

**SRM Institute of Science and Technology SET -D**

**College of Engineering and Technology**

**School of Computing**

SRM Nagar, Kattankulathur – 603203, Chengalpattu District, Tamil Nadu

**Academic Year: 2024 (EVEN)**

**School of Computing**

**Test: CLA-T2** **Date: 26/3/2024**

**Course Code & Title: 18CSC304J & Compiler Design Duration:12.30 PM to 2.15 PM (2 periods)**

**Year & Sem: III Year /VI Sem** **Max. Marks: 50**

**ANSWER KEY**

| **S.No.** | **Course Outcome** | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6** | **PO7** | **PO8** | **PO9** | **PO10** | **PO11** | **PO12** | **PSO1** | **PSO2** | **PSO3** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **CO1** | **3** | **2** | **2** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **3** |
| **2** | **CO2** | **-** | **3** | **3** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **1** |
| **3** | **CO3** | **-** | **3** | **3** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **1** |
| **4** | **CO4** | **-** | **3** | **3** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **2** |
| **5** | **CO5** | **-** | **3** | **3** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **-** | **3** |

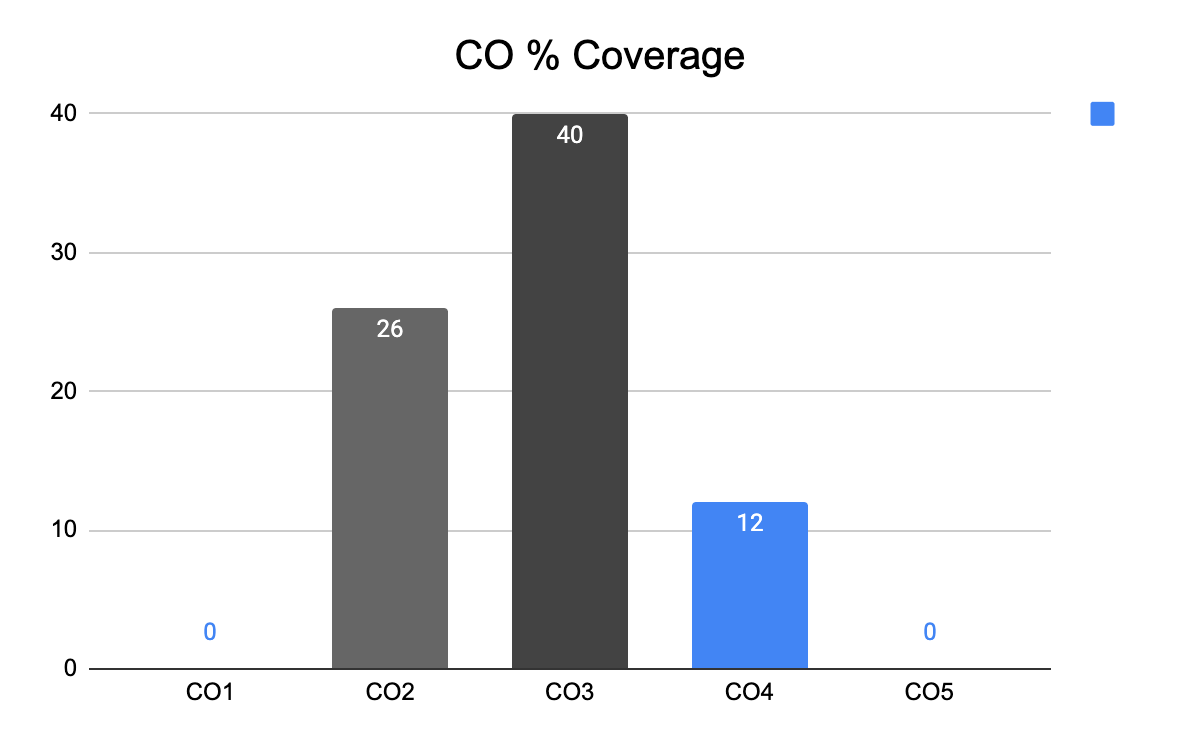
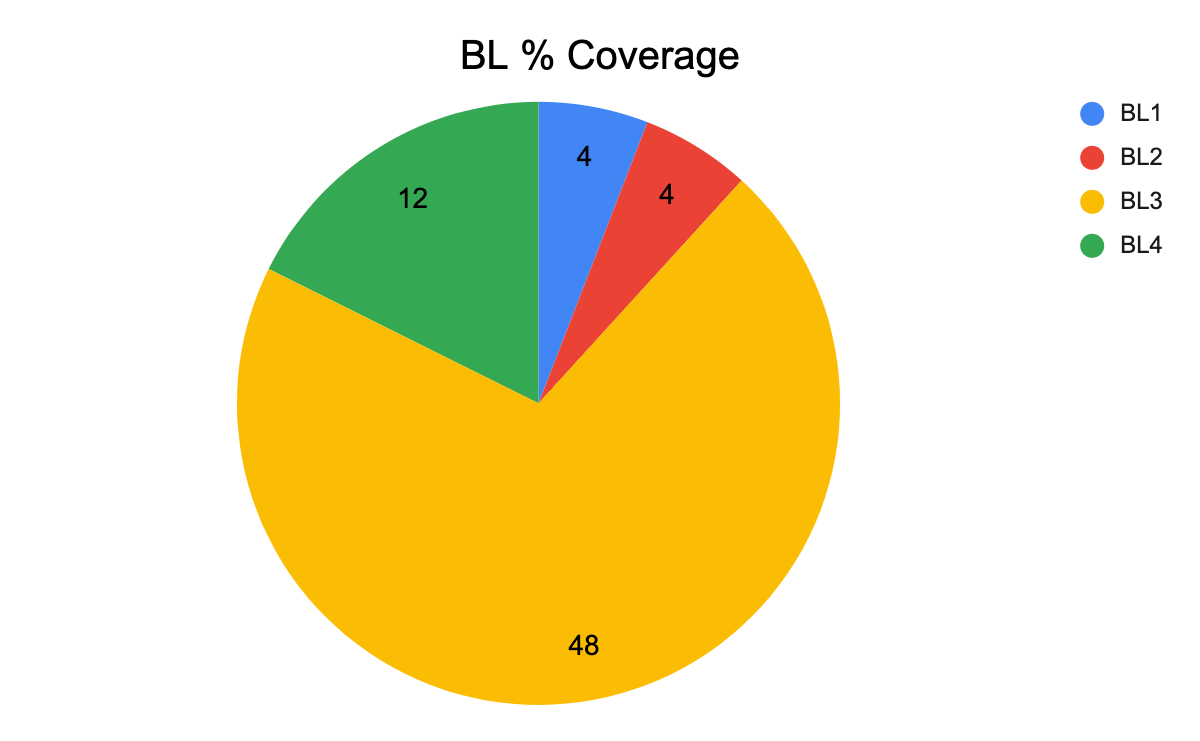
| **Part – A Answer All Questions**  **Answer all (10 x 1= 10 marks)** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| **Q. No** | **Question** | **Mark** | **BL** | **CO** | **PO** | **PI Code** |
| 1 | Which of the following is true about left factoring  **a) It is the process of factoring out of common prefixes of production rules**  b) It is the process of factoring out of common suffixes of production rules  c) It is the process of factoring out of terminals of production rules  d) It is the process of factoring out of terminals of production rules | 1 | 2 | 2 | 3 | 3.6.1 |
| 2 | Search in sidebar query  **Ans: (D)** | 1 | 2 | 2 | 3 | 3.6.1 |
| 3 | In a predictive parsing table if any cell has multiple entries, the grammar is:  **a) Ambiguous**  b) Erroneous  c) Both ambiguous and erroneous  d) unambiguous | 1 | 2 | 2 | 3 | 3.6.1 |
| 4 | Assuming that lower-case characters are terminals and upper-case ones are non-terminals in a grammar, which rule prevents the grammar from being an operator grammar?  a) S → aA  **b) S → AB**  c) S → AbB  d) S → Ab | 1 | 1 | 3 | 6 | 6.3.1 |
| 5 | In operator precedence parsing, the shift and reduce operations are done based on the priority between:  a) The symbols referred by stack[top] and stack[top-1]  **b) The symbol referred by stack[top] and the current input symbol**  c) The current input symbol and the immediately next input symbol  d) The current output symbol and the immediately next input symbol | 1 | 1 | 3 | 1 | 1.6.1 |
| 6 | In LR(1) parsing, we reduce by A → α only on those input symbols a for which [A → α., a] is an LR(1) item in the state on top of the stack. The set of such a’s will always be:  a) Equal to the set FOLLOW(A)  b) A subset of FOLLOW(A)  **c) A proper subset of FOLLOW(A)**  d) A proper subset of FOLLOW(B) | 1 | 2 | 2 | 3 | 1.6.1 |
| 7 | Consider the following grammar G, compute FIRST (S)  S → A | B  A →a | c  B → b | ε  **(a) {a,b,c,ε}**  (b) {a,c}  (c) {b,c}  (d) { b,c,ε} | 1 | 2 | 2 | 6 | 1.6.1 |
| 8 | Check the order of LR Parsers  (a) CLR<SLR<LALR  (b)SLR<CLR<LALR  **(c)SLR<LALR<CLR**  (d)LALR<CLR<SLR | 1 | 1 | 2 | 6 | 3.6.1 |
| 9 | Which one from the following is false?  a) LALR parser is Bottom-up parser  **b) A parsing algorithm which performs a left to right scanning and a rightmost derivation in reverse is RL(1)**  c) LR parser is Bottom-up parser  d) In LL(1), the 1 indicates that there is a one-symbol look-ahead | 1 | 1 | 3 | 6 | 3.6.1 |
| 10 | Which of the following statements is false?  **a) An ambiguous grammar may have LL(1) parser without multiple entries in parsing table**  b) An LL(1) parser is a top-down parser  c) LALR is more powerful than SLR  d) An ambiguous grammar can never be LR(k) for any k | 1 | 1 | 3 | 6 | 3.6.1 |

| **Part –B**  **Answer any Four (4x 4=16 marks)** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| **Q. No** | **Question** | **Mark** | **BL** | **CO** | **PO** | **PI Code** |
| 11 | Consider the given grammar to compute LEADING & TRAILING  S -> L=R  S -> R  L-> \*R  L -> id  R -> L | 4 | 1 | 2 | 1 | 1.6.1 |
| 12 | List out the steps involved in identifying FOLLOW of the variables present in the production rules.  FOLLOW(A) is the set of terminals a, that appear immediately to the right of A. For rightmost sentential form of A, $ will be in FOLLOW(A). • Rules  – For the FOLLOW(start symbol) place $, where $ is the input end marker.  – If there is a production A → αBβ, then everything in FIRST(β) except ε is in FOLLOW(B).  – If there is a production A → αB, or a production A → αBβ where FIRST(β) contains ε, then everything in FOLLOW(A) is in FOLLOW(B) | 4 | 3 | 2 | 6 | 6.3.1 |
| 13 | Summarize the steps involved in constructing an operator precedence parsing table | 4 | 2 | 2 | 6 | 6.3.1 |
| 14 | Compare panic mode error recovery with phrase level recovery  Panic Mode Recovery  • Panic mode error recovery is the easiest method of error-recovering strategy which prevents the parser from developing infinite loops. • When parser finds an error in the statement, it ignores the rest of the statement by not processing the input.  • The parser intends to find designated set of synchronizing tokens by discarding input symbols one at a time.  • Synchronizing tokens may be delimiters, semicolon or } whose role in source program is clear.  • Advantages – Simplicity. Never get into infinite loop.  • Disadvantage – Additional errors cannot be checked as some of the input symbols will be skipped  Phrase Level Recovery  • Parser performs local correction on the remaining input when an error is detected.  – When a parser finds an error, it tries to take corrective measures so that the rest of inputs of statement allow the parser to parse ahead.  – One wrong correction will lead to an infinite loop.  – The local correction may be  • Replacing a prefix by some string.  • Replacing comma by semicolon.  • Deleting extraneous semicolon.  • Insert missing semicolon | 4 | 3 | 2 | 6 | 6.3.1 |
| 15 | In SLR parsing for the grammar  S’ → S  S → aSbS  S → bSaS | ε. Check whether shift/reduce conflict occurs, if so initially in which state the conflict occurs and identify which terminal encounters the conflict  **Answer: In state 0 both ‘a’ and ‘b’ have shift/reduce conflicts** | 4 | 3 | 2 | 6 | 6.3.1 |

| **Part –C**  **Answer all ( 2 x 12=24 marks)** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| **Q. No** | **Questions** | **Marks** | **BL** | **CO** | **PO** | **PI Code** |
| 16 | (i) Construct Predictive parsing table and parse the input string “cdcd”  S -> CC  C ->cC  C -> d | (12) | 4 | 4 | 6 | **6.3.1** |
|  | **Or** |  |  |  |  |  |
| 17 | (i) Consider the following Grammar and construct the operator precedence parsing table  E -> E+T|T  T -> T\*F|F  F-> (E)|id  (ii) Parse the input string id+id\*id by the operator precedence table | (4+8) | 3 | 3 | 5,6 | **6.3.1** |
|  |  |  |  |  |  |  |
| 18 | Consider the Following Grammar and prepare the CLR Parsing Table.  A -> B = C  A -> C  B -> \* C  B -> id  C -> B | 12 | 3 | 3 | 5,6 | **6.3.1** |
|  | **Or** |  |  |  |  |  |
| 19 | Consider the Following Grammar and prepare the LALR parsing table  A -> BB  B ->bB  B -> c  ii) Parse the input string bcc with the help of LALR parsing table above | (8+4) | 3 | 3 | 6 | **6.3.1** |

**\*Performance Indicators are available separately for Computer Science and Engineering in AICTE examination reforms policy.**

**Course Outcome (CO) and Bloom’s level (BL) Coverage in Questions**

****

**Approved by the Audit Professor/Course Coordinator**