Heaven's Light is our Guide

RAJSHAHI UNIVERSITY OF ENGINEERING & TECHNOLOGY

Department of Computer science & Engineering

Lab-02

Submitted by:
MD.Abu Farhad

Course no:CSE -2205
Roll no : 1503116

Submitted to:
Sadia zaman mishu

Lecturer

Experiment no: 2

Experiment name: DFA Deign with decimal digit.

Theory: In the theory of computation, a branch of theoretical computer science, a deterministic finite automaton (DFA)—also known as a deterministic finite acceptor (DFA) and a deterministic finite state machine (DFSM)—is a finite-state machine that accepts and rejects strings of symbols and only produces a unique computation (or run) of the automaton for each input string. Deterministic refers to the uniqueness of the computation.

A deterministic finite automaton consists of a set of finitely many states, which one might number 0, 1, 2, and where one might assume that the machine is initially in the state 0.

The second ingredient of the DFA is an update function which tells for each state and symbol read what the succeeding state is.

Furthermore, some of the states are called accepting states; the other states are called rejecting states.

Now a DFA processes a word w by starting in the state 0 and updating the state for each symbol in the word according to the update function; if the state reached this way is "accepting" then the DFA accepts the word w else the DFA rejects the word w.

Code:

```
#include<bits/stdc++.h>

using namespace std;

int dfa_table[100][100];

int main()

{
```

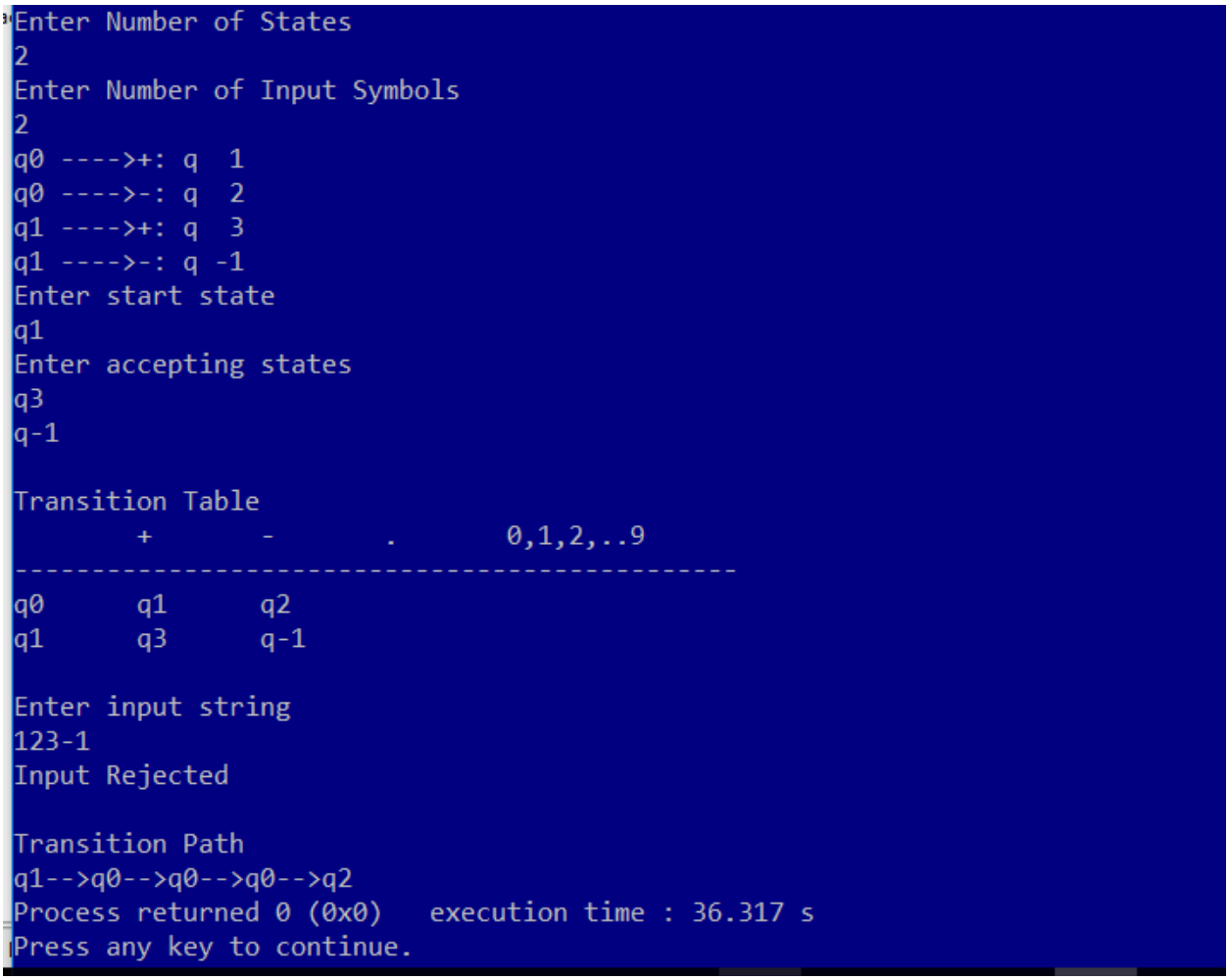
```
map<char, int> charMap;
charMap['+'] = 0;
charMap['-'] = 1;
charMap['.'] = 2;
charMap['0'] = 3;
charMap['1'] = 3;
charMap['2'] = 3;
charMap['3'] = 3;
charMap['4'] = 3;
charMap['5'] = 3;
charMap['6'] = 3;
charMap['7'] = 3;
charMap['8'] = 3;
charMap['9'] = 3;
//////////
int no_of_states, no_of_symbol;
cout<<"Enter Number of States "<<endl;
cin>>no_of_states;
cout<<"Enter Number of Input Symbols "<<endl;
cin>>no_of_symbol;
for(int i=0;i<no_of_states; i++)
{
for(int j=0; j<no_of_symbol; j++)
```

```
{
char c;
switch(j)
{
case 0: c = '+';break;
case 1: c = '-';break;
case 2: c = '.';break;
case 3: c = '#';break;
}
printf("q%d ---->%c: q",i, c);
cin>>dfa_table[i][j];   }}
int start_state, accepting_state;
cout<<"Enter start state"<<endl;
cout<<"q";
cin>>start_state;
vector<int> accepting_states;

cout<<"Enter accepting states"<<endl;
while(1)
{
cout<<"q";
int x;
cin>>x;
```

```
if(x==-1)
break;
accepting_states.push_back(x);
}
int current_state = start_state;
//////////////
cout<<endl<<"Transition Table"<<endl;
cout<<"\t+\t-\t.\t0,1,2,..9"<<endl;
cout<<"-----------------------------------------------"<<endl;
for(int i=0;i<no_of_states; i++)
{
printf("q%d",i);
for(int j=0; j<no_of_symbol; j++)
printf("\tq%d", dfa_table[i][j]);
cout<<endl;
}
///////////////
cout<<endl<<"Enter input string"<<endl;
string input;
cin>>input;
vector<int> path;
for(int i=0; i<input.size(); i++)
{
```

```
int ch = charMap[input[i]];
path.push_back(current_state);
current_state = dfa_table[current_state][ch];
}
if(current_state==accepting_states[0] || current_state==accepting_states[1] )
cout<<"Input Accepted"<<endl;
else
cout<<"Input Rejected"<<endl;
cout<<endl<<"Transition Path"<<endl;

for(int i=0;i<path.size();i++) {
if(i!=path.size()-1)
cout<<"q"<<path[i]<<"-->";
else
cout<<"q"<<path[i];    }
return 0;  }
```

Output:



Conclusions: For Decimal DFA a string consist of decimal digit. From one input Starting state to accepted input process the string with the decimal digit. And finally print the transition path of the string which process. Input string may with decimal and without decimal digit. Any of input string process it from starting state to accepting state.