Michael Scobee

UMGC

CMSC 430

12/14/2021

Project 4 Documentation

**Approach**

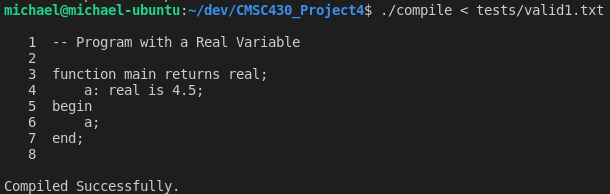
For project 4 I approached it by following the recommended approach document step by step. I started by adding the real literal tokens and then proceeded to add type coercion and boolean types. The next step involved editing the productions for logical operations and exponentiation. Added c++ functionality to the rem operator production. I then proceeded to add logic to check the semantics on the if statements, and finally check for narrowing variable initialization.

**Lessons Learned**

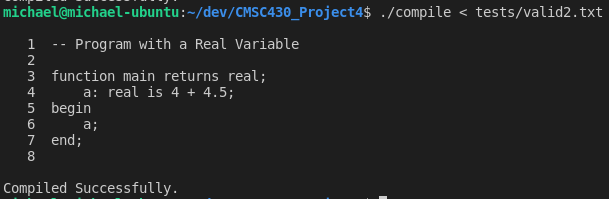
I learned quite a few things in this project but the largest thing I believe to be how a compiler can handle type coercion and report semantic errors due to type mistakes.

**Test Plan**

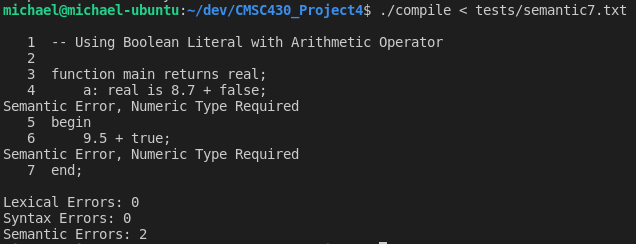
| **Test** | **Input** | **Expected Output** | **Actual Output** | **Pass/Fail** | **Figure** |
| --- | --- | --- | --- | --- | --- |
| Real literal tokens added | valid1.txt | Compiled successfully | Compiled successfully | Pass | 1 |
| Integer/Real type coercion | valid2.txt | Compiled successfully | Compiled successfully | Pass | 2 |
| Boolean Type | semantic7.txt | Semantic Error: 2 | Semantic Error: 2 | Pass | 3 |
| Edited productions of not or exp | semantic8.txt | Semantic Error: 1 | Semantic Error: 1 | Pass | 4 |
| Check types on rem operator | semantic11.txt | Semantic Error: 1 | Semantic Error: 1 | Pass | 5 |
| If statement semantic checks | semantic12.txt | Semantic Error: 1 | Semantic Error: 1 | Pass | 6 |
| If then type mismatch | semantic13.txt | Semantic Error :1 | Semantic Error: 1 | Pass | 7 |
| Narrowing Variable initialization | semantic14.txt | Semantic Error: 1 | Semantic Error: 1 | Pass | 8 |
| Narrowing function return | semantic15.txt | Illegal Narrowing Function Return | Illegal Narrowing Function Return | Fail | 9 |
| Duplicate Variables | semantic16.txt | Duplicate identifier:b | Duplicate identifier:b | Pass | 10 |

****

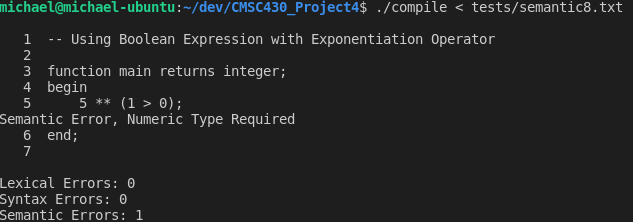
**Figure 1 Real Literal Tokens Added**

****

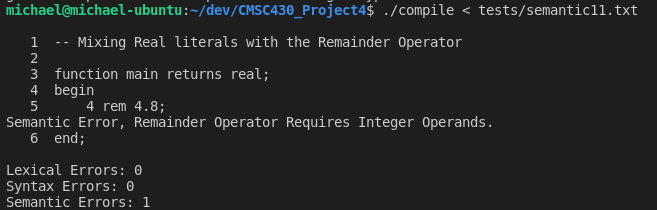
**Figure 2 Integer/Real Type Coercion**

****

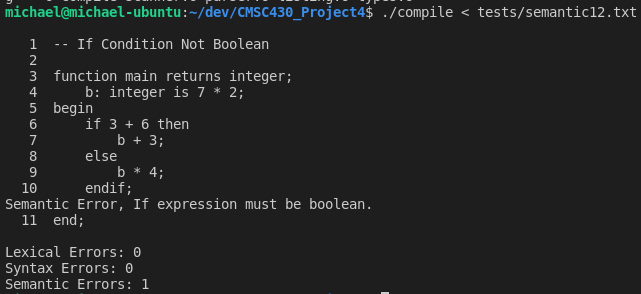
**Figure 3 Boolean type**

****

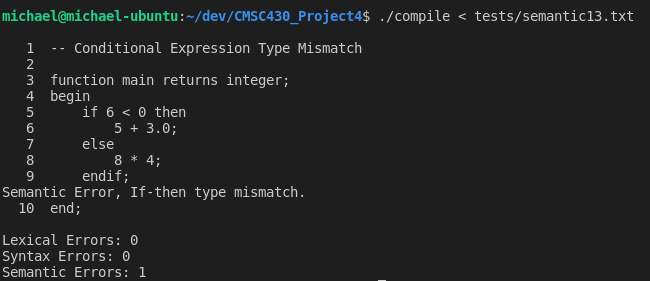
**Figure 4 Added productions to exponentiation**

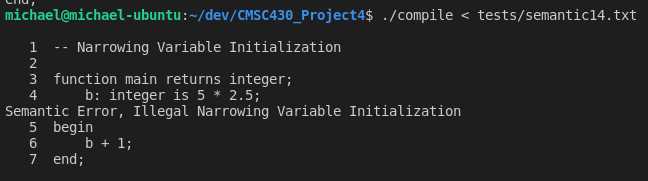
****

**Figure 5 Remainder Operator requires integers**

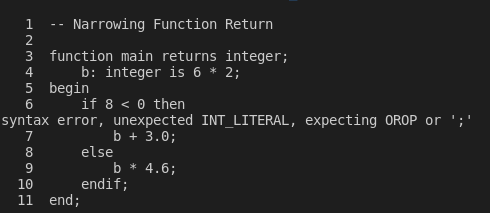
****

**Figure 6 If statement semantic checks**

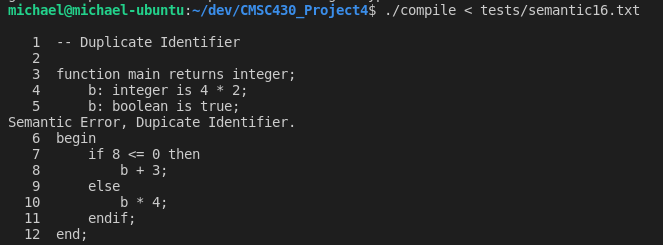
**Figure 7 if then type mismatch**

****

**Figure 8 narrowing variable initialization**

****

**Figure 9 Narrowing Function Return**

**Figure 10 duplicate identifier**