

|  |  |
| --- | --- |
| **NAME OF THE PROGRAMME** | **CSE** |
| **YEAR** | **II** |
| **SEMESTER** | **IV** |
| **REGULATIONS** | **2022 REVISED** |
| **COURSE CODE** | **CS3401** |
| **COURSE NAME** | **Theory of Computation** |

|  |  |  |  |
| --- | --- | --- | --- |
| **FACULTY NAME (Prepared by)** | **M.MARIYAMMAL** | **Contact** | **9626125381** |
| **NAME OF SUBJECT EXPERT(Verified by)** | **BIJU BALAKRISHNAN, MUTHUMARI LAKSHMI S** | **Contact** | **9486669918**  **8270014618** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **REVISED BLOOMS TAXONOMY(RBT)** | | | | | |
| **L1- Remembering** | **L2 - Understanding** | **L3 - Applying** | **L4 - Analyzing** | **L5 - Evaluating** | **L6 – Creating** |

**Course Objectives:**

|  |  |
| --- | --- |
| **COSNO** | **CourseObjectives** |
| 1. | To understand the foundations of computation, including automata theory |
| 2. | To construct models of regular expressions and languages |
| 3. | To design context-free grammar and push down automata |
| 4. | To understand Turing machines and their capability |
| 5. | To understand Undecidability and NP class problems |

**Course Outcomes:**

On Completion of the course, the students will be able to

|  |  |  |
| --- | --- | --- |
| **CO.NO** | **CourseOutcomes** | **RBTLevel** |
| 1. | Construct automata theory using Finite Automata | L3 |
| 2. | Write regular expressions for any pattern | L3 |
| 3. | Design context-free grammar and Pushdown Automata | L6 |
| 4. | Design a Turing machine for computational functions | L6 |
| 5. | Differentiate between decidable and undecidable problems | L4 |

|  |  |
| --- | --- |
| **UNIT** | **TITLE** |
| **I** | AUTOMATA AND REGULAR EXPRESSIONS |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SNO** | **QUESTIONS** | **RBT** | **CO** | **MARK** | **TOPIC** | **IMAGES** |
|  | Define ε - closure (q) with an example. | L1 | CO 1 | 2 | ε-closure |  |
|  | Find the closure of the states 1, 2 and 4 in the following transition diagram. | L2 | CO 1 | 2 | ε-closure | https://www.poriyaan.in/media/imgPori/images52/A9JQuRA.jpg |
|  | Find the language accepted by the DFA given below. | L3 | CO 1 | 2 | DFA Language | https://www.poriyaan.in/media/imgPori/images52/nC6EHdy.jpg |
|  | Construct a DFA for the language accepting strings ending with ‘abba’over input alphabets Σ = (a, b). | L3 | CO 1 | 2 | DFA Design |  |
|  | Construct a DFA that accepts a language L over input alphabets Σ = (a, b) such that L is the set of all strings starting with ‘aa’ or ‘bb’. | L3 | CO 1 | 2 | Introduction to Formal Proofs |  |
|  | Obtain the NFA without ε transition to the following NFA with ε transition. | L3 | CO 1 | 2 | Conversion of NFA to NFA without ε-moves | https://www.poriyaan.in/media/imgPori/images52/oKnIIh8.jpg |
|  | Obtain the DFA equivalent to the following NFA. | L2 | CO 1 | 2 | Conversion of NFA to DFA | https://www.poriyaan.in/media/imgPori/images52/H6767Rt.jpg |
|  | Draw a non-deterministic automata to accept a string containing substring 0101. | L2 | CO 1 | 2 | NFA |  |
|  | Define Deterministic Finite Automata (DFA). | L1 | CO 1 | 2 | DFA Definition |  |
|  | Design DFA to accept strings over Σ = (0,1) with two consective O's. | L3 | CO 1 | 2 | DFA Design |  |
|  | Define NFA with ε transition. Is the NFA's with ε transitions more powerful than the NFA's without ε transitions? | L2 | CO 1 | 2 | NFA with ε-transitions |  |
|  | Design DFA to accept all Binary Strings which are divisible by 5. L={w/w mod 5=0} | L3 | CO 1 | 2 | DFA Design |  |
|  | Find the set of strings accepted by the finite automata. | L2 | CO 1 | 2 | Language Accepted by Automata | https://www.poriyaan.in/media/imgPori/images52/JemONrq.jpg |
|  | Construct a DFA for the following:  a) All strings that contain exactly 4 - zeros.  b) All strings that don't contain the substring 110. | L2 | CO 1 | 2 | DFA Design |  |
|  | Construct a DFA for the language over {0, 1}\* such that it contains "000" as a substring. | L2 | CO 1 | 2 | DFA Design |  |
|  | Construct deterministic finite automata to recognise odd number of 1's and even number of zero's. | L2 | CO 1 | 2 | DFA Design |  |
|  | Construct a DFA over Σ = (a, b) which produces not more than 3a's. | L2 | CO 1 | 2 | DFA Design |  |
|  | Enumerate the differences between DFA and NFA. | L2 | CO 1 | 2 | DFA vs. NFA |  |
|  | What is a finite automation? Give two examples. | L1 | CO 1 | 2 | Finite Automaton Definition |  |
|  | Construct a finite automata for the language {0n | n mod 3 = 2, n ≥ 0}. | L2 | CO 1 | 2 | Design of Finite Automata (DFA/NFA) |  |
|  | i) Construct an NFA for the set of strings with {0,1} ending with 01 draw the transition table for the same and check whether the input string 00101 is accepted by above NFA.  ii) Construct NFA for set of all strings {0,1} that ends with three consecutive 1‘s at its end.  iii) Construct NFA for set of all strings {a,b} with abb as substring. | L3 | CO 1 | 12 | NFA Design and Validation |  |
|  | Convert to a DFA, the following NFA | L2 | CO 1 | 12 | NFA to DFA Conversion |  |
|  | Convert the following NFA-with ε, to a NFA- without ε | L2 | CO 1 | 12 | Conversion of ε-NFA to NFA |  |
|  | Construct a DFA equivalent to the the NFA M=({a,b,c,d},{0,1}, δ,a,{b,d}) where δ is  defined as | L3 | CO 1 | 12 | DFA Construction |  |
|  | Convert the following E-NFA to NFA and then convert the resultant NFA to DFA. | L2 | CO 1 | 12 | Conversion of E-NFA to NFA and DFA |  |
|  | Prove the following by induction for all n≥0 (i) 12+22+32+42+…….+n2=(n(n+1)(2n+1))/6 (ii) 13+23+….+n3=(n2(n+1)2)/4 | L3 | CO 1 | 12 | Mathematical Induction Proofs |  |
|  | Prove the following by mathematical induction method (i) 2n > n for all n≥0 (ii) x ≥4, 2x ≥ x2 | L3 | CO 1 | 12 | Inductive Proof |  |
|  | Construct DFA equivalent to the NFA given below | L3 | CO 1 | 12 | NFA to DFA Conversion |  |
|  | Design DFA to accept the Language L={w/w has both even number of 0‘s and even number of 1‘s} | L4 | CO 1 | 12 | Design DFA |  |
|  | Give NFA accepting the set of strings in (0+1)\* such that two 0‘s are separated by a string  whose length is 4i, for some i>=0. | L4 | CO 1 | 12 | NFA Design with Specific Conditions |  |
|  | i) If a Regular language L is accepted by a non – deterministic finite automata then there exists a Deterministic Finite Automata that accepts L.  ii) A Language ‗L‘ is accepted by some ε – NFA if and only if L is accepted by NFA without ε transition | L3 | CO 1 | 12 | Regular Language Properties and Proofs |  |
|  | Construct NFA without ε transitions for the NFA given below | L3 | CO 1 | 12 | NFA Conversion |  |
|  | i) Construct a DFA that accepts all strings on {0, 1} except those containing the substring 101.  ii) Construct a NFA accepting the set of strings over {a,b} ending in aba. Use it to constructa DFA accepting the same set of strings. | L3 | CO 1 | 12 | DFA/NFA Design and Conversion |  |
|  | i) Convert the following NFA-with ε, to a NFA- without ε  ii) Convert to a DFA, the following NFA | L3 | CO 1 | 12 | NFA Conversion to DFA |  |
|  | Define ε-NFA. Consider the following ε-closure of each state and find it‘s equivalent DFA. | L3 | CO 1 | 12 | ε-Closure and DFA Conversion |  |
|  | Design a NFA accept the following strings over the alphabets {0,1} that begins with 01 and ends with 11. Check for the validity of 01111 and 0110 strings. | L4 | CO 1 | 12 | NFA Design and Validation |  |
|  | Convert to a DFA, the following NFA. | L2 | CO 1 | 12 | NFA to DFA Conversion |  |
|  | Give NFA to accept the following languages over {0,1} (i) L = {String that contains either 101 or 110 as a substring} (ii) L = {Strings such that every 1 is following immediately by 00} | L3 | CO 1 | 12 | NFA Design |  |
|  | Give NFA to accept the following languages over {0,1} (i) L = {String that contains either 101 or 110 as a substring}  (ii) L = {Strings such that every 1 is following immediately by 00} | L3 | CO 1 | 12 | NFA Design |  |
|  | Design a DFA to accept the language L = {w/w has both an even number of 0’s and even number of 1’s} and illustrate from its transition function to check the string w= 110101. | L4 | CO 1 | 12 | NFA Design |  |
|  | Minimize the following DFA | L3 | CO 1 | 16 | DFA Minimization |  |
|  | Minimize the following DFA. | L3 | CO 1 | 16 | DFA Minimization |  |
|  | Minimize the following DFA. | L3 | CO 1 | 16 | DFA Minimization |  |
|  | Convert ∈ - NFA to NFA | L3 | CO 1 | 16 | Conversion of ∈-NFA to NFA |  |
|  | Convert ∈ - NFA to NFA | L3 | CO 1 | 16 | Conversion of ∈-NFA to NFA |  |

|  |  |
| --- | --- |
| **UNIT** | **TITLE** |
| **II** | REGULAR EXPRESSIONS AND LANGUAGES |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SNO** | **QUESTIONS** | **RBT** | **CO** | **MARK** | **TOPIC** | **IMAGES** |
|  | Differentiate regular expression and regular language | L2 | CO 2 | 2 | Basics of Regular Expressions and Regular Languages |  |
|  | Give the regular expression for set of all strings ending in 00. | L1 | CO 2 | 2 | Regular Expressions Design |  |
|  | State pumping lemma for regular language and its advantages. | L1 | CO 2 | 2 | Pumping Lemma Definition and Properties |  |
|  | Give the regular expression for the following L1= set of all strings of 0 and 1 ending in 00 L2= set of all string 0 and 1 beginning with 0 and ending with 1 | L2 | CO 2 | 2 | Regular Expressions Design |  |
|  | Is it true that the language accepted by any NFA is different from the regular language? Justify your answer. | L2 | CO 2 | 2 | Properties of Regular Languages and NFAs |  |
|  | Is regular set is closed under complement? Justify. | L2 | CO 2 | 2 | Closure Properties of Regular Languages |  |
|  | Construct NFA for the regular expression (0+1)01 | L3 | CO 2 | 2 | NFA Construction |  |
|  | Prove or disprove that (r+s)\*=r\*+s\*. | L2 | CO 2 | 2 | Properties of Regular Expressions |  |
|  | Give English description of the following language (0+10)\*1\*. | L2 | CO 2 | 2 | Regular Language Interpretation |  |
|  | Write RE for the set of strings over {0,1} that have atleast one. | L1 | CO 2 | 2 | Regular Expressions Design |  |
|  | Show whether a language L=(0n12n/n>0} is regular or not using pumping Lemma. | L3 | CO 2 | 2 | pumping Lemma. |  |
|  | Generate NFA - ε to represent a\* b|c | L3 | CO 2 | 2 | NFA - ε |  |
|  | Write regular expression for the set of all strings of 0's and 1's not containing 101 as substring. Provide justification that your regular expression is correct. | L2 | CO 2 | 2 | Regular Expressions Design with Validation |  |
|  | What are closure properties of regular languages. | L1 | CO 2 | 2 | NFA Construction |  |
|  | Construct NFA for regular expression a\*b\*. | L3 | CO 2 | 2 | regular expression |  |
|  | Let L= {W: W ϵ {0, 1}\* W does not contain 00 and is not empty}. Construct a regular expression that generates L} | L3 | CO 2 | 2 | regular expression |  |
|  | Construct an NDFA for all strings over alphabet = {a, b} that contains a substring 'ab'. | L3 | CO 2 | 2 | NFA Construction |  |
|  | Construct a DFA for the regular expression aa\*bb\* | L3 | CO 2 | 2 | DFA Construction |  |
|  | Define regular expression. Give an example. | L1 | CO 2 | 2 | Basics of Regular Expressions |  |
|  | Describe the following by regular expression  a) L1 = The set of all strings of 0's and 1's ending in 00.  b) L2 The set of all strings of 0's and 1's beginning with 0 and ending with 1. | L2 | CO 2 | 2 | Regular Expressions Design |  |
|  | Construct an - NFA for the regular expression b+ba\*. | L3 | CO 2 | 12 | NFA Construction |  |
|  | Which of the following language is regular? Justify. i. L={ nbm/n,m>0} ii. L={ anbn/n,>0} | L3 | CO 2 | 12 | Regular Expression Extraction |  |
|  | Obtain the regular expression for the finite automata. | L3 | CO 2 | 12 | Regular Expressions and Finite Automata |  |
|  | i) Using pumping lemma for the regular sets, prove that the language L={ambn/m>n} is notregular. ii) Prove any two closure properties of regular languages. | L4 | CO 2 | 12 | Regular Languages and Pumping Lemma |  |
|  | Convert the following NFA into a R.E | L3 | CO 2 | 12 | NFA to Regular Expression Conversion |  |
|  | Construct a finite automata for the RE 10+(0+11)0\*1 | L3 | CO 2 | 12 | Finite Automaton Construction |  |
|  | Prove that the following languages are not regular: L= {w€{a,b}\*/w = wr} | L4 | CO 2 | 12 | Non-Regular Languages and Pumping Lemma |  |
|  | Show that the regular language are closed under : Union ,Intersection, Kleene closure, Complement, Difference | L3 | CO 2 | 12 | Closure Properties of Regular Languages |  |
|  | State pumping lemma and hence find whether the given language is regular or not L={anbn/m>n} | L4 | CO 2 | 12 | pumping lemma |  |
|  | Prove the following statement with justification. “the language L={ aibjci|i ,j >0}  is not regular” | L4 | CO 2 | 12 | Proving Non-Regular Languages Using Pumping Lemma |  |
|  | What is Regular Expression? Write a regular expression for set of strings that consists of alternating 0's and 1's. | L2 | CO 2 | 12 | Regular Expressions Design |  |
|  | Minimize the FA shown in fig below and show both the given and the reduced one areequivalent. | L4 | CO 2 | 12 | DFA Minimization |  |
|  | Write and explain the algorithm for minimization of a DFA. Using the above algorithm minimize the following DFA. | L4 | CO 2 | 12 | DFA Minimization Algorithm |  |
|  | Construct NFA with epsilon for the R.E=(a/b)\*ab and convert into DFA and further find theminimized DFA. | L4 | CO 2 | 12 | NFA to DFA Conversion and Minimization |  |
|  | Minimize the following automaton | L4 | CO 2 | 12 | DFA Minimization |  |
|  | Prove that any language accepted by a DFA can be represented by regular expression. | L4 | CO 2 | 12 | Regular Expressions and DFA Equivalence |  |
|  | Explain the DFA minimization algorithm with an example. | L4 | CO 2 | 12 | DFA Minimization Algorithm |  |
|  | Construct a minimized DFA for the RE 10+(0+11)0\*1. | L4 | CO 2 | 12 | DFA Minimization |  |
|  | Prove that the class of regular sets is closed under complementation. | L3 | CO 2 | 12 | Closure Properties of Regular Languages |  |
|  | i. Define regular expression.  ii. Show that (1+00\*1)+(1+00\*1)(0+10\*1)\*(0+10\*1)=0\*1(0+10\*1)\* | L4 | CO 2 | 12 | Regular Expressions Equivalence |  |
|  | Convert the following to regular expression | L4 | CO 2 | 16 | Automata to Regular Expression Conversion |  |
|  | Convert the following to regular expression | L4 | CO 2 | 16 | Automata to Regular Expression Conversion |  |
|  | For every DFA A= ,, 0 ,  there is a regular expression r such that L(r) = L(A). | L3 | CO 2 | 16 | DFA and Regular Expression Equivalence |  |
|  | Prove that there exists an NFA with €- transition that accepts the regular expression r. | L3 | CO 2 | 16 | NFA Construction from Regular Expressions |  |
|  | Construct RE for | L3 | CO 2 | 16 | Regular Expressions Design |  |

|  |  |
| --- | --- |
| **UNIT** | **TITLE** |
| **III** | CONTEXT-FREE GRAMMAR AND PUSH DOWN AUTOMATA |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SNO** | **QUESTIONS** | **RBT** | **CO** | **MARK** | **TOPIC** | **IMAGES** |
|  | Define CFG .Give an example. | L1 | CO 3 | 2 | Context-Free Grammar Basics |  |
|  | What is CFL? | L2 | CO 3 | 2 | Context-Free Languages |  |
|  | What is ambiguous grammar? Or When do you say grammar is ambiguous? | L2 | CO 3 | 2 | Ambiguity in Grammars |  |
|  | Consider the alphabet Σ = { a,b,(,),+,\*,-, . ,ξ }. Construct a CFG that generate all thestrings in Σ\* that are regular expression on the alphabet, Σ. | L3 | CO 3 | 2 | CFG for Regular Expressions |  |
|  | Find LMD & RMD, parse tree for the following grammar. | L3 | CO 3 | 2 | Derivation and Parse Trees |  |
|  | Write a grammar to recognize all prefix expressions involving all binary arithmeticoperators. Construct the parse tree for the sentence “-\*+abc/de” from your grammar. | L3 | CO 3 | 2 | Grammar for Prefix Expressions |  |
|  | Write the CFG for the following CFL L(G) = { / m+n=p, m&n>1} | L3 | CO 3 | 2 | CFG Construction for CFLs |  |
|  | Construct a grammar for the language L which has all the strings which are all palindrome over \_={a, b}. | L3 | CO 3 | 2 | Context-Free Grammar (CFG) Construction for Palindromes |  |
|  | Give the general forms of CNF. (Or) State CNF. | L1 | CO 3 | 2 | Chomsky Normal Form |  |
|  | Construct a parse tree of (a+b)\*c for the grammar EE+E/E\*E/(E)/id | L3 | CO 3 | 2 | Parse Tree Construction |  |
|  | Construct a CFG fro set of strings that contain equal number of a's and b's over Σ={a,b}. C | L3 | CO 3 | 2 | CFG for Balanced Strings |  |
|  | Define PDA. | L1 | CO 3 | 2 | PDA Basics |  |
|  | Draw PDA accepting the language L= { ancbn/n>0} | L3 | CO 3 | 2 | Pushdown Automata (PDA) Design |  |
|  | What are the different ways of language acceptance by a PDA and define them? | L2 | CO 3 | 2 | PDA Acceptance Methods |  |
|  | What is the additional feature PDA has when compared with NFA?Is PDA superiorover NFA in the sense of language acceptance? justify? (Or)Compare NFA & PDA. | L4 | CO 3 | 2 | PDA vs NFA |  |
|  | Construct a RMD of (a+b)\*c using the grammar and also state that whether a givengrammar is ambiguous or not. | L4 | CO 3 | 2 | Ambiguity in Grammars |  |
|  | What is an instantaneous description of PDA? | L2 | CO 3 | 2 | PDA Definitions |  |
|  | Differentiate PDA acceptance by empty stack with acceptance by final state. | L2 | CO 3 | 2 | PDA Acceptance Comparison |  |
|  | Derive the string “aabaab” for the following CFG SaSX/b  XXb/a | L3 | CO 3 | 2 | CFG Derivations |  |
|  | Generate CFG for a signed integer constant in C language. | L3 | CO 3 | 2 | CFG for Specific Constructs |  |
|  | Define Deterministic PDA.Is NPDA and DPDA equivalent? Illustrate with an eg. | L4 | CO 3 | 12 | Deterministic vs Non-Deterministic PDA |  |
|  | i) Let L is a context free language. Prove that there exists a PDA that accepts L. ii) Construct PDA for the Language L= {WWR | W is in (a+b)\*}. | L4 | CO 3 | 12 | Pushdown Automata and Context-Free Languages |  |
|  | i)Construct a transition table for PDA which accepts the language L={ (a2nbn/n>0} Trace your PDA for the input with n=3. ii)Find the PDA equivalent to the give CFG with the following productions.S→A, A→BC, B→ba, C→ac | L3 | CO 3 | 12 | Pushdown Automata for Specific Context-Free Languages |  |
|  | Construct the PDA accepting the language L= { anbn/n>0} by empty stack and final state. | L4 | CO 3 | 12 | Pushdown Automata |  |
|  | Convert the grammar S0S1/A: A1A0/S/ ε into PDA that accepts the same language by empty stack. Check whether 0101 belongs to N(M). | L3 | CO 3 | 12 | PDA Construction from Grammar |  |
|  | Convert the grammar SaSb/A, AbSa/S/ε to PDA that accepts the same language byempty stack. | L3 | CO 3 | 12 | PDA Conversion from Grammar |  |
|  | Construct a PDA for the language L={x€{a,b}\*/ na(x)>nb(x)} | L3 | CO 3 | 12 | Pushdown Automata |  |
|  | Design a PDA to accept {021|n>1}. Draw the transition diagram for the PDA. Show byinstantaneous description that the PDA accepts the strings‘0011’. | L3 | CO 3 | 12 | Pushdown Automata |  |
|  | Construct the PDA accepting the language L= { an/m,n>0} | L3 | CO 3 | 12 | PDA for Special Languages |  |
|  | Construct the PDA accepting the language L= { an/m,n>0} | L3 | CO 3 | 12 | Repeated PDA for Specific CFLs |  |
|  | Construct the PDA accepting the language L= { ai+/ i,j>0} | L3 | CO 3 | 12 | PDA for Complex Context-Free Patterns |  |
|  | Construct the PDA accepting the language L= { ai/ i >0} | L3 | CO 3 | 12 | PDA for Counting 'a' |  |
|  | Construct a PDA for the grammar S→ aB/bA, A → a/aS/bAA, B→b/bS/aBB. | L3 | CO 3 | 12 | PDA Equivalent to Given Grammar |  |
|  | Let G be the grammar given by S →0BB,B→0S/1S/0. Construct a PDA and test whether it satisfies 010000 the language. | L3 | CO 3 | 12 | PDA Verification |  |
|  | Construct a PDA equivalent to the following productions: S →aAA, A→aS/bS/a and hence whether the PDA accepts the string abaaaa. | L4 | CO 3 | 12 | PDA Validation with Specific String |  |
|  | Construct a CFG for the regular expression (011+1) (01). | L3 | CO 3 | 12 | Constructing a Context-Free Grammar (CFG) for a Regular Expression |  |
|  | Construct CFG for the language L={an/n is odd} | L4 | CO 3 | 12 | Constructing a Context-Free Grammar (CFG) for a Regular Expression |  |
|  | Show that the following grammar is ambiguous: SSbS/a. | L2 | CO 3 | 12 | Ambiguity in Context-Free Grammar (CF |  |
|  | Give the regular expression of the language generated by the CFG given below:  SaS/bS/a/b. Convert the RE to E-NFA | L4 | CO 3 | 12 | Regular Expression and Epsilon-NFA (E-NFA) Conversion |  |
|  | Explain about parse tree. For the following grammar. | L5 | CO 3 | 12 | Parse Tree and Derivation with Context-Free Grammar (CFG) |  |
|  | Convert the PDA to CFG | L5 | CO 3 | 16 | Converting PDA to CFG |  |
|  | A language is context free if and only if some PDA recognizes it. | L3 | CO 3 | 16 | PDA |  |
|  | Convert the grammar S→0S1/A, A→1A0/S/ into an equivalent PDA and check whether 0101 belongs to it. | L3 | CO 3 | 16 | PDA |  |
|  | Given the Grammar G=(V,Σ, R,E), Where  V= {E,D,1,2,3,4,5,6,7,8,9,0,+,-,\*,/,(,)}  Σ={1, 2,3,4,5,6,7,8,9,0,+,-,\*,/,(,)} and  R contains the following rules:  ED|(E)|E+E|E-E|E\*E|E/E  D0|1|2…..9. Find a parse tree for the string 1+2\*3 | L3 | CO 3 | 16 | Parse Tree |  |
|  | Show the derivation steps and construct derivation tree for the string ababbb' by using leftmost derivation with the grammar. S —> AB / ξ, A—> aB, B—> Sb | L3 | CO 3 | 16 | Parse Tree |  |