Assignment 1
Lex

Note: All code must be compilable and executable on Linux systems.
The assignments will be checked on a system running Ubuntu. Code that does not compile will receive 0 marks. There will be no evaluations. C/C++ and JAVA will be accepted.

This document introduces a new programming language C--. Your task is to develop a Lexical Analyzer for this language.

The Lex shall take a text file containing C-- source code as input, and shall generate two text files: the first file shall contain the sequence of token-lexeme pairs, and the second shall list down all the identifiers found (Symbol Table).

Following is asample program written in c--. This is in fact an implementation of the Selection-Sort algorithm.

```
int numPrint(int num, int length)
{
        int i, j, first, temp;
        char a;
        a “= ‘x’;
        out "enter number";
        int i;
        i := length;
        while (i> 0)
        {
                first:= 0;              #this line contains a comment
                j :=1;
                while (j := i)
                {
                        print j;
                        j := j + 1;
                }
                # this is a comment
                i:= i - 1;
                ##This is a
                Multiline
                Comment##
        }
        out “temp is “;
        out temp;
        ret i;
}
```

The language contains the following elements:
data types: int char
Keywords: if else while ret in out

arithmetic operators: + - * /
relational operators: < <= > >= == !=
line comments: start the line with #
multi line comments: ## enclose commentin ##
identifier: a letter followed by any number of letters or digits
numeric constants: only integers
literal constants: a letter enclosed in single quotes
strings: no need to store as variables, only used in out statements
parenthesis, braces, square brackets
assignment operator :=
semi colon
comma

Requirements:

- Complete description of language tokens along with regular definitions and transition diagrams for any complex tokens (those with more than one lexeme). This will be part of the documentation of your language and compiler. Name this documentation.pdf
- A test file that successfully runs your lex. Name this test.cmm
- Complete code of your Lex, which will generate the following text files:
- words.txt: generated sequence of all token-lexeme pairs encountered in the given .cm file
- symboltable.txt: the names of all the program identifiers

Like any good compiler, your lex must print appropriate and helpful errors when incorrect programs are provided to it.

Interface:

Your compiler shall be invoked from the command prompt as follows:

```
> cmm prog1.cm
```

Here cmm is the executable file containing your lex, and prog1.cm is a text file containing any sample program written in C--. Note that the prog1.cm can be replaced with name of any other text file, containing C-- source code.

Make sure you follow all given instructions, and name the files as instructed.

Submission Instructions:
Submit files unzipped
Submit exactly once per group
Do not submit executables.
This deliverable will be marked without an evaluation, so make sure the code is compilable **ON**

<u>LINUX</u>.

Marks will be awarded to registered groups only, so make sure the person you are submitting the assignment with is the person you submitted the form with.