##### Theory of Computation

##### Assignment Questions

|  |  |  |
| --- | --- | --- |
|  | **UNIT I** | **Marks** |
|  | Construct a finite automaton for strings containing at least one *a* and at least one *b*. | 5 |
|  | Construct a finite automaton for binary strings with no consecutive 0’s. Show the computation for the input string *w*1 = 1011101 and for *w*2 = 1001. | 5 |
|  | Construct a finite automaton for some number of *a’*s followed by some number of *b’*s with the total length of the string being odd. | 5 |
|  | Construct a finite automaton for strings over the alphabet {*a*, *b*} of the form (*ab*)*n*, for example, *ababab*. | 5 |
|  | Construct a finite automaton for strings over the alphabet {*0, 1*} where every 00 is followed immediately by a 1, For example, the strings 1001, 0010, 0010011001 are in the language, but 0001 and 0100 are not. | 5 |
|  | For the transition table specified describe the language:   |  |  |  | | --- | --- | --- | | State | Input = *a* | Input = *b* | | -> *q*0 | *q*1 | *q*3 | | *q*1 | *q*3 | *q*2 | | \**q*2 | *q*2 | *q*2 | | *q*3 | *q*3 | *q*3 | | 5 |
|  | For the transition table specified describe the language:   |  |  |  | | --- | --- | --- | | **State** | **Input = *a*** | **Input = *b*** | | -> *q*0 | *q*2 | *q*1 | | *q*1 | *q*1 | *q*1 | | *q*2 | *q*3 | *q*2 | | \**q*3 | *q*3 | *q*2 | | 5 |
|  | Construct a Deterministic Finite Automaton to accept binary strings in which the substring 101 occurs somewhere in the string and the part before that has an odd number of zeros. E.g., 010010100, 1010100 and 011101110101 are in the language but 00101110 is not | 8 |
|  | Explain why we cannot construct a finite automatonfor theunequal numbers of *a’*s and *b’*s, in any order, for the alphabet {*a*, *b*}. | 8 |
|  | Construct a non-deterministic finite automaton (NFA) for Binary strings in which the first part of each string contains at least four 1 s and the second part contains at least three 0 s. | 8 |
|  | 1. Construct a non-deterministic finite automaton (NFA) for strings over {a, b} where the last 2 symbols in each string is a reversal of the first 2 symbols (i.e., last symbol = first symbol and penultimate symbol = second symbol). The NFA must contain only 10 states (not including reject states). | 8 |
|  | Construct a non-deterministic finite automaton (NFA) for Strings over {a, b} that contain at least three a s or at least two b s. | 8 |
|  | Explain why we cannot construct a finite automatonfor the palindromes (i.e., strings that read the same forward or backward, for example, *malayalam*) of arbitrary length. | 5 |
|  | Construct a non-deterministic finite automaton (NFA) for Strings over {*a, b, c*} that contain at least one *a* and at least one *b*. | 5 |
|  | Write an algorithm for reducing the number of states in a deterministic finite automaton. | 6 |
|  | What is the use of λ–transitions in a non-deterministic automaton? | 5 |
|  | Binary strings in which the sum of the last 4 digits is odd (e.g., 00101011 but not 00101001). | 8 |
|  | Construct a non-deterministic finite automaton (NFA) for Binary strings of any length with alternating 0 s and 1 s. The NFA must have just three states (not including reject states). How many states does an equivalent minimal DFA have? | 5 |
|  | Binary strings in which the first part of each string contains at least four 1 s and the second part contains at least three 0 s. | 8 |
|  | Convert the given table to an NFA diagram and then to an equivalent DFA.  The NFA specified by:   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **State** | **Input = *a*** | **Input = *b*** | **Input = *c*** | ***λ*** | | *q*0 | {} | {*q*1} | {*q*2} | {*q*1, *q*2} | | *q*1 | {*q*0} | {*q*2} | {*q*0, *q*1} | {} | | \* *q*2 | {} | {} | {} | {} | | 8 |
|  | Convert the given table to an NFA diagram and then to an equivalent DFA.  The NFA specified by:   |  |  |  |  | | --- | --- | --- | --- | | **State** | **Input = *a*** | **Input = *b*** | **Λ** | | *q*0 | {*q*0, *q*1} | {*q*1} | {} | | *q*1 | {*q*2} | {*q*1, *q*2} | {} | | \* *q*2 | {*q*0} | {*q*2} | {*q*1} | | 8 |
|  | Convert the following NFA to equivalent DFA and minimize the resulting DFA. | 10 |
|  | Convert the following NFA to equivalent DFA and minimize the resulting DFA. | 10 |
|  | Minimize the following DFA. | 6 |
|  | Minimize the following DFA. | 6 |

|  |  |  |
| --- | --- | --- |
|  | **UNIT II** | **Marks** |
|  | Construct a regular expression (RegEx) for Binary strings containing at least one 00 and at least one 11. | 2 |
|  | Construct a regular expression (RegEx) for Binary strings with at least two occurrences of at least two consecutive 1’s, the two occurrences not being adjacent (i.e., 011011 is acceptable but 011111 is not) | 2 |
|  | Construct a regular expression (RegEx) for Strings over {*a*, *b*, *c*} in which the fourth symbol from the beginning is a ‘*c’*. | 2 |
|  | Construct a regular expression (RegEx) for strings over {*a*, *b*} whose length is divisible by 2 but not by 3. | 2 |
|  | For the given, describe the language of the RegEx as concisely as possible:  1(0 + 1)(0 + 1)(0 + 1)(0 + 1)\*0 | 5 |
|  | For the given, describe the language of the RegEx as concisely as possible:  (b+λ)(a(a+λ)\*(b+λ))\*(a+λ)\* | 5 |
|  | For the given, describe the language of the RegEx as concisely as possible:  (aa)\*(bb)\*(b+λ) | 5 |
|  | For “Strings over {*a*, *b*} with an odd number of *a’*s and an odd number of *b’*s, Construct a finite automaton (DFA or NFA) first and then convert it to a RegEx. | 6 |
|  | For the Binary strings not containing the keyword 011, construct a finite automaton (DFA or NFA) first and then convert it to a RegEx. | 6 |
|  | For the Binary strings representing positive integers divisible by 3, construct a finite automaton (DFA or NFA) first and then convert it to a RegEx. | 6 |
|  | Convert the given RegEx to an equivalent NFA: (0+λ)(1+λ)(1+2)\*0(2+1)\* | 2 |
|  | Convert the given RegEx to an equivalent NFA: (0+11+10(1+00)\*01)\* | 2 |
|  | Convert the given RegEx to an equivalent NFA: ((a+b+c)c)\*(a+b+c+λ) | 2 |
|  | Are the given pairs of RegEx’s equivalent (i.e., do they represent the same set of strings)? (0+λ)(11\*0)\*(1+λ) and (1+λ)(011\*)\*(0+λ) | 6 |
|  | Are the given pairs of RegEx’s equivalent (i.e., do they represent the same set of strings)? (0\*1\*)\* and (0+1)\* | 6 |
|  | Are the given pairs of RegEx’s equivalent (i.e., do they represent the same set of strings)? (1+λ)(00\*1)\*0\* and (0+λ)(11\*0)1\* | 6 |
|  | Are the given pairs of RegEx’s equivalent (i.e., do they represent the same set of strings)? (0+1)\*(0+λ) and (1+λ)(1+0)\*(0+1+λ) | 6 |
|  | Can a RegEx with only union and concatenation operators (i.e., with no \* closure) represent an infinite number of strings? Explain. | 5 |
|  | Debug and correct the given incorrect RegEx’s: Binary strings with a 0 in every third position: 110(0 + 1)\* | 5 |
|  | Debug and correct the given incorrect RegEx’s:  Student letter grades represented as strings made up of the symbols {*A*, *B*, *C*, *D*, *F*} with no more than two *F* grades: (A+B+C+D)\*F(A+B+C+D)\*F | 6 |
|  | Construct a regular expression to accept all binary strings containing any of these patterns: 011 or 010 or 100. | 5 |
|  | Construct right-linear or left-linear grammars for the following regular languages: Binary strings in which every 0 is followed by 11. Construct a parse tree for the string 0111011 | 6 |
|  | Write a regular expression for all strings over {0, 1} in which the 3rd and the 4th symbols are the same as the 1st and the 2nd (respectively), the 7th & 8th symbols are the same as the 5th & 6th, and so on. Strings may be of any length. For eg, 01011010101 is in the language. | 8 |
|  | Prove that the given language is not regular  L={0n1m|m is not equal to n} | 6 |
|  | Prove that the language of binary strings of even length having the same number of 1’s in its two halves is not regular. | 6 |

It is mandatory to write each of the following.

* 2marks questions-5
* 5 marks question-4
* 6 Marks questions 2
* 8 marks question-1