MILITARY INSTITUTE OF SCIENCE AND TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
COURSE CODE: CSE-304, COURSE TITLE: COMPILER SESSIONAL
DATE: 22-04-2021
SPRING 2021

ASSIGNMENT 3 FOR SECTION-A

- Write a lexical analyzer for C.
  - Ignore white space
  - Match all keywords
  - Match all identifiers(variables)
  - Match all operators
  - Match all numbers
  - Match parentheses, curly braces
  - Match comma, semicolon, colon etc.
  - Match literals
  - Count line numbers

Lexemes	**Example**	**Token**
Variables	Variables start with a letter or underscore (_) Ex : a, a9bc, _abc but not 8cde.	Token name is Identifier Ex: Token: <Identifier, _abc>
Numbers	Numbers may contain optional fraction or exponent Ex: 3, 3.056, 3.45E5, 3.45E-2, 3E+2	Token name is Number Ex: Token: <Number, 3.056>
Arithmetic operators	+, -, *, %, /	Token name is Arithmetic Operator Ex: Token :<Arithmetic Operator, +>
Relational operators	< , <=, >=, >	Token name is Relational Operator Ex: Token :< Relational Operator, <=>
Assignment Operator	=	Token name is Assignment Operator Ex: Token :< Assignment Operator, =>
Increment Operator	++	Token name is Increment Operator Ex: Token :< Increment Operator, ++>
Decrement Operator	--	Token name is Decrement Operator Ex: Token :< Decrement Operator, -->
Keywords	If, else, switch, case, while, for, int, break, default, main, printf	Token name is Keyword Ex: Token is <Keyword, if>
Parentheses	(or)	Token name is Parentheses Ex: Token: < Parentheses , (>
curly braces	{ or }	Token name is curly Ex: Token: < curly , }>

Lexemes	Example	Token
comma, semicolon, colon	, ; :	Token name is separator Ex: Token: <Separator, :>
literals	“Anything…..”	Token name is literal Ex: Token: <Literal, “Anything…”>

Now, For each token print the corresponding token name and line no of occurring.
Take input from file and give output to file
See Sample Input/Output For Further Reference