**REQUIREMENTS**

**Software Used:**

Code Blocks IDE version 20.03

**Functional Requirements**:

Since this project uses C language, it is designed only for the subset of C. The Lexical Analyzer is designed as a procedure that will be called every time a token is needed by the parser. The routine consists of at least two parameters that are returned to the calling routine: “Code” which is the numeric representation of the token and “String” which is the character representation of the token.

On reading the characters from the input given by the user:

* It should identify all the keywords.
* It should identify all the operators.
* It should identify all the valid Identifiers.
* It should identify the literals such as integer and real numbers (in this case).

And should print the tokens along with their symbol types.

For identification, the input tokens given by the user that are taken up the parser are compared with the entities within the defined function. On return of a Boolean value the further action takes place. Another, function is written wherein the Boolean values are compared using conditional loops and finally the tokens along with their types are printed.

## Working:

Lexical analyzer scans the entire source code of the program. It takes the input in the form of a character stream. And further produces a token stream as shown in the figure below. It identifies each token one by one.

For Example:

In this project when the user provides the following input:

**int a = 9.7 + 2**

It would result in giving the following output based on the programming done and the working explained above.

 ‘int’ is a keyword

 ‘a’ is an identifier

  ‘=’ is an operator

 ‘9.7’ is a real number

  ‘+’ is an operator

  ‘2’ is an integer