Program 1:

WAP to implement a program that takes an input string from the console and verifies it against a Deterministic Finite Automaton which is given through a separate file.

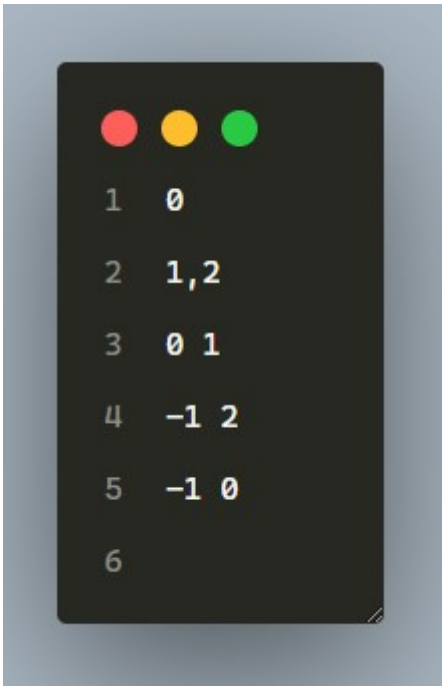
```c
#include <stdio.h>
#include <string.h>

#define MAX_TRANSISTION 100
#define MAX_VARIABLES 100
#define MAX_FINAL_STATES MAX_VARIABLES

int isInFinalState(int arr[], int len, int state)
{
 for (int i = 0; i < len; ++i)
  if (arr[i] == state)
   return 1;
 return 0;
}

int Read(char *str)
{
 int i = 0;
 while (1)
 {
  str[i] = getchar();
  if (str[i] == '\n' || str[i] == '\r')
  {
   str[i] = '\0';
   return i;
  }
  i++;
 }
}

int main()
{

 int i = 0, j = 0;
 int tempInt1, tempInt2;
 char ch1, ch2;

 FILE *filePointer = NULL;
 filePointer = fopen("DFA.txt", "r");

 int finalStates[MAX_FINAL_STATES] = {0};
 unsigned int finalStatesLen = 0;
 int DFA_Table[MAX_TRANSISTION][MAX_VARIABLES];
```

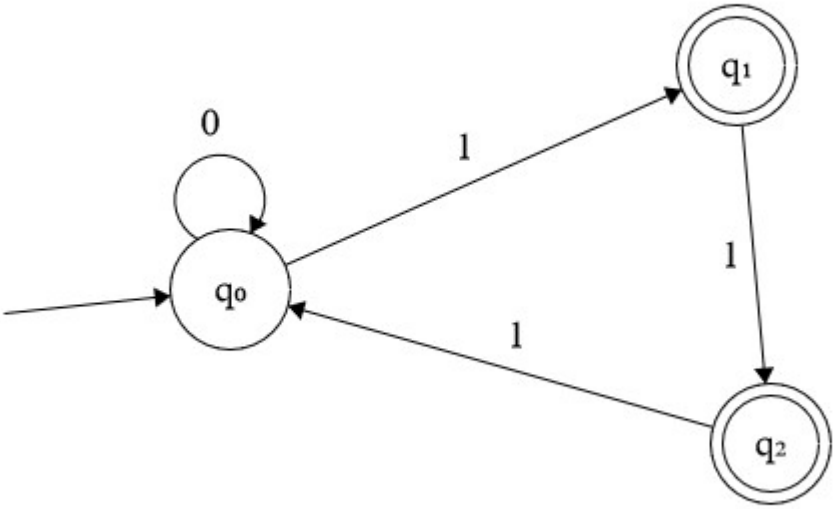
```
for (i = 0; i < MAX_TRANSISTION; ++i)
 for (j = 0; j < MAX_VARIABLES; ++j)
  DFA_Table[i][j] = -1;

if (filePointer == NULL)
{
 printf("Unable to open DFA.txt\n");
 return 1;
}

int initialState = -1;
int currentState = -1;
int numberOfStates = 0;
int numberOfInputs = 0;

fscanf(filePointer, "%d%c", &initialState, &ch1);
printf("Initial state : %d\n", initialState);

printf("Final states : ");

do
{
 fscanf(filePointer, "%d%c", &tempInt1, &ch1);
 finalStates[finalStatesLen] = tempInt1;
 finalStatesLen++;
 printf("%d, ", tempInt1);
} while (ch1 != '\n');
printf("\n");

i = j = 0;
while (fscanf(filePointer, "%d%c", &tempInt1, &ch1) != EOF)
{
 DFA_Table[i][j] = tempInt1;
 j++;
 if (ch1 == '\n')
  numberOfInputs = j, j = 0, i++;
}
numberOfStates = i;
fclose(filePointer);

printf("Transistion Table\n");
printf("| State | Input(0) | Input(1) |\n");
for (i = 0; i < numberOfStates; ++i)
{
```

```
  printf("| %3d    | ", i);
  for (j = 0; j < numberOfInputs; j++)
  {
   printf("   %3d    | ", DFA_Table[i][j]);
  }
  printf("\n");
 }

 int flag = 0;
 char inputString[100] = {0};
 int inputStringLen = 0;

 while (1)
 {
  printf("\n-----------------------\nEnter input string ('#' to
exit) : ");
  inputStringLen = Read(inputString);

  if (inputString[0] == '#')
   break;

  printf("Input string : %s, Input string len : %d\n\n",
inputString, inputStringLen);

  currentState = initialState;
  flag = 0;
  for (i = 0; i < inputStringLen; ++i)
  {
   tempInt1 = inputString[i] - '0';

   printf("string and state transition : %d , q%d --> ", tempInt1,
currentState);
   currentState = DFA_Table[currentState][tempInt1];

   if (currentState == -1)
   {
    printf("NO TRANSISTION\n");
    flag = 1;
    break;
   }
   printf("q%d\n", currentState);
  }

  if (flag == 1 || !isInFinalState(finalStates, finalStatesLen,
```

```
currentState))
    printf("\nString %s is NOT accepted by the DFA!\n", inputString);
  else
    printf("\nString %s is accepted by the DFA!\n", inputString);
 }

 printf("\nExiting ... \n");

 return 0;
}
```

DFA.txt





Output :

```
Initial state : 0
Final states : 1, 2,
Transistion Table
| State | Input(0) | Input(1) |
|   0   |      0   |      1   |
|   1   |     -1   |      2   |
|   2   |     -1   |      0   |

-----------------------
Enter input string ('#' to exit) : 0011
Input string : 0011, Input string len : 4

string and state transition : 0 , q0 -> q0
string and state transition : 0 , q0 -> q0
string and state transition : 1 , q0 -> q1
string and state transition : 1 , q1 -> q2

String 0011 is accepted by the DFA!

-----------------------
Enter input string ('#' to exit) : 00100
Input string : 00100, Input string len : 5

string and state transition : 0 , q0 -> q0
string and state transition : 0 , q0 -> q0
string and state transition : 1 , q0 -> q1
string and state transition : 0 , q1 -> NO TRANSISTION

String 00100 is NOT accepted by the DFA!

-----------------------
Enter input string ('#' to exit) :
Input string : , Input string len : 0


String  is NOT accepted by the DFA!

-----------------------
Enter input string ('#' to exit) : #

Exiting...
```