Compiler Construction
Oblig 2: Type Checking and Code Generation

Team Members : Selleban Farah (Sellebaf)

Solution

Type Checking/Semantic analysis

In order to properly type check Variables and return types a SymbolTable class was made that generates that holds a set of hashtables that hold the various types of declarations. These being: variables, records/structs, parameters, and procedure declarations. This symbol table is generated in the Program class and passed down to the syntaxtree.

With the symbol table presence within the syntax tree , typechecking is a simple as looking up the declaration name up in the symbol table and returning the declaration. From there we can return the declaration type, which we can use for simple comparison. For handling comparison, the class Type checking was made that hosts a set of static procedures that aid in comparing the returned types, as well as for returning bytecode types later.

Code Generation

Code generation starts with the program class and adding the Standard library declarations to the set of bytecode instruction. From here code generation for subsequent declarations are called down the syntax tree.

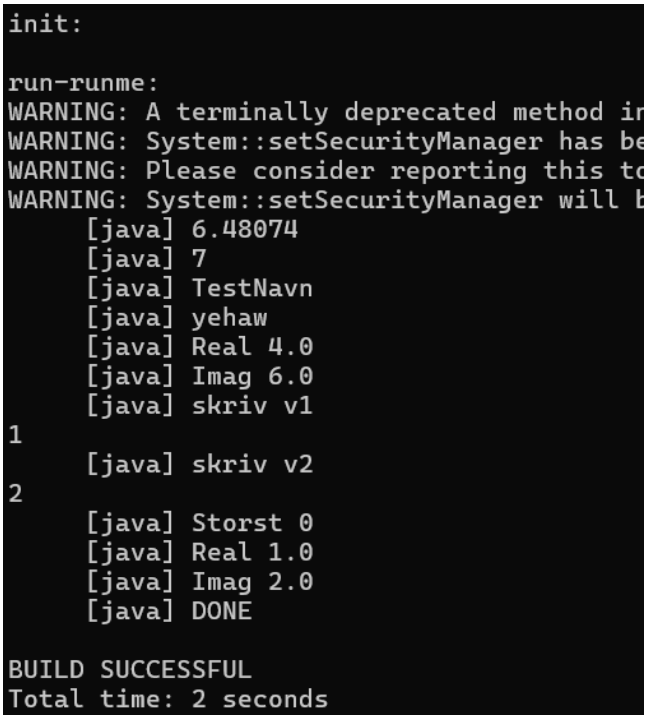
Code generation errors

It was during code generation i hade some problems. I hade trouble understanding the proper flow of and use of the JMP and JMP false instruction. If statements when true tend to give the correct results, it is when specifically jumping to the false instructions i meet some errors. There's also the possibility that the error is caused by a call further down the syntax

tree, which makes the error even more confusing, identify and properly handle. The goal of the oblig is to properly run the runme.cp, a test file that only holds an if-statement, meaning other statements that take use of jump instructions such as while-statements were never properly tested. The point im getting at is that there may be more underlying errors within the code generation that are hiding because the runme.cp test file doesn't take use of them.

Tests and Results

Within the outputs folder, there should be two new folders. One named “bin” that holds the generated code(binary file). The second named “printouts” folder simply holds a text file of the list-runme results from runme.cp



Printout from run-runme