DETERMINISTIC FINITE AUTOMATA [DFA]

* AIM :

To write a C Program to simulate a Deterministic Finite Automata.

* ALGORTIHM

1. Draw a DFA for the given language and construct the transition table
2. Store the transition table in a two-dimensional array.
3. Initialize present-state, next-state & final state
4. Get the input string from the user
5. Find the length of the input string
6. Read the input string characters by character
7. Repeat step 6 for every characters
8. Refer the transition table for the entry corresponding to the present state and the current input symbol and update the next state
9. When we reach the end of the input if the final state is reached, the input is accepted. Otherwise the input is not accepted.

* PROGRAM:

```
#include <stdio.h>
#include <string.h>
#define max 20
int main()
{
    int trans_table[4][2] = {{1,3},{1,2},{1,2},{3,3}};
    int final_state = 2, i;
    int present_state = 0;
    int next_state = 0;
    int invalid = 0;
    char input_string[max];
    printf("Enter a string: ");
    scanf("%s", input_string);
    int l = strlen(input_string);
    for(i=0; i<l; i++)
    {
        if(input_string[i] == 'a')
        next_state = trans_table[present_state][0];
        else if(input_string[i] == 'b')
        next_state = trans_table[present_state][1];
        else
        invalid = 1
        present_state = next_state;
    }
```

```
if (invalid == 1)
{
  printf("Invalid input");
}
  else if (present_state == final_state)
  printf("Accepted");
  else
   printf("Don't Accept");
}
```

* OUTPUT

i) Enter a string : abaab
Accept

ii) Enter a string : abbbaaaba
Don't Accept.

CHECK WHETHER A STRING BELONGS TO THE GRAMMER

* AIM :

To write a C program to check whether a string belongs to the grammer

$S \longrightarrow 0A|1$

$A \longrightarrow 0A|1A|\epsilon$

* Language define by the Grammer

Set of all strings over $\Sigma = \{0,1\}$ that start with 0 and end with 1

* ALGORTIHM :

1. Get the input string from the user
2. Find the length of the string
3. Check whether all the symbols in the input are either 0 or 1. If so, Print "string is valid" and go to step 4. otherwise Print "string not valid" and quit the program
4. If the first symbol is 0 and the last symbol is 1, Print "String accepted", otherwise Print "string not Accepted".

* PROGRAM :-

```
#include <stdio.h>
# include <string.h>
int main()
char s[100];
int i, flag, l;
Printf("Enter a string to check")
Scanf("%s", s);
l = Strlen(s)
flag = 0
for (i=0; i<l; i++)
{
   if (s[i] != '0' && s[i] != '1')
   {
     flag = 0
   }
}
if (flag != 1)
     Printf("String is not valid \n");
if (flag == 1)
{
    if (s[0] == '0' && s[l-1] == '1')
                                  Printf("string is accepted \n");
    else
                                  Printf("string is not Accepted \n");
}
}
```

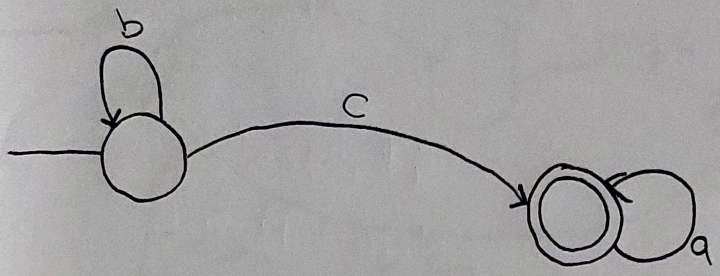
* OUTPUT :-

Enter a string to check : 0101011110

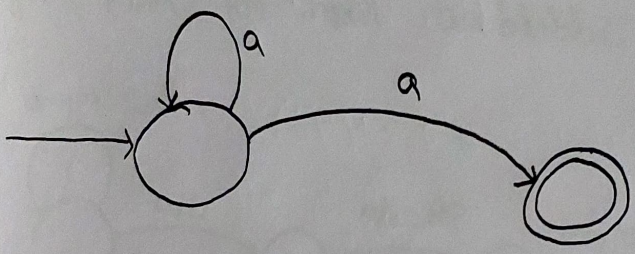
String is accepted.

1. Design DFA to Accept bcaaaaaa, bc and c

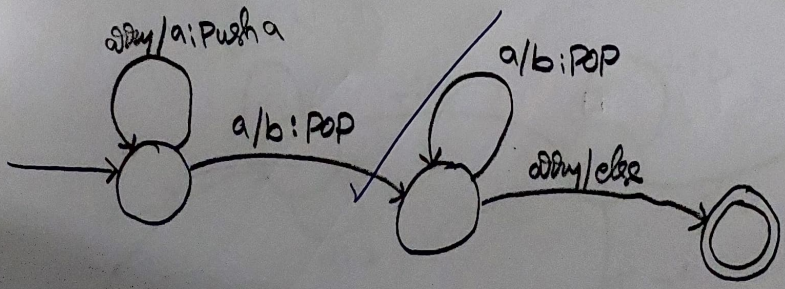
Ans.



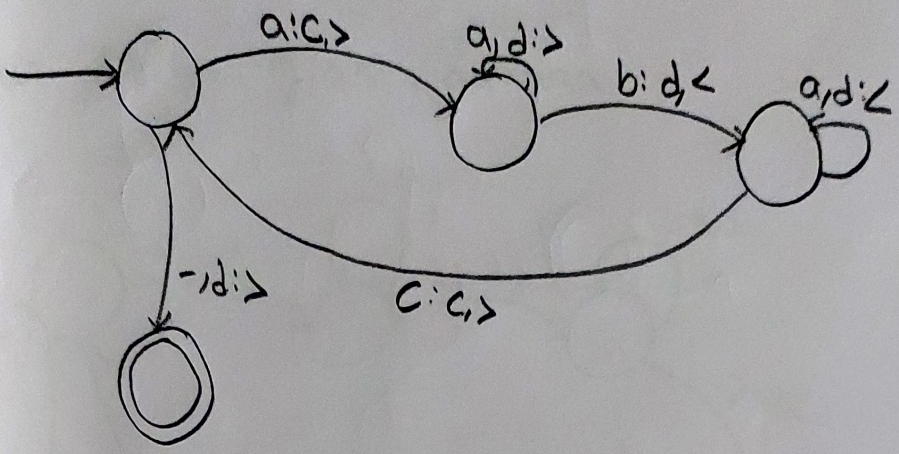
2. Design NFA to accept aaaaa



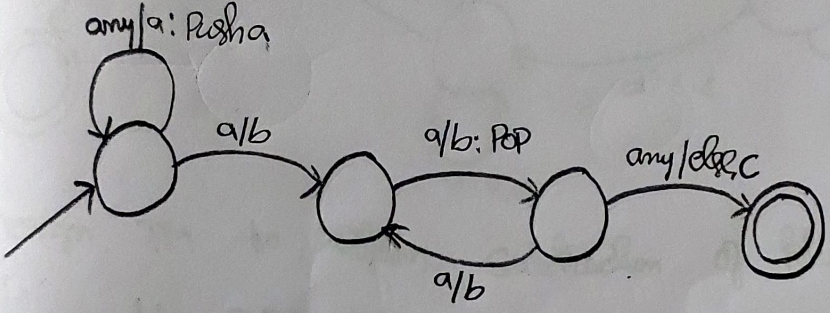
3. Design PDA for the input a^n b^n



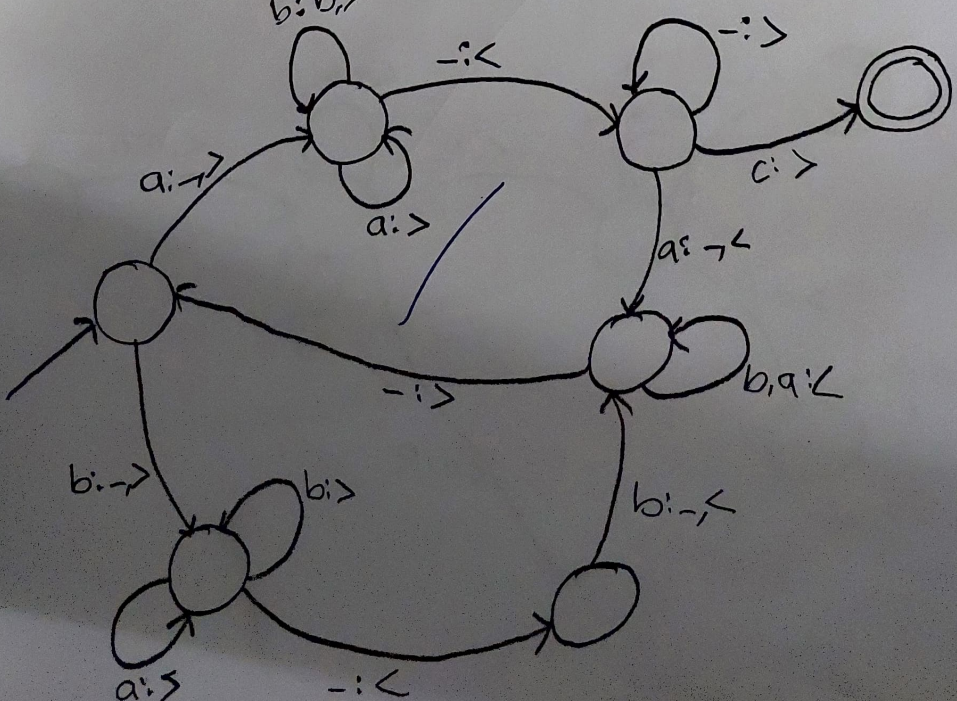
4. Design TM for input $a^n b^n$



5 Design PDA for input aa bbbbC ($a^n b^{n2n}$)



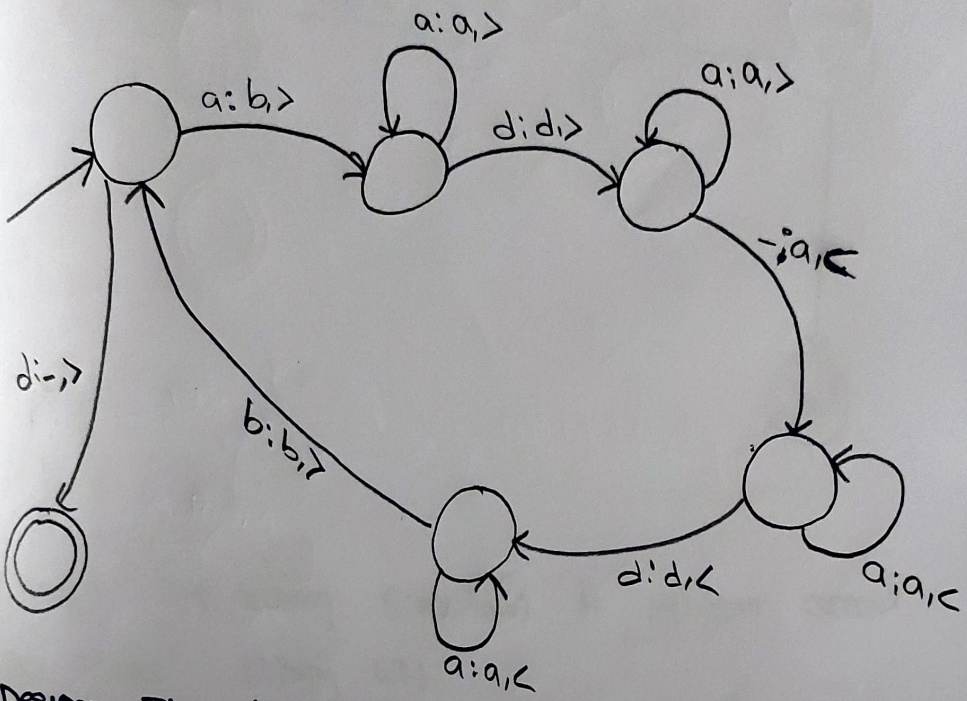
6. Design TM Simulation for palindrome W= ababa c.



7. Design TM to perform addition of following

W = aa + aaaa

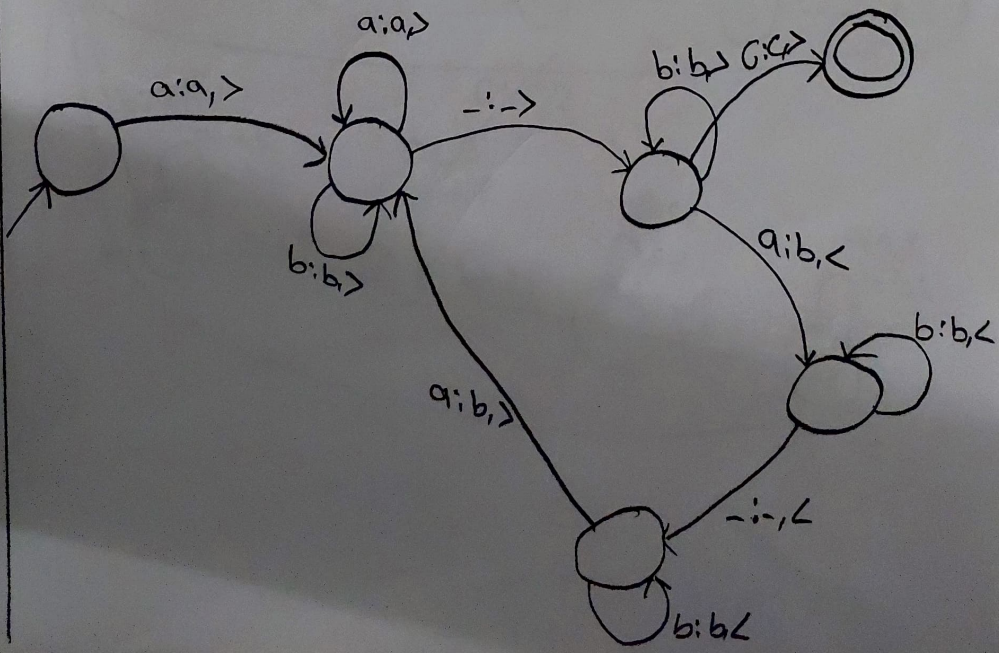
After Addition of a's = aaaaaa



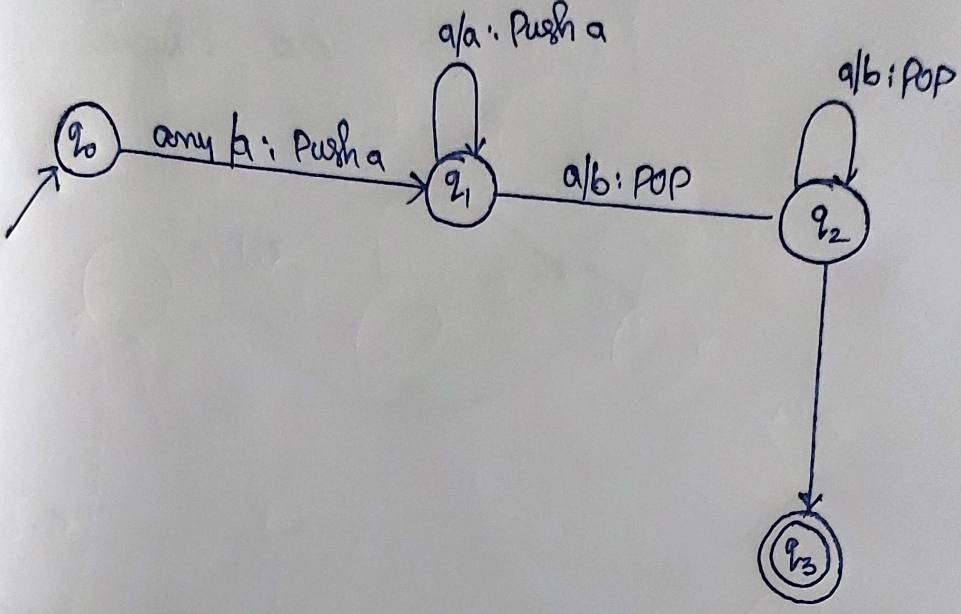
8 Design TM to perform Subtraction of following

W = aaa - a

The Result of Subtraction is = a

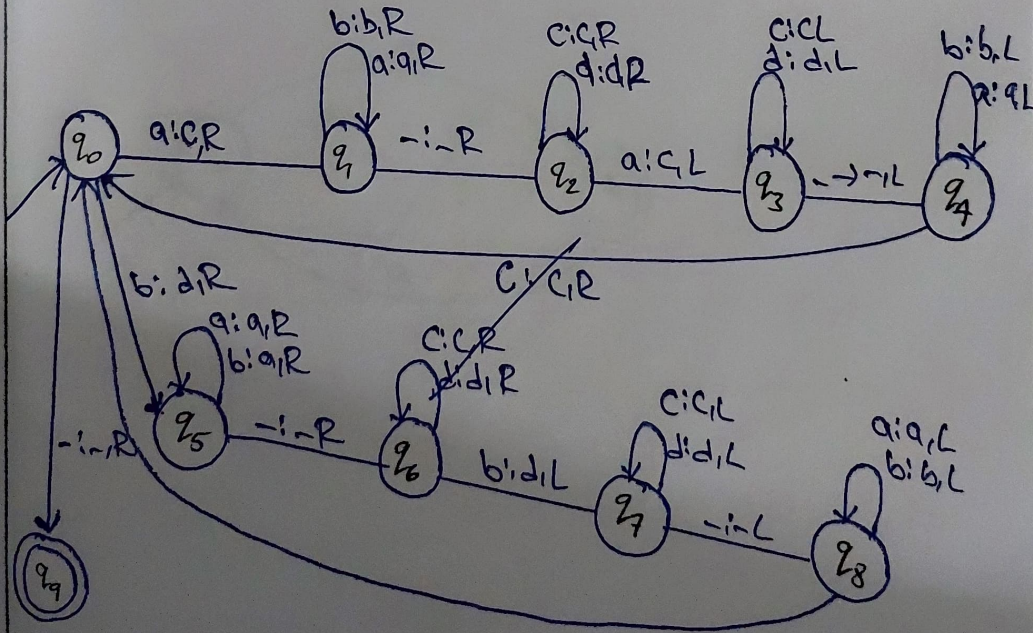


9. Design PDA using simulator to accept the input string aabb



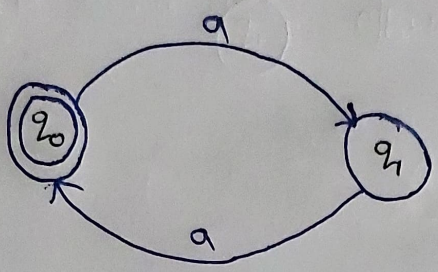
10. Design TM using simulator to perform accept the input string WW

Input : aba aba



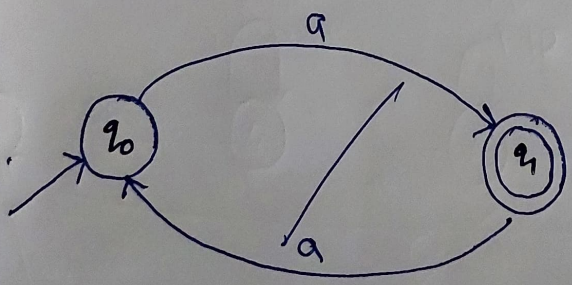
11 Design DFA using simulator to accept Even number of a's

Input : aa



12. Design DFA using simulator to accept odd number of a's

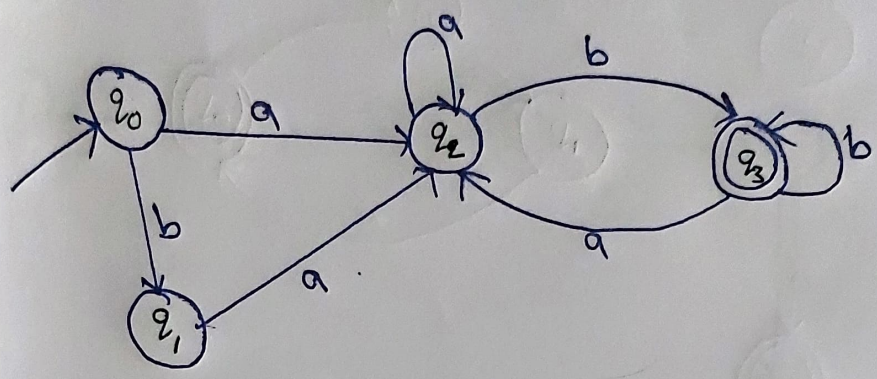
Input : a a a



13. Design DFA using simulator to accept the string the end with ab over set {a, b}

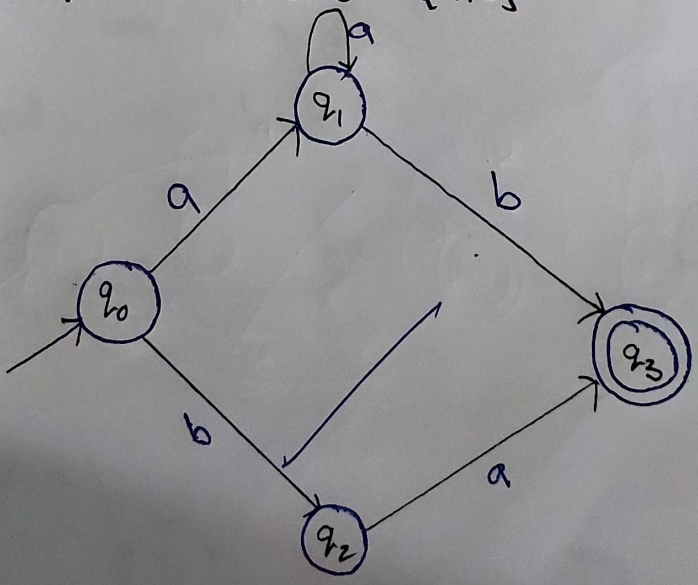
W = aaa bab

Input : aaa bab



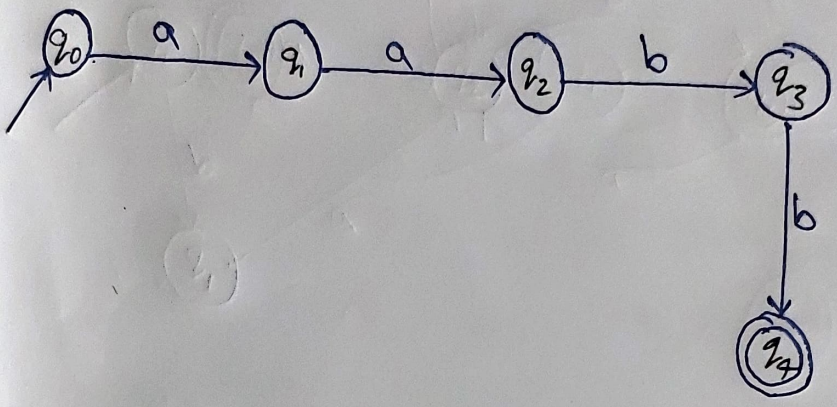
14. Design DFA Using simulator to accept the string start with a & b over set {a, b}

Input : a & b {a, b}



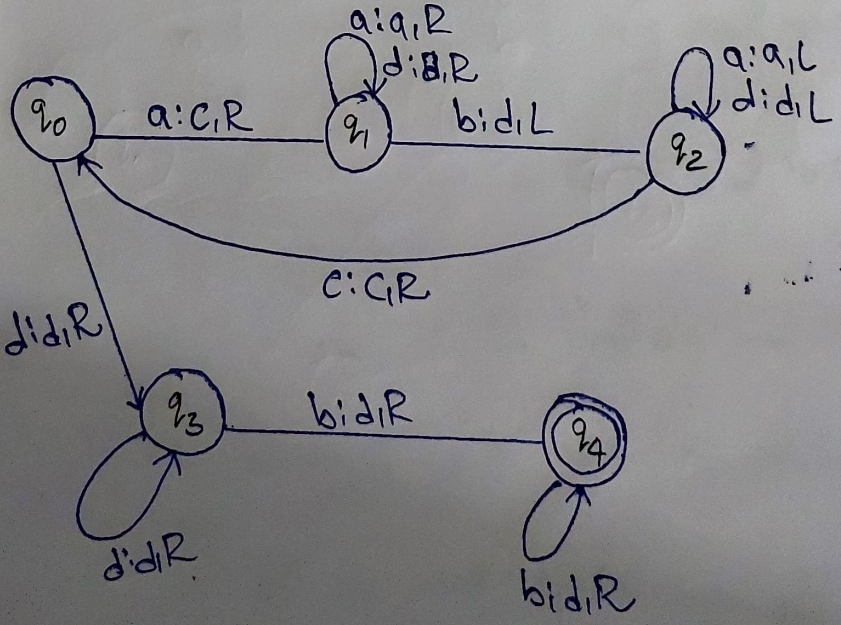
15. Design DFA using simulator to accept the string having 'ab's substring over the set {a,b}

Input : a a b b



16. Design TM using simulator to accept the input string A^n B^2n

Input : a a b b b b



TURING MACHINE

* AIM:

To write a C Program to simulate a Turing Machine for the language $L = \{0^n 1^{2n} / n \geq 1\}$ in which n number of 0's are followed by 2n number of 1's

* ALGORITHM

1. Get the input string from the user.
2. Find the length of the string
3. Read a '0', change it to A and move one position to the right
4. Skip all 0's and B's if any and move in the right
5. Read a '1' change it to B and move one position to the right
6. Skip all 1's and B's if any and move in the Right
7. Read a '2', change it to 'c' and move one position to the left
8. Skip all C's, 1's, B's and 0's and move left
9. When we read a 'A' move one position to the right
   a. If the next symbol is 0, goto step 3.

6. Otherwise, if the next symbol is B, then skip all B's and move right. After skipping all B's

i. If there is a C, then skip all C's and move in the right.

ii If there is a symbol other than C, print "String not accepted".

10. End the program.

* PROGRAM:

```
# include <stdio.h>
# include <string.h>
 void main()
{
   int i,j,le, flag, flag1, flag2;
   char str[20];
   Printf(" program to show how a
turing Machine will Process on\n\n");
   Printf(" Enter a string");
   Scanf("%s", str);
    le = strlen(str);
    j=0
    while (1)
   {
     flag =0; flag1=0; flag2=0, i=0;
     while (i<le)
     {
       if ((str[i] == '0' && (flag == 0))
       {
         str[i] = 'A';
         Printf("%s\n", str);
         flag =1;
          i=i+1;
       }
       else if ((str[i] == '0' && (flag == 1))
       {
         i=i+1;
       } else if (str[i] == 'A')
```

```
    i=i+1;
  }
  else if (Str[i]=='1') && (flag1==0)
  {
      Str[i]='B';
      Printf("%s\n", Str);
      flag1=1;
      i=i+1;
  }
  else if ((Str[i]=='1') && (flag1==1))
  {
     i=i+1;
  }else if (Str[i]=='B')
  {
   i=i+1;
  }
  else if ((Str[i]=='0') && (flag2==0))
  {
      Str[i]='C';
      Printf("%s\n", Str);
      flag2=1;
      i=i+1;
  }
  else if ((Str[i]=='0') && (flag2==1))
  {
      i=i+1;
    }  else if (Str[i]=='C')
    {
     i=i+1;
    }
  }
  j=j+1;
  if (j==e)
```

```
        {
            break;
        }
    }
}
```

*OUTPUT

Program to show how a turning machine will process on 1s 2s

Enter a string : 000 111 222

A 00 111 222
A 00 B11 222
A 00 B11 C22
AA 0 B11 C22
AA 0 BB1 C22
AA 0 BB1 CC2
AA A BB1 CC2
AA A BBB CC2
AA A BBB CCC