Slip test - 2 10/02/2023

Part - A :-

1. Differentiate tokens, patterns, lexeme.

Tokens :-

A token is a sequence of characters with collective meaning.

the tokens are keywords, identifiers literals, variables, constants.

Lexeme :-

A lexeme is defined as a value for identifiers. It is a sequence of alphanumeric characters in a token.

~~lexeme~~ patterns :-

A pattern is a rule for defining a token.

Eg :-

Lexemes	tokens	patterns
int	keyword	-

2. What is lexeme? define a regular set :

lexeme :-

lexeme is defined a value for identifiers. It is a sequence

of alphanumeric characters in a token.

Eg:- int
a
*

Regular set :-

Any set that represents the value of a regular Expression is called regular set. the regular expressi are ϵ, ϕ and any Symbol.

3. what are the error recovery actions in a lexical analyzer?

* Removes one character from the remaining input.

* In panic mode, the successi characters are always ignored until we reach a well-formed token.

* By inserting the missing character into the remaining input.

* Replacing a character with another.

* transpose two serial characters.

4. Write short notes on buffer pair :-

Buffer pair is a specialized buffering technique can decrease the amount of overhead, which is need to process an input character in transferring characters.

It consists of two identifiers

It has two pointers lexem, forward, which points the begining.

5. Define the regular definition for Number :-

digit → 0|1 . . . |9.

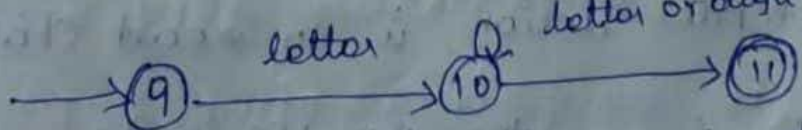
digits → digit digit *

optional fraction → . digits | ∈

optional Exponent → E(+|-|∈) digits | ∈

Num → digits optionalfraction optional Exponent.

6. Transition diagram for identifier :-



tokens are recognized by finite automata.

Part - B :

7. Analyze the roles of lexical analyzer with suitable Example?
Draw the transition diagram for that recognizes the lexemes matching the tokens relop (relational operator and identifier).

Lexical analysis :

* Lexical analysis is the very first phase in the compiler designing.

* program that perform lexical analysis in compiler design are called lexical analyzers.

Role of lexical analyzer :

The role of lexical analyzer in compiler design is to read character streams from the source code, check for legal tokens and pass the data to the Syntax analyzer.

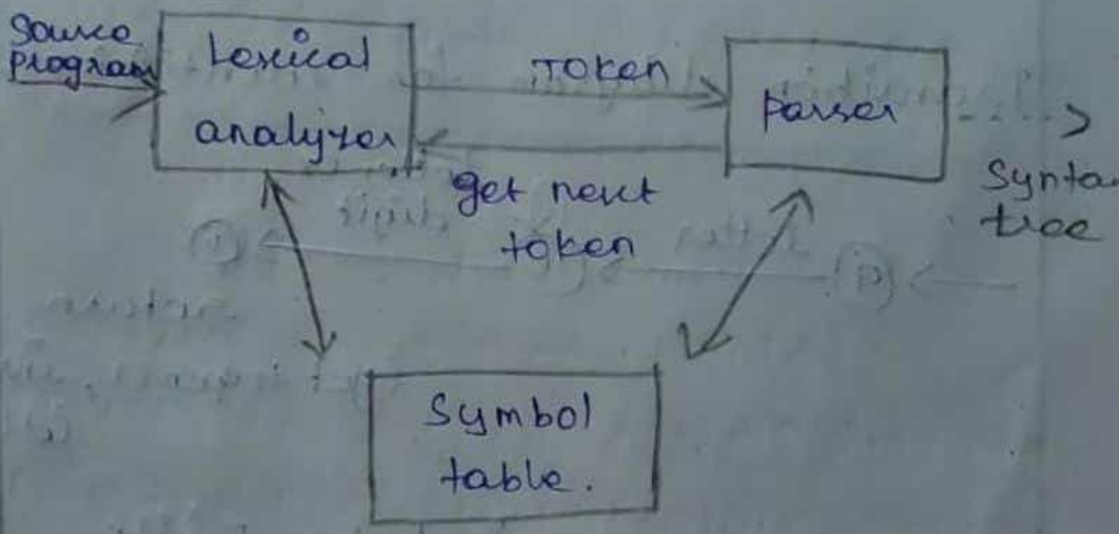
* It helps to identify token into the Symbol table.

* Removes white spaces and Comments from the source program.

* It correlates error messages with source program.

* It helps you to read input Characters from source program.

Diagram:-

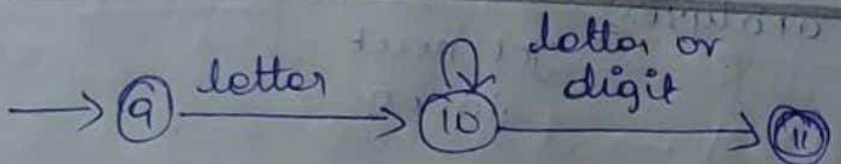


Example:-

```
#include <stdio.h>
int maximum (int x, int y).
int main().
{
    if (x > y)
        return x;
    else
{ return y;
}
}
```

lexeme	Token
int	keyword
(	operator
x	identifier
,	identifier
if	keyword
else	keyword
)	operator
,	operator

Transition diagram for identifiers :-



token are recognized by Finite automata.

* Identifiers are symbols used to uniquely identify a program element in the code.

* they are also used to refer to types, constants, parameters

Transition diagram for relational operators:-