EXPERIMENT NO: 1

21/09/23 LEXICAL ANALYZER USING C

AIM

Design & Implement a lexical analyzer for given language using C & the lexical analyzer should ignore redundant spaces, tabs & new lines.

THEORETICAL BACKGROUND

A lexical analyzer, also known as a lexer or a scanner, is the first phase of a compiler or interpreter. It's main task is to read the source code of a program & break it down into meaningful units called tokens. These tokens are the building blocks of the program & represent different types of elements such as keywords, identifiers, literals, operators & punctuations.

(i) Tokenization :-

* process of dividing the source code into tokens.
* tokens can include keywords, identifiers, literals, operators & special symbols.

(ii) Regular expressions:

* Regular expressions are used to define search patterns that match tokens in the source code

(iii) Lexical Rules :-

* They specify how tokens are recognized & constructed.
* These rules specify how ~~tokens are~~ characters for different token types & the order in which they should appear.

(iv) Ignoring white spaces

* One crucial function of the lexer is to ignore redundant whitespace characters like spaces, tabs & newlines.
* whitespace is essential for code readability but doesn't

impact the meaning of the program.

* Ignoring whitespace ensures the code's formatting doesn't interfere with tokenization.

CONCLUSION

The program was successfully implemented & executed.

ALGORITHM

1. Start
2. Include necessary header files.
3. Define a function 'iskeyword' to check if a given string is a keyword in C. It takes a character array as input & returns 1 if it is a keyword 0 otherwise.
4. In the main function:-
   4.1 Declare necessary variables
   4.2 Open a file named "compression.txt" for reading
   4.3 Start a loop to read characters from the file until EOF.
   4.4 Check for '\n'. If true, increment newline counter.
   4.5 Check for alphanumeric:-
      * If first character digit :- error
      * else, add to buffer.
   4.6 Check the buffer if it is keyword or identifier.
   4.7 Close the file
5. Stop.