Code Generator 2

Note: please use the new version of ASTFunction::Output; otherwise you cannot use parameters correctly.

**Rules that must be fully implemented for part 2 (partial implementation to make part 1 work):**

* ASTFunctionCall - this one needs to be implemented to work for the general case of function calls, NOTE: requires the new implementation of ASTFunction - there is a missing line in ASTFunction from part 1.
* ASTBoolExprA
* ASTBoolExprB
* ASTTerm
* ASTExpr

**Rules for part 2 that are not implemented at all in Part 1.**

* ASTIf
* ASTWhile
* ASTBlock
* ASTAssign

Example input:

test(float z)

{

print(z);

}

main()

{

float x;

time(x);

x = x / 100;

x = x % 400;

if(FALSE)

{

test(x);

}

drawsquare(1,0,0,x,50,250,250);

}

Example output:

function test params f0 endparams

print f0

endfunction

function main params endparams

float f1

time f1

float f2

float f3

= f3 100

/ f2 f1 f3

= f1 f2

float f4

float f5

= f5 400

% f4 f1 f5

= f1 f4

bool b6

= b6 FALSE

! b6 b6

jumpif l1 b6

callfunction test args f1 endargs

label l1

float f7

= f7 1

float f8

= f8 0

float f9

= f9 0

float f10

= f10 50

float f11

= f11 250

float f12

= f12 250

callfunction drawsquare args f7 f8 f9 f1 f10 f11 f12 endargs

endfunction

Code Generator 1

This program will output a program that works with the interpreter from the first part of the project.

This part of the program functions similarly to the type checker -> each rule of the program will implement its own operations and (sometimes) return a result to a node above it. You also have the luxury of assuming the correctness of the program at this point. I have included a partial implementation of ASTFunctionCall so that you can see how that can work.

Keep in mind whatever you implement needs to work with the interpreter. That means just like how the bytecode programs of our interpreter used registers, so too will the code generator.

**Rules that are implemented**

ASTProgram

ASTFunction (this might change a part 2, but it is not your concern for part 1)

**Rules that must be implemented:**

ASTStatements

ASTStatement

ASTDeclaration

ASTFactor

ASTElement

Example run (NOTE: your output may be different and still correct. I’m going to test it by running your output through my interpreter to see if it gives the correct output for things like prints.)

Input:

main()

{

float x = 10;

time(x);

print(x);

bool y = TRUE;

bool z = FALSE;

bool t = z;

bool t2 = y;

print(t);

print(t2);

drawsquare(1,0.3,0.8,1,1,50,50);

}

Output:

function main params endparams

float f0

float f1

= f1 10

= f0 f1

time f0

print f0

bool b2

bool b3

= b3 TRUE

= b2 b3

bool b4

bool b5

= b5 FALSE

= b4 b5

bool b6

= b6 b4

bool b7

= b7 b2

print b6

print b7

float f8

= f8 1

float f9

= f9 0.3

float f10

= f10 0.8

float f11

= f11 1

float f12

= f12 1

float f13

= f13 50

float f14

= f14 50

callfunction drawsquare args f8 f9 f10 f11 f12 f13 f14 endargs

endfunction