

C-programm for dfg

Code:

```
#include<stdio.h>

#include<string.h>

#define max 20

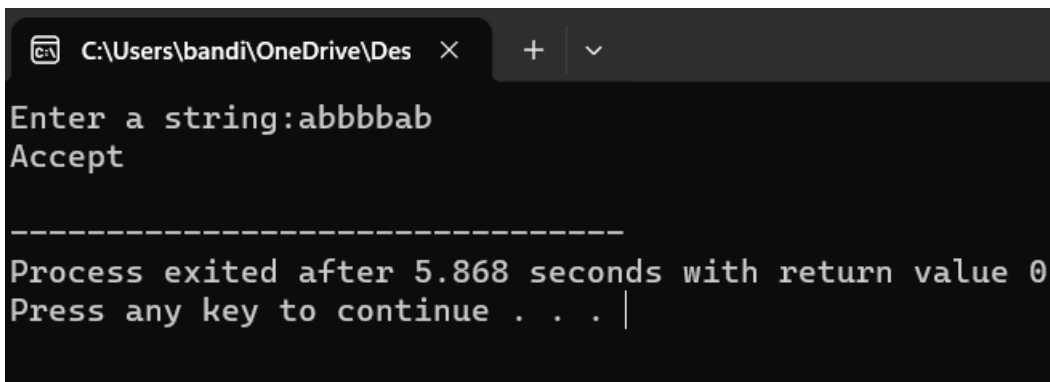
int main()
{
int trans_table[4][2]={{1,3},{1,2},{1,2},{3,3}};
int final_state=2,i;
int present_state=0;
int next_state=0;
int invalid=0;
char input_string[max];
printf("Enter a string:");
scanf("%s",input_string);
int l=strlen(input_string);
for(i=0;i<l;i++)
{
if(input_string[i]=='a')
next_state=trans_table[present_state][0];
else if(input_string[i]=='b')
next_state=trans_table[present_state][1];
else
invalid=i;
present_state=next_state;
}
```

```

if(invalid==l)
{
printf("Invalid input");
}
else if(present_state==final_state)
printf("Accept\n");
else
printf("Don't Accept\n");
}

```

Output:



```

C:\Users\bandi\OneDrive\Des
Enter a string:abbbbab
Accept

-----
Process exited after 5.868 seconds with return value 0
Press any key to continue . . . |

```

c-programm for nfa

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

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

```
#include<stdlib.h> // Include stdlib.h for exit()
```

```
int main() {
```

```

    int i, j, k, l, m, next_state[20], n, mat[10][10][10],
    flag, p;

```

```
int num_states, final_state[5], num_symbols,  
num_final;
```

```
int present_state[20], prev_trans, new_trans;
```

```
char input[20];
```

```
int symbol[5], inp, inp1;
```

```
printf("How many states in the NFA : ");
```

```
scanf("%d", &num_states);
```

```
printf("How many symbols in the input alphabet : ");
```

```
scanf("%d", &num_symbols);
```

```
for (i = 0; i < num_symbols; i++) {
```

```
    printf("Enter the input symbol %d : ", i + 1);
```

```
    scanf("%d", &symbol[i]);
```

```
}
```

```
printf("How many final states : ");
```

```
scanf("%d", &num_final);
```

```
for (i = 0; i < num_final; i++) {  
    printf("Enter the final state %d : ", i + 1);  
    scanf("%d", &final_state[i]);  
}
```

```
for (i = 0; i < 10; i++) {  
    for (j = 0; j < 10; j++) {  
        for (k = 0; k < 10; k++) {  
            mat[i][j][k] = -1;  
        }  
    }  
}
```

// Get input from the user and fill the 3D transition table

```
for (i = 0; i < num_states; i++) {  
    for (j = 0; j < num_symbols; j++) {  
        printf("How many transitions from state %d for  
the input %d: ", i, symbol[j]);  
        scanf("%d", &n);  
        for (k = 0; k < n; k++) {
```

```

        printf("Enter the transition %d from state %d
for the input %d : ", k + 1, i, symbol[j]);

        scanf("%d", &mat[i][j][k]);

    }

}

}

```

```

printf("The transitions are stored as shown
below\n");

```

```

for (i = 0; i < 10; i++) {
    for (j = 0; j < 10; j++) {
        for (k = 0; k < 10; k++) {
            if (mat[i][j][k] != -1)
                printf("mat[%d][%d][%d] = %d\n", i, j, k,
mat[i][j][k]);
        }
    }
}

```

```

while (1) {
    printf("Enter the input string : ");

```

```
scanf("%s", input);
present_state[0] = 0;
prev_trans = 1;
l = strlen(input);

for (i = 0; i < l; i++) {
    if (input[i] == '0')
        inp1 = 0;
    else if (input[i] == '1')
        inp1 = 1;
    else {
        printf("Invalid input\n");
        exit(0);
    }
    for (m = 0; m < num_symbols; m++) {
        if (inp1 == symbol[m]) {
            inp = m;
            break;
        }
    }
}
```

```

new_trans = 0;
for (j = 0; j < prev_trans; j++) {
    k = 0;
    p = present_state[j];
    while (mat[p][inp][k] != -1) {
        next_state[new_trans++] = mat[p][inp][k];
        k++;
    }
}
for (j = 0; j < new_trans; j++) {
    present_state[j] = next_state[j];
}
prev_trans = new_trans;
}

flag = 0;
for (i = 0; i < prev_trans; i++) {
    for (j = 0; j < num_final; j++) {
        if (present_state[i] == final_state[j]) {
            flag = 1;
            break;

```

```
        }  
    }  
}  
if (flag == 1)  
    printf("Accepted\n");  
else  
    printf("Not accepted\n");  
    printf("Try with another input\n");  
}  
return 0;  
}
```

Output:


```

How many states in the NFA : 4
How many symbols in the input alphabet : 2
Enter the input symbol 1 : 0
Enter the input symbol 2 : 1
How many final states : 1
Enter the final state 1 : 2
How many transitions from state 0 for the input 0: 1
Enter the transition 1 from state 0 for the input 0 : 1
How many transitions from state 0 for the input 1: 1
Enter the transition 1 from state 0 for the input 1 : 3
How many transitions from state 1 for the input 0: 2
Enter the transition 1 from state 1 for the input 0 : 1
Enter the transition 2 from state 1 for the input 0 : 2
How many transitions from state 1 for the input 1: 1
Enter the transition 1 from state 1 for the input 1 : 1
How many transitions from state 2 for the input 0: 0
How many transitions from state 2 for the input 1: 0
How many transitions from state 3 for the input 0: 1
Enter the transition 1 from state 3 for the input 0 : 3
How many transitions from state 3 for the input 1: 2
Enter the transition 1 from state 3 for the input 1 : 2
Enter the transition 2 from state 3 for the input 1 : 3
The transitions are stored as shown below
mat[0][0][0] = 1
mat[0][1][0] = 3
mat[1][0][0] = 1
mat[1][0][1] = 2
mat[1][1][0] = 1
mat[3][0][0] = 3
mat[3][1][0] = 2
mat[3][1][1] = 3
Enter the input string : 0111101
Not accepted
Try with another input
Enter the input string : 011110
Accepted

```

c-prograame for e-closure

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

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

```
int trans_table[10][5][3];
```

```
char symbol[5],a;
```

```
int e_closure[10][10],ptr,state;
```

```
void find_e_closure(int x);
```

```
int main()
```

```
{
int i,j,k,n,num_states,num_symbols;
for(i=0;i<10;i++)
{
for(j=0;j<5;j++)
{
for(k=0;k<3;k++)
{
trans_table[i][j][k]=-1;
}
}
}

printf("How may states in the NFA with e-moves:");
scanf("%d",&num_states);

printf("How many symbols in the input alphabet
including e :");
scanf("%d",&num_symbols);

printf("Enter the symbols without space. Give 'e'
first:");
scanf("%s",symbol);
for(i=0;i<num_states;i++)
```

```
{
for(j=0;j<num_symbols;j++)
{
printf("How many transitions from state %d for the
input %c:",i,symbol[j]);
scanf("%d",&n);
for(k=0;k<n;k++)
{
printf("Enter the transitions %d from state %d for the
input %c :", k+1,i,symbol[j]);
scanf("%d",&trans_table[i][j][k]);
}
}
}
for(i=0;i<10;i++)
{
for(j=0;j<10;j++)
{
e_closure[i][j]=-1;
}
}
}
```

```
for(i=0;i<num_states;i++)
e_closure[i][0]=i;
for(i=0;i<num_states;i++)
{
if(trans_table[i][0][0]==-1)
continue;
else
{
state=i;
ptr=1;
find_e_closure(i);
}
}
for(i=0;i<num_states;i++)
{
printf("e-closure(%d)= {",i);
for(j=0;j<num_states;j++)
{
if(e_closure[i][j]!=-1)
{
```

```
printf("%d, ",e_closure[i][j]);  
}  
}  
printf("}\n");  
}  
}  
void find_e_closure(int x)  
{  
    int i,j,y[10],num_trans;  
    i=0;  
    while(trans_table[x][0][i]!=-1)  
    {  
        y[i]=trans_table[x][0][i];  
        i=i+1;  
    }  
    num_trans=i;  
    for(j=0;j<num_trans;j++)  
    {  
        e_closure[state][ptr]=y[j];  
        ptr++;  
    }  
}
```

```

find_e_closure(y[j]);
}
}

```

Output:

```

C:\Users\bandi\OneDrive\Des  ×  +  v
How may states in the NFA with e-moves:3
How many symbols in the input alphabet including e :3
Enter the symbols without space. Give 'e' first:e01
How many transitions from state 0 for the input e:1
Enter the transitions 1 from state 0 for the input e :1
How many transitions from state 0 for the input 0:0
How many transitions from state 0 for the input 1:1
Enter the transitions 1 from state 0 for the input 1 :1
How many transitions from state 1 for the input e:1
Enter the transitions 1 from state 1 for the input e :2
How many transitions from state 1 for the input 0:2
Enter the transitions 1 from state 1 for the input 0 :0
Enter the transitions 2 from state 1 for the input 0 :1
How many transitions from state 1 for the input 1:0
How many transitions from state 2 for the input e:0
How many transitions from state 2 for the input 0:0
How many transitions from state 2 for the input 1:0
e-closure(0)= {0, 1, 2, }
e-closure(1)= {1, 2, }
e-closure(2)= {2, }

-----
Process exited after 26.1 seconds with return value 0
Press any key to continue . . . |

```

c-programme for cfg:

```

#include<stdio.h>

#include<string.h>

int main(){

```

```
char s[100];
int i,flag;
int l;
printf("enter a string to check:");
scanf("%s",s);
l=strlen(s);
flag=1;
for(i=0;i<l;i++)
{
if(s[i]!='0' && s[i]!='1')
{
flag=0;
}
}
if(flag!=1)
printf("string is Not Valid\n");
if(flag==1)
{
if (s[0]=='0'&& s[l-1]=='1')
printf("string is accepted\n");
```

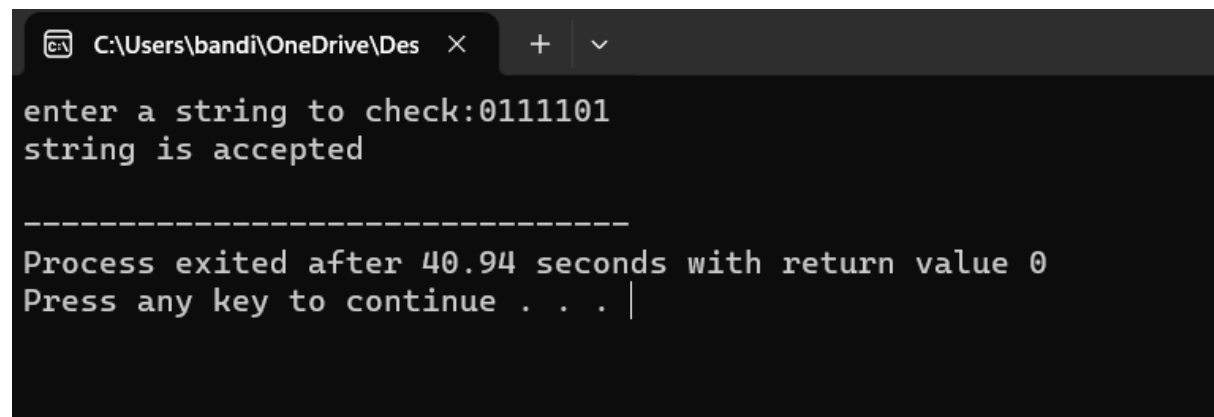
else

```
printf("string is Not accepted\n");
```

```
}
```

```
}
```

Output:



The screenshot shows a Windows command prompt window with a dark background. The title bar at the top indicates the file path 'C:\Users\bandi\OneDrive\Des' and includes standard window controls. The command prompt displays the following text: 'enter a string to check:0111101', 'string is accepted', a separator line of dashes, 'Process exited after 40.94 seconds with return value 0', and 'Press any key to continue . . . |'.

```
C:\Users\bandi\OneDrive\Des > enter a string to check:0111101
string is accepted

-----
Process exited after 40.94 seconds with return value 0
Press any key to continue . . . |
```