ASSIGNMENT -01

NAME : MD . NAHID BABU

ID : 1511874642

SECTION: 01

SYBJECT: CSE473(THEORY OF COMPUTATION)

SUBMITTED TO: SFM1

DATE: 15/04/2019

SEMESTER: SPRING-2019

1.Give the formal definition of a DFA

ANS: A deterministic finite automaton (DFA) is a 5-tuple (Q; Σ; δ; $q0$; F), where

- Q is a finite set called the state,
- Σ is a finite set called the alphabet,
- δ: Q × Σ! Q is the transition function,
- q0 2 Q is the start state, and
- F ⊆ Q is the set of accept states

2. Design a DFA that accepts strings over Σ = {0,1} such that

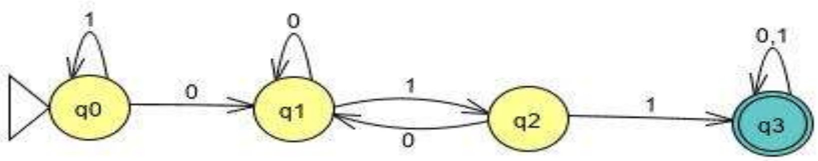
(a) 011 is a substring

(b) the string starts with a 01 and ends with 10

ANSWER:

(A)

This DFA will accept any string with ‘011’ substring

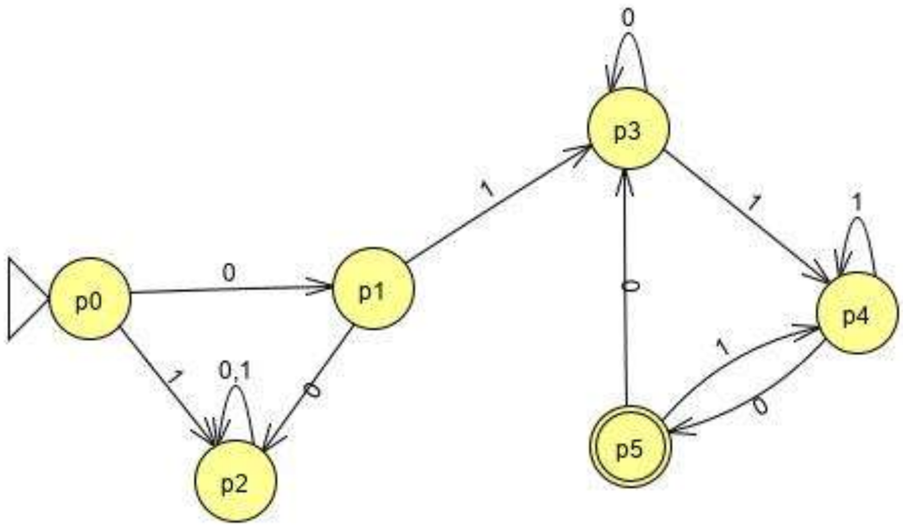


Test with inputs:

Input	Result
100110	Accept
1101111	Accept
01010101110	Accept
01010101	Reject
10101010	Reject

(B):

This DFA will accept any strings which start with ‘01’ and end with ‘10’.



Test with inputs:

Input	Result
11111111	Reject
0000000	Reject
10101010101	Reject
0101010101	Reject
000011111	Reject
01011010	Accept
0111111110	Accept

3. Find the regular expressions for 2(a) and 2(b).

ANSWER:

(2A):

$P_o = p_o1 + \varepsilon$ -------------------------------- (1)
$p_1 = p_o0 + p_10 + p_20$ ---------------------------(2)
$p_2 = p_11$ ---------------------------------------(3)
$p_3 = p_21 + p_30 + p_31$ -------------------------- (4)

For (1) we got,
$P_o = P_o 1 + \varepsilon$
$= \varepsilon 1 *$
$= 1^*$
For (2) we got,
$p_1 = P_o 0 + p_1 0 + p_2 0$
$= P_o 0 + p_1 0 + p_1 10$ [$p_2 = p_1 1$]
$= p_1(0+10) + P_o 0$
$= p_1(0+10) + 1^*0$ [$P_o = 1^*$]
$= 1^*0(0+10)^*$
For (3) we got,
$p_2 = p_1 1$
$= (1^*0(0+10)^*)1$ [$p_1 = 1^*0(0+10)^*$]
For (4) we got,
$p_3 = p_2 1 + p_3 0 + p_3 1$
$= p_3 (0+1) + p_2 1$
$= p_3 (0+1) + (1^*0(0+10)^*)1$ [$p_2 = (1^*0(0+10)^*)1$]
$= ((1^*0(0+10)^*)1) (0+1)^*$
REGULAR EXPRESSION = p_3
$= ((1^*0(0+10)^*1)1) (0+1)^*$

2(B):

$p_o = \varepsilon$……..(1)

$p_1 = p_o 0$……..(2)

$p_2 = p_o 1 + p_1 0 + p_2 0 + p_1 1$………(3)

$p_3 = p_1 1 + p_3 0 + p_5 0$……..(4)

$p_4 = p_3 1 + p_4 1 + p_5 1$……(5)

$p_5 = p_4 0$…………(6)

from (4)……

$p_3 = p_1 1 + p_3 0 + p_5 0$

$p_3 = p_o 01 + p_3 0 + p_4 00$ [from (2) and (6)]

$p_3 = \varepsilon 01 + p_3 0 + p_4 00$ [from (1)]

$p_3 = (01 + p_4 00) + p_3 0$

$p_3 = (01 + p_4 00)0^*$

$p_3 = 010^* + p_4 000^*$………(7)

from (5)

$p_4 = p_3 1 + p_4 1 + p_5 1$

$p_4 = (010^* + p_4 000^*)1 + p_4 1 + p_4 01$ [from (7) and (6)]

$p_4 = 010^*1 + p_4 000^*1 + p_4 (1+01)$

$p_4 = 010^*1 + p_4 (000^*1 + (1+01))$

$p_4 = 010^*1(000^*1 + (1+01))^*$ ……..(8)

from (6)

$p_5 = p_4 0$

$p_5 = (010^*1(000^*1 + (1+01))^*)0$

So,the regular expression for 2(b) is = $(010^*1(000^*1+(1+01))^*)0$.

4. Use pumping lemma to show that the language L = {1^n0^n; n ≥ 1} is nonregular

ANSWER:

To show that the language L = {1^n0^n; n ≥ 1} is non-regular we must follow below the steps.

STEP 1: Let us assume that L is not regular

STEP 2: If, therefore has a pumping length is P

STEP 3: So, S= 1^p 0^p where |s| = 2P>P

STEP 4 and STEP 5: Divided S into xyz

If P=4, then S = 11110000 let, x=11 , y=11 , z=0000

CASE 1: y has the one part

If P=4, then S = 11110000 let, x=111 , y=10 , z=000

CASE 2: y has the zero and one part

If P=4, then S = 1111 00 00 let, x=1111 , y=00 , z=00

CASE 3: y has the zero part

STEP 6:

CASE1:

If P=4, then S = 11 11 0000
I=3 xy3z = 11 111111 0000 ∉ L

CASE1:

If P=4, then S = 111 10 000
I=2 xy2z = 111 1010 000 ∉ L

CASE1:

If P=4, then S = 1111 00 00
I=3 xy3z = 1111 000000 00 ∉ L

So, L = {$\mathbf{1^n0^n}$; n ≥ 1} is non-regular
[SHOWED]

5. Implement the DFA designed in 2(a) using programming language of your own choice.

to implement 2(a) DFA I prefer java programming language.

I implement code that will take an input from user and accept these string which have '011' substring.

```
import java.util.Scanner;
```

```
import java.util.regex.Pattern;
import java.util.Arrays;
import java.util.regex.Matcher;

public class BinaryString {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
			Scanner input = new Scanner(System.in);
			 System.out.println("Enter any String:");
			 String s = input.nextLine();
			 char[] c = s.toCharArray();
			 boolean flag2 = false;
			 boolean flag1 = false;

				if(isStringOnlyAlphabet(s)) {

					flag1=true;
				}
				else if(isStringBinary(s)==false) {
					flag1=true;
```

```
        }

        else if(s.contains("011")) {
            flag2 =true;
        }

    if(flag1) {
        System.out.println("Invalid");
    }

    else if(flag2) {
        System.out.println("ACCEPT");
    }
    else {
        System.out.println("NOT ACCEPT");
    }

}


public static boolean isStringOnlyAlphabet(String str)
```

```
    {
       return ((str != null)
              && (!str.equals(""))
              && (str.matches("^[a-zA-Z]*$")));
    }

  public static boolean isStringBinary(String str)
    {


         long number = Long.parseLong(str);


         long targetInput = number;
         while (targetInput != 0) {
                if (targetInput % 10 > 1){
                       return false;
                       }
                targetInput = targetInput / 10;
                }
  return true;
    }
}
```

Test with different input:

```
Enter any String:
1010011100010101
ACCEPT
```

```
Enter any String:
10101010101010000
NOT ACCEPT
```

```
<terminated> BinaryStr
Enter any String:
24558
Invalid
```

---------------------------THE END-----------------------------