|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **P.V.P Siddhartha Institute of Technology** | | | | | | |
| **Department of Computer Science and Engineering** | | | | | | |
| **Course: B.Tech.** | | **Year: III** | **Semester: II** | **A.Y:2024–25** | |  |
| **Subject Code:** 20**CS3601** | | **Subject Name: Compiler Design** | | | **Regulation:PVP20** | |
| **ASSIGNMENT-1** | | | | | | |
| **Q. No** | **QUESTION** | | | | | |
| |  |  |  |  | | --- | --- | --- | --- | | 1. | Test whether the grammar is LL(1) or not, and  construct a predictive parsing table for following  grammar  S→A  A→aB | Ad  B→bBC | f  C→g |  |  | | 2. | Explain about Left factoring and Left Recursion with examples. |  |  | | 3. | Eliminate the left recursion for the following grammar  E→E+T | T  T→T\*F | F  F→(E) | id |  |  | | 4. | Calculate FIRST and FOLLOW for the following grammar:  E→E+T | T  T→T\*F | F  F→(E) | id |  |  | | 5. | Sketch syntax tree for the expression a=b\*– c+b\*– c. |  |  | | 6. | Consider the grammar.  S->AB|Abad  A->d  E->b  D->b| ε  B->c  Construct the predictive parsing table. Show that the given grammar is LL(1) or not. |  |  | | 7. | Show the following Grammar  S->AaAb|BbBa  A-> ε  B-> ε  Is LL(1) and parse the input string “ba”. |  |  | | 8. | Consider the following Grammar:  A-> ABd|Aa|a  B-> Be|b  Remove left recursion. |  |  | | 9. | Consider the following grammar:  E->E+E  E->E\*E  E->id  Construct shift reduce parsing of the input  string “id1+id2+id3”. |  |  | | 10. | Do left factoring in the following grammar:  A-> aAB|aA|a  B-> bB|b |  |  | | 11. | Let G be a Context Free Grammar for which the production Rules are given below:  S -> aB|bA  A -> a|aS|bAA  B -> b|bS|aBB  Drive the string aaabbabbba using the above grammar (using Left Most Derivation and Right most  Derivation). |  |  | | 12. | Given the CFG G = {S, {S, U, V, W}, {a, b, c, d}, P} with P given as shown below :  S -> UVW  U -> (S) | aSb | d  V -> aV | ε  W -> cW | ε  Construct a table-based LL(1) predictive parser. |  |  | | 13. | .Construct predictive parser for the following grammar  S-->(L)/a  L-->L,S/S and parse any input string. |  |  | | 14. | Construct the Predictive parse table (LL parse table) for the following grammar:  S->aBDh,  B->cC,  C->bc| ε ,  D->EF ,  E->g| ε ,F->f| ε |  |  | | 15. | Eliminate left recursion and left factoring for the following grammar.  E → E + T | E - T | T  T → a | b | ( E ). |  |  | | 16. | Eliminate left factoring for the following grammar.  S->iEtS|iEtSeS|a  E->b |  |  | | 17. | Construct Predictive Parsing Table for the following grammar.  S->iEtSS’ | a  S’->eS| ε  E->b  Check whether the given grammar is in LL(1) or not. |  |  | | 18. | Construct Predictive Parsing Table for the following grammar.  X->d semi X | sY  Y->semi s Y | ε |  |  | | 19. | Calculate FIRST and FOLLOW for the following grammar:  S’->S#  S->ABd  A->dB  A->aS  A->d  B->AS  B->b |  |  | | | | | | | |