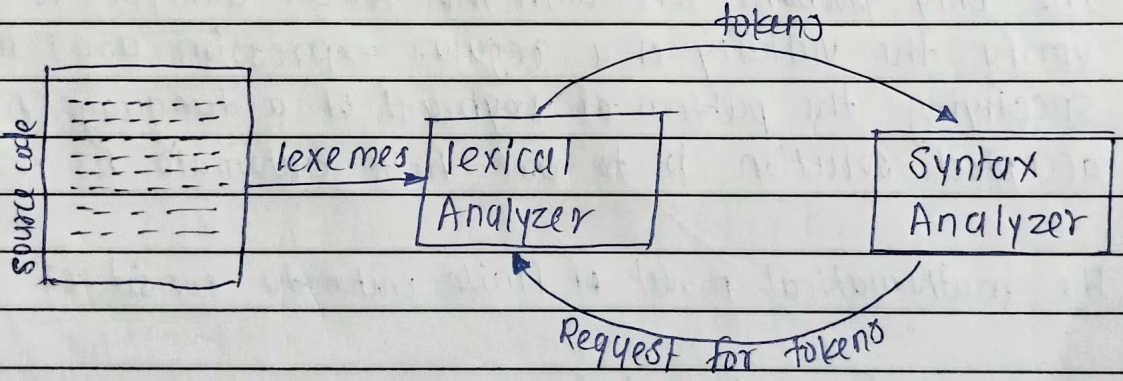
Page No.
Class
Roll No.
Date: / /202

Practical No 01

Title:- Design a lexical analyzer for given language and the lexical analyzer should ignore redundant spaces tabs and new lines.

Theory:-

Lexical analysis is the first phase of a compiler It takes the modified source code from languages preprocessors that are written in the form of sentences the lexical analyzer breaks these syntaxes into a series of token by removing any whitespaces or comments in the source code



Tokens:

lexemes are said to be sequence of characters (alphabets in tokens there are some predefined rules for every lexemes to be identified as a valid token.

In programming language keywords constant identifier strings numbers operator and punctuation symbol can be considered as a tokens

Page No.
Class
Roll No.
Date: / /202

for ex. in C language the variable declaration line
Int val 100;
Contain the tokens;

in (keyword), value (identifier) = (operator), 100 (constant)
and ; (systems)

Representing language tokens using regular expression
Decimals = (sign)* (digit)+
Identifier = (letter) (letter/digit)*

The only problem left with the lexical analyzer is how
verify the validity of a regular expression used in
specifying the pattern of keyboard of a language A well
accepted solution is to use finite automato for verifica

The mathematical model of finite automata consist of:

- finit set of state (Q)
- finite set of input symbol (Σ)
- One start state (q0)
- set of final state (qf)

Page No.
Class
Roll No.
Date: / /202

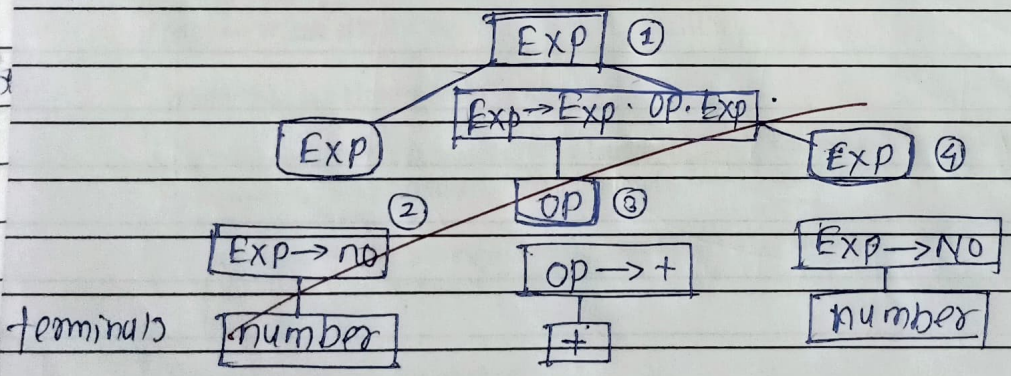
- Transition function (δ)

The transition function (δ) maps the finit set of state Q to a finite set of input symbol (Σ), $Q \times \Sigma \rightarrow Q$

Importance of lexical Analyzer.

- lexical analyzer makes writing a parser much easier
- Instead of having to build up names such as "networth - future" from their individual characters
- this leads to efficiency of programming

Ex.



Results :-

Thus the program for designing of lexical analyzer for a given language is performed successfully.