.out

```
1686
```

test-mod1.fj

```
func int main() {
    print(7%2);
    print(9%3);
    print(10%11);

    int a = 8;
    int b = 3;
    int c = a%b;
    print(c);

    print(0%8);
}
```

test-mod1.out

```
1
0
10
2
0
```

test-power.fj

```
func int power(int base, int pow) {
    int result;
    if (base == 0) {
        result = 0;
    }
    else {
        result = 1;
    }
    while (pow > 0) {
        result *= base;
        pow--;
    }
    return result;
}

func int main() {
    int x = power(2, 10);
    print(x);
    print(power(3,3));
    print(power(0,3));
    x = power(9,0);
    print(x);
}
```

test-power.out

```
1024
27
0
1
```

test-print1.fj

```
func int main() {
    print("hello");
    print('c');
    print(4.5);
    print(8);
}
```

test-print1.out

```
"hello"
c
4.500000
8
```

test-print2.fj

```
func int main() {
    print(3+4);
    print(6-3);
    print(4*4);
    print(16/2);
}
```

test-print2.out

```
7
3
16
8
```

test-print3.fj

```
func int main() {
    print(3.1+4.4);
    print(6.23-3.1);
    print(5.0*8.2);
    print(8.4/2.1);
}
```

test-print3.out

```
7.500000
3.130000
41.000000
4.000000
```

test-printbool1.fj

```
func int main() {
    print(1>2);
    print(4==4);
    print(9 >= 9);
    print(true);
    print(false);
}
```

test-printbool1.out

```
false
true
true
true
false
```

test-printbool2.fj

```
func int main() {
    print(true and true);
    print(true and false);
    print(false or true);
    print(not true);
    print(not false);
}
```

test-printbool2.out

```
true
false
true
false
true
```

test-printbool3.fj

```
func int main() {
    bool ba = 4 > 3;
    bool bb = true or true;
    print(ba);
    print(bb and ba);

    bool bc = not ba;
    print(bc);
}
```

test-printbool3.out

```
true
true
false
```

test-return1.fj

```
func int funcyints(int a) {
    int result = 0;
    int b = 0;
    while (b < a) {
        if (result < 1) { result++; }
        result *= b;
        b++;
    }
    return result;
}

func float funcyfloats(float a) {
    float result = 0.0;
    float b = 0.0;
```

```
    while (b < a) {
        if (result < 1.0) { result+=1.0; }
        result *= b;
        b+=1.0;
    }
    return result;
}

func int main() {
    int ia = funcyints(8);
    int ib = funcyints(-2);
    print(ia);
    print(ib);

    float fa = funcyfloats(8.0);
    float fb = funcyfloats(-2.0);
    print(fa);
    print(fb);
}
```

test-return1.out

```
5040
0
5040.000000
0.000000
```

test-return2.fj

```
func char funcychars(int a) {
    char c = '0';
    if (a > 10) {
        c = 'a';
    }
    else {
        c = 'b';
    }
    return c;
}

func str funcystrs(int a) {
    str s = "";
    if (a > 10) {
        s = "BIG";
    }
    else {
        s = "small";
    }
    return s;
}

func int main() {
    char ca = funcychars(8);
    char cb = funcychars(18);
    print(ca);
    print(cb);
```

```
28
29      str sa = funcystrs(8);
30      str sb = funcystrs(18);
31      print(sa);
32      print(sb);
33  }
```

test-return2.out

```
1  b
2  a
3  "small"
4  "BIG"
```

test-return3.fj

```
1   func int retint() {
2       return 8;
3   }
4
5   func float retfloat() {
6       return -4.5;
7   }
8
9   func char retchar() {
10      return 'p';
11  }
12
13  func str retstr() {
14      return "funky!";
15  }
16
17  func int reti(int a) {
18      if (a < 6) {
19          return a;
20      }
21      else {
22          return 6;
23      }
24  }
25
26  func float retf(float a) {
27      if (a < 9.0) {
28          return a;
29      }
30      else {
31          return 9.0;
32      }
33  }
34
35  func char retc(int a) {
36      if (a < 2) {
37          return 'y';
38      }
39      else {
40          return 'n';
```

```
    }
}

func str rets(int a) {
    if (a < 6) {
        return "hello";
    }
    return "goodbye";
}

func int main() {
    print(retint());
    print(retfloat());
    print(retchar());
    print(retstr());

    print(reti(8));
    print(reti(4));

    print(retf(10.00));
    print(retf(-4.41));

    print(retc(1));
    print(retc(4));

    print(rets(8));
    print(rets(4));
}
```

test-return3.out

```
8
-4.500000
p
"funky!"
6
4
9.000000
-4.410000
y
n
"goodbye"
"hello"
```

test-return4.fj

```
func int subints(int a, int b) {
    if (a > b) {
        return a - b;
    }
    else {
        return b - a;
    }
}

func int multints(int a, int b) {
```

```
    while (b > 0) {
        a *= b;
        b--;
    }
    return a;
}

func int main() {
    int a = subints(4,1); /* returns 3 */
    int b = 5;
    int c = multints(a, b);
    print(c);

    int d = subints(c, a*b);
    print(multints(d, b));

    print(subints(subints(4, 5), 10));
    print(multints(multints(2,4), subints(8,2)));
}
```

test-return4.out

```
360
41400
9
34560
```

test-stringarray.fj

```
func int main(){
  array<str, 3> llist = ["hi", "hello", "bye"];
  str x = llist[0];
  print(x);
  return 0;
}
```

test-stringarray.out

```
"hi"
```

test-stringarray2.fj

```
func int main(){
    array<str, 2> ss = ["you", "what"];
    str result;
    int i;
    for(i=0; i < 2; i++){
        print(ss[i]);
    }
    print("?");
}
```

test-stringarray2.out

```
"you"
"what"
"?"
```

test-stringarray3.fj

```
func int main(){
    array<str, 4> st = ["I", "printing","alternating","actually"];
    array<str, 4> ss = ["am", "from","arrays","!"];

    int i = 0;
    while (i<4) {
        print(st[i]);
        print(ss[i]);
        i++;
    }
    return 0;
}
```

test-stringarray3.out

```
"I"
"am"
"printing"
"from"
"alternating"
"arrays"
"actually"
"!"
```

test-stringarray4.fj

```
func int main(){

    array<str, 6> st = ["y !","r a","a r","n g","r i","s t"];
    array<str, 1> spacer = ["."];

    int i;
    for (i=5; i>-1; i--){
        str g = st[i];
        print(g);
        g = spacer[0];
        print(g);
    }
    return 0;
}
```

test-stringarray4.out

```
"s t"
"."
"r i"
"."
"n g"
"."
"a r"
"."
"r a"
"."
"y !"
"."
```

test-stringarray5.fj

```
func int main(){
    array<str, 5> caller = ["right","no","nope","yes","wrong"];
    int i;
    for (i = 0; i < 5; i++) {
        str ss = caller[i];
        print(helper(ss));
    }
    return 0;
}

func str helper(str st) {
    if (st == "right" or st == "yes") {
        return "ok";
    }
    return "not ok";
}
```

test-stringarray5.out

```
"ok"
"not ok"
"not ok"
"ok"
"not ok"
```

test-uop1.fj

```
func int main() {
    print(9);
    print(-9);
    print(-3.4);
    int x = -4;
    print(x);
    int y = - x;
    print(y);
}
```

test-uop1.out

```
9
-9
-3.400000
-4
4
```

test-uop2.fj

```
func int main() {
    float f = 7.5;
    print(negate(f));
}

func float negate(float num) {
    return -num;
}
```

test-uop2.out

```
-7.500000
```

test-uop3.fj

```
func int main() {
    float f = -7.5;
    print(-f);
    print(- -f);
}
```

test-uop3.out

```
7.500000
-7.500000
```

test-var1.fj

```
func int main() {
    int ia;
    ia = 8;
    int ib;
    ib = -9;
    print(ia);
    print(ib);

    float fa;
    fa = 8.9;
    float fb;
    fb = -9.0;
    print(fa);
    print(fb);

    char ca;
    ca = '9';
    char cb;
    cb = 'o';
    print(ca);
    print(cb);

    str sa;
    sa = "FUNC-y JAVA";
    str sb;
    sb = "";
    print(sa);
    print(sb);
}
```

test-var1.out

```
8
-9
8.900000
-9.000000
9
o
"FUNC-y JAVA"
""
```

test-var2.fj

```
func int main() {
    int ia = 7;
    print(ia);
    int ib = -8;
    print(ib);
    int ic = 0;
    print(ic);

    float fa = 7.7;
    print(fa);
    float fb = -4.8;
    print(fb);
    float fc = 0.0;
    print(fc);

    char ca = 'a';
    print(ca);
    char cb = ';';
    print(cb);
    char cc = '&';
    print(cc);


    str sa = "hello world";
    print(sa);
    str sb = "0";
    print(sb);
    str sc = "";
    print(sc);
}
```

test-var2.out

```
7
-8
0
7.700000
-4.800000
0.000000
a
;
&
"hello world"
"0"
""
```

test-var3.fj

```
func int main() {
    int ia = 9;
    print(ia);

    ia = 8;
    print(ia);
```

```
    float fa = 9.9;
    print(fa);

    fa = 2.3;
    print(fa);
    print(ia);
}
```

test-var3.out

```
9
8
9.900000
2.300000
8
```

test-var4.fj

```
func int main() {
    str s;
    char c;
    s = "hello";
    print(s);

    str sa = s;

    s = "bonjour";
    print(s);
    print(sa);

    c = 'e';
    print(c);
    char ca = c;
    c = 'c';
    print(s);
    print(c);
    print(ca);
}
```

test-var4.out

```
"hello"
"bonjour"
"hello"
e
"bonjour"
c
e
```

test-var5.fj

```
func int main() {
    int ia = 9;
    int ib = 7;
    print(ia+ib);
    int ic = ia + ib;
```

```
    print(ic);

    print(ia-ib);
    int id = ia - ib;
    print(id);

    print(ia-4);
    int ie = ia-4;
    print(ie);

    print(ib/3);
    int ig = ib/3;
    print(ig);

    print(ie*8);
    int ih = ie*8;
    print(ih);
}
```

test-var5.out

```
16
16
2
2
5
5
2
2
40
40
```

test-var6.fj

```
func int main() {
    float fa = 8.4;
    float fb = 2.9;
    print(fa+fb);
    float fc = fa + fb;
    print(fc);

    print(fc-fb);
    float fd = fc-fb;
    print(fd);

    print(fa-4.0);
    float fe = fa-4.0;
    print(fe);

    print(fe/2.2);
    float fg = fe/2.2;
    print(fg);

    print(fg*fc);
    float fh = fg*fc;
    print(fh);
```

```
}
```

test-var6.out

```
11.300000
11.300000
8.400000
8.400000
4.400000
4.400000
2.000000
2.000000
22.600000
22.600000
```

test-var7.fj

```
func int main() {
    int a = 9;
    print(a);
    int b;
    int c = 10;
    b = a + c;
    print(b);
    a = 4;
    print(a);
    print(b);
}
```

test-var7.out

```
9
19
4
19
```

test-while1.fj

```
func int main(){
    int i = 0;
    while (i < 6){
        print(i);
        i++;
    }

    i = 80;
    while (i >= 70){
        print(i);
        i--;
    }
}
```

test-while1.out

```
0
1
2
3
```

```
4
5
80
79
78
77
76
75
74
73
72
71
70
```

test-while2.fj

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

    result = 1;
    i = 6;
    while (i > 0) {
        result = result * i;
        i--;
    }
    print(result);

    result = 10;
    i = 1;
    while (i < 4) {
        result -= i;
        i++;
    }
    print(result);
}
```

test-while2.out

```
15
720
4
```

test-while3.fj

```
func int main(){
    int i = 1;
    while (i < 7) {
        int j = 1;
        while (j < 7){
            print(i + j);
            j += 2;
```

```
        }
        i+= 2;
    }
}
```

test-while3.out

```
2
4
6
4
6
8
6
8
10
```

test-while4.fj

```
func int main(){
    float result = 0.0;
    while (result < 8.0) {
        result += 0.5;
        print(result);
    }

    result = 100.0;
    float fl = 0.2;
    while (fl < 2.0) {
        result = result * fl;
        fl *= 2.0;
    }
    print(result);
}
```

test-while4.out

```
0.500000
1.000000
1.500000
2.000000
2.500000
3.000000
3.500000
4.000000
4.500000
5.000000
5.500000
6.000000
6.500000
7.000000
7.500000
8.000000
10.240000
```

7.4 Test Log

```
/usr/bin/lli

###### Testing test-assnop1
_build/funcyjava.native tests/test-assnop1.fj > test-assnop1.ll
llc -relocation-model=pic test-assnop1.ll > test-assnop1.s
cc -o test-assnop1.exe test-assnop1.s
./test-assnop1.exe
diff -b test-assnop1.out tests/test-assnop1.out > test-assnop1.diff
###### SUCCESS

###### Testing test-assnop2
_build/funcyjava.native tests/test-assnop2.fj > test-assnop2.ll
llc -relocation-model=pic test-assnop2.ll > test-assnop2.s
cc -o test-assnop2.exe test-assnop2.s
./test-assnop2.exe
diff -b test-assnop2.out tests/test-assnop2.out > test-assnop2.diff
###### SUCCESS

###### Testing test-binop1
_build/funcyjava.native tests/test-binop1.fj > test-binop1.ll
llc -relocation-model=pic test-binop1.ll > test-binop1.s
cc -o test-binop1.exe test-binop1.s
./test-binop1.exe
diff -b test-binop1.out tests/test-binop1.out > test-binop1.diff
###### SUCCESS

###### Testing test-binop2
_build/funcyjava.native tests/test-binop2.fj > test-binop2.ll
llc -relocation-model=pic test-binop2.ll > test-binop2.s
cc -o test-binop2.exe test-binop2.s
./test-binop2.exe
diff -b test-binop2.out tests/test-binop2.out > test-binop2.diff
###### SUCCESS

###### Testing test-boolop-predicate1
_build/funcyjava.native tests/test-boolop-predicate1.fj > test-boolop-
   predicate1.ll
llc -relocation-model=pic test-boolop-predicate1.ll > test-boolop-
   predicate1.s
cc -o test-boolop-predicate1.exe test-boolop-predicate1.s
./test-boolop-predicate1.exe
diff -b test-boolop-predicate1.out tests/test-boolop-predicate1.out > test
   -boolop-predicate1.diff
###### SUCCESS

###### Testing test-boolop-predicate2
_build/funcyjava.native tests/test-boolop-predicate2.fj > test-boolop-
   predicate2.ll
llc -relocation-model=pic test-boolop-predicate2.ll > test-boolop-
   predicate2.s
cc -o test-boolop-predicate2.exe test-boolop-predicate2.s
./test-boolop-predicate2.exe
```

```
diff -b test-boolop-predicate2.out tests/test-boolop-predicate2.out > test-boolop-predicate2.diff
###### SUCCESS

###### Testing test-boolop-predicate3
_build/funcyjava.native tests/test-boolop-predicate3.fj > test-boolop-predicate3.ll
llc -relocation-model=pic test-boolop-predicate3.ll > test-boolop-predicate3.s
cc -o test-boolop-predicate3.exe test-boolop-predicate3.s
./test-boolop-predicate3.exe
diff -b test-boolop-predicate3.out tests/test-boolop-predicate3.out > test-boolop-predicate3.diff
###### SUCCESS

###### Testing test-boolop-predicate4
_build/funcyjava.native tests/test-boolop-predicate4.fj > test-boolop-predicate4.ll
llc -relocation-model=pic test-boolop-predicate4.ll > test-boolop-predicate4.s
cc -o test-boolop-predicate4.exe test-boolop-predicate4.s
./test-boolop-predicate4.exe
diff -b test-boolop-predicate4.out tests/test-boolop-predicate4.out > test-boolop-predicate4.diff
###### SUCCESS

###### Testing test-comments1
_build/funcyjava.native tests/test-comments1.fj > test-comments1.ll
llc -relocation-model=pic test-comments1.ll > test-comments1.s
cc -o test-comments1.exe test-comments1.s
./test-comments1.exe
diff -b test-comments1.out tests/test-comments1.out > test-comments1.diff
###### SUCCESS

###### Testing test-comments2
_build/funcyjava.native tests/test-comments2.fj > test-comments2.ll
llc -relocation-model=pic test-comments2.ll > test-comments2.s
cc -o test-comments2.exe test-comments2.s
./test-comments2.exe
diff -b test-comments2.out tests/test-comments2.out > test-comments2.diff
###### SUCCESS

###### Testing test-demo1
_build/funcyjava.native tests/test-demo1.fj > test-demo1.ll
llc -relocation-model=pic test-demo1.ll > test-demo1.s
cc -o test-demo1.exe test-demo1.s
./test-demo1.exe
diff -b test-demo1.out tests/test-demo1.out > test-demo1.diff
###### SUCCESS

###### Testing test-demo2
_build/funcyjava.native tests/test-demo2.fj > test-demo2.ll
llc -relocation-model=pic test-demo2.ll > test-demo2.s
cc -o test-demo2.exe test-demo2.s
```

```
./test-demo2.exe
diff -b test-demo2.out tests/test-demo2.out > test-demo2.diff
###### SUCCESS

###### Testing test-demo3
_build/funcyjava.native tests/test-demo3.fj > test-demo3.ll
llc -relocation-model=pic test-demo3.ll > test-demo3.s
cc -o test-demo3.exe test-demo3.s
./test-demo3.exe
diff -b test-demo3.out tests/test-demo3.out > test-demo3.diff
###### SUCCESS

###### Testing test-equalop1
_build/funcyjava.native tests/test-equalop1.fj > test-equalop1.ll
llc -relocation-model=pic test-equalop1.ll > test-equalop1.s
cc -o test-equalop1.exe test-equalop1.s
./test-equalop1.exe
diff -b test-equalop1.out tests/test-equalop1.out > test-equalop1.diff
###### SUCCESS

###### Testing test-equalop2
_build/funcyjava.native tests/test-equalop2.fj > test-equalop2.ll
llc -relocation-model=pic test-equalop2.ll > test-equalop2.s
cc -o test-equalop2.exe test-equalop2.s
./test-equalop2.exe
diff -b test-equalop2.out tests/test-equalop2.out > test-equalop2.diff
###### SUCCESS

###### Testing test-equalop3
_build/funcyjava.native tests/test-equalop3.fj > test-equalop3.ll
llc -relocation-model=pic test-equalop3.ll > test-equalop3.s
cc -o test-equalop3.exe test-equalop3.s
./test-equalop3.exe
diff -b test-equalop3.out tests/test-equalop3.out > test-equalop3.diff
###### SUCCESS

###### Testing test-fact1
_build/funcyjava.native tests/test-fact1.fj > test-fact1.ll
llc -relocation-model=pic test-fact1.ll > test-fact1.s
cc -o test-fact1.exe test-fact1.s
./test-fact1.exe
diff -b test-fact1.out tests/test-fact1.out > test-fact1.diff
###### SUCCESS

###### Testing test-fib1
_build/funcyjava.native tests/test-fib1.fj > test-fib1.ll
llc -relocation-model=pic test-fib1.ll > test-fib1.s
cc -o test-fib1.exe test-fib1.s
./test-fib1.exe
diff -b test-fib1.out tests/test-fib1.out > test-fib1.diff
###### SUCCESS

###### Testing test-floatarray
_build/funcyjava.native tests/test-floatarray.fj > test-floatarray.ll
```

```
llc -relocation-model=pic test-floatarray.ll > test-floatarray.s
cc -o test-floatarray.exe test-floatarray.s
./test-floatarray.exe
diff -b test-floatarray.out tests/test-floatarray.out > test-floatarray.
    diff
###### SUCCESS

###### Testing test-floatarray2
_build/funcyjava.native tests/test-floatarray2.fj > test-floatarray2.ll
llc -relocation-model=pic test-floatarray2.ll > test-floatarray2.s
cc -o test-floatarray2.exe test-floatarray2.s
./test-floatarray2.exe
diff -b test-floatarray2.out tests/test-floatarray2.out > test-floatarray2
    .diff
###### SUCCESS

###### Testing test-floatarray3
_build/funcyjava.native tests/test-floatarray3.fj > test-floatarray3.ll
llc -relocation-model=pic test-floatarray3.ll > test-floatarray3.s
cc -o test-floatarray3.exe test-floatarray3.s
./test-floatarray3.exe
diff -b test-floatarray3.out tests/test-floatarray3.out > test-floatarray3
    .diff
###### SUCCESS

###### Testing test-floatarray4
_build/funcyjava.native tests/test-floatarray4.fj > test-floatarray4.ll
llc -relocation-model=pic test-floatarray4.ll > test-floatarray4.s
cc -o test-floatarray4.exe test-floatarray4.s
./test-floatarray4.exe
diff -b test-floatarray4.out tests/test-floatarray4.out > test-floatarray4
    .diff
###### SUCCESS

###### Testing test-floatarray5
_build/funcyjava.native tests/test-floatarray5.fj > test-floatarray5.ll
llc -relocation-model=pic test-floatarray5.ll > test-floatarray5.s
cc -o test-floatarray5.exe test-floatarray5.s
./test-floatarray5.exe
diff -b test-floatarray5.out tests/test-floatarray5.out > test-floatarray5
    .diff
###### SUCCESS

###### Testing test-for1
_build/funcyjava.native tests/test-for1.fj > test-for1.ll
llc -relocation-model=pic test-for1.ll > test-for1.s
cc -o test-for1.exe test-for1.s
./test-for1.exe
diff -b test-for1.out tests/test-for1.out > test-for1.diff
###### SUCCESS

###### Testing test-for2
_build/funcyjava.native tests/test-for2.fj > test-for2.ll
llc -relocation-model=pic test-for2.ll > test-for2.s
```

```
cc -o test-for2.exe test-for2.s
./test-for2.exe
diff -b test-for2.out tests/test-for2.out > test-for2.diff
###### SUCCESS

###### Testing test-for3
_build/funcyjava.native tests/test-for3.fj > test-for3.ll
llc -relocation-model=pic test-for3.ll > test-for3.s
cc -o test-for3.exe test-for3.s
./test-for3.exe
diff -b test-for3.out tests/test-for3.out > test-for3.diff
###### SUCCESS

###### Testing test-for4
_build/funcyjava.native tests/test-for4.fj > test-for4.ll
llc -relocation-model=pic test-for4.ll > test-for4.s
cc -o test-for4.exe test-for4.s
./test-for4.exe
diff -b test-for4.out tests/test-for4.out > test-for4.diff
###### SUCCESS

###### Testing test-for6
_build/funcyjava.native tests/test-for6.fj > test-for6.ll
llc -relocation-model=pic test-for6.ll > test-for6.s
cc -o test-for6.exe test-for6.s
./test-for6.exe
diff -b test-for6.out tests/test-for6.out > test-for6.diff
###### SUCCESS

###### Testing test-forwhile1
_build/funcyjava.native tests/test-forwhile1.fj > test-forwhile1.ll
llc -relocation-model=pic test-forwhile1.ll > test-forwhile1.s
cc -o test-forwhile1.exe test-forwhile1.s
./test-forwhile1.exe
diff -b test-forwhile1.out tests/test-forwhile1.out > test-forwhile1.diff
###### SUCCESS

###### Testing test-forwhile2
_build/funcyjava.native tests/test-forwhile2.fj > test-forwhile2.ll
llc -relocation-model=pic test-forwhile2.ll > test-forwhile2.s
cc -o test-forwhile2.exe test-forwhile2.s
./test-forwhile2.exe
diff -b test-forwhile2.out tests/test-forwhile2.out > test-forwhile2.diff
###### SUCCESS

###### Testing test-func1
_build/funcyjava.native tests/test-func1.fj > test-func1.ll
llc -relocation-model=pic test-func1.ll > test-func1.s
cc -o test-func1.exe test-func1.s
./test-func1.exe
diff -b test-func1.out tests/test-func1.out > test-func1.diff
###### SUCCESS

###### Testing test-func2
```

```
_build/funcyjava.native tests/test-func2.fj > test-func2.ll
llc -relocation-model=pic test-func2.ll > test-func2.s
cc -o test-func2.exe test-func2.s
./test-func2.exe
diff -b test-func2.out tests/test-func2.out > test-func2.diff
###### SUCCESS

###### Testing test-func3
_build/funcyjava.native tests/test-func3.fj > test-func3.ll
llc -relocation-model=pic test-func3.ll > test-func3.s
cc -o test-func3.exe test-func3.s
./test-func3.exe
diff -b test-func3.out tests/test-func3.out > test-func3.diff
###### SUCCESS

###### Testing test-func4
_build/funcyjava.native tests/test-func4.fj > test-func4.ll
llc -relocation-model=pic test-func4.ll > test-func4.s
cc -o test-func4.exe test-func4.s
./test-func4.exe
diff -b test-func4.out tests/test-func4.out > test-func4.diff
###### SUCCESS

###### Testing test-func5
_build/funcyjava.native tests/test-func5.fj > test-func5.ll
llc -relocation-model=pic test-func5.ll > test-func5.s
cc -o test-func5.exe test-func5.s
./test-func5.exe
diff -b test-func5.out tests/test-func5.out > test-func5.diff
###### SUCCESS

###### Testing test-func6
_build/funcyjava.native tests/test-func6.fj > test-func6.ll
llc -relocation-model=pic test-func6.ll > test-func6.s
cc -o test-func6.exe test-func6.s
./test-func6.exe
diff -b test-func6.out tests/test-func6.out > test-func6.diff
###### SUCCESS

###### Testing test-gcd
_build/funcyjava.native tests/test-gcd.fj > test-gcd.ll
llc -relocation-model=pic test-gcd.ll > test-gcd.s
cc -o test-gcd.exe test-gcd.s
./test-gcd.exe
diff -b test-gcd.out tests/test-gcd.out > test-gcd.diff
###### SUCCESS

###### Testing test-if1
_build/funcyjava.native tests/test-if1.fj > test-if1.ll
llc -relocation-model=pic test-if1.ll > test-if1.s
cc -o test-if1.exe test-if1.s
./test-if1.exe
diff -b test-if1.out tests/test-if1.out > test-if1.diff
###### SUCCESS
```

```
306
307 ###### Testing test-if2
308 _build/funcyjava.native tests/test-if2.fj > test-if2.ll
309 llc -relocation-model=pic test-if2.ll > test-if2.s
310 cc -o test-if2.exe test-if2.s
311 ./test-if2.exe
312 diff -b test-if2.out tests/test-if2.out > test-if2.diff
313 ###### SUCCESS
314
315 ###### Testing test-if3
316 _build/funcyjava.native tests/test-if3.fj > test-if3.ll
317 llc -relocation-model=pic test-if3.ll > test-if3.s
318 cc -o test-if3.exe test-if3.s
319 ./test-if3.exe
320 diff -b test-if3.out tests/test-if3.out > test-if3.diff
321 ###### SUCCESS
322
323 ###### Testing test-if4
324 _build/funcyjava.native tests/test-if4.fj > test-if4.ll
325 llc -relocation-model=pic test-if4.ll > test-if4.s
326 cc -o test-if4.exe test-if4.s
327 ./test-if4.exe
328 diff -b test-if4.out tests/test-if4.out > test-if4.diff
329 ###### SUCCESS
330
331 ###### Testing test-if5
332 _build/funcyjava.native tests/test-if5.fj > test-if5.ll
333 llc -relocation-model=pic test-if5.ll > test-if5.s
334 cc -o test-if5.exe test-if5.s
335 ./test-if5.exe
336 diff -b test-if5.out tests/test-if5.out > test-if5.diff
337 ###### SUCCESS
338
339 ###### Testing test-ifelse1
340 _build/funcyjava.native tests/test-ifelse1.fj > test-ifelse1.ll
341 llc -relocation-model=pic test-ifelse1.ll > test-ifelse1.s
342 cc -o test-ifelse1.exe test-ifelse1.s
343 ./test-ifelse1.exe
344 diff -b test-ifelse1.out tests/test-ifelse1.out > test-ifelse1.diff
345 ###### SUCCESS
346
347 ###### Testing test-ifelse2
348 _build/funcyjava.native tests/test-ifelse2.fj > test-ifelse2.ll
349 llc -relocation-model=pic test-ifelse2.ll > test-ifelse2.s
350 cc -o test-ifelse2.exe test-ifelse2.s
351 ./test-ifelse2.exe
352 diff -b test-ifelse2.out tests/test-ifelse2.out > test-ifelse2.diff
353 ###### SUCCESS
354
355 ###### Testing test-ifelse3
356 _build/funcyjava.native tests/test-ifelse3.fj > test-ifelse3.ll
357 llc -relocation-model=pic test-ifelse3.ll > test-ifelse3.s
358 cc -o test-ifelse3.exe test-ifelse3.s
359 ./test-ifelse3.exe
```

```
diff -b test-ifelse3.out tests/test-ifelse3.out > test-ifelse3.diff
###### SUCCESS

###### Testing test-intarray
_build/funcyjava.native tests/test-intarray.fj > test-intarray.ll
llc -relocation-model=pic test-intarray.ll > test-intarray.s
cc -o test-intarray.exe test-intarray.s
./test-intarray.exe
diff -b test-intarray.out tests/test-intarray.out > test-intarray.diff
###### SUCCESS

###### Testing test-intarray2
_build/funcyjava.native tests/test-intarray2.fj > test-intarray2.ll
llc -relocation-model=pic test-intarray2.ll > test-intarray2.s
cc -o test-intarray2.exe test-intarray2.s
./test-intarray2.exe
diff -b test-intarray2.out tests/test-intarray2.out > test-intarray2.diff
###### SUCCESS

###### Testing test-intarray3
_build/funcyjava.native tests/test-intarray3.fj > test-intarray3.ll
llc -relocation-model=pic test-intarray3.ll > test-intarray3.s
cc -o test-intarray3.exe test-intarray3.s
./test-intarray3.exe
diff -b test-intarray3.out tests/test-intarray3.out > test-intarray3.diff
###### SUCCESS

###### Testing test-intarray4
_build/funcyjava.native tests/test-intarray4.fj > test-intarray4.ll
llc -relocation-model=pic test-intarray4.ll > test-intarray4.s
cc -o test-intarray4.exe test-intarray4.s
./test-intarray4.exe
diff -b test-intarray4.out tests/test-intarray4.out > test-intarray4.diff
###### SUCCESS

###### Testing test-intarray5
_build/funcyjava.native tests/test-intarray5.fj > test-intarray5.ll
llc -relocation-model=pic test-intarray5.ll > test-intarray5.s
cc -o test-intarray5.exe test-intarray5.s
./test-intarray5.exe
diff -b test-intarray5.out tests/test-intarray5.out > test-intarray5.diff
###### SUCCESS

###### Testing test-mod1
_build/funcyjava.native tests/test-mod1.fj > test-mod1.ll
llc -relocation-model=pic test-mod1.ll > test-mod1.s
cc -o test-mod1.exe test-mod1.s
./test-mod1.exe
diff -b test-mod1.out tests/test-mod1.out > test-mod1.diff
###### SUCCESS

###### Testing test-power
_build/funcyjava.native tests/test-power.fj > test-power.ll
llc -relocation-model=pic test-power.ll > test-power.s
```

```
cc -o test-power.exe test-power.s
./test-power.exe
diff -b test-power.out tests/test-power.out > test-power.diff
###### SUCCESS

###### Testing test-print1
_build/funcyjava.native tests/test-print1.fj > test-print1.ll
llc -relocation-model=pic test-print1.ll > test-print1.s
cc -o test-print1.exe test-print1.s
./test-print1.exe
diff -b test-print1.out tests/test-print1.out > test-print1.diff
###### SUCCESS

###### Testing test-print2
_build/funcyjava.native tests/test-print2.fj > test-print2.ll
llc -relocation-model=pic test-print2.ll > test-print2.s
cc -o test-print2.exe test-print2.s
./test-print2.exe
diff -b test-print2.out tests/test-print2.out > test-print2.diff
###### SUCCESS

###### Testing test-print3
_build/funcyjava.native tests/test-print3.fj > test-print3.ll
llc -relocation-model=pic test-print3.ll > test-print3.s
cc -o test-print3.exe test-print3.s
./test-print3.exe
diff -b test-print3.out tests/test-print3.out > test-print3.diff
###### SUCCESS

###### Testing test-printbool1
_build/funcyjava.native tests/test-printbool1.fj > test-printbool1.ll
llc -relocation-model=pic test-printbool1.ll > test-printbool1.s
cc -o test-printbool1.exe test-printbool1.s
./test-printbool1.exe
diff -b test-printbool1.out tests/test-printbool1.out > test-printbool1.
   diff
###### SUCCESS

###### Testing test-printbool2
_build/funcyjava.native tests/test-printbool2.fj > test-printbool2.ll
llc -relocation-model=pic test-printbool2.ll > test-printbool2.s
cc -o test-printbool2.exe test-printbool2.s
./test-printbool2.exe
diff -b test-printbool2.out tests/test-printbool2.out > test-printbool2.
   diff
###### SUCCESS

###### Testing test-printbool3
_build/funcyjava.native tests/test-printbool3.fj > test-printbool3.ll
llc -relocation-model=pic test-printbool3.ll > test-printbool3.s
cc -o test-printbool3.exe test-printbool3.s
./test-printbool3.exe
diff -b test-printbool3.out tests/test-printbool3.out > test-printbool3.
   diff
```

```
###### SUCCESS

###### Testing test-return1
_build/funcyjava.native tests/test-return1.fj > test-return1.ll
llc -relocation-model=pic test-return1.ll > test-return1.s
cc -o test-return1.exe test-return1.s
./test-return1.exe
diff -b test-return1.out tests/test-return1.out > test-return1.diff
###### SUCCESS

###### Testing test-return2
_build/funcyjava.native tests/test-return2.fj > test-return2.ll
llc -relocation-model=pic test-return2.ll > test-return2.s
cc -o test-return2.exe test-return2.s
./test-return2.exe
diff -b test-return2.out tests/test-return2.out > test-return2.diff
###### SUCCESS

###### Testing test-return3
_build/funcyjava.native tests/test-return3.fj > test-return3.ll
llc -relocation-model=pic test-return3.ll > test-return3.s
cc -o test-return3.exe test-return3.s
./test-return3.exe
diff -b test-return3.out tests/test-return3.out > test-return3.diff
###### SUCCESS

###### Testing test-return4
_build/funcyjava.native tests/test-return4.fj > test-return4.ll
llc -relocation-model=pic test-return4.ll > test-return4.s
cc -o test-return4.exe test-return4.s
./test-return4.exe
diff -b test-return4.out tests/test-return4.out > test-return4.diff
###### SUCCESS

###### Testing test-stringarray
_build/funcyjava.native tests/test-stringarray.fj > test-stringarray.ll
llc -relocation-model=pic test-stringarray.ll > test-stringarray.s
cc -o test-stringarray.exe test-stringarray.s
./test-stringarray.exe
diff -b test-stringarray.out tests/test-stringarray.out > test-stringarray
    .diff
###### SUCCESS

###### Testing test-stringarray2
_build/funcyjava.native tests/test-stringarray2.fj > test-stringarray2.ll
llc -relocation-model=pic test-stringarray2.ll > test-stringarray2.s
cc -o test-stringarray2.exe test-stringarray2.s
./test-stringarray2.exe
diff -b test-stringarray2.out tests/test-stringarray2.out > test-
    stringarray2.diff
###### SUCCESS

###### Testing test-stringarray3
_build/funcyjava.native tests/test-stringarray3.fj > test-stringarray3.ll
```

```
llc -relocation-model=pic test-stringarray3.ll > test-stringarray3.s
cc -o test-stringarray3.exe test-stringarray3.s
./test-stringarray3.exe
diff -b test-stringarray3.out tests/test-stringarray3.out > test-
   stringarray3.diff
###### SUCCESS

###### Testing test-stringarray4
_build/funcyjava.native tests/test-stringarray4.fj > test-stringarray4.ll
llc -relocation-model=pic test-stringarray4.ll > test-stringarray4.s
cc -o test-stringarray4.exe test-stringarray4.s
./test-stringarray4.exe
diff -b test-stringarray4.out tests/test-stringarray4.out > test-
   stringarray4.diff
###### SUCCESS

###### Testing test-stringarray5
_build/funcyjava.native tests/test-stringarray5.fj > test-stringarray5.ll
llc -relocation-model=pic test-stringarray5.ll > test-stringarray5.s
cc -o test-stringarray5.exe test-stringarray5.s
./test-stringarray5.exe
diff -b test-stringarray5.out tests/test-stringarray5.out > test-
   stringarray5.diff
###### SUCCESS

###### Testing test-uop1
_build/funcyjava.native tests/test-uop1.fj > test-uop1.ll
llc -relocation-model=pic test-uop1.ll > test-uop1.s
cc -o test-uop1.exe test-uop1.s
./test-uop1.exe
diff -b test-uop1.out tests/test-uop1.out > test-uop1.diff
###### SUCCESS

###### Testing test-uop2
_build/funcyjava.native tests/test-uop2.fj > test-uop2.ll
llc -relocation-model=pic test-uop2.ll > test-uop2.s
cc -o test-uop2.exe test-uop2.s
./test-uop2.exe
diff -b test-uop2.out tests/test-uop2.out > test-uop2.diff
###### SUCCESS

###### Testing test-uop3
_build/funcyjava.native tests/test-uop3.fj > test-uop3.ll
llc -relocation-model=pic test-uop3.ll > test-uop3.s
cc -o test-uop3.exe test-uop3.s
./test-uop3.exe
diff -b test-uop3.out tests/test-uop3.out > test-uop3.diff
###### SUCCESS

###### Testing test-var1
_build/funcyjava.native tests/test-var1.fj > test-var1.ll
llc -relocation-model=pic test-var1.ll > test-var1.s
cc -o test-var1.exe test-var1.s
./test-var1.exe
```

```
diff -b test-var1.out tests/test-var1.out > test-var1.diff
###### SUCCESS

###### Testing test-var2
_build/funcyjava.native tests/test-var2.fj > test-var2.ll
llc -relocation-model=pic test-var2.ll > test-var2.s
cc -o test-var2.exe test-var2.s
./test-var2.exe
diff -b test-var2.out tests/test-var2.out > test-var2.diff
###### SUCCESS

###### Testing test-var3
_build/funcyjava.native tests/test-var3.fj > test-var3.ll
llc -relocation-model=pic test-var3.ll > test-var3.s
cc -o test-var3.exe test-var3.s
./test-var3.exe
diff -b test-var3.out tests/test-var3.out > test-var3.diff
###### SUCCESS

###### Testing test-var4
_build/funcyjava.native tests/test-var4.fj > test-var4.ll
llc -relocation-model=pic test-var4.ll > test-var4.s
cc -o test-var4.exe test-var4.s
./test-var4.exe
diff -b test-var4.out tests/test-var4.out > test-var4.diff
###### SUCCESS

###### Testing test-var5
_build/funcyjava.native tests/test-var5.fj > test-var5.ll
llc -relocation-model=pic test-var5.ll > test-var5.s
cc -o test-var5.exe test-var5.s
./test-var5.exe
diff -b test-var5.out tests/test-var5.out > test-var5.diff
###### SUCCESS

###### Testing test-var6
_build/funcyjava.native tests/test-var6.fj > test-var6.ll
llc -relocation-model=pic test-var6.ll > test-var6.s
cc -o test-var6.exe test-var6.s
./test-var6.exe
diff -b test-var6.out tests/test-var6.out > test-var6.diff
###### SUCCESS

###### Testing test-var7
_build/funcyjava.native tests/test-var7.fj > test-var7.ll
llc -relocation-model=pic test-var7.ll > test-var7.s
cc -o test-var7.exe test-var7.s
./test-var7.exe
diff -b test-var7.out tests/test-var7.out > test-var7.diff
###### SUCCESS

###### Testing test-while1
_build/funcyjava.native tests/test-while1.fj > test-while1.ll
llc -relocation-model=pic test-while1.ll > test-while1.s
```

```
cc -o test-while1.exe test-while1.s
./test-while1.exe
diff -b test-while1.out tests/test-while1.out > test-while1.diff
###### SUCCESS

###### Testing test-while2
_build/funcyjava.native tests/test-while2.fj > test-while2.ll
llc -relocation-model=pic test-while2.ll > test-while2.s
cc -o test-while2.exe test-while2.s
./test-while2.exe
diff -b test-while2.out tests/test-while2.out > test-while2.diff
###### SUCCESS

###### Testing test-while3
_build/funcyjava.native tests/test-while3.fj > test-while3.ll
llc -relocation-model=pic test-while3.ll > test-while3.s
cc -o test-while3.exe test-while3.s
./test-while3.exe
diff -b test-while3.out tests/test-while3.out > test-while3.diff
###### SUCCESS

###### Testing test-while4
_build/funcyjava.native tests/test-while4.fj > test-while4.ll
llc -relocation-model=pic test-while4.ll > test-while4.s
cc -o test-while4.exe test-while4.s
./test-while4.exe
diff -b test-while4.out tests/test-while4.out > test-while4.diff
###### SUCCESS

###### Testing fail-assign1
_build/funcyjava.native < tests/fail-assign1.fj 2> fail-assign1.err >>
   testall.log
diff -b fail-assign1.err tests/fail-assign1.err > fail-assign1.diff
###### SUCCESS

###### Testing fail-assign2
_build/funcyjava.native < tests/fail-assign2.fj 2> fail-assign2.err >>
   testall.log
diff -b fail-assign2.err tests/fail-assign2.err > fail-assign2.diff
###### SUCCESS

###### Testing fail-assign3
_build/funcyjava.native < tests/fail-assign3.fj 2> fail-assign3.err >>
   testall.log
diff -b fail-assign3.err tests/fail-assign3.err > fail-assign3.diff
###### SUCCESS

###### Testing fail-assign4
_build/funcyjava.native < tests/fail-assign4.fj 2> fail-assign4.err >>
   testall.log
diff -b fail-assign4.err tests/fail-assign4.err > fail-assign4.diff
###### SUCCESS

###### Testing fail-assignop1
```

```
_build/funcyjava.native < tests/fail-assignop1.fj 2> fail-assignop1.err >> testall.log
diff -b fail-assignop1.err tests/fail-assignop1.err > fail-assignop1.diff
###### SUCCESS

###### Testing fail-assignop2
_build/funcyjava.native < tests/fail-assignop2.fj 2> fail-assignop2.err >> testall.log
diff -b fail-assignop2.err tests/fail-assignop2.err > fail-assignop2.diff
###### SUCCESS

###### Testing fail-assignop3
_build/funcyjava.native < tests/fail-assignop3.fj 2> fail-assignop3.err >> testall.log
diff -b fail-assignop3.err tests/fail-assignop3.err > fail-assignop3.diff
###### SUCCESS

###### Testing fail-assignop4
_build/funcyjava.native < tests/fail-assignop4.fj 2> fail-assignop4.err >> testall.log
diff -b fail-assignop4.err tests/fail-assignop4.err > fail-assignop4.diff
###### SUCCESS

###### Testing fail-binop1
_build/funcyjava.native < tests/fail-binop1.fj 2> fail-binop1.err >> testall.log
diff -b fail-binop1.err tests/fail-binop1.err > fail-binop1.diff
###### SUCCESS

###### Testing fail-binop2
_build/funcyjava.native < tests/fail-binop2.fj 2> fail-binop2.err >> testall.log
diff -b fail-binop2.err tests/fail-binop2.err > fail-binop2.diff
###### SUCCESS

###### Testing fail-binop3
_build/funcyjava.native < tests/fail-binop3.fj 2> fail-binop3.err >> testall.log
diff -b fail-binop3.err tests/fail-binop3.err > fail-binop3.diff
###### SUCCESS

###### Testing fail-binop4
_build/funcyjava.native < tests/fail-binop4.fj 2> fail-binop4.err >> testall.log
diff -b fail-binop4.err tests/fail-binop4.err > fail-binop4.diff
###### SUCCESS

###### Testing fail-binop5
_build/funcyjava.native < tests/fail-binop5.fj 2> fail-binop5.err >> testall.log
diff -b fail-binop5.err tests/fail-binop5.err > fail-binop5.diff
###### SUCCESS

###### Testing fail-boolop1
```

```
_build/funcyjava.native < tests/fail-boolop1.fj 2> fail-boolop1.err >>
   testall.log
diff -b fail-boolop1.err tests/fail-boolop1.err > fail-boolop1.diff
###### SUCCESS

###### Testing fail-boolop2
_build/funcyjava.native < tests/fail-boolop2.fj 2> fail-boolop2.err >>
   testall.log
diff -b fail-boolop2.err tests/fail-boolop2.err > fail-boolop2.diff
###### SUCCESS

###### Testing fail-boolop3
_build/funcyjava.native < tests/fail-boolop3.fj 2> fail-boolop3.err >>
   testall.log
diff -b fail-boolop3.err tests/fail-boolop3.err > fail-boolop3.diff
###### SUCCESS

###### Testing fail-equalop1
_build/funcyjava.native < tests/fail-equalop1.fj 2> fail-equalop1.err >>
   testall.log
diff -b fail-equalop1.err tests/fail-equalop1.err > fail-equalop1.diff
###### SUCCESS

###### Testing fail-equalop2
_build/funcyjava.native < tests/fail-equalop2.fj 2> fail-equalop2.err >>
   testall.log
diff -b fail-equalop2.err tests/fail-equalop2.err > fail-equalop2.diff
###### SUCCESS

###### Testing fail-floatarray
_build/funcyjava.native < tests/fail-floatarray.fj 2> fail-floatarray.err
   >> testall.log
diff -b fail-floatarray.err tests/fail-floatarray.err > fail-floatarray.
   diff
###### SUCCESS

###### Testing fail-floatarray2
_build/funcyjava.native < tests/fail-floatarray2.fj 2> fail-floatarray2.
   err >> testall.log
diff -b fail-floatarray2.err tests/fail-floatarray2.err > fail-floatarray2
   .diff
###### SUCCESS

###### Testing fail-floatarray3
_build/funcyjava.native < tests/fail-floatarray3.fj 2> fail-floatarray3.
   err >> testall.log
diff -b fail-floatarray3.err tests/fail-floatarray3.err > fail-floatarray3
   .diff
###### SUCCESS

###### Testing fail-floatarray4
_build/funcyjava.native < tests/fail-floatarray4.fj 2> fail-floatarray4.
   err >> testall.log
diff -b fail-floatarray4.err tests/fail-floatarray4.err > fail-floatarray4
```

```
    .diff
###### SUCCESS

###### Testing fail-floatarray5
_build/funcyjava.native < tests/fail-floatarray5.fj 2> fail-floatarray5.
    err >> testall.log
diff -b fail-floatarray5.err tests/fail-floatarray5.err > fail-floatarray5
    .diff
###### SUCCESS

###### Testing fail-floatarray6
_build/funcyjava.native < tests/fail-floatarray6.fj 2> fail-floatarray6.
    err >> testall.log
diff -b fail-floatarray6.err tests/fail-floatarray6.err > fail-floatarray6
    .diff
###### SUCCESS

###### Testing fail-for1
_build/funcyjava.native < tests/fail-for1.fj 2> fail-for1.err >> testall.
    log
diff -b fail-for1.err tests/fail-for1.err > fail-for1.diff
###### SUCCESS

###### Testing fail-for2
_build/funcyjava.native < tests/fail-for2.fj 2> fail-for2.err >> testall.
    log
diff -b fail-for2.err tests/fail-for2.err > fail-for2.diff
###### SUCCESS

###### Testing fail-for3
_build/funcyjava.native < tests/fail-for3.fj 2> fail-for3.err >> testall.
    log
diff -b fail-for3.err tests/fail-for3.err > fail-for3.diff
###### SUCCESS

###### Testing fail-for4
_build/funcyjava.native < tests/fail-for4.fj 2> fail-for4.err >> testall.
    log
diff -b fail-for4.err tests/fail-for4.err > fail-for4.diff
###### SUCCESS

###### Testing fail-func1
_build/funcyjava.native < tests/fail-func1.fj 2> fail-func1.err >> testall
    .log
diff -b fail-func1.err tests/fail-func1.err > fail-func1.diff
###### SUCCESS

###### Testing fail-func2
_build/funcyjava.native < tests/fail-func2.fj 2> fail-func2.err >> testall
    .log
diff -b fail-func2.err tests/fail-func2.err > fail-func2.diff
###### SUCCESS

###### Testing fail-func3
```

```
_build/funcyjava.native < tests/fail-func3.fj 2> fail-func3.err >> testall
    .log
diff -b fail-func3.err tests/fail-func3.err > fail-func3.diff
###### SUCCESS

###### Testing fail-func4
_build/funcyjava.native < tests/fail-func4.fj 2> fail-func4.err >> testall
    .log
diff -b fail-func4.err tests/fail-func4.err > fail-func4.diff
###### SUCCESS

###### Testing fail-func5
_build/funcyjava.native < tests/fail-func5.fj 2> fail-func5.err >> testall
    .log
diff -b fail-func5.err tests/fail-func5.err > fail-func5.diff
###### SUCCESS

###### Testing fail-if1
_build/funcyjava.native < tests/fail-if1.fj 2> fail-if1.err >> testall.log
diff -b fail-if1.err tests/fail-if1.err > fail-if1.diff
###### SUCCESS

###### Testing fail-if2
_build/funcyjava.native < tests/fail-if2.fj 2> fail-if2.err >> testall.log
diff -b fail-if2.err tests/fail-if2.err > fail-if2.diff
###### SUCCESS

###### Testing fail-intarray
_build/funcyjava.native < tests/fail-intarray.fj 2> fail-intarray.err >>
    testall.log
diff -b fail-intarray.err tests/fail-intarray.err > fail-intarray.diff
###### SUCCESS

###### Testing fail-intarray2
_build/funcyjava.native < tests/fail-intarray2.fj 2> fail-intarray2.err >>
     testall.log
diff -b fail-intarray2.err tests/fail-intarray2.err > fail-intarray2.diff
###### SUCCESS

###### Testing fail-intarray3
_build/funcyjava.native < tests/fail-intarray3.fj 2> fail-intarray3.err >>
     testall.log
diff -b fail-intarray3.err tests/fail-intarray3.err > fail-intarray3.diff
###### SUCCESS

###### Testing fail-intarray4
_build/funcyjava.native < tests/fail-intarray4.fj 2> fail-intarray4.err >>
     testall.log
diff -b fail-intarray4.err tests/fail-intarray4.err > fail-intarray4.diff
###### SUCCESS

###### Testing fail-mod1
_build/funcyjava.native < tests/fail-mod1.fj 2> fail-mod1.err >> testall.
    log
```

```
diff -b fail-mod1.err tests/fail-mod1.err > fail-mod1.diff
###### SUCCESS

###### Testing fail-return1
_build/funcyjava.native < tests/fail-return1.fj 2> fail-return1.err >>
   testall.log
diff -b fail-return1.err tests/fail-return1.err > fail-return1.diff
###### SUCCESS

###### Testing fail-return2
_build/funcyjava.native < tests/fail-return2.fj 2> fail-return2.err >>
   testall.log
diff -b fail-return2.err tests/fail-return2.err > fail-return2.diff
###### SUCCESS

###### Testing fail-stringarray
_build/funcyjava.native < tests/fail-stringarray.fj 2> fail-stringarray.
   err >> testall.log
diff -b fail-stringarray.err tests/fail-stringarray.err > fail-stringarray
   .diff
###### SUCCESS

###### Testing fail-stringarray2
_build/funcyjava.native < tests/fail-stringarray2.fj 2> fail-stringarray2.
   err >> testall.log
diff -b fail-stringarray2.err tests/fail-stringarray2.err > fail-
   stringarray2.diff
###### SUCCESS

###### Testing fail-stringarray3
_build/funcyjava.native < tests/fail-stringarray3.fj 2> fail-stringarray3.
   err >> testall.log
diff -b fail-stringarray3.err tests/fail-stringarray3.err > fail-
   stringarray3.diff
###### SUCCESS

###### Testing fail-uop1
_build/funcyjava.native < tests/fail-uop1.fj 2> fail-uop1.err >> testall.
   log
diff -b fail-uop1.err tests/fail-uop1.err > fail-uop1.diff
###### SUCCESS

###### Testing fail-while1
_build/funcyjava.native < tests/fail-while1.fj 2> fail-while1.err >>
   testall.log
diff -b fail-while1.err tests/fail-while1.err > fail-while1.diff
###### SUCCESS

###### Testing fail-while2
_build/funcyjava.native < tests/fail-while2.fj 2> fail-while2.err >>
   testall.log
diff -b fail-while2.err tests/fail-while2.err > fail-while2.diff
###### SUCCESS

```

```
###### Testing fail-while3
_build/funcyjava.native < tests/fail-while3.fj 2> fail-while3.err >>
   testall.log
diff -b fail-while3.err tests/fail-while3.err > fail-while3.diff
###### SUCCESS
```

7.5 References/Acknowledgement

During the process of developing our language, there were times when we would run into roadblocks with our implementation. We received guidance on these issues from our fantastic TA Xijiao Li and from looking at other languages from past semesters. We would like to thank Xijiao for all of her guidance and cite the languages we referenced for certain features of FUNC-y java here:

1. AP++ (2018)
2. Strux (2017)
3. PixMix (2017)
4. Coral (2018)
5. F.I.R.E (2018)
6. GOLD (2017)
7. Pseudo (2017)