Name – Bhargav Shamuvel Gurav | **PRN** – 2041009

Class – L.Y. B-Tech (Computer) | **Batch** – B1

Course Code – CO406U | **Course Name -** CDL

Practical no. 10

Aim: Implement Deterministic Finite Automata.

Theory :

Finite state machine

- Finite state machine is used to recognize patterns.
- Finite automata machine takes the string of symbol as input and changes its state accordingly. In the input, when a desired symbol is found then the transition occurs.
- While transition, the automata can either move to the next state or stay in the same state.
- FA has two states: accept state or reject state. When the input string is successfully processed and the automata reached its final state then it will accept.

A finite automata consists of following:
Q: finite set of states
∑: finite set of input symbol
q0: initial state
F: final state
δ: Transition function

Transition function can be define as

1. δ: Q x ∑ →Q

   FA is characterized into two ways:
   1. DFA (finite automata)
   2. NDFA (non deterministic finite automata)

DFA

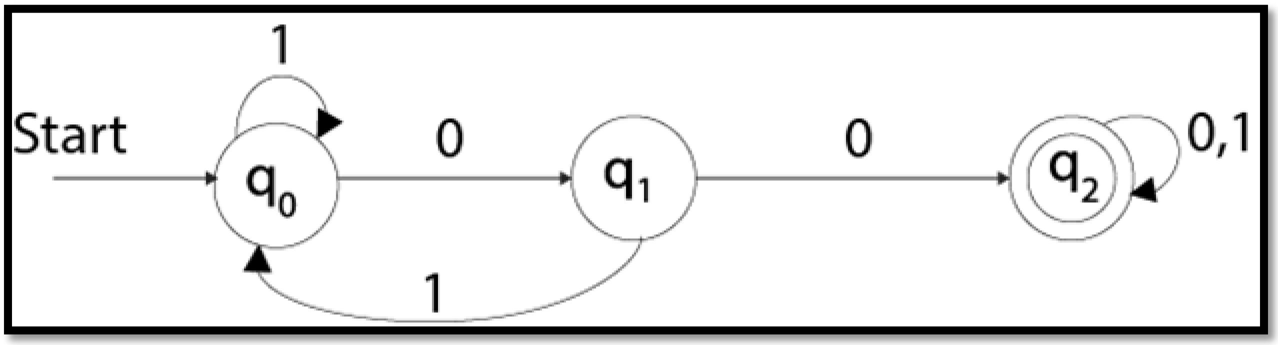
DFA stands for Deterministic Finite Automata. Deterministic refers to the uniqueness of the computation. In DFA, the input character goes to one state only. DFA doesn't accept the null move that means the DFA cannot change state without any input character.

DFA has five tuples {Q, ∑, q0, F, δ}

Q: set of all states
∑: finite set of input symbol where δ: Q x ∑ →Q
q0: initial state
F: final state
δ: Transition function

Example
See an example of deterministic finite automata:

1. Q = {q0, q1, q2}
2. ∑ = {0, 1}
3. q0 = {q0}
4. F = {q3}



NDFA
NDFA refer to the Non Deterministic Finite Automata. It is used to transit the any number of states for a particular input. NDFA accepts the NULL move that means it can change state without reading the symbols.

NDFA also has five states same as DFA. But NDFA has different transition function.

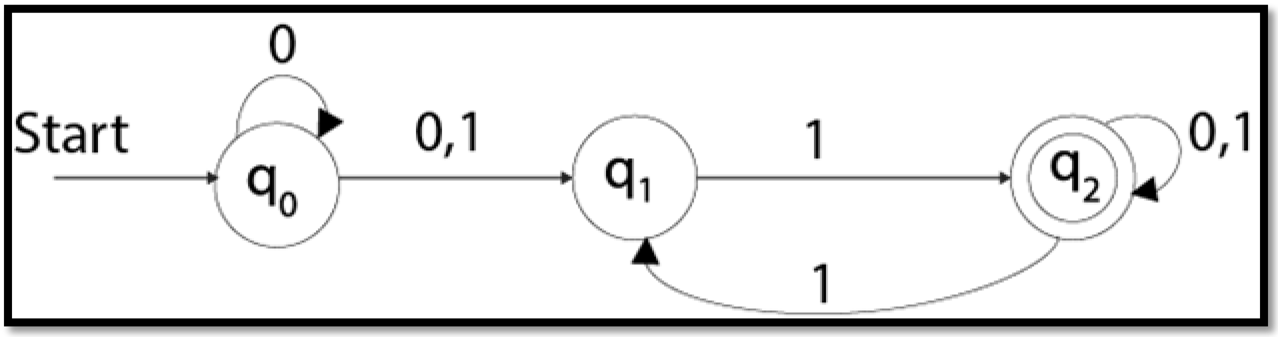
Transition function of NDFA can be defined as:

δ: Q x ∑ →2Q

Example
See an example of non deterministic finite automata:

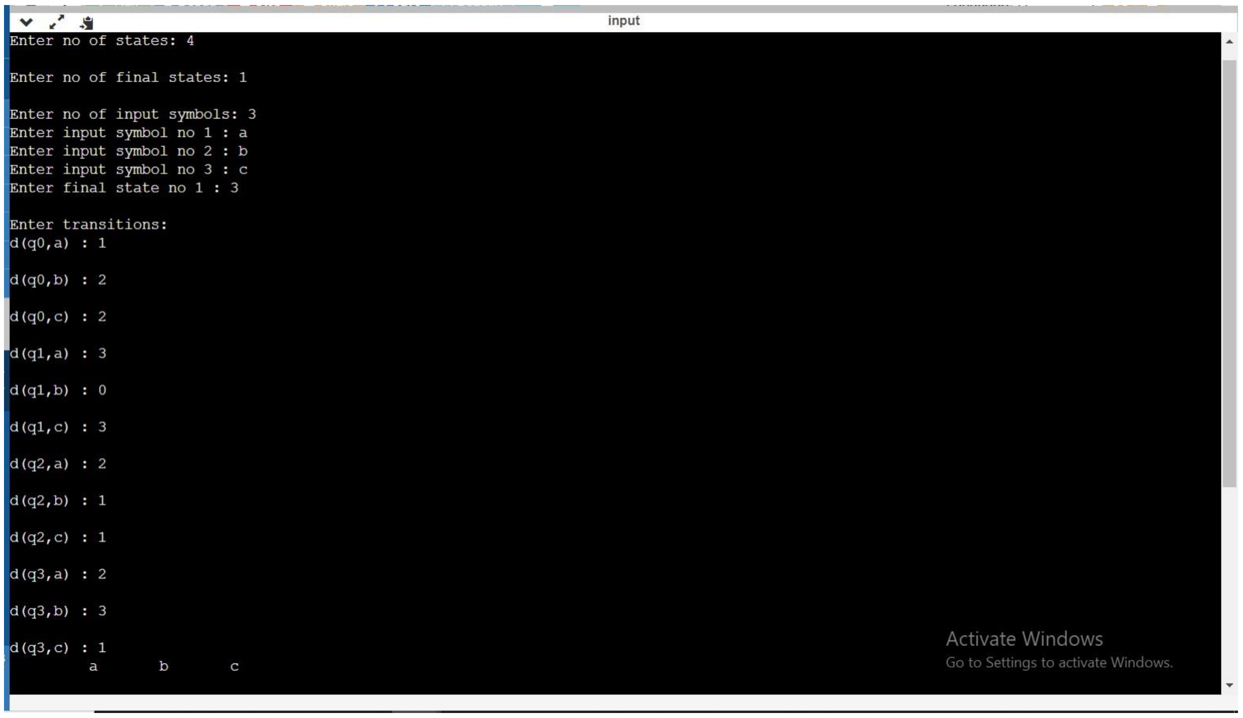
1. Q = {q0, q1, q2}
2. ∑ = {0, 1}
3. q0 = {q0}

4. F = {q3}

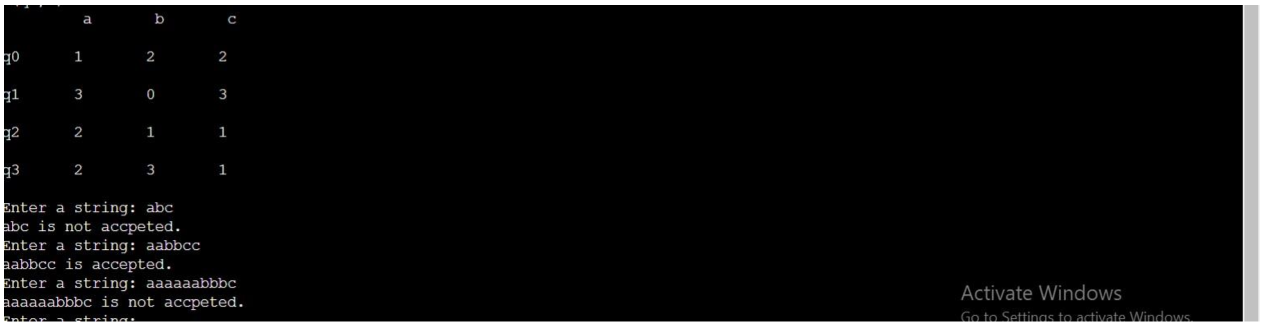


Program Code:

```
#include<stdio.h>
#include<string.h>
typedef struct DFA
{
 int nos;
 int  noi;
 int nof;
 int delta[10][10];
 int final[10];
 char inputSymbols[10];
}DFA;
int checkSymbol(char ch,DFA d)
{
 for(int i=0;i<d.noi;i++)
 {
  if(ch == d.inputSymbols[i])
  {
   return i;
  }
 }
 return -1;
}
int checkFinalState(int st,DFA d)
{
 for(int i=0;i<d.nof;i++)
 {
  if(st == d.final[i])
  {
   return 1;
  }
 }
 return 0;
}
int main()
{
 DFA d;
 printf("\nEnter no of states: ");
 scanf(" %d",&d.nos);
 printf("\nEnter no of final states: ");
 scanf(" %d",&d.nof);
 printf("\nEnter no of input symbols: ");
 scanf(" %d",&d.noi);
 // accept the input symbols
 for(int i=0;i<d.noi;i++)
 {
  printf("Enter input symbol no %d : ",i+1);
  scanf(" %c",&d.inputSymbols[i]);
 }
 // accept the final states
 for(int i=0;i<d.nof;i++)
 {
  printf("Enter final state no %d : ",i+1);
  scanf(" %d",&d.final[i]);
 }
 printf("\nEnter transitions: ");

 for(int i=0;i<d.nos;i++)
  for(int j=0;j<d.noi;j++)
  {
   printf("\nd(q%d,%c) : ", i,d.inputSymbols[j]);
   scanf(" %d",&d.delta[i][j]);
  }
 // print the transition table
 // print the symbols as columns of transition table
 for(int i=0;i<d.noi;i++)
  printf("\t %c",d.inputSymbols[i]);
 printf("\n");
 for(int i=0;i<d.nos;i++)
 {
  printf("\nq%d",i);
  for(int j=0;j<d.noi;j++)
  {
   printf("\t%d",d.delta[i][j]);
  }
  printf("\n");
 }
 do{
  char string[10];
  printf("\nEnter a string: ");
  scanf("%s",string);
  int stateCounter = 0;
  int flag = 1;
  for(int i=0;i<strlen(string);i++)
  {
   int symPos = checkSymbol(string[i],d);
   if(symPos==-1)
   {
    flag = 0;
    break;
   }
   stateCounter = d.delta[stateCounter][symPos];
  }
  if(flag==1 && checkFinalState(stateCounter,d)==1)
  {
   printf("%s is accepted. ",string);
  }
  else
  {
   printf("%s is not accpeted. ",string);
  }
 }while(1);

 return 0;
}
```
Output:



```
Enter no of states: 4

Enter no of final states: 1

Enter no of input symbols: 3
Enter input symbol no 1 : a
Enter input symbol no 2 : b
Enter input symbol no 3 : c
Enter final state no 1 : 3

Enter transitions:
d(q0,a) : 1

d(q0,b) : 2

d(q0,c) : 2

d(q1,a) : 3

d(q1,b) : 0

d(q1,c) : 3

d(q2,a) : 2

d(q2,b) : 1

d(q2,c) : 1

d(q3,a) : 2

d(q3,b) : 3

d(q3,c) : 1
         a       b       c
```



```
         a       b       c

q0       1       2       2

q1       3       0       3

q2       2       1       1

q3       2       3       1

Enter a string: abc
abc is not accpeted.
Enter a string: aabbcc
aabbcc is accepted.
Enter a string: aaaaaabbbc
aaaaaabbbc is not accpeted.
```

Conclusion : In this practical we implemented finite automata.