EXPERIMENT 2

1. Write a C program to simulate a Deterministic Finite Automata (DFA) for the given language representing strings that start with 0 and end with 1

# AIM:

To write a C program to simulate a Deterministic Finite Automata (DFA) for the given language representing strings that start with 0 and end with 1

# APPARATUS:

DEV C++

# ALGORITHM:

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 and final\_state

4. Get the input string from the user.

5. Find the length of the input string.

6. Read the input string character by character.

7. Repeat step 8 for every character

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

# C PROGRAM:

#include<stdio.h>

#include<conio.h>

#include<string.h>

int validate(char[]);

int s\_table[3][2]={{1,5}

,{2,1}, {2,1}};

char str[10];

int main(){

int i,j,k;

printf("\n\tTHIS DFA WILL HANDLE THE STRINGS WHICH START AND END WITH A");

printf("\nEnter any string: ");

scanf("%s",str);

validate(str);

}

int validate(char str[]){

int c\_state=0,f\_state=2,i,x=1;

int l=strlen(str);

if(str[0]=='0')

printf("0");

for(i=0;i<l;i++){

if(str[i]=='0'){

x=0;

}

else if(str[i]=='1'){

x=1;

}

c\_state=s\_table[c\_state][x];

if(c\_state==5)

break;

printf("---->%d",c\_state);

}

if(c\_state==f\_state)

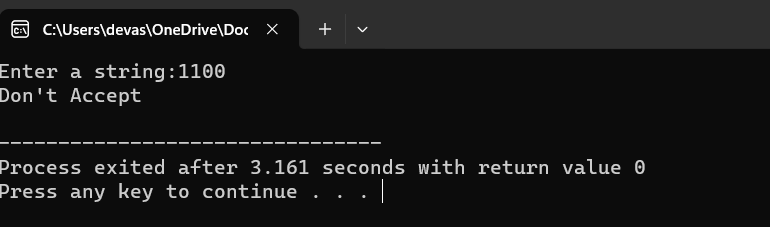
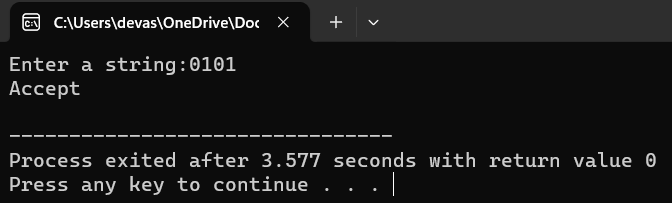
printf("\nVALID STRING");

else

printf("\nINVALID STRING");

}

# OUTPUT:



# RESULT:

Thus the C program is executed successfully and verified string.