Department of Computer Science and Engineering

Course Code: CSE 420	**Credits: 1.5**
Course Name: Compiler Design	**Semester:** **Spring 2024**

1 Introduction

So far, we have built a syntax analyzer of a compiler and generated a symbol table with tokens encountered in a given c code. Now, we will perform semantic analysis on our code to detect any semantic errors.

2 Tasks

In this assignment , you have to perform the following tasks:

• **Type Checking:** You have to perform different types of type checking in this part.
You have to perform the following semantic checks:

1. Generate error message if operands of an assignment operator are not consistent with each other. Note that, the second operand of the assignment operator will be an expression that may contain numbers, variables, function calls, etc.
2. Generate an error message if the index of an array is not an integer.
3. Both the operands of the modulus operator should be integers
4. Second operator of modulus and division should not be 0.
5. During a function call all the arguments should be consistent with the function definition.
6. A void function cannot be called as a part of an expression.

• **Type Conversion**:
You have to perform some type-conversion. For example,

1. you have to generate error/warning message if floating point number is assigned to an integer type variable.
2. Also, the result of RELOP and LOGICOP operation should be an integer.

• **Uniqueness Checking:**

1. You should check whether a variable used in an expression is declared or not.
2. Check whether there are multiple declarations of variables with the same ID in the same scope.

• **Array Index:**

1. You have to check whether there is an index used with array and vice versa.

• **Function Parameter:**

1. Check whether a function is called with appropriate number of parameters with appropriate types.
2. A function call cannot be made with non-function type identifier. You have already added necessary info function name symbols in the previous assignment. You can use that.

4 Input

The input will be a file with .c extension containing a c source program. File name will be given from the command line.

5 Output

In this assignment, there will be two output files. The output file should be named as **<Your_student_ID>_log.txt** and **<Your_student_ID>_error.txt**.
The log file will contain **matching grammar rules, and corresponding segment of source code** as the previous assignment. Print the **line count and the error count** at the end of the log file. Print the **error count** at the end of the error file.

The error file will contain all **error/warning messages** encountered while compiling the code. Please make sure not to give errors in multiple lines for one error in an expression.

Please note that your code should not stop working if one error is encountered. Rather, compile till the end of the given c code and capture all errors in the code.

For more clarification about input-output check the supplied sample I/O files given in the lab folder. You are highly encouraged to produce output exactly like the sample one.

6 Submission

1. In your local machine create a new folder whose **name is your student id.**
2. Put the lex file named as **<your_student_id>.l**, the Yacc file **<your_student_id>.y** and a script named **script.sh** (modifying with your own filenames), the **symbol_info.h** file, the **scope_table.h** file, the **symbol_table.h** a suitable input file names **input.c** in a folder **named with your student id**. **DO NOT** put any output file, generated lex.yy.c/y.tab.h/y.tab.c file or any executable file in this folder.
3. Compress the folder in a **.zip file** which should be **named as your student id**.
4. Submit the .zip file.

Failure to follow these instructions will result in penalty.