|  |
| --- |
| 1. Why is buffering used in lexical analysis? What are the commonly used buffering methods? |
| 1. What are the various representations of Intermediate languages? |
| 1. An arithmetic expression with unbalanced parenthesis is lexical or syntax error. Comment on it.- |
| 1. Describe the language denoted by the regular expressions b\*ab\*ab\*ab\*. |
| 1. Construct the syntax tree for the following assignment statement: a:=b\*- c+b\*-c. |
| 1. Obtain the regular expressions for the following sets: 2. The set of all strings over {a, b} beginning and ending with ‘a’.   The set of all strings over {b} such that string belong to {b2 , b5 , b8 , …}. |
| 1. Identify the type of error in the following statements.   y = 3 + \* 5;  int a = "hello";  Justify this with the suitable points. |
| 1. Comment on the efficiency of the compiler if the number of passes in compilation is increased. |
| 1. What are the various representations of Intermediate languages? |
| 1. A compiler that translates a high-level language into another high-level language is called a source- to-source translator. What advantages are there to use C as a target language for a compiler? |
| 11. Comment on the efficiency of the compiler if the number of passes in compilation is increased |
| 1. Would it be better if the phases of the compiler are combined into a single phase? |
| 1. Differentiate compilers from interpreters. |
| 1. Explain how a language is processed with the suitable diagram and assume the c/c++ language |
| 1. Obtain the regular expressions for the following having Σ={0,1}. 2. Strings of 0’s and 1’s beginning with 0 and ending with 1. 3. {x | x contains an even number of 0’s or an odd number of 1’s}. |
| 1. Generate the target code for the following code fragment. Assume a=(b+c)\* (b+c)\* as input to the lexical analyzer and try to show the intermediate outputs of all the phases from lexical analyzer to code generator. |
| 1. Discuss the working of various phases of Compiler. Interpret the output for each phase for the following assignment statement a: = b + c \* 50. |
| 1. A compiler can choose one of the two options.    * 1. Translate the input source into intermediate code and then convert it to final machine code.      2. Directly generate the final machine code from the input source.   What is the preferred option and why? Justify this with the suitable claims. |
| 1. Compare tokens, patterns and lexemes. 2. Count the number of tokens in the following code snippet.   int main()  {  a=10,b=20;  printf(“ sum is : %d”, a+b;  return 0;  } |
| 1. Explain in detail the structure of the compilers. Compare language dependent and independent phases. |
| 1. How to solve the source program to target machine code by using language processing system? |
| 1. Why do we separate the analysis phase into lexical and parsing phases? |
| int fact;  int factorial (int n)  {  int val;  if(n>1){  val=n\*factorial (n-1);  return (val);  }  else  {  return(1);  }  }  int main()  { printf (“factorial program”);  Fact 5=factorial(5);  Printf(“fact=5=%d \n”, fact5);  }  Consider the above program, what are the data structures used to store the information, how the symbol table management is handled? |
| 1. Construct transition diagram for keyword, identifiers and relational operators. |
| 1. Obtain the required DFA from the regular expression (a/b)\* a (a/b) using Minimization of DFA using Π new constructions. Write the suitable algorithm. |
| 1. Explain in detail the structure of the compilers. Compare language dependent and independent phases |
|  |
| 1. Obtain the NFA from the following regular expression (a+b) abb (a/b)\* using Thompson construction. Convert it into DFA and then minimize the DFA. Describe the sequence of moves made by each in processing the input string ababba. |
| 1. What are the advantages and disadvantages of intermediate language? |
| 1. Explain the role performed by lexical analysis of the compiler |
| 1. Write down the possible error recovery actions taken by lexical analyzer |
| 1. Define the purpose of gettoken( ) function |
| 1. Define the purpose of installid( ) function |
| 1. State the difference between linear, hierarchical and syntax analysis. |
| 1. Explain Symbol table management and error handling |
| 1. Explain construction tools /cousins of compiler. |