

# School of Engineering (CSE)

# III Year B. Tech - I Semester

**Subject Name: Compiler Design**

**Subject Code: MR22-1CS0107**

**Question Bank**

|  |  |  |  |
| --- | --- | --- | --- |
| **Qno** | **Question** | **Marks** | **Section** |
| 1 | a) Define Finite Automata and explain its representations with example.  b) Design a FA which i) starts with 1 and ends with 0  ii) contains 2 consecutive a’s and 2 consecutive b’s | 4  4 | Section-I |
| 2 | 1. Consider the CFG:   S → aB | bA,  A → a | aS | bAA,  B → b | bS | aBB Find LMD and RMD for the string  w= aabbabba.   1. Differentiate between DFA and NFA. | 4  4 | Section-I |
| 3 | Consider the following NFA and convert into DFA. | 8 | Section-I |
| 4 | Convert the given NFA with epsilon to NFA without epsilon.  https://www.tutorialspoint.com/assets/questions/media/53119/without_epsilon.jpg | 8 | Section-I |
| 5 | Describe the terms with examples.   1. Regular Language 2. Regular Expression 3. Context free grammar 4. Derivation trees. 5. Ambiguous Grammar | 8 | Section-I |
| 6 | a) Draw NFA for the Regular expression: i) a(a+b)\*ab  ii) (0+1)\*1(0+1)  b) Draw DFA for the Regular expression: 10+(0+11)0\*1 | 4  4 | Section-I |
| 7 | What are the simplified forms in CFG? Explain each form in detail with example. | 8 | Section-I |
| 8 | a) Convert the following CFG into CNF.  S → ASB  A → aAS|a|ε  B → SbS|A|bb  b) Convert the following CFG into GNF.  S 🡪 AA | 0  A 🡪 SS | 1 | 4  4 | Section-I |
| 9 | a) Design a FA from given regular expression 10 + (0 + 11)0\* 1.  b) Write the regular expression for the language accepting all the string which are starting with 1 and ending with 0, over ∑ = {0, 1}. | 4  4 | Section-I |
| 10 | Describe the terms with examples.   1. Regular Language 2. Regular Expression 3. Context free grammar 4. Derivation trees 5. Ambiguous Grammar | 8 | Section-I |
| 11 | Discuss the different phases of a compiler. | 8 | Section-II |
| 12 | a) Explain in detail about input buffering with example.  b) Discuss about One-buffer scheme and Two buffer scheme with examples. | 4  4 | Section-II |
| 13 | Explain various operations on string with example. | 8 | Section-II |
| 14 | Explain about specification and Recognition of Tokens. | 8 | Section-II |
| 15 | Explain LEX program structure and Write a LEX program to recognize tokens.  (Identifiers, keywords, relational operators and numbers). | 8 | Section-II |
| 16 | Explain in detail about design of lexical analyzer generator. | 8 | Section-II |
| 17 | a) Explain the roles of lexical analyser.  b)Explain LEX program structure with example LEX program | 8 | Section-II |
| 18 | Describe the terms.   1. Symbol Table 2. Tokens 3. Lexemes 4. Patterns 5. Parse Tree | 8 | Section-II |
| 19 | Differentiate between Compiler and Interpreter. | 8 | Section-II |
| 20 | Define the Phases of a compiler which indicating the inputs and outputs of each phase in translating the statement “x=y+z\*40”. | 8 | Section-II |
| 21 | What is back tracking? Write recursive procedures for each non terminal by taking suitable example. | 8 | Section-III |
| 22 | Calculate FIRST and FOLLOW from the following grammar.  S🡪ABCDE  A🡪 a | Є  B🡪 b | Є  C🡪 c  D🡪 d | Є  E🡪 e | Є | 8 | Section-III |
| 23 | Construct LR parsing table for the following grammar.  S → CC  C → aC | d | 8 | Section-III |
| 24 | Construct SLR parsing table for the following grammar.  S → AA  A → aA | b. | 8 | Section-III |
| 25 | Calculate FIRST and FOLLOW from the following grammar.  E -> TE’  E’ -> +T E’|Є  T -> F T’  T’ -> \*F T’ | Є  F -> (E) | id |  | Section-III |
| 26 | Discuss about Bottom up parsing with an example. | 8 | Section-III |
| 27 | Discuss about Top down parsing and Bottom up parsing with an example. | 8 | Section-III |
| 28 | 1. Explain in detail about the Role of Parser in Compiler. 2. Write a short note on error recovery with LR parsers. How is it different from LL parsers? | 4  4 | Section-III |
| 29 | What is meant by Parsing? Discuss about Top down parsing with an example. | 8 | Section-III |
| 30 | Discuss about Shift reduce parsing with an example. | 8 | Section-III |
| 31 | What are different intermediate code forms? Explain with example. | 8 | Section-IV |
| 32 | Construct a Three-address code, quadruples, triples and Syntax tree for the following expression:  a = b\* - c + b \* -c ; | 8 | Section-IV |
| 33 | Define SDT and write it’s applications. | 8 | Section-IV |
| 34 | What are different types of attributes? Explain with example. | 8 | Section-IV |
| 35 | Describe the construction of syntax tree. Construct a syntax tree for the expression: x\*y-5+z | 8 | Section-IV |
| 36 | Explain the Variants of Syntax tree. | 8 | Section-IV |
| 37 | 1. Distinguish between synthesized attributes and inherited attributes. 2. Construct the syntax tree for the expression   a-4+c\*d. | 4  4 | Section-IV |
| 38 | Discuss different Three Address code types and implementations of Three Address statements. | 8 | Section-IV |
| 39 | a) Elaborate the role of intermediate code generator in compilation process.  b) Explain in detail about Abstract syntax tree. | 4  4 | Section-IV |
| 40 | What are Boolean Expressions? And Write Syntax Directed Translation for the Boolean Expressions:  𝐄 → 𝐄 𝐎𝐑 𝐄  𝐄 → 𝐄 𝐀𝐍𝐃 𝐄  𝐄 → 𝐢𝐝𝟏 𝐫𝐞𝐥𝐨𝐩 𝐢𝐝𝟐  𝐄 → 𝐍𝐎𝐓 𝐄  𝐄 → (𝐄)  𝐄 → 𝐅𝐀𝐋𝐒𝐄  𝐄 → 𝐓𝐑𝐔𝐄 | 8 | Section-IV |
| 41 | What is the role of code Optimizer in compiler? Explain various machine independent code optimization techniques. | 8 | Section-V |
| 42 | Explain in detail about various peephole optimizations. | 8 | Section-V |
| 43 | a) Explain the role of DAG in optimization with suitable example.  b) Explain the Redundant sub expression elimination with an example. | 4  4 | Section-V |
| 44 | What is the purpose of loop optimization? Explain in detail loop optimization with example. | 8 | Section-V |
| 45 | Describe the terms with examples.   1. Code motion 2. Copy propagation 3. Dead code Elimination 4. DAG 5. Flow Graphs | 8 | Section-V |
| 46 | What is the purpose of optimization of building blocks? Explain different sources of optimization techniques with suitable examples. | 8 | Section-V |
| 47 | Explain how loop optimization can be done? How they are different from local optimizations. | 8 | Section-V |
| 48 | Explain the importance of register allocation with respect to optimization. | 8 | Section-V |
| 49 | a) Explain in detail about simple code generator.  b) Discuss in detail the role of dead code elimination and strength reduction during code optimization. | 4  4 | Section-V |
| 50 | Explain in detail about basic blocks and flow graphs with an example. | 8 | Section-V |