**Experiment No. 4**

**Implement lexical analyzer.**

Lexical Analysis is the first phase of the compiler also known as a scanner. It converts the High level input program into a sequence of **Tokens**.
- Lexical Analysis can be implemented with the Deterministic finite Automata.
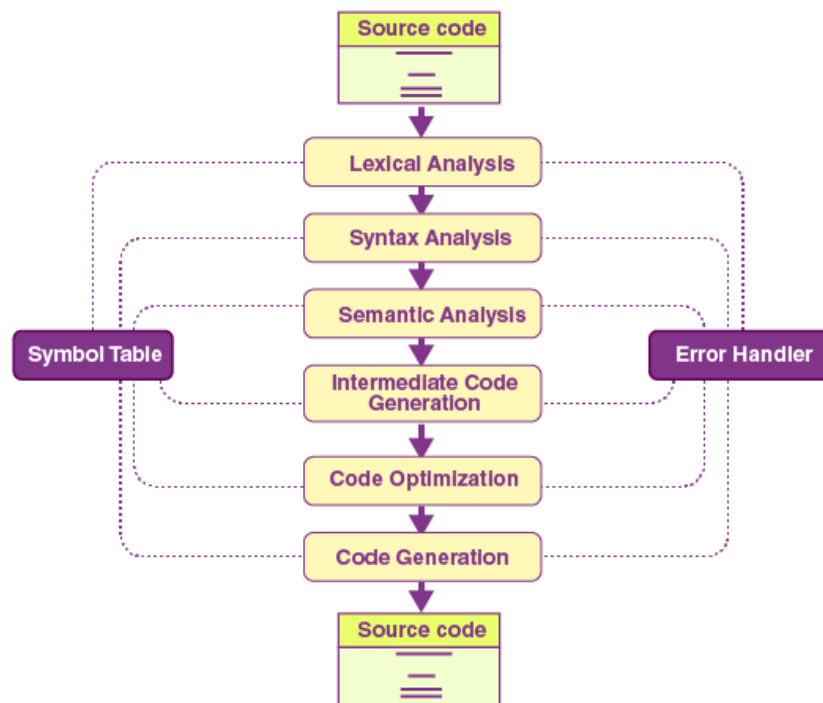- The output is a sequence of tokens that is sent to the parser for syntax analysis
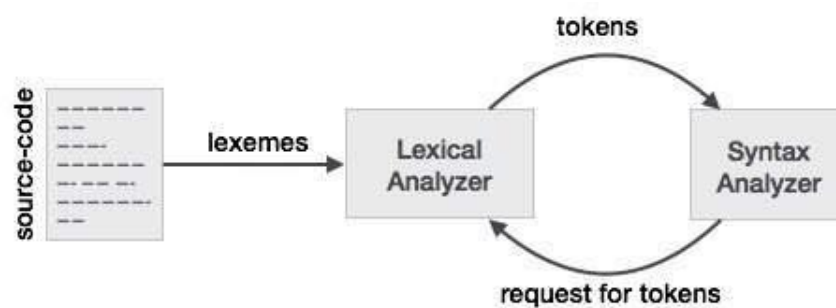


**Fig. Phases of Compiler**



**Fig. Lexical Analyzer**

**Token** : A lexical token is a sequence of characters that can be treated as a unit in the grammar of the programming languages.

**Example of tokens:**
- Type token (id, number, real, . . . )
- Punctuation tokens (IF, void, return, . . . )
- Alphabetic tokens (keywords)

**Example of Non-Tokens:**
- Comments, preprocessor directive, macros, blanks, tabs, newline, etc.

**Lexeme**: The sequence of characters matched by a pattern to form the corresponding token or a sequence of input characters that comprises a single token is called a lexeme. eg- "float", "ty_b_rollno_01", "=", "-", "273", ";" .

**How Lexical Analyzer works-**
1. Input preprocessing: This stage involves cleaning up the input text and preparing it for lexical analysis. This may include removing comments, whitespace, and other non-essential characters from the input text.
2. Tokenization: This is the process of breaking the input text into a sequence of tokens. This is usually done by matching the characters in the input text against a set of patterns or regular expressions that define the different types of tokens.
3. Token classification: In this stage, the lexer determines the type of each token. For example, in a programming language, the lexer might classify keywords, identifiers, operators, and punctuation symbols as separate token types.
4. Token validation: In this stage, the lexer checks that each token is valid according to the rules of the programming language. For example, it might check that a variable name is a valid identifier, or that an operator has the correct syntax.
5. Output generation: In this final stage, the lexer generates the output of the lexical analysis process, which is typically a list of tokens. This list of tokens can then be passed to the next stage of compilation or interpretation.

Conclusion: