8. VIVA QUESTIONS

1. **Define system software**.
   System software is computer software designed to operate the computer hardware and to provide a platform for running application software. Eg: operating system, assembler, and loader.
2. **What is an Assembler?**
   Assembler for an assembly language, a computer program to translate between lower-level representations of computer programs.
3. **Explain lex and yacc tools**
   α. Lex: - scanner that can identify those tokens
   β. Yacc: - parser.yacc takes a concise description of a grammar and produces a C routine that can parse that grammar.
4. **Explain yyleng?**
   yyleng-contains the length of the string our lexer recognizes.
5. **What is a Parser?**
   A Parser for a Grammar is a program which takes in the Language string as it's input and produces either a corresponding Parse tree or an Error.
6. **What is the Syntax of a Language?**
   The Rules which tells whether a string is a valid Program or not are called the Syntax.
7. **What is the Semantics of a Language?**
   The Rules which gives meaning to programs are called the Semantics of a Language.
8. **What are tokens?**
   When a string representing a program is broken into sequence of substrings, such that each substring represents a constant, identifier, operator, keyword etc of the language, these substrings are called the tokens of the Language.
9. **What is the Lexical Analysis?**
   The Function of a lexical Analyzer is to read the input stream representing the Source program, one character at a time and to translate it into valid tokens.
10. **How can we represent a token in a language?**
    The Tokens in a Language are represented by a set of Regular Expressions. A regular expression specifies a set of strings to be matched. It contains text characters and operator characters. The Advantage of using regular expression is that a recognizer can be automatically generated.
11. **How are the tokens recognized?**
    The tokens which are represented by an Regular Expressions are recognized in an input string by means of a state transition Diagram and Finite Automata.
12. **Are Lexical Analysis and Parsing two different Passes?**
    These two can form two different passes of a Parser. The Lexical analysis can store all the recognized tokens in an intermediate file and give it to the Parser as an input. However it is more convenient to have the lexical Analyzer as a co routine or a subroutine which the Parser calls whenever it requires a token.
13. **What are the Advantages of using Context-Free grammars?**
    α. It is precise and easy to understand.
    β. It is easier to determine syntactic ambiguities and conflicts in the grammar.
14. **If Context-free grammars can represent every regular expression, why do one needs R.E at all?**
    α. Regular Expression are Simpler than Context-free grammars.

β. It is easier to construct a recognizer for R.E than Context-Free grammar.
χ. Breaking the Syntactic structure into Lexical & non-Lexical parts provide better front end for the Parser.
δ. R.E are most powerful in describing the lexical constructs like identifiers, keywords etc while Context-free grammars in representing the nested or block structures of the Language.

15. **What are the Parse Trees?**
Parse trees are the Graphical representation of the grammar which filters out the choice for replacement order of the Production rules.

16. **What are Terminals and non-Terminals in a grammar?**
Terminals:- All the basic symbols or tokens of which the language is composed of are called Terminals. In a Parse Tree the Leafs represents the Terminal Symbol.
Non-Terminals:- These are syntactic variables in the grammar which represents a set of strings the grammar is composed of. In a Parse tree all the inner nodes represents the Non-Terminal symbols.

17. **What are Ambiguous Grammars?**
A Grammar that produces more than one Parse Tree for the same sentences or the Production rules in a grammar is said to be ambiguous.

18. **What is bottom up Parsing?**
The Parsing method is which the Parse tree is constructed from the input language string beginning from the leaves and going up to the root node.
Bottom-Up parsing is also called shift-reduce parsing due to its implementation. The YACC supports shift-reduce parsing.

19. **What is the need of Operator precedence?**
The shift reduce Parsing has a basic limitation. Grammars which can represent a left-sentential parse tree as well as right-sentential parse tree cannot be handled by shift reduce parsing. Such a grammar ought to have two non-terminals in the production rule. So the Terminal sandwiched between these two non-terminals must have some associability and precedence. This will help the parser to understand which non-terminal would be expanded first.

20. **What is exit status command?**
Exit 0- return success, command executed successfully.
Exit 1 – return failure.

21. **Define API's**
An application programming interface (API) is a source code based specification intended to be used as an interface by software components to communicate with each other.