SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Ramapuram Campus, BharathiSalai, Ramapuram, Chennai - 600089

**FACULTY OF ENGINEERING AND TECHNOLOGY**

# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

****

**QUESTIONBANK**

**DEGREE / BRANCH: B.TECH-CSE**

**VISEMESTER**

**18CSC304J – COMPILER DESIGN**

**2018 Regulation**

**Academic Year 2022-2023 EVEN SEMESTER**

# SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

**Ramapuram Campus, BharathiSalai, Ramapuram, Chennai-600089**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**QUESTIONBANK**

**SUBJECT : COMPILER DESIGN**

**SEM/YEAR: VI/III**

**Course Outcomes**

CLR-1: Utilize the mathematics and engineering principles for the Design of Compilers

CLR-2: Acquire knowledge of Lexical Analyzer from a specification of a language's lexical rules

CLR-3: Acquire knowledge of Syntax Analyzer for parsing the sentences in a compiler grammar

CLR-4: Gain knowledge to translate a system into various intermediate codes

CLR-5: Analyze the methods of implementing a Code Generator for compilers

CLR-6: Analyze and Design the methods of developing a Code Optimizer

| **UNIT- II** | | | |
| --- | --- | --- | --- |
| Syntax Analysis Definition - Role of parser Lexical versus Syntactic Analysis- Representative Grammars Syntax Error Handling- Elimination of Ambiguity, Left Recursion Left Factoring - Top down parsing Recursive Descent Parsing, back tracking Computation of FIRST Problems related to FIRST Computation of FOLLOW Problems related to FOLLOW Construction of a predictive parsing table- Predictive Parsers LL(1) Grammars Transition Diagrams for Predictive Parser Error Recovery in Predictive Parsing Predictive Parsing Algorithm Non Recursive Predictive Parser | | | |
| **PART-A (Multiple Choice Questions)** | | | |
| **Q.**  **No** | **Questions** | **Course Outcome** | **Competence**  **BT Level** |
| **1** | **A bottom-up parser generates**  a) Right most derivation b) Right most derivation in reverse c) Leftmost derivation d) Leftmost derivation in reverse  **Answer :b** | CO3 | BT1 |
| **2** | **A grammar A → AA | (A) | e is not suitable for predictive-parsing because the grammar is**  a) Ambiguous b) Left recursive c) Right recursive d) An operator grammar  **Answer: b** | CO3 | BT3 |
| **3** | **Shift reduce parsers are**  a) Top-down Parser b) Bottom-Up parser c) May be top down or bottom up d) Both top down and Bottom up  **Answer:b** | CO3 | BT1 |
| **4** | **Which phases of compiler is Syntax Analysis?**  a) First b) Second c) Third d) Fourth  **Answer :b** | CO3 | BT1 |
| **5** | Consider the following grammar  S🡪 L=R  S🡪R  L🡪\*R  L🡪id  R🡪L  Find the First(S) and Follow (R)   1. {\*,id} {=, $} 2. {\*} {=} 3. {id} { $} 4. {id,id} {$}   Ans: a | CO3 | BT5 |
| **6** | **Which of the following derivations does a top-down parser use while parsing an input string?**  a) Leftmost derivation b) Leftmost derivation in reverse c) Rightmost derivation d) Rightmost derivation in reverse  **Answer: a** | CO3 | BT1 |
| **7** | Which phase of the compiler is also known as Parser?  a)Code Optimization  b)Semantic Analysis  c)Syntax Analysis  d)Lexical Analysis  **Answer:** c. | CO3 | BT2 |
| **8** | **The left factored grammar for the following grammar is--------- A→∝β1|∝β2**  a) A→∝A' A'→β1|β2 b) A→∝A'|∈ A'→β1|β2 c) A→∝A' A'→β1|β2|∈ d) A→A' A'→β1|β2 **SHW ANSWER**  Ans: A | CO3 | BT1 |
| **9** | **----------grammar can have more than one parse tree generating a given string of a terminals.**   1. Ambiguous 2. Context free 3. Context sensitive 4. Unambiguous   Answer: a | CO3 | BT2 |
| **10** | **Which of the following is not example of Top-Down Parser?**   1. Recursive Descent Parser 2. Predictive Parsing 3. Shift reduce parser 4. Non-recursive predictive parsing   Answer: c | CO3 | BT1 |
| **11** | **What does a Syntactic Analyzer do?** a) Maintain Symbol Table b) Collect type of information c) Create parse tree d) Create table  **Answer: c** | CO3 | BT1 |
| **12** | **Eliminate left-recursion from the following grammar : S → (L) | a L → L,S | S After eliminating left-recursion we get,------**  a) S → (L) | a        L → SL'        L'→ ,SL'|∈ b) S → (L) | a        L → SL'        L'→ ,SL' c) S → (L) | a        L → L'S        L'→ ,SL' d) S → (L) | a        L → SL'L’        L'→ ,SL'|∈  Answer: a | CO3 | BT4 |
| **13** | **Consider the following grammar S→iEtSS′|a S′→eS|ε E→b In the predictive parse table, M, of this grammar, the entries M[S′,e] and M[S′,ϕ] respectively are----------**   1. {S′→eS} and {S′→ε} 2. {S′→eS} and { } 3. c) {S′→ε}and {S′→ε}code 4. {S′→eS, S′→ε} and {S′→ε}   Answer: d | CO3 | BT4 |
| **14** | **Which one of the following is a top-down parser?** a) Recursive descent parser b) Operator precedence parser c) An LR(k) parser d) An LALR(k) parser **Answer: a** | CO3 | BT1 |
| **15** | **Consider the grammar** E → E + n | E × n | n For a sentence n + n × n, the handles in the right-sentential form of the reduction are a) n, E + n and E + n × n b) n, E + n and E + n × n c) n, n + n and n + n × n d) n, E + n and E × n **Answer: d** | CO3 | BT1 |
| **16** | **The variable which produces ∈ is called-----------**   1. Empty variable 2. Nullable variable 3. Terminal   d)All of the above  Answer:b | CO3 | BT1 |
| **17** | **A parser with the valid prefix property is advantageous because it** a) Detects errors b) syntax error c) Errors are passed to the text phase d) All of the mentioned **Answer: c** | CO3 | BT1 |
| **18** | **The action of parsing the source program into proper syntactic classes is called** a) Syntax Analysis b) Lexical Analysis c) Interpretation analysis d) General Syntax Analysis **Answer: b** | CO3 | BT1 |
| **19** | **Consider the grammar**  **S 🡪 AbBaCc | €**  **A 🡪 aAb | ba**  **B 🡪 bBC | cb**  **C 🡪 cCa | ac**  **Find the first () of S**   1. {a,b,c, €} 2. {a,b, €} 3. {€} 4. {a, €}   Ans:b | CO3 | BT4 |
| **20** | What is the maximum number of reduce moves that can be taken by a bottom-up parser for a grammar with no epsilon- and unit-production (i.e., of type A -> є and A -> a) to parse a string with n tokens? **a)** n/2 **b)** n-1 **c)** 2n-1 **d)** 2n **Answer:** **b** | CO3 | BT2 |
| **21.** | **Consider the following grammar**  **P🡪xQRS**  **Q🡪yz|z**  **R🡪W| | €**  **S🡪y**  **What is follow(Q)?**   1. {R} 2. (W} 3. {W,y} 4. {W,$}   Ans:c | CO3 | BT5 |
| **22.** | **Consider the following grammar**  **S🡪 (L)**  **S🡪a**  **L🡪L,S**  **L🡪S**  **find the Follow (L)**   1. { ( } 2. { ) } 3. { $,) ) 4. { $}   Ans: b | CO3 | BT5 |
| **PART B (4 Marks)** | | | |
| **1** | Define parser and describe its role. | CO3 | BT 1 |
| **2** | Why lexical & syntax analyzer are separated out? | CO3 | BT 1 |
| **3** | Explain ambiguity with example |  |  |
| **4** | Compare syntax tree and Parse tree | CO3 | BT 1 |
| **5** | Differentiate top-down approach and bottom-up approach to parsing with an example | CO3 | BT 2 |
| **6** | Explain the necessary conditions to be carried out before construction of predictive parsing? | CO3 | BT 1 |
| **7** | Eliminate left recursion for the grammar E->E+T, T->T\*F|F, F->(E)|id | CO3 | BT 4 |
| **8** | Explain the types of parsing |  |  |
| **9** | Derive the string and construct a syntax tree for the input string ceaedae using the grammar | CO3 | BT 4 |
| **10** | Write the algorithm for FIRST and FOLLOW in parser. | CO3 | BT 2 |
| PART –C 12 MARKS | | | |
| **1** | Construct a Predictive parsing for the grammar,E->E+T/T,  T->T\*F/F,F->(E)/id | CO3 | BT 5 |
| **2** | Explain in detail, error Recovery in Predictive parsing | CO3 | BT 5 |
| **3** | Construct the predictive parser for the following grammar   1. S->(L)|a L->L, S|S   ii) Construct the behavior of the parser on the sentence (a,a) using the grammar specified above | CO3 | BT 5 |
| **4** | Explain Recursive predictive parsing with example | CO3 | BT 5 |
| **5** | Compute FIRST and FOLLOW for the Grammar, E->E+T/T,  T->T\*F/F,F->(E)/id | CO3 | BT 5 |

| **UNIT -III** | | | |
| --- | --- | --- | --- |
| Bottom Up Parsing – Reductions-Handle Pruning- Shift Reduce ParsingProblems related to Shift Reduce Parsing- Conflicts During Shift Reduce ParsingLR Parsers- Why LR Parsers- Items and LR(0) Automaton, Closure of Item Sets-LR Parsing Algorithm- Operator Precedence Parser -Computation of LEADING- Computation of TRAILING- Problems related to LEADING AND TRAILING-SLR Grammars- SLR Parsing TablesProblems related to SLR- Construction of Canonical LR(1) and LALRConstruction of LALR- Problems related to Canonical LR(1) and LALR -Parsing Table | | | |
| **PART-A (Multiple Choice Questions)** | | | |
| **Q.**  **No** | **Questions** | **Course Outcome** | **Competence**  **BT Level** |
| **1** | which is not a shift reduce parser action.   1. Shift 2. Goto 3. Reduce 4. Accept   **Answer: b** | CO3 | **BT1** |
| **2** | Reverse of a right most derivation is called ------------.   1. reduction 2. production 3. handle 4. base   **Answer: c** | CO3 | **BT1** |
| **3** | The canonical reduction sequence is obtained by ------.   1. reduction 2. handle 3. production 4. handle pruning   **Answer: c** | CO3 | **BT1** |
| **4** | **YACC builds up** a) SLR parsing table b) Canonical LR parsing table c) LALR parsing table d) operator precedence **Answer: c** | CO3 | **BT1** |
| **5** | **What is the similarity between LR, LALR and SLR?**  a) Use same algorithm, but different parsing table. b) Same parsing table, but different algorithm. c) Their Parsing tables and algorithm are similar but uses top down approach. d) Both Parsing tables and algorithm are different.  **Answer:a** | CO3 | **BT2** |
| **6** | **In operator precedence parsing whose precedence relations are defined**  a) For all pair of non-terminals b) For all pair of terminals c) To delimit the handle d) Both terminal and non-terminal  **Answer:a** | CO3 | **BT2** |
| **7** | **Which grammar rules violate the requirements of an operator grammar? i)P → Q R ii)P → Q s R iii)P → ε iv)P → Q t R r** a) i only b) i and iii only c)ii and iii only d) iii and iv only  **Answer:b** | CO3 | **BT3** |
| **8** | **The most powerful parser is** a) SLR b) LALR c) Canonical LR d) Operator Precedence  **Answer:c** | CO3 | **BT2** |
| **9** | **S → C C C → c C | d The grammar is**  a) LL(1) b) SLR(1) but not LL(1) c) LALR(1) but not SLR(1) d) LR(1) but not LALR(1)  **Answer:a** | CO3 | BT4 |
| **10** | **Which of the following suffices to convert an arbitrary CFG to an LL(1) grammar?** a) Removing left recursion only b) Factoring the grammar alone c) Factoring & left recursion removal d) Left factoring **Answer: d** | CO3 | **BT2** |
| **11** | **Assume that the SLR parser for a grammar G has n1 states and the LALR parser for G has n2 states.** a) n1 is necessarily less than n2 b) n1 is necessarily equal to n2 c) n1 is necessarily greater than n2 d) n2 is greater than n1  **Answer: b** | CO3 | **BT2** |
| **12** | LR Parser is a ------------parser.   1. Bottom-Up 2. Top-Down 3. forward 4. Reverse   **Answer: a** | CO3 | **BT2** |
| **13** | LR parser construct a -----------type of derivation.   1. MMD 2. RMD 3. LMD 4. CLR   **Answer: b** | CO3 | **BT2** |
| **14** | A bottom-up parser generates--------   1. Left-most derivation in reverse 2. Left-most derivation 3. Right-most derivation in reverse 4. Right–most derivation   **Answer: c** | CO3 | **BT2** |
| **15** | which of the following grammar rules violate the requirements of an operator grammar? P, Q, R are nonterminal and s, r, s, t are terminals. 1.P→QR 2. P→QsR 3. P→ε 4. P→QtRr   1. 1 only 2. 1 and 3 only 3. 2 and 3 only 4. 3 and 4 only   **Answer: b** | CO3 | **BT4** |
| **16** | type of conflicts can occur in the shift-reduce parsing?   1. What reduce/reduce 2. shift/reduce 3. Both shift/reduce and reduce/reduce 4. None of the above   **Answer: b** | CO3 | **BT2** |
| **17** | 1. **Which of the following statements is false?** 2. An LL(1) parser is a top-down parser 3. An unambiguous grammar has same leftmost and rightmost derivation 4. LALR is more powerful than SLR 5. An ambiguous grammar can never be LR(k) for any k   **Answer: b** | CO3 | **BT2** |
| 18 | The grammar S→aSa|bS|c is-------   1. LL(1) but not LR(1) 2. LR(1) but not LL(1) 3. Both LL(1) and LR(1) 4. Neither LL(1) nor LR(1)   **Answer: c** | CO3 | **BT3** |
| **19** | We can get an LL(1) grammar by ----------   1. Removing left recurrence 2. Applying left factoring 3. Removing left recurrence and applying left factoring 4. None of the above.  **Answer: c** | CO3 | **BT2** |
| **20** | Which of the following option is not a function of the shift-reduce parser?   1. Reduce 2. Accept 3. Go 4. Shift   **Answer:** **c.** | CO3 | **BT1** |
| **21** | --------is the most powerful parser.   1. operator precedence 2. LALR 3. canonical LR 4. SLR   **Answer: c.** | CO3 | **BT2** |
| **22** | What is the LEADING(S) for the following grammar? S➔S-B|B B➔B\*A|A A➔(S)|id  1. LEADING(S)={-,\*,),id} 2. LEADING(S)={-,\*,(,)} 3. LEADING(S)={-,\*,(,id} 4. LEADING(S)={-,\*,(}   **Answer: c.** | CO3 | BT3 |
| **23** | What is the TRAILING(S) for the following grammar? S➔S-B|B B➔B\*A|A A➔(S)|id  1. TRAILING(S)={-,\*,),id} 2. TRAILING(S)={-,\*,(,)} 3. TRAILING(S)={-,\*,(,id} 4. TRAILING(S)={-,\*,(}   **ANSWER**  **Answer: a** | CO3 | BT3 |
| **24** | The precedence relation between ) and ( will be --------  1. < 2. > 3. = 4. Undefined   **Answer: d** | CO3 | BT2 |
| **25** | What is the TRAILING(T) for the following grammar? E➔E+T|T T➔T\*F|F F➔id|(E)  1. TRAILING(T)={+,\*,(} 2. TRAILING(T)={\*,id,(} 3. TRAILING(T)={\*,id,)} 4. TRAILING(T)={+,\*,id}   **Answer: c.** | CO3 | BT3 |
| **26.** | **Which of the following describes a handle (as applicable to LR-parsing) appropriately?** a) Position where next reduce or shift operation will occur b) The next step has use of Non-terminal for reduction c) used for reduction in a coming-up step along with a position in the sentential form where the next shift or reduce operation will occur d) used in the next step for reduction along with a position in the sentential form where the right hand side of the production may be found **Answer: d** | CO3 | BT |
| **PART B (4 Marks)** | | | |
| **1** | Define– Handle. What is handle pruning? | CO3 | BT 1 |
| **2** | Define – LR (0) items. | CO3 | BT 2 |
| **3** | Example for right sentential form and Handle for grammar for the following  E🡪E + E  E🡪E \* E  E🡪( E )  E🡪 id |  | BT3 |
| **4** | Write the conflicts arises during Shift -reduce parser. | CO3 | BT2 |
| **5** | Construct LR (1) for the grammar S🡪 CC, C 🡪cC | d | CO3 | BT3 |
| **6** | List out the difference between CLR and LALR parser. | CO3 | BT4 |
| **7** | Enumerate the concepts of Operator Precedence parser with an example. | CO3 | BT2 |
| **8** | Define augumented grammar. | CO3 | BT2 |
| **PART C (12 Marks)** | | | |
| **1** | List all LR (0) items for the following grammar S->;AS/b, A-   * SA/a | CO3 | BT 5 |
| **2** | Construct the SLR(1) parsing table for E->E+T |T T->T\*F|F F->(E)|id | CO3 | BT 5 |
| **3** | Construct CLR parsing table from S->AA, A->Aa|b | CO3 | BT 5 |
| **4** | Consider the following grammar S->AS|b A->SA|a Construct SLR parse table for the grammar.  Show the action of the parser for the input string “abab”. | CO3 | BT 5 |
| **5** | Consider the following grammar S->L=R|R, L->\*R|id, R->L Discuss LALR parsing method for this grammar. List out canonical collections and also construct the parse table | CO3 | BT 5 |

**Note:**

1. **BT Level –** Blooms Taxonomy Level
2. **CO – Course Outcomes**

BT1 –RememberBT2 – Understand BT3 – Apply BT4 – Analyze BT5 – Evaluate BT6 – Create