# PROJECT 2 REPORT

CMSC 430 – COMPILER THEORY AND DESIGN

Robert D. Carswell

6 February 2024

## Contents

## Executive Summary

This project involves enhancing the syntactic analyzer for a given compiler by extending its grammar by modifying and adding the existing grammar. This included new terminals and non-terminals to eliminate the EBNF brace and bracket meta-symbols. At the same time, it introduces the meanings of parentheses, operator precedence, and associativity rules for arithmetic, logical, and relational operators. While adding error productions to find and fix syntax errors using the semicolon as the synchronization token. To ensure the correct parsing of a syntactically valid program is done without errors. To achieve these goals, attention was given to operator precedence, associativity, and the introduction of error productions. In turn, it creates an effective syntactic analyzer capable of handling additional syntax errors and producing correct parse results.

## Testing

### Test Case Table

| Test Cases | Description | Expected Output | Actual Output | Pass/Fail |
|---|---|---|---|---|
| **Test Case 0** | Compile program | Compiles with no shift/reduce conflicts | Test Case 1 | Read test file syntax1.txt |
| **Test Case 1** | Read test file syntax1.txt | File contents with:<br><br>Msg (Under line 5): syntax Error, unexpected INT_LITERAL, Expecting ‘;’<br><br>Msg (Bottom of file):<br>Lexical Errors: 0<br>Syntax Errors: 1<br>Semantic Errors: 0 | File contents with:<br><br>Msg (Under line 5): Syntax Error, Unexpected INT_LITERAL, Expecting ‘;’<br><br>Msg (Bottom of file):<br>Lexical Errors: 0<br>Syntax Errors: 1<br>Semantic Errors: 0 | Pass |
| **Test Case 2** | Read test file syntax2.txt | File contents with:<br><br>Msg (Under line 3): syntax error, unexpected INTEGER, expecting ':'<br><br>Msg (Bottom of file):<br>Lexical Errors: 0<br>Syntax Errors: 1<br>Semantic Errors: 0 | File contents with:<br><br>Msg (Under line 3): syntax error, unexpected INTEGER, expecting ':'<br><br>Msg (Bottom of file):<br>Lexical Errors: 0<br>Syntax Errors: 1<br>Semantic Errors: 0 | Pass |
| **Test Case 3** | Read test file Syntax3.txt | File contents with:<br><br>Msg (Under line 4): syntax error, unexpected INTEGER, expecting ':'<br><br>Msg (Bottom of file):<br>Lexical Errors: 0<br>Syntax Errors: 1<br>Semantic Errors: 0 | File contents with:<br><br>Msg (Under line 4): syntax error, unexpected INTEGER, expecting ':'<br><br>Msg (Bottom of file):<br>Lexical Errors: 0<br>Syntax Errors: 1<br>Semantic Errors: 0 | Pass |
| **Test Case 4** | Read test file syntax4.txt | File contents with:<br><br>Msg (Under line 6): syntax error, | File contents with:<br><br>Msg (Under line 6): syntax error, | Pass |

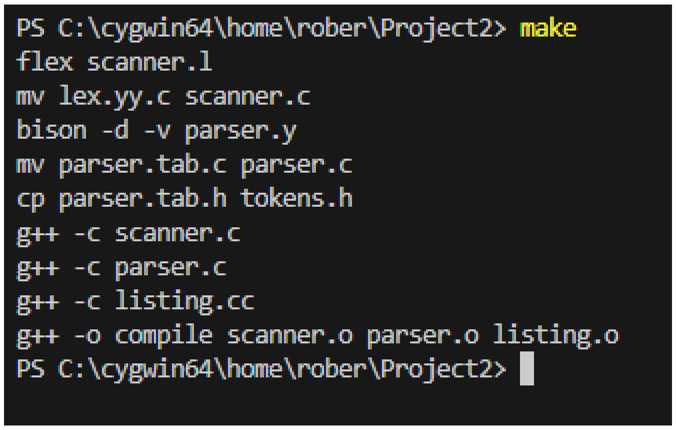
| Test Cases | Description | Expected Output | Actual Output | Pass/Fail |
|---|---|---|---|---|
| | | unexpected INT_LITERAL, expecting ';'<br><br>Msg (Under line 8): syntax error, unexpected ENDSWITCH, expecting CASE or OTHERS'<br><br>Msg (Bottom of file):<br>Lexical Errors: 0<br>Syntax Errors: 2<br>Semantic Errors: 0 | unexpected INT_LITERAL, expecting ';'<br><br>Msg (Under line 8): syntax error, unexpected ENDSWITCH, expecting CASE or OTHERS'<br><br>Msg (Bottom of file):<br>Lexical Errors: 0<br>Syntax Errors: 2<br>Semantic Errors: 0 | |
| **Test Case 5** | Read test file syntax5.txt | File contents with:<br><br>Msg (Under line 3): syntax error, unexpected INTEGER, expecting ':'<br><br>Msg (Under line 4): syntax error, unexpected MULOP<br><br>Msg (Under line 8): syntax error, unexpected MULOP<br><br>Msg (Under line 11): syntax error, unexpected ARROW, expecting INT_LITERAL<br><br>Msg (Under line 13): syntax error, unexpected ENDSWITCH, expecting CASE or OTHERS<br><br>Msg (Bottom of file):<br>Lexical Errors: 0<br>Syntax Errors: 5 | File contents with:<br><br>Msg (Under line 3): syntax error, unexpected INTEGER, expecting ':'<br><br>Msg (Under line 4): syntax error, unexpected MULOP<br><br>Msg (Under line 8): syntax error, unexpected MULOP<br><br>Msg (Under line 11): syntax error, unexpected ARROW, expecting INT_LITERAL<br><br>Msg (Under line 13): syntax error, unexpected ENDSWITCH, expecting CASE or OTHERS<br><br>Msg (Bottom of file):<br>Lexical Errors: 0<br>Syntax Errors: 5 | Pass |

| Test Cases | Description | Expected Output | Actual Output | Pass/Fail |
|---|---|---|---|---|
| | | Semantic Errors: 0 | Semantic Errors: 0 | |
| **Test Case 6** | Read test file test1.txt | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | Pass |
| **Test Case 7** | Read test file test2.txt | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | Pass |
| **Test Case 8** | Read test file test3.txt | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | Pass |
| **Test Case 9** | Read test file test4.txt | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | Pass |
| **Test Case 10** | Read test file test5.txt | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | Pass |
| **Test Case 11** | Read test file test6.txt | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | Pass |
| **Test Case 12** | Read test file test7.txt | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | Pass |
| **Test Case 13** | Read test file test8.txt | File contents with :<br><br>Msg (Bottom of file): | File contents with :<br><br>Msg (Bottom of file): | Pass |

| Test Cases | Description | Expected Output | Actual Output | Pass/Fail |
|---|---|---|---|---|
| | | Compilation Successful | Compilation Successful | |
| **Test Case 14** | Read test file test9.txt | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | Pass |
| **Test Case 15** | Read test file test10.txt | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | Pass |
| **Test Case 16** | Read test file test11.txt | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | Pass |
| **Test Case 17** | Read test file test12.txt | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | Pass |
| **Test Case 18** | Read test file test13.txt | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | Pass |
| **Test Case 19** | Read test file test14.txt | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | Pass |
| **Test Case 20** | Read test file test15.txt | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | File contents with :<br><br>Msg (Bottom of file): Compilation Successful | Pass |

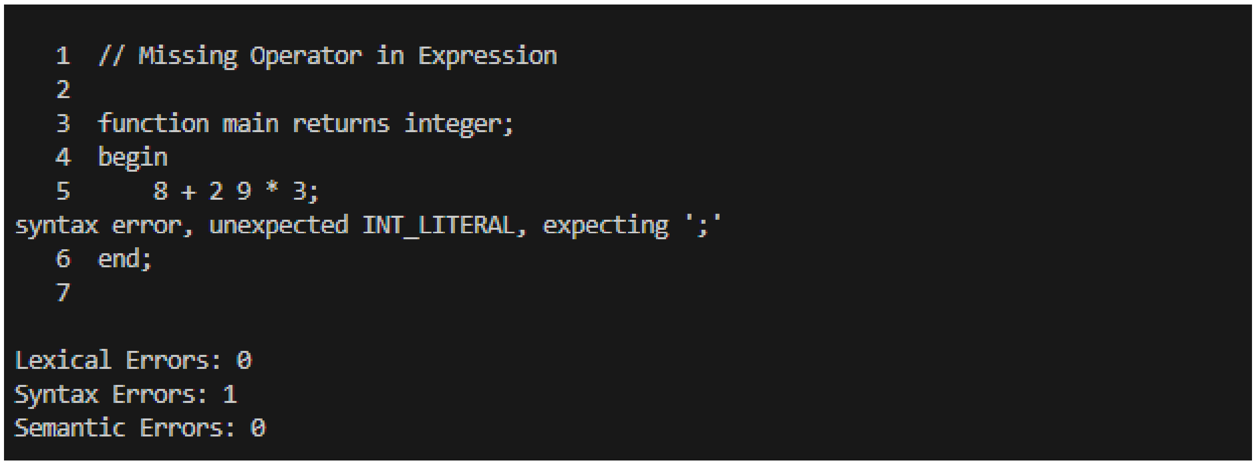
## Test Case Screenshots

### Test Case 0



```
PS C:\cygwin64\home\rober\Project2> make
flex scanner.l
mv lex.yy.c scanner.c
bison -d -v parser.y
mv parser.tab.c parser.c
cp parser.tab.h tokens.h
g++ -c scanner.c
g++ -c parser.c
g++ -c listing.cc
g++ -o compile scanner.o parser.o listing.o
PS C:\cygwin64\home\rober\Project2>
```

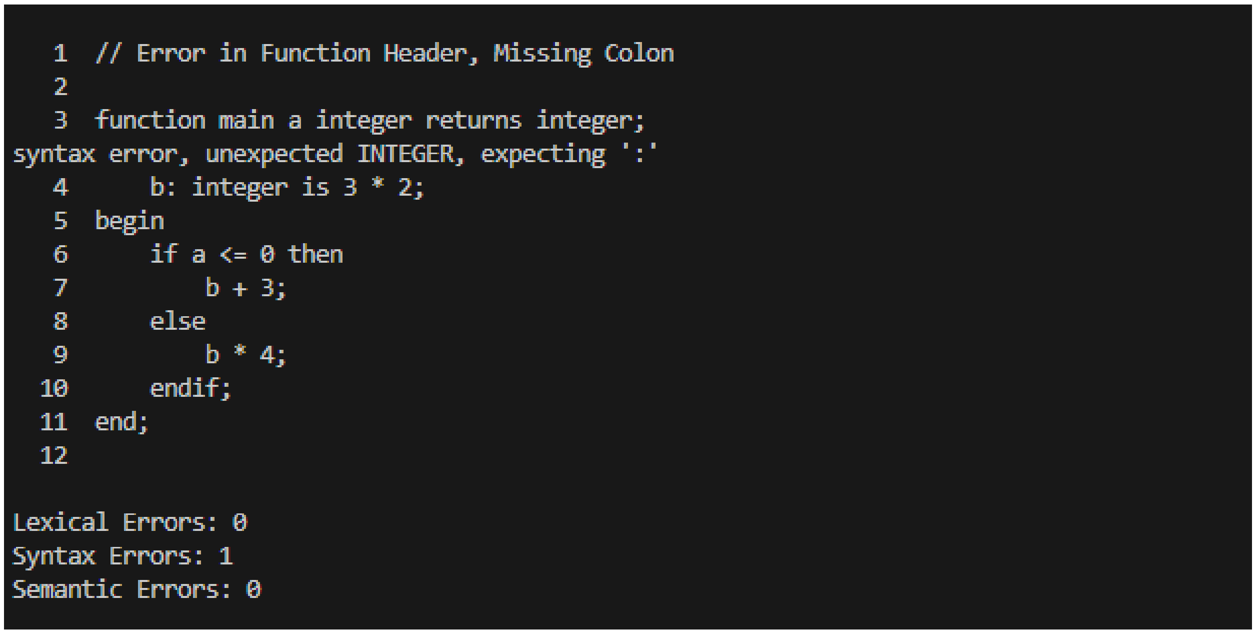
### Test Case 1



```
   1  // Missing Operator in Expression
   2
   3  function main returns integer;
   4  begin
   5      8 + 2 9 * 3;
syntax error, unexpected INT_LITERAL, expecting ';'
   6  end;
   7

Lexical Errors: 0
Syntax Errors: 1
Semantic Errors: 0
```

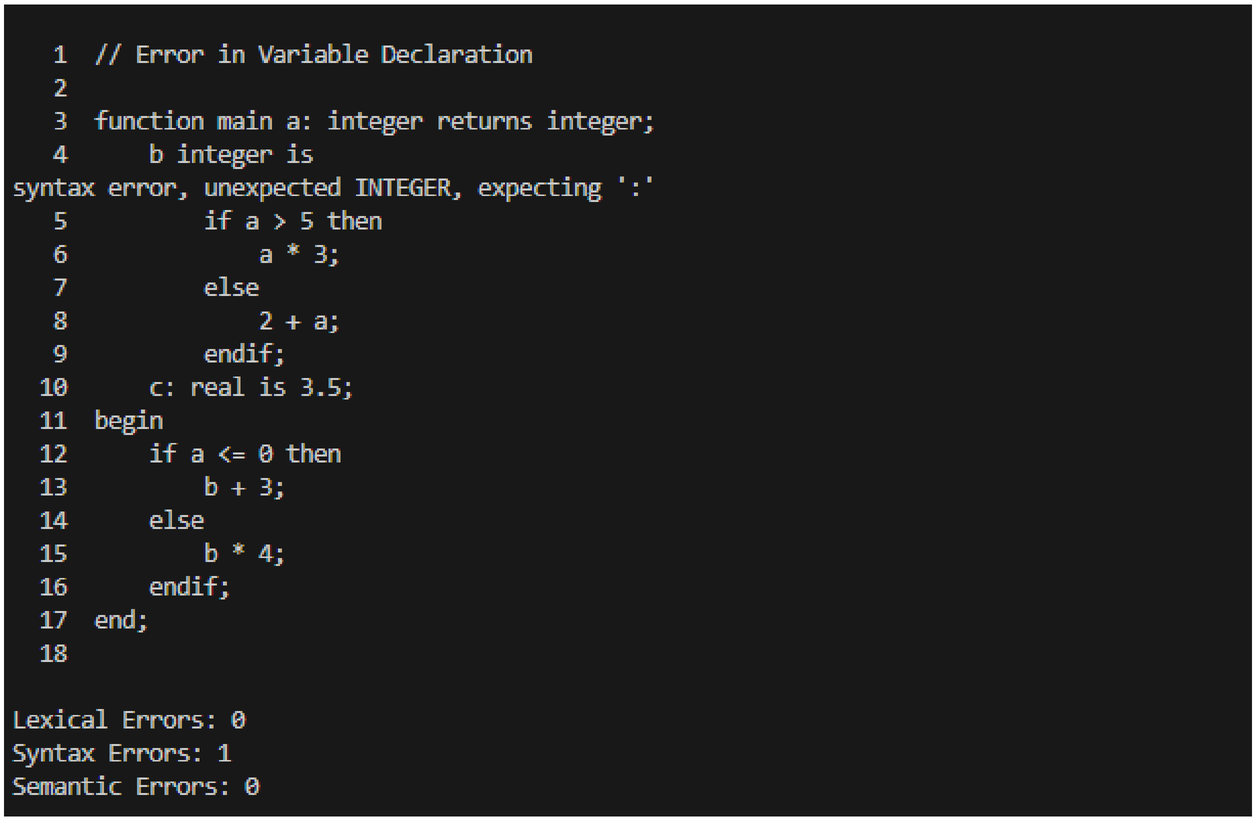
## Test Case 2



```
   1  // Error in Function Header, Missing Colon
   2
   3  function main a integer returns integer;
syntax error, unexpected INTEGER, expecting ':'
   4      b: integer is 3 * 2;
   5  begin
   6      if a <= 0 then
   7          b + 3;
   8      else
   9          b * 4;
  10      endif;
  11  end;
  12

Lexical Errors: 0
Syntax Errors: 1
Semantic Errors: 0
```

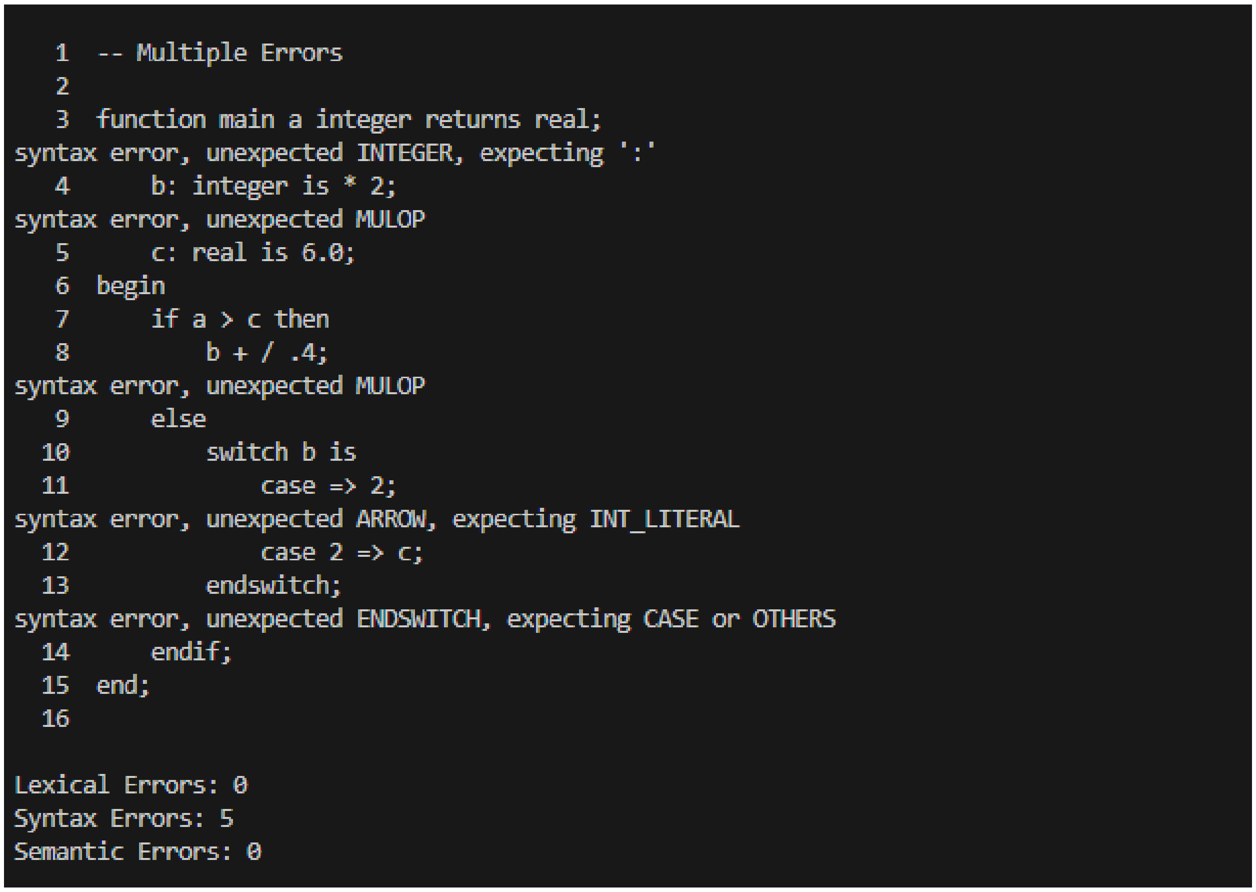
## Test Case 3



```
   1  // Error in Variable Declaration
   2
   3  function main a: integer returns integer;
   4      b integer is
syntax error, unexpected INTEGER, expecting ':'
   5          if a > 5 then
   6              a * 3;
   7          else
   8              2 + a;
   9          endif;
  10      c: real is 3.5;
  11  begin
  12      if a <= 0 then
  13          b + 3;
  14      else
  15          b * 4;
  16      endif;
  17  end;
  18

Lexical Errors: 0
Syntax Errors: 1
Semantic Errors: 0
```

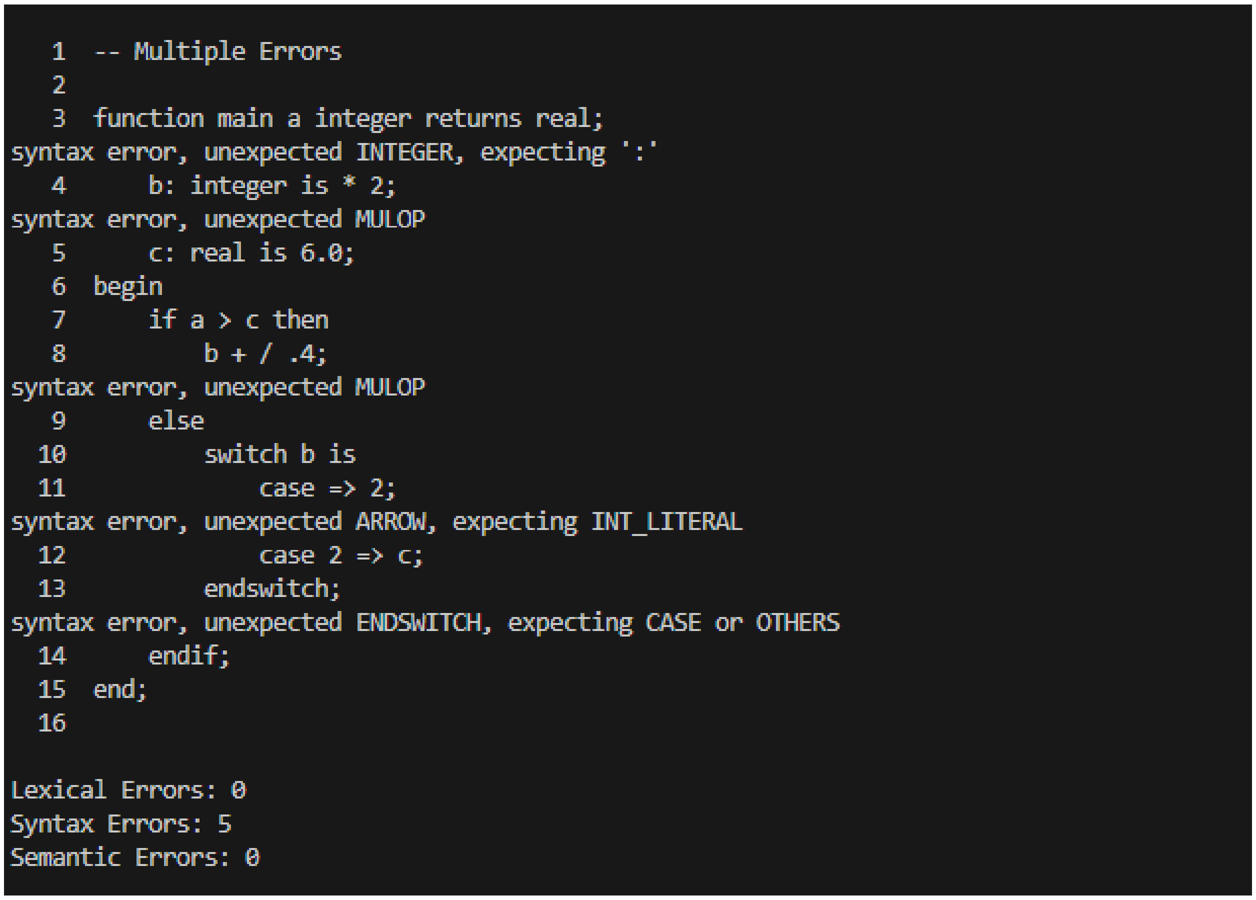
## Test Case 4



```
   1  -- Multiple Errors
   2
   3  function main a integer returns real;
syntax error, unexpected INTEGER, expecting ':'
   4      b: integer is * 2;
syntax error, unexpected MULOP
   5      c: real is 6.0;
   6  begin
   7      if a > c then
   8          b + / .4;
syntax error, unexpected MULOP
   9      else
  10          switch b is
  11              case => 2;
syntax error, unexpected ARROW, expecting INT_LITERAL
  12              case 2 => c;
  13          endswitch;
syntax error, unexpected ENDSWITCH, expecting CASE or OTHERS
  14      endif;
  15  end;
  16

Lexical Errors: 0
Syntax Errors: 5
Semantic Errors: 0
```

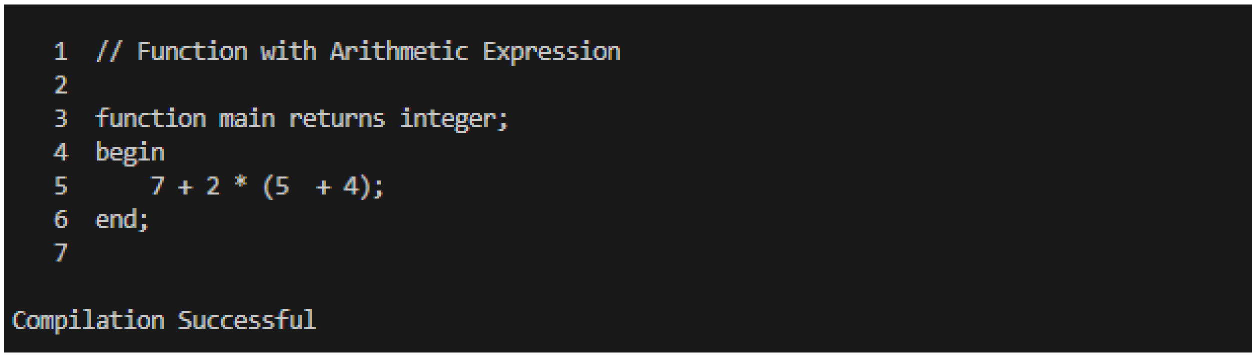
## Test Case 5



```
   1  -- Multiple Errors
   2
   3  function main a integer returns real;
syntax error, unexpected INTEGER, expecting ':'
   4      b: integer is * 2;
syntax error, unexpected MULOP
   5      c: real is 6.0;
   6  begin
   7      if a > c then
   8          b + / .4;
syntax error, unexpected MULOP
   9      else
  10          switch b is
  11              case => 2;
syntax error, unexpected ARROW, expecting INT_LITERAL
  12              case 2 => c;
  13          endswitch;
syntax error, unexpected ENDSWITCH, expecting CASE or OTHERS
  14      endif;
  15  end;
  16

Lexical Errors: 0
Syntax Errors: 5
Semantic Errors: 0
```

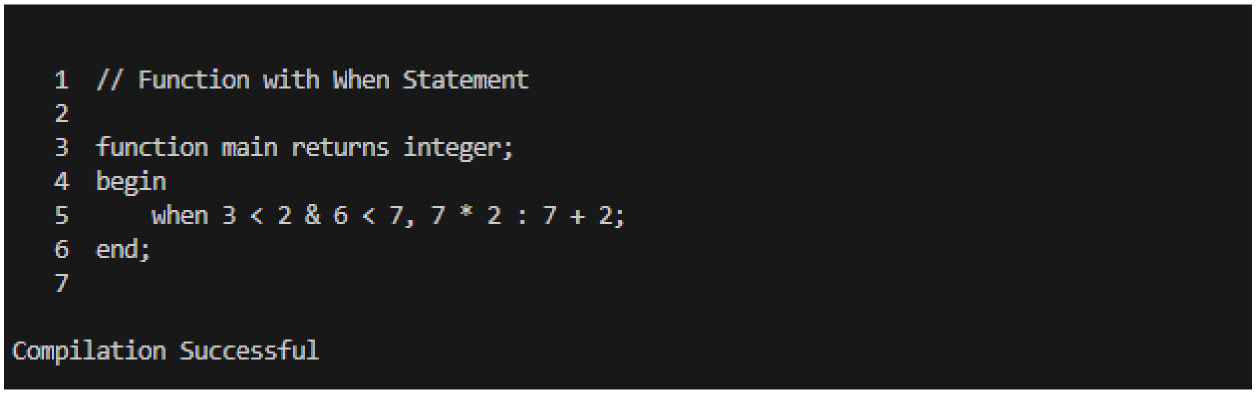
## Test Case 6



```
   1  // Function with Arithmetic Expression
   2
   3  function main returns integer;
   4  begin
   5      7 + 2 * (5  + 4);
   6  end;
   7

Compilation Successful
```

## Test Case 7

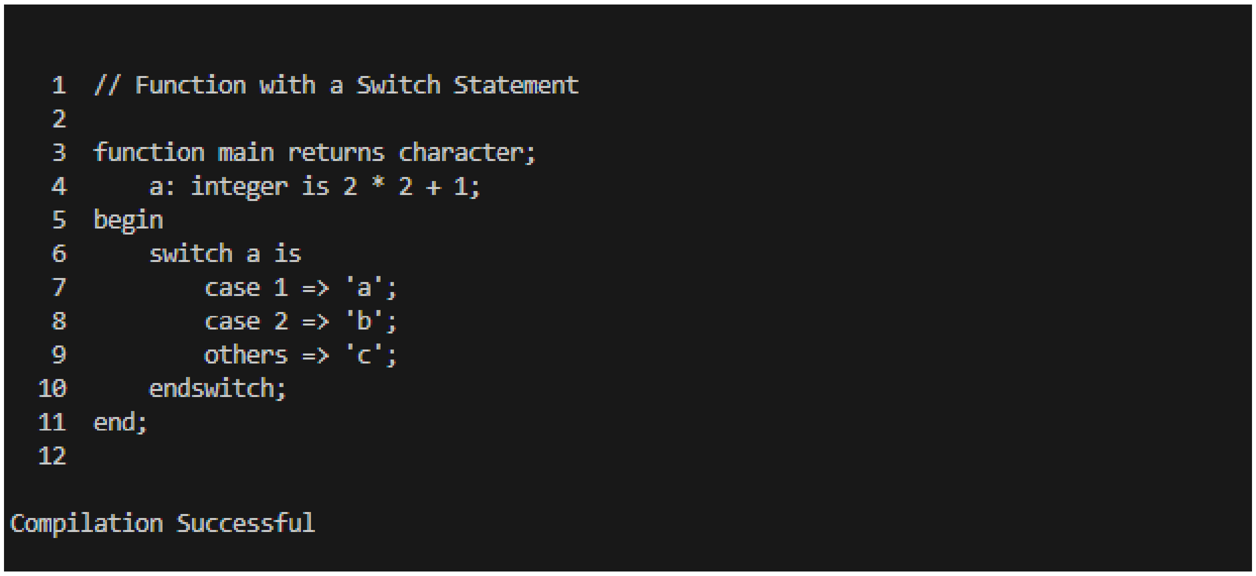


```
// Function with When Statement

function main returns integer;
begin
    when 3 < 2 & 6 < 7, 7 * 2 : 7 + 2;
end;

Compilation Successful
```

## Test Case 8

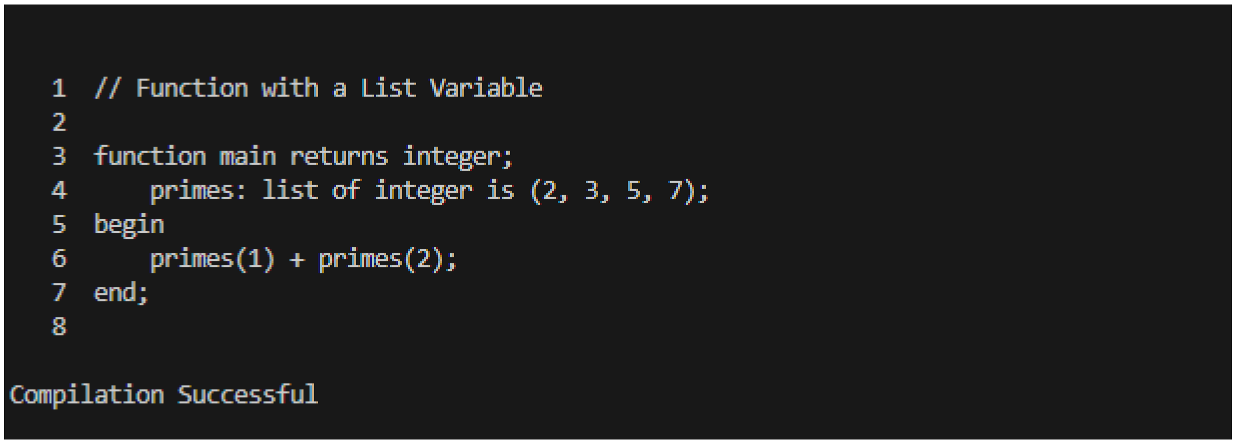


```
// Function with a Switch Statement

function main returns character;
    a: integer is 2 * 2 + 1;
begin
    switch a is
        case 1 => 'a';
        case 2 => 'b';
        others => 'c';
    endswitch;
end;

Compilation Successful
```

## Test Case 9

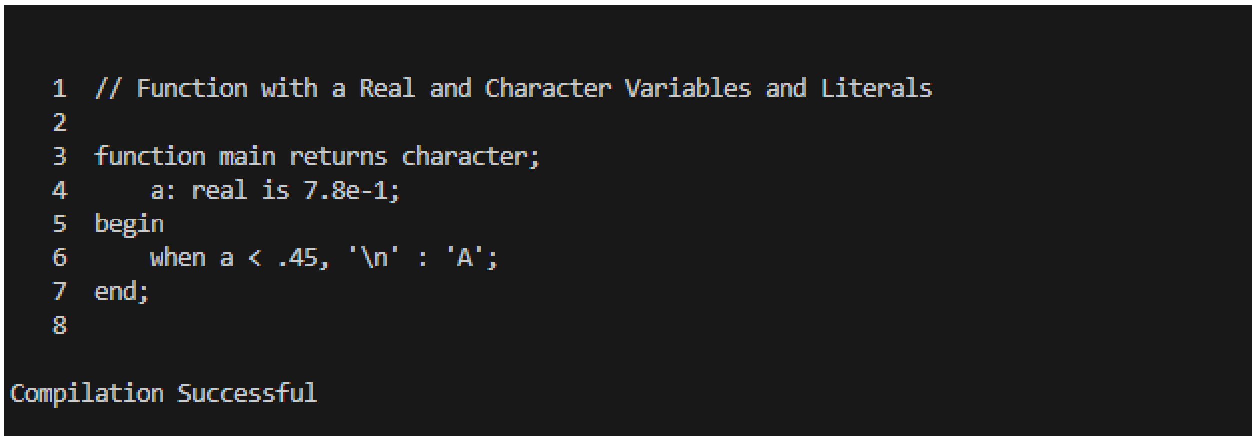


```
// Function with a List Variable

function main returns integer;
    primes: list of integer is (2, 3, 5, 7);
begin
    primes(1) + primes(2);
end;

Compilation Successful
```

## Test Case 10

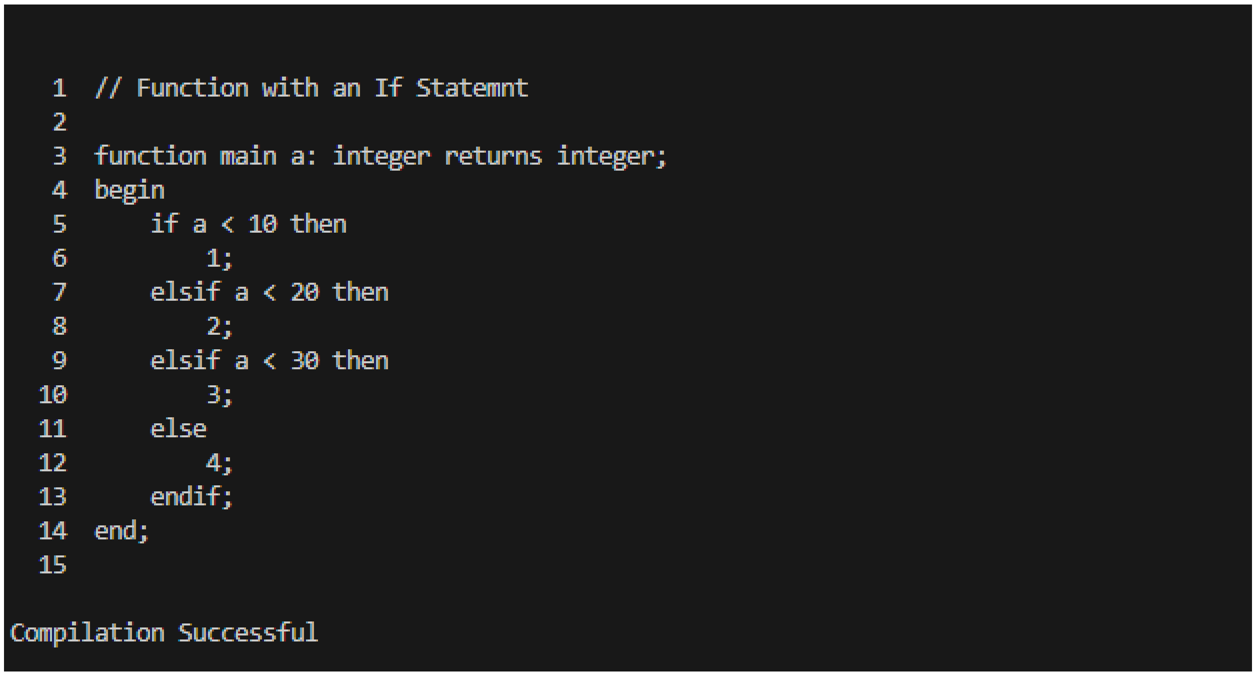


```
// Function with a Real and Character Variables and Literals

function main returns character;
    a: real is 7.8e-1;
begin
    when a < .45, '\n' : 'A';
end;

Compilation Successful
```

## Test Case 11

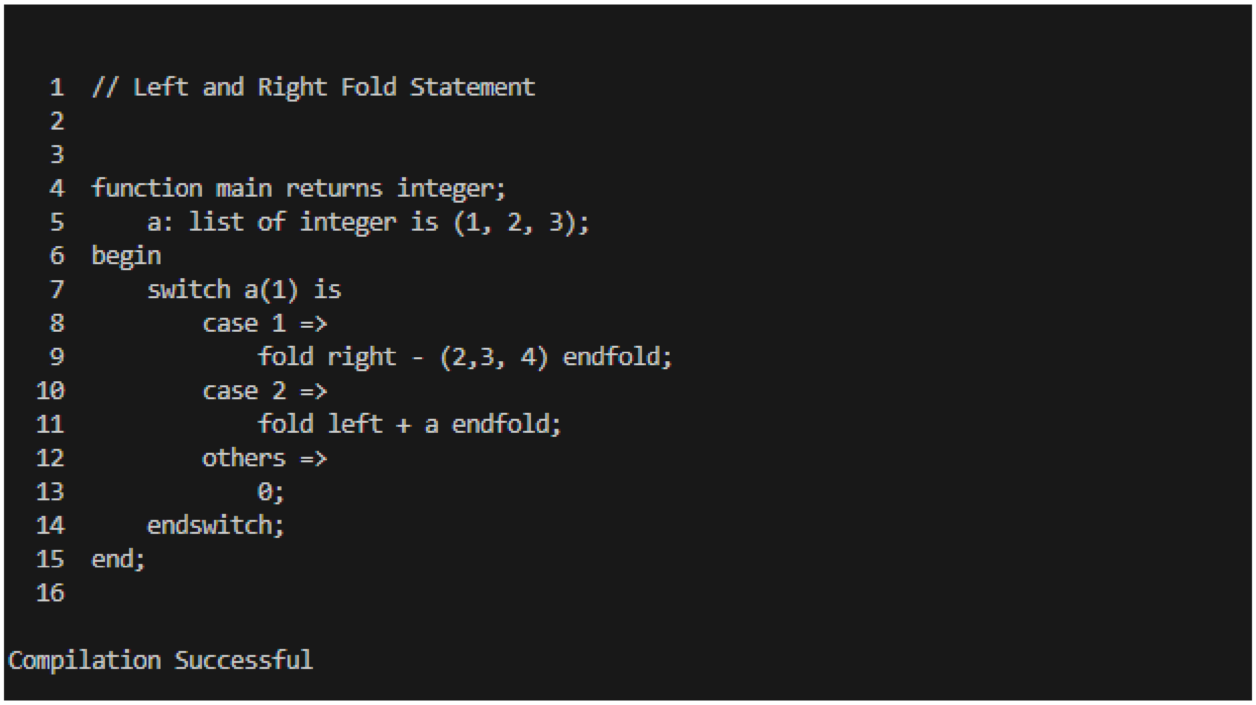


```
// Function with an If Statemnt

function main a: integer returns integer;
begin
    if a < 10 then
        1;
    elsif a < 20 then
        2;
    elsif a < 30 then
        3;
    else
        4;
    endif;
end;

Compilation Successful
```

## Test Case 12



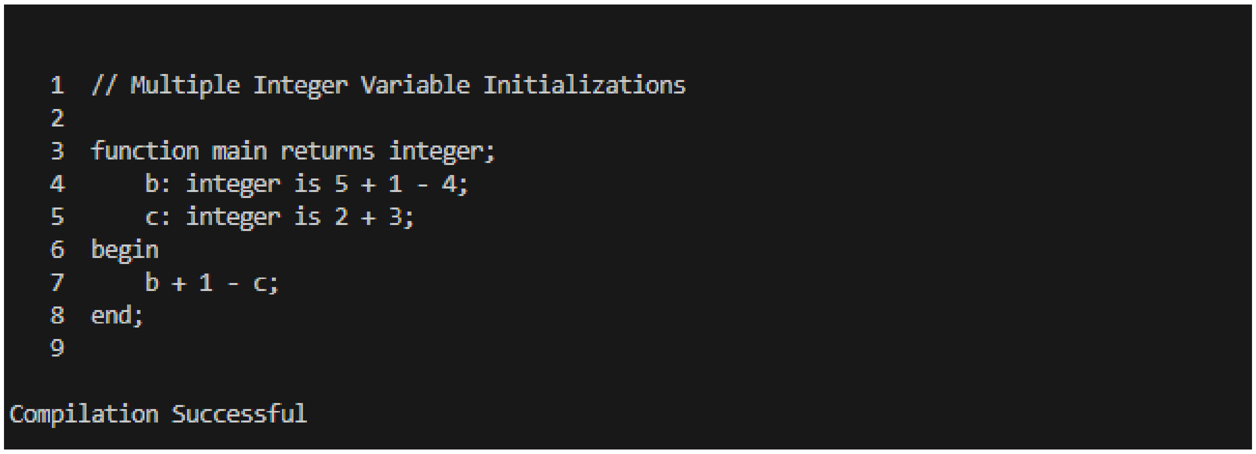
```
// Left and Right Fold Statement


function main returns integer;
    a: list of integer is (1, 2, 3);
begin
    switch a(1) is
        case 1 =>
            fold right - (2,3, 4) endfold;
        case 2 =>
            fold left + a endfold;
        others =>
            0;
    endswitch;
end;

```

```
Compilation Successful
```

## Test Case 13



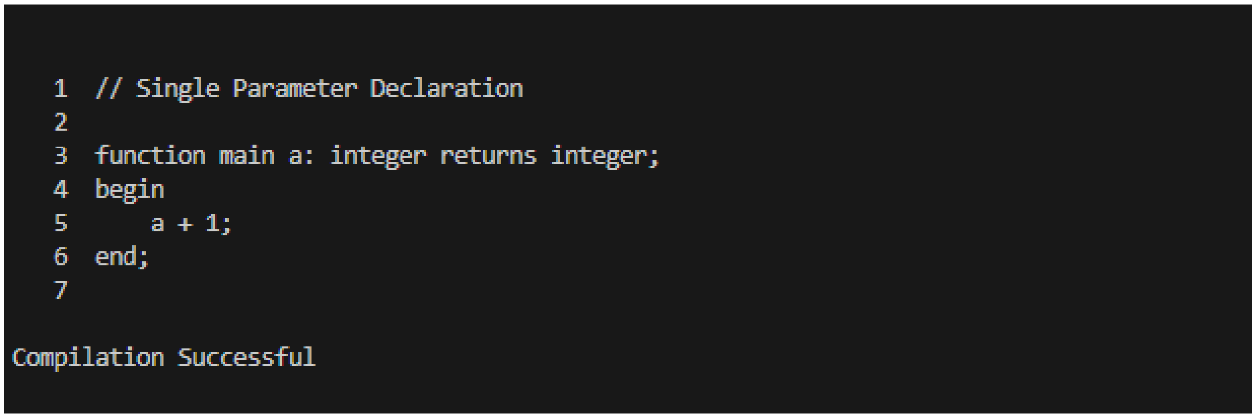
```
// Multiple Integer Variable Initializations

function main returns integer;
    b: integer is 5 + 1 - 4;
    c: integer is 2 + 3;
begin
    b + 1 - c;
end;

```

```
Compilation Successful
```

## Test Case 14



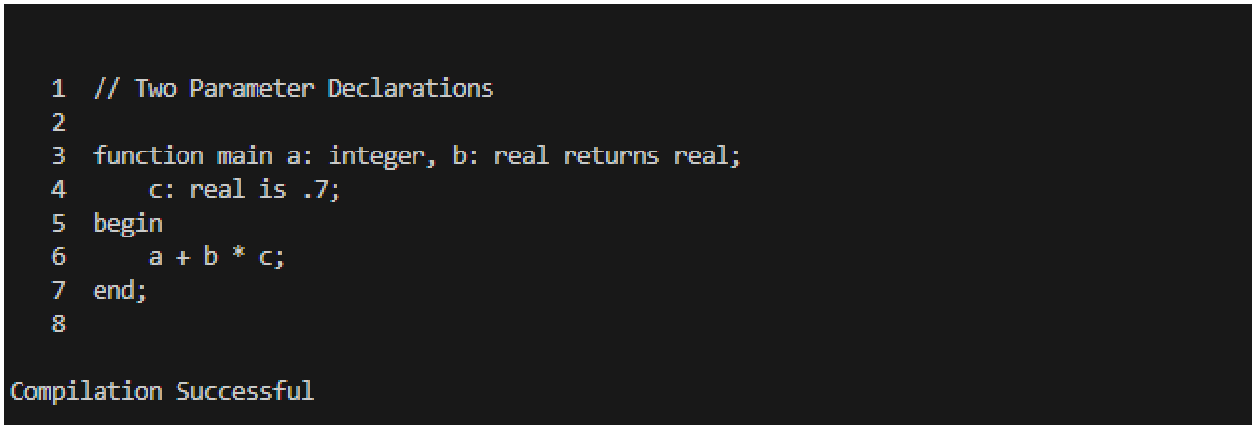
```
// Single Parameter Declaration

function main a: integer returns integer;
begin
    a + 1;
end;

```

```
Compilation Successful
```

## Test Case 15



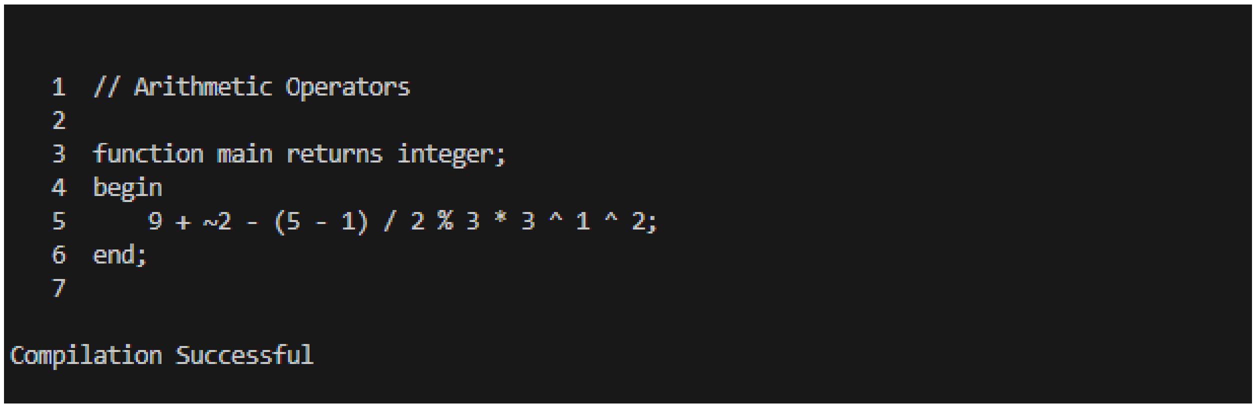
```
// Two Parameter Declarations

function main a: integer, b: real returns real;
    c: real is .7;
begin
    a + b * c;
end;

```

```
Compilation Successful
```

## Test Case 16



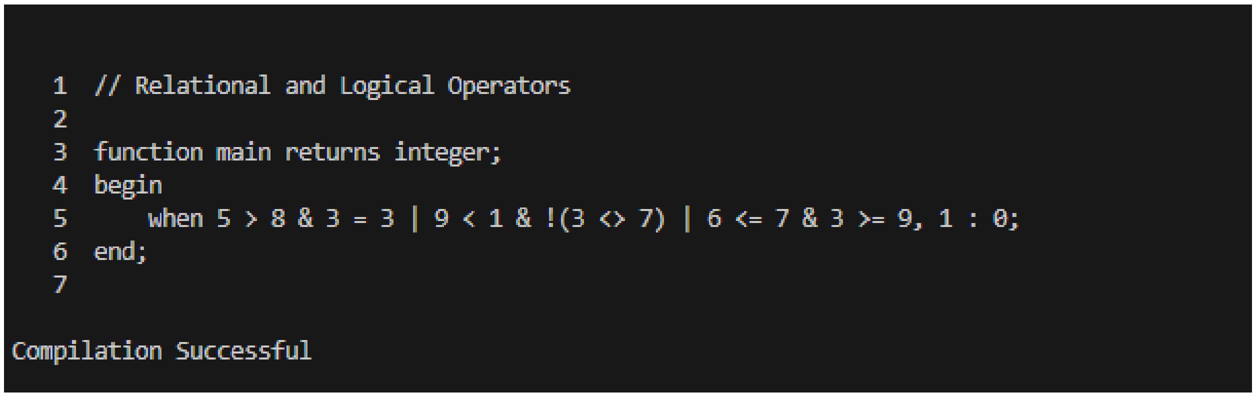
```
// Arithmetic Operators

function main returns integer;
begin
    9 + ~2 - (5 - 1) / 2 % 3 * 3 ^ 1 ^ 2;
end;

```

```
Compilation Successful
```

## Test Case 17



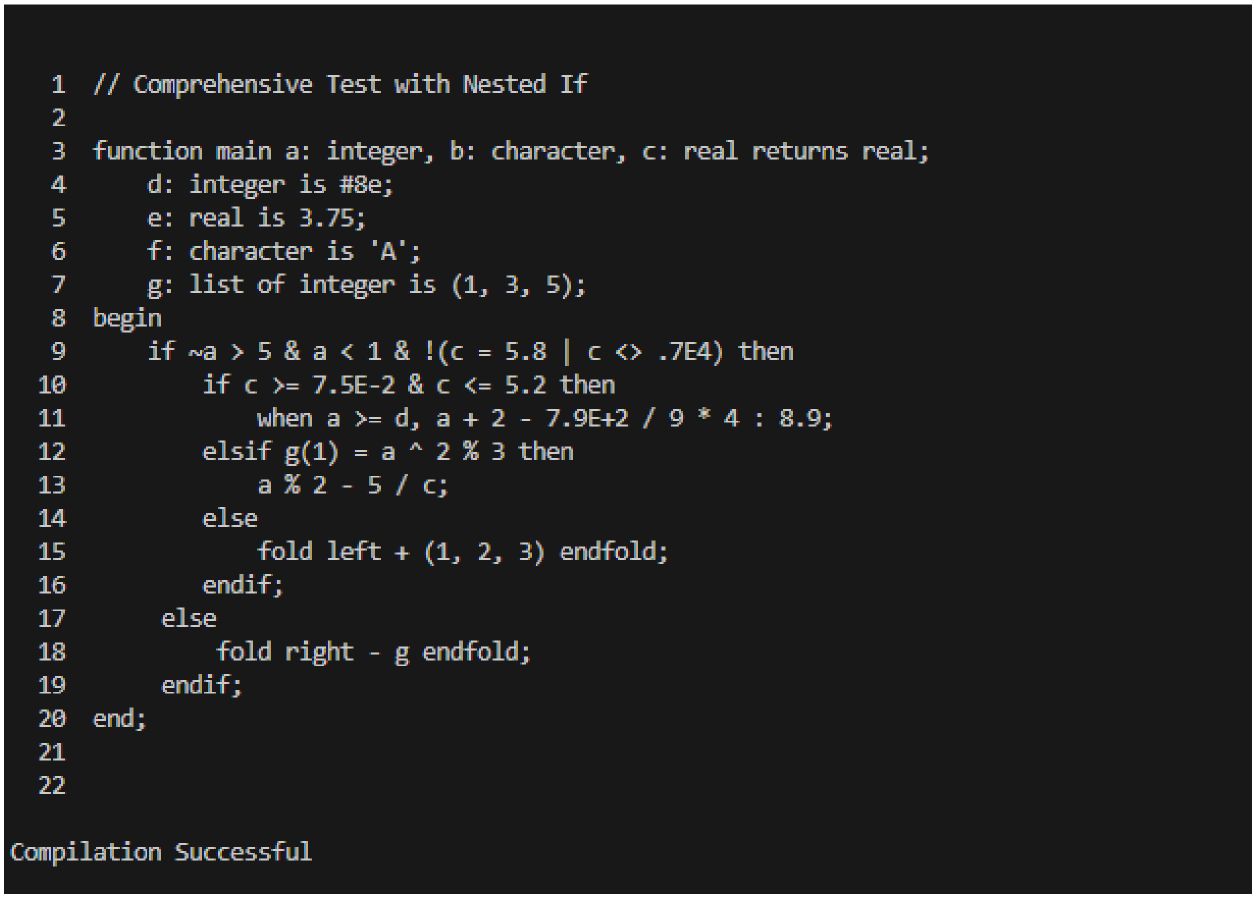
```
// Relational and Logical Operators

function main returns integer;
begin
    when 5 > 8 & 3 = 3 | 9 < 1 & !(3 <> 7) | 6 <= 7 & 3 >= 9, 1 : 0;
end;

```

```
Compilation Successful
```

Test Case 18



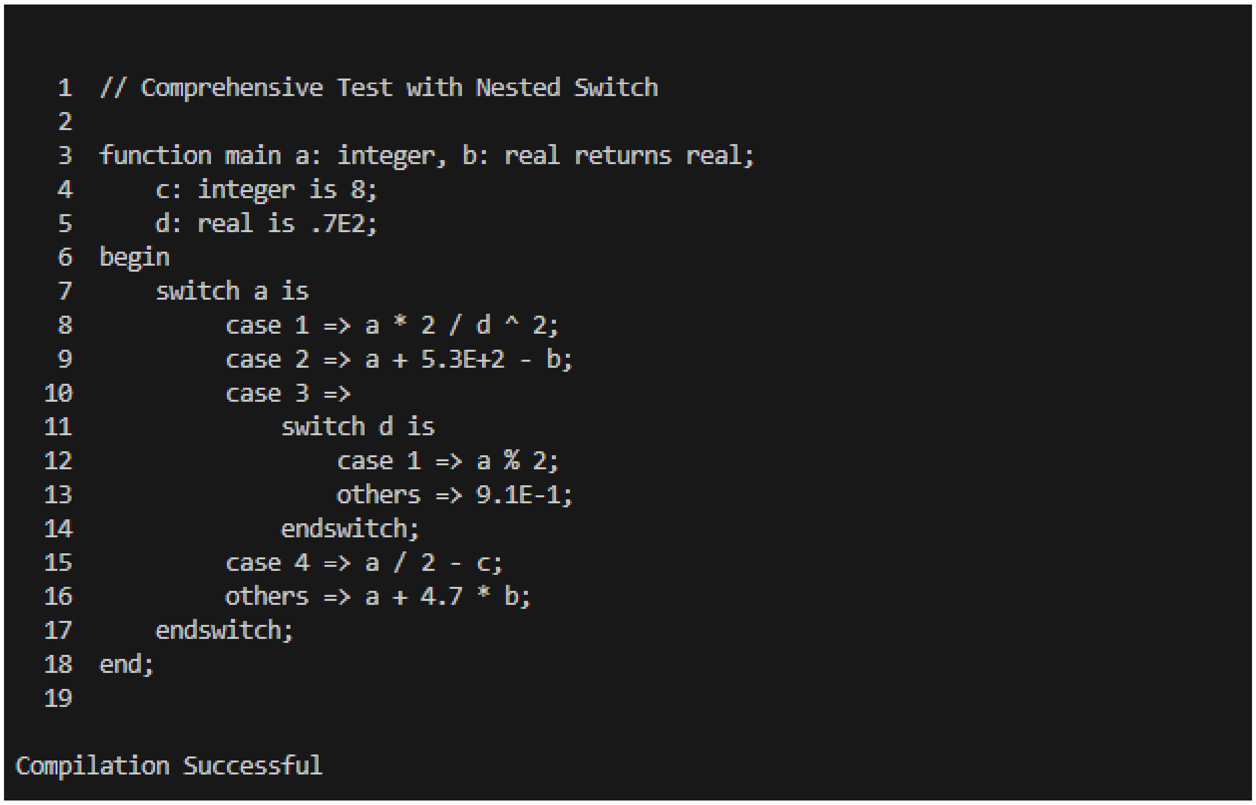
```
// Comprehensive Test with Nested If

function main a: integer, b: character, c: real returns real;
    d: integer is #8e;
    e: real is 3.75;
    f: character is 'A';
    g: list of integer is (1, 3, 5);
begin
    if ~a > 5 & a < 1 & !(c = 5.8 | c <> .7E4) then
        if c >= 7.5E-2 & c <= 5.2 then
            when a >= d, a + 2 - 7.9E+2 / 9 * 4 : 8.9;
        elsif g(1) = a ^ 2 % 3 then
            a % 2 - 5 / c;
        else
            fold left + (1, 2, 3) endfold;
        endif;
     else
         fold right - g endfold;
     endif;
end;



Compilation Successful
```

Test Case 19

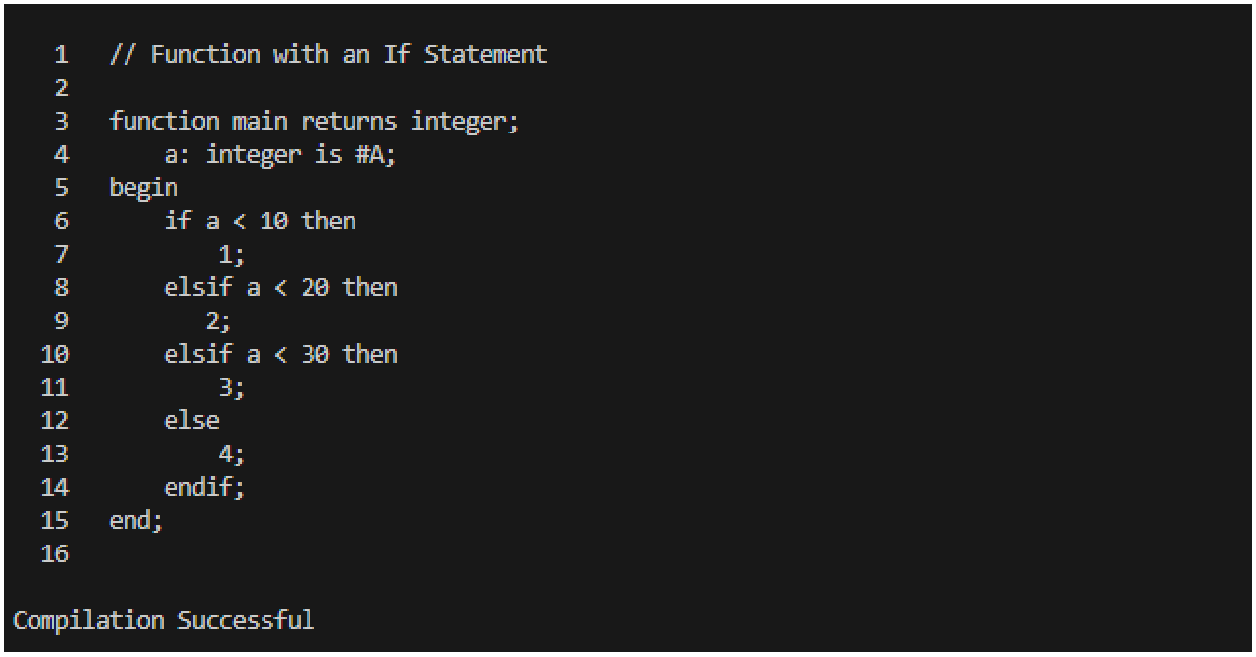


```
// Comprehensive Test with Nested Switch

function main a: integer, b: real returns real;
    c: integer is 8;
    d: real is .7E2;
begin
    switch a is
         case 1 => a * 2 / d ^ 2;
         case 2 => a + 5.3E+2 - b;
         case 3 =>
             switch d is
                 case 1 => a % 2;
                 others => 9.1E-1;
             endswitch;
         case 4 => a / 2 - c;
         others => a + 4.7 * b;
    endswitch;
end;


Compilation Successful
```

### Test Case 20



```
// Function with an If Statement

function main returns integer;
    a: integer is #A;
begin
    if a < 10 then
        1;
    elsif a < 20 then
       2;
    elsif a < 30 then
        3;
    else
        4;
    endif;
end;


Compilation Successful
```

## Approach

It was said that slow and steady wins the race; well, attention to detail doesn't hurt, and both were needed on Project 2. I started by analyzing the requirements and the make file. Taking note of what was being generated and how to get a firm grasp on the flow of information. I started by moving my scanner.l, listing.cc, listing.h, and updating the parser.y to handle the new tokens since it produced the token.h. However, I had trouble with my scanner.l, so I used the skelleton and added my old one as I progressed. I slowly worked my way down the new grammar, adding and modifying the terminals and non-terminals while testing the current and all previous passable tests. At the same time, I took note of the original structure, using it to eliminate the EBNF braces and brackets, add the error productions, and account for the operator precedence and associativity rules.

## Lessons Learned

I learned that being attentive, slow, and steady does win the race. In the early stages of the project, I started over three times. I was having problems with my scanner.l file, so I opted to cut and paste my old one in as I went along since the skeleton was the baseline. Well, I came to find out the main problem was with the '%' token. In project 1, it was called REMOP, and in project 2, it was called MODOP. Everything moved along nicely until the end, when I ran syntax5.txt. It was getting an error in the header, which was checked in a previous test. The error was the comment double dash, so I went through the parser with a fine toothcomb. It wasn't until I noticed some of the tests in the approach were slightly different that I ran the tests off each page. The approach led me to test 5, which is my test 20. It used the integer literals and was failing when it shouldn't, but it led me to the scanner.l literals that I forgot to cut and paste from Project 1. This revelation helped me identify the source of both issues and fix them accordingly.