INSTITUTE OF ENGINEERING & MANAGEMENT

Department of Computer Science & Engineering

Name : Subhadip Roy

Section : A

Class Roll No. : 19

Enrollment No. : 12019002002027

Subject Name : Formal Language and Automata Theory

Assignment : Mini Project

Subject Code : PCCCS502

Date : 25 / 12 / 2021

Problem Statement

Given a regular expression, write down a program which shows the step by step labeled graph of corresponding finite automaton. Moreover, perform this task preferably using Java language in order to construct a graphical tool for visualization of the above process.

Solution ->

In this project, we have developed a program which shows the step-by-step labeled graph of the finite state automaton for any given regular expression as user input. To construct a graphical tool for visualization of the above process, we have used C language.

Now, we know that :-

Finite Automata (FA) is the simplest machine to recognize patterns. The finite automata or finite state machine is an abstract machine having five elements or tuples.

Deterministic Finite Automata (DFA) consists of 5 tuples {Q, **Σ**, q, F, **δ**}.

Q : set of all states.

Σ : set of input symbols. (Symbols which machine takes as input)

q : Initial state. (Starting state of a machine)

F : set of final state.

δ : Transition Function, defined as **δ** : Q X **Σ** -> Q.

In a DFA, for a particular input character, the machine goes to 1 state only.

A transition function is defined on every state for every input symbol. Also in DFA null (or ε) move is not allowed i.e. DFA cannot change state without any input character.

Keeping these concepts in mind, let's proceed to the source code.

Source Code ->

```
// Header file inclusions :-
#include <stdio.h>
#include <string.h>
#include <graphics.h> // helps us to draw the graph

int n = 0;
int mattrix[30][4];

// prototype for matrix insertion method
void insert(int p, int q, int r, int s);

// Driver method:-
int main(void)
{
    char str[30];
    int i, pre_do, post_do, hold, idx, idx2, spc;
    int a[30];
    int a2[30];
    pre_do = 0, post_do = 1, idx = 0, idx2 = 0, i = 0;
    printf("Please Enter any Regular Expression :-\n");
    gets(str);
    printf("\n\nHere is the step-by-step explanation of the process
:-");

    // printing the steps :-
    while (i != strlen(str))
    {
        if (str[i] == '(')
        {
```

```
            hold = pre_do;
            while (str[i + 1] != ')')
            {
                i++;
                if (str[i + 1] != '*' && str[i] != '+')
                {
                    insert(pre_do, post_do, str[i], 1);
                    printf("q%d -----> q%d by character %c\n", pre_do,
post_do, str[i]);
                    pre_do++;
                    post_do++;
                    if (pre_do != post_do - 1)
                        pre_do = post_do - 1;
                }
                else if (str[i + 1] == '*')
                {
                    insert(pre_do, post_do, str[i], 2);
                    printf("q%d -----> q%d and a self-loop at q%d by
character %c\n", pre_do, post_do, post_do, str[i]);
                    i++;
                    pre_do++;
                    post_do++;
                    if (pre_do != post_do - 1)
                        pre_do = post_do - 1;
                }
                else if (str[i] == '+')
                {
                    a[idx] = pre_do;
                    pre_do = hold;
                    idx++;
                }
            }
            if (str[i + 2] == '+')
            {
                hold = 0;
                a2[idx2] = pre_do;
                if (idx2 == 0)
```

```
                spc = pre_do;
            idx2++;
            pre_do = hold;
            int k = i;
            while (str[k] != '(')
                k++;
            i = k;
        }
        else if (str[i + 2] != '+')
        {
            insert(pre_do, post_do, 48, 0);
            printf("q%d -----> q%d by null\n", pre_do, post_do);

            for (int j = 0; j < idx; j++)
            {
                insert(a[j], post_do, 48, 0);
                printf("q%d -----> q%d by null\n", a[j], post_do);
            }
            a[20] = {0};
            idx = 0;
            pre_do = post_do;
            post_do++;
            int k = i;
            while (str[k] != '(')
            {
                if (str[k] == ')')
                    break;
                else
                    k++;
            }
            i = k + 1;
            if (i == strlen(str) - 1)
                break;
        }
    }
    else
        break;
```

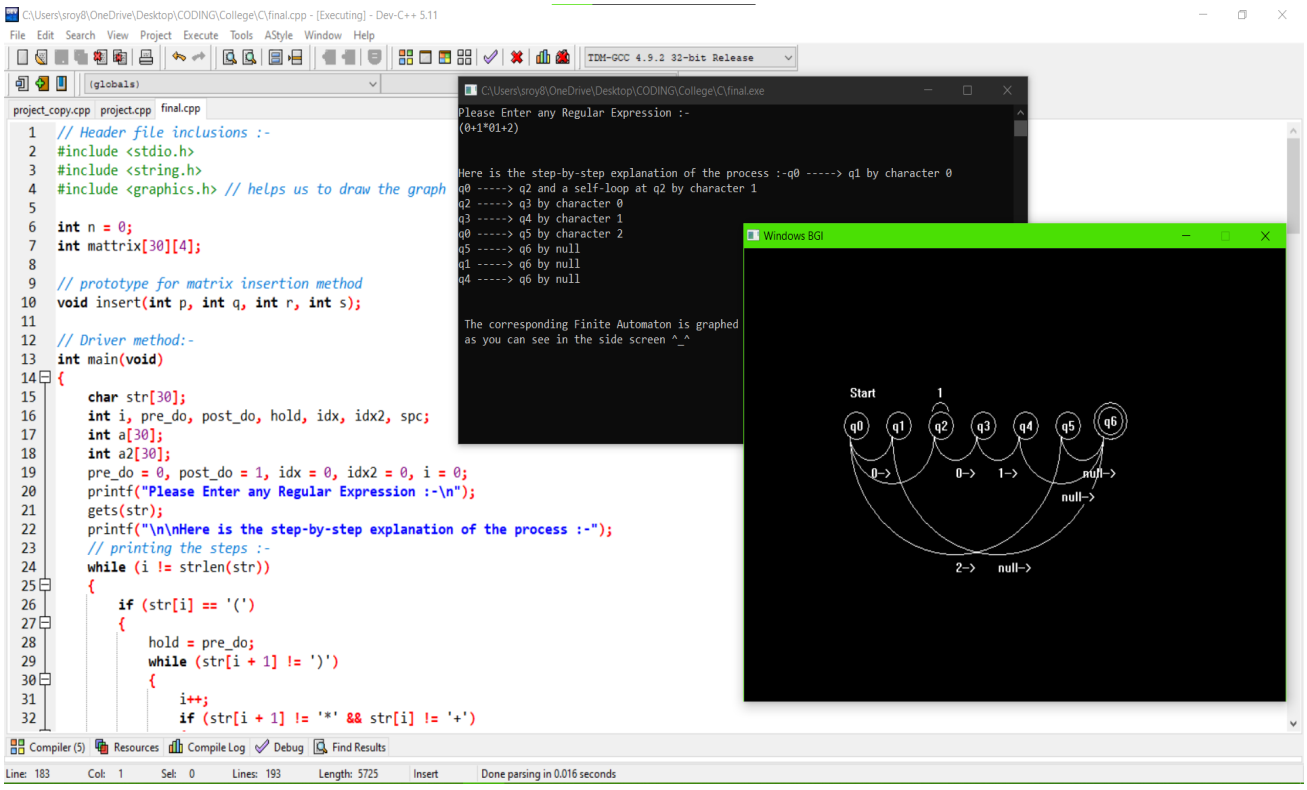
```
}
a2[0] = spc;
for (int m = 0; m < idx2; m++)
{
    insert(a2[m], post_do - 1, 0, 0);
    printf("q%d -----> q%d by null\n", a2[m], post_do - 1);
}
printf("\n\n The corresponding Finite Automaton is graphed");
printf("\n as you can see in the side screen ^_^");
int gd = DETECT, gm, hor_shift = 125;
initgraph(&gd, &gm, "");
outtextxy(0 + hor_shift, 145, "Start");
// printing the graph :-
for (int p = 0; p <= mattrix[n - 1][1]; p = p + 50)
{
    char sno[10] = "q";
    char sn[5];
    itoa(p / 50, sn, 10);
    strcat(sno, sn);
    if (p != mattrix[n - 1][1])
    {
        if (p / 50 > 9)
        {
            arc(p + 8 + hor_shift, 183, 0, 360, 20);
            outtextxy(p - 3 + hor_shift, 175, sno);
        }
        else
        {
            arc(p + 8 + hor_shift, 188, 0, 360, 15);
            outtextxy(p + hor_shift, 180, sno);
        }
    }
    else
    {
        if (p / 50 > 9)
        {
            arc(p + 8 + hor_shift, 183, 0, 360, 20);
```

```
            arc(p + 8 + hor_shift, 183, 0, 360, 25);
            outtextxy(p - 3 + hor_shift, 175, sno);
        }
        else
        {
            arc(p + 8 + hor_shift, 183, 0, 360, 15);
            arc(p + 8 + hor_shift, 183, 0, 360, 20);
            outtextxy(p + hor_shift, 175, sno);
        }
    }
}
for (int p = 0; p < n; p++)
{
    int mid = (mattrix[p][1] + mattrix[p][0]) / 2;
    int rad = mid - mattrix[p][0];
    arc(mid + hor_shift, 200, 180, 360, rad);
    char sno1[10] = "-->";
    char sn1[10];
    if (mattrix[p][3] == 1)
    {
        itoa(mattrix[p][2], sn1, 10);
        strcat(sn1, sno1);
        outtextxy(mid + hor_shift, rad + 205, sn1);
    }
    else if (mattrix[p][3] == 0)
        outtextxy(mid + hor_shift, rad + 205, "null-->");
    else
    {
        arc(mattrix[p][1] + 7 + hor_shift, 173, 0, 180, 10);
        char sn2[10];
        itoa(mattrix[p][2], sn2, 10);
        outtextxy(mattrix[p][1] + 3 + hor_shift, 145, sn2);
    }
}
getch();
closegraph(); // graphing completes
return 0;
```

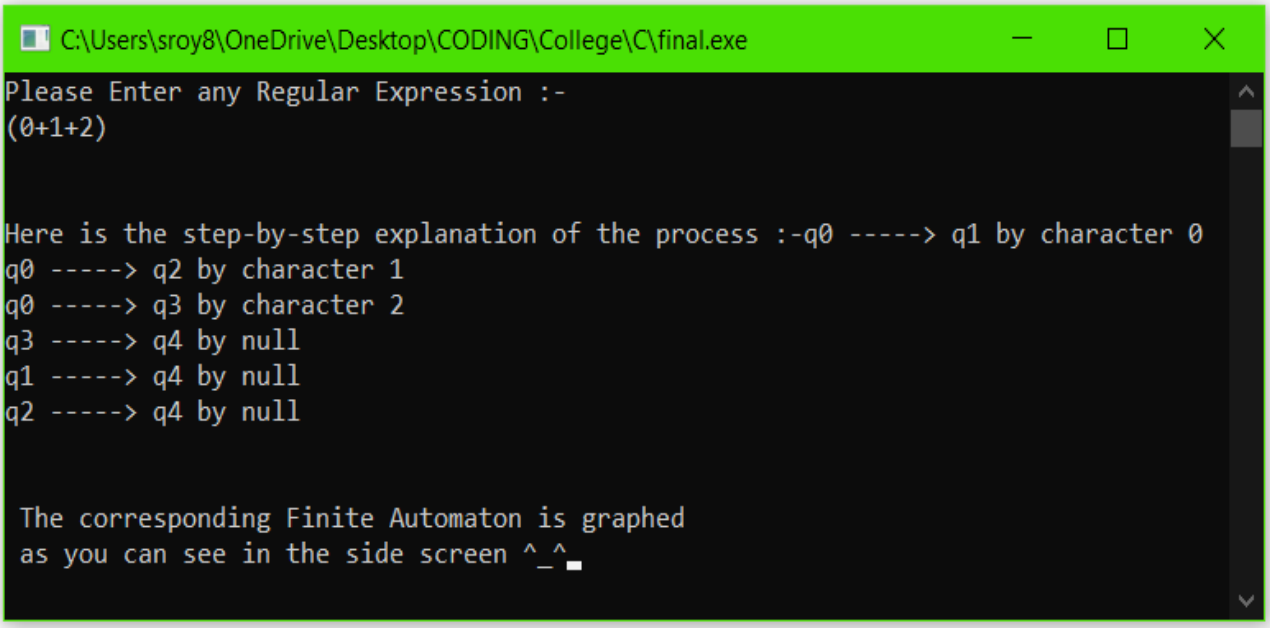
```
}

// Defining matrix insertion:-
void insert(int p, int q, int r, int s)
{
    mattrix[n][0] = p * 50;
    mattrix[n][1] = q * 50;
    mattrix[n][2] = r - 48;
    mattrix[n][3] = s;
    n++;
    return;
}
```

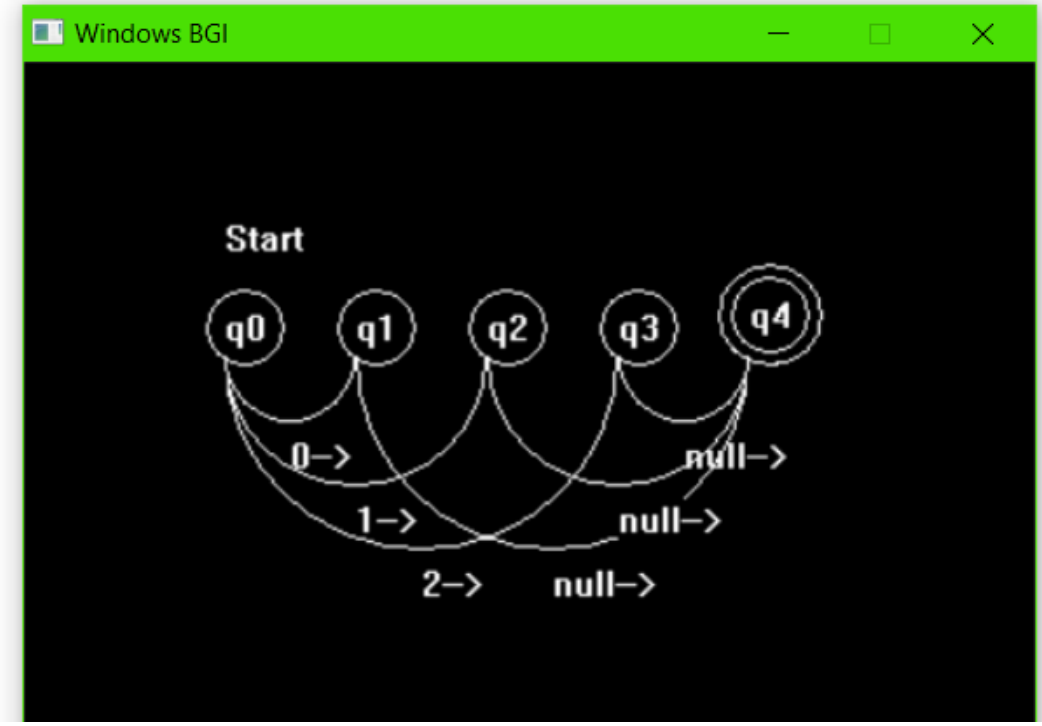
Output ->



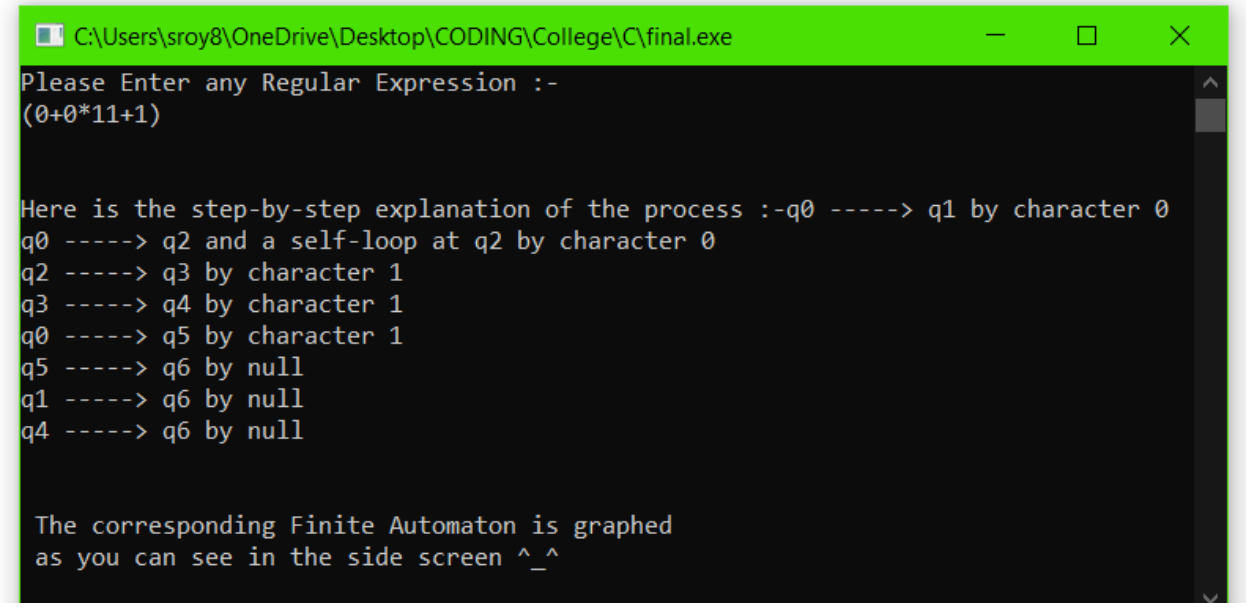
A snapshot of the workspace where the program is run



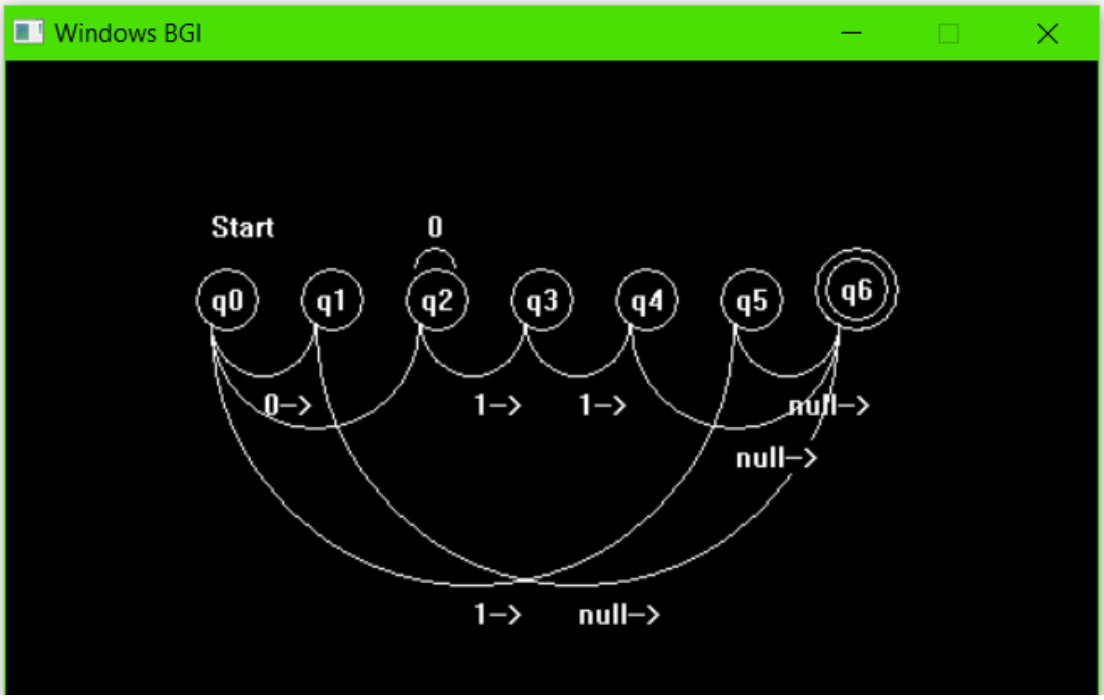
Step-by-step explanation of the process (example 1)



Graph of corresponding Finite Automaton (example 1)



Step-by-step explanation of the process (example 2)



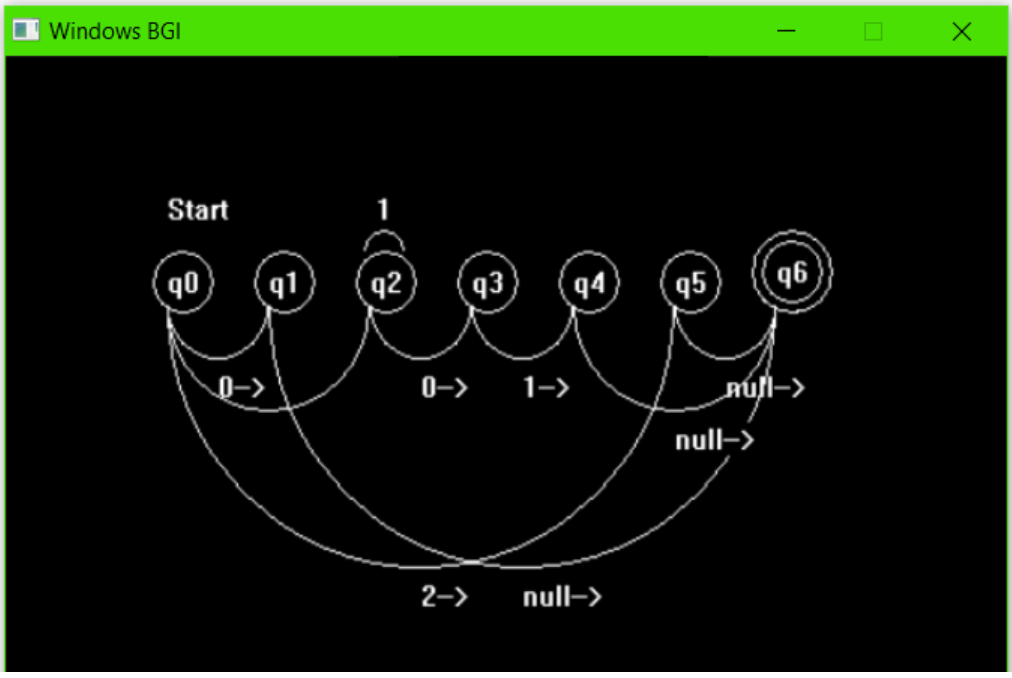
Graph of corresponding Finite Automaton (example 2)

```
C:\Users\sroy8\OneDrive\Desktop\CODING\College\C\final.exe
Please Enter any Regular Expression :-
(0+1*01+2)


Here is the step-by-step explanation of the process :-q0 -----> q1 by character 0
q0 -----> q2 and a self-loop at q2 by character 1
q2 -----> q3 by character 0
q3 -----> q4 by character 1
q0 -----> q5 by character 2
q5 -----> q6 by null
q1 -----> q6 by null
q4 -----> q6 by null


 The corresponding Finite Automaton is graphed
 as you can see in the side screen ^_^
```

Step-by-step explanation of the process (example 3)



Graph of corresponding Finite Automaton (example 3)