|  |  |
| --- | --- |
| **Date:** | **Tutorial-1** |
|  | **Basic concepts & Finite Automata** |

1. Design a FSM to accept binary numbers that are divisible by 3.
2. Design a DFA to accept strings over {a,b} that end with abb or bba.
3. Design a Moore and Mealy machine that receives input over {a,b} and generates output same as input except if the input contains “aba” then it is converted in output as “abb”.
4. Convert following Regular expression to NFA:
   * 1. (a+b)\*abb
     2. (aba+bbb)\*(ab)+

|  |  |
| --- | --- |
| **Date:** | **Tutorial-2** |
|  | **Regular Expression** |

1. Write regular expression for following expression over the set {0,1}
2. Set of strings that starts and ends with different symbols.
3. Set of strings that contain 3 consecutive number of 1’s or 0’s.
4. Set of strings containing even number if 1’s and no 0’s.
5. Set of strings that contain a ‘1’ at every even position.
6. Set of strings that contains a ‘1’ as third symbol from right.
7. Set of strings that do not end with 00 and 11.
8. Convert following DFA to RE:

Diagram

Description automatically generated

|  |  |
| --- | --- |
| **Date:** | **Tutorial-3** |
|  | **Grammars** |

1. Consider the given CFG S →aSa | bSb | a | b | Ɛ

Give LMD, RMD and Parse Treee for “abaaba”

1. Convert following into CNF

S→aAbB

A→aAa|Ɛ B→c|bbB|b C→AaA

1. Convert to GNF

S → AA|0

A →SS|1

|  |  |
| --- | --- |
| **Date:** | **Tutorial-4** |
|  | **PDA** |

Design PDA for

1. { 0n 1m 0n | n,m > 0 }
2. { (aba)n (cdc)n | n > 0 }
3. Well formed parentheses

|  |  |
| --- | --- |
| **Date:** | **Tutorial-5** |
|  | **TM** |

Design TM for

1. Well formed parentheses

1. 1’s and 2’s complement of binary number.
2. L={anbn |n>=1}