DATE:

Nikkita Gurani
D12A
20

# LAB-02

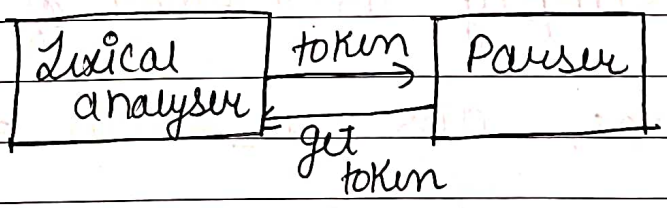
**Aim:** To implement lexical analyser.

**Theory:-**

1) What is lexical analysis?

→ Lexical analysis is the first phase of compiler. It is also known as scanner. It converts high level input program into a sequence of Tokens.

- Lexical Analysis can be implemented with DFA (Deterministic Finite Automata)
- The output is then sent to parser for syntax analysis



DATE:

Token → A lexical token is a sequence of characters that can be treated as a unit in grammar of programming language.

Example of token →

- Type token (id, number, real, ...)
- Punctuation tokens ( IF, void, return)
- Alphabet tokens (keywords)

Keywords; Examples- for, while, if etc.
Identifier: Examples- Variable name, function name, etc.
Operators: Examples- '+', '++', '-' etc.
Separators: Examples- ',', ';', etc.

* Lexeme:- The sequence of characters matched by pattern to form corresponding token / sequence of input characters that comprises a single token is called lexeme. It is simply an instance of token.

Eg. - float, abs_zero_kelvin, =, -, 273;

* Perimeter = 2 * (l+b);

Eg-, Perimeter, =, 2, *, (, l, +, b, );

DATE:

Pattern → It specifies a set of rules that a scanner follows to create a token. The sequence of characters that make a keyword.

It must start with the alphabet, followed by alphabet or a digit.

These rules describing all those lexeme that can describe a token.

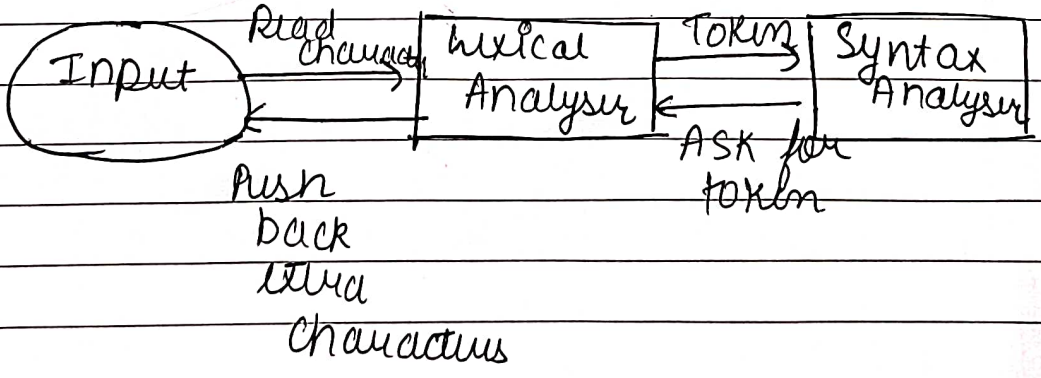
Type of operator : +, =

Type of punctuation : (, ), {, }

Any string of characters (Except ") between " and ".

Example → Identifier → start with alphabet/ underscore followed by any alphanumeric char ^ht → (0-9)*

# Diagram with example-



- The lexical analyser is responsible for removing white spaces and comments f source program.
- It corresponds to error messages with s program.
- It helps to identify the tokens
- The input characters are read by lexic analyser from source code.

Conclusion -