CMSC 430: Compiler Theory and Design

Professor Gregory Williams

Project 4

Due Date: December 15, 2020

Bedemariam Degef

Project 4

Approach To Project

Before beginning this project, I watched the four-part video series on Semantic Analyzers. I then read Chapter 7 of Mogensen’s Textbook. After going through all of the assigned material, I started by importing my project 2 code to the project 4 skeleton code. I then made the necessary changes to types.h, types.cc, scanner.l, and parser.y according to the project requirements, starting from the elements evaluated from the bottom of the parse tree and working my way up. After each change to the code, I checked my work with a specific test plan to test that change.

Test Case Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Input File** | **Expected Result** | **Result** | **Pass/Fail** |
| Test1.txt | Integer operand required | Integer operand required | Pass |
| Test2.txt | Numeric Type Required | Numeric Type Required | Pass |
| Test3.txt | Boolean Type Required | Boolean Type Required | Pass |
| Test4.txt | Illegal Narrowing and Type Mismatch | Illegal Narrowing and Type Mismatch | Pass |
| Test5.txt | If Expression Must be Boolean | If Expression Must be Boolean | Pass |
| Test6.txt | Illegal Narrowing Function | Illegal Narrowing Function | Pass |
| Test7.txt | Type Mismatch on Variable Initialization | Type Mismatch on Variable Initialization | Pass |
| Test8.txt | Undeclared Variable | Undeclared Variable | Pass |
| Test9.txt | Duplicate Identifier | Duplicate Identifier | Pass |
| Test10.txt | Illegal Narrowing | Illegal Narrowing | Pass |
| Test11.txt | Case expression errors | Case expression errors | Pass |
| Test12.txt | Case Expression Must Match | Case Expression Must Match | Pass |
| Test13.txt | Reduction operation error | Reduction operation error | Pass |

Test Plan Results

**Test Case #1 Result – Testing with modulus operator with non-integer operands**

Text

Description automatically generated

**Test Case #2 Result – Testing with invalid relational operator**

Text

Description automatically generated

**Test Case #3 Result – Testing with invalid logical operator**

Text

Description automatically generated

**Test Case #4 Result – Testing with invalid if/else statement**

Text

Description automatically generated

**Test Case #5 Result – Testing with invalid if/else statement**

Text

Description automatically generated

**Test Case #6 Result – Testing with invalid variable declaration**

Text

Description automatically generated

**Test Case #7 Result – Testing with invalid variable declaration**

Text

Description automatically generated

**Test Case #8 Result – Testing with undeclared variable**

Text

Description automatically generated

**Test Case #9 Result - Testing with duplicate identifier**

**Text

Description automatically generated**

**Test Case #10 Result - Testing with illegal narrowing**

**Text

Description automatically generated**

**Test Case #11 Result - Testing with case statement with Boolean**

**Text

Description automatically generated**

**Test Case #12 Result - Testing with case statement with non-integer**

**Text

Description automatically generated**

**Test Case #13 Result -Testing with reduction with non-numeric**

Text

Description automatically generated

Lessons Learned

This project taught me the concepts of type checking, type coercion, narrowing, and widening. Throughout the course of Compiler Theory and Design, I have learned how to use Flex, Bison, and learned the basics of compiling lexical, syntactical, and semantic analyzers.