**Assignment 3 Object Code Generation**

**Problem Statement**

The problem is to determine whether the given source code is semantically correct and to translate it to object code if so. If the parse and semantic analysis pass, the equivalent assembly instructions for the source code will be added to an instruction table. If not, an error message will be printed. There is also type checking that must be done when the source code assigns or manipulates variables. We do this by using a symbol table for keeping track of declared variables. At the end of the parse, the instructions table is output.

**How to use the Program**

To execute the program:

Windows (Visual Studio)

1. Navigate to the folder containing Compiler.exe and RUN “compiler.exe /c pause” from Command Prompt.
2. Enter the .txt file name if it’s in the current directory, or the relative path if it’s not.

Linux

1. Within command prompt, open ‘compiler’ directory.
2. Compile it with the command “g++ compiler.cpp -o compiler”.
3. Run it with “./compiler”
4. When the program runs it will prompt you to enter the source file name’s relative path (ex. ../test.txt).

**Design of the Program**

The list of lexemes from the Lexical Analysis is created in the file named result\_lexemes.txt in the same directory. The list of syntax rules from the Syntactical Analysis is created in the file named result\_syntax.txt in the same directory

The data structure symbol table keeps track of declared variables’ details such as the symbol, memory\_address, size, and type. It has the functions lookup, insert, and print. There is an instructions table that holds all the assembly instructions of the object code. It is printed to the console at the end of the parse.

The algorithm parses the inputted source code for lexical and syntactic correctness. As tokens get parsed successfully, declarations are added to the symbol table and appropriate assembly instructions are added to the instruction table. If the parse of a token is unsuccessful, an error message is outputted. Type checking is done where necessary in the course of the parse.

**Limitations:**

* The symbol table holds a maximum of 200 entries.
* The instructions table holds a maximum of 400 instructions.

**Shortcomings**

None