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=l;

present\_state=next\_state;

}

if(invalid==l)

{

printf("Invalid input");

}

else if(present\_state==final\_state)

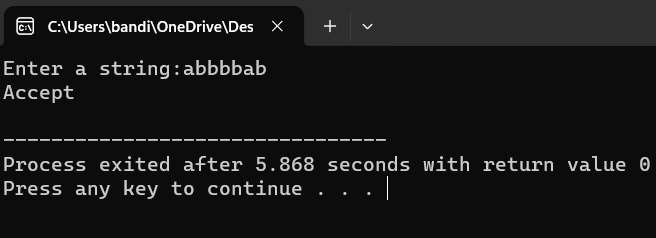
printf("Accept\n");

else

printf("Don't Accept\n");

}

Output:



c-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");

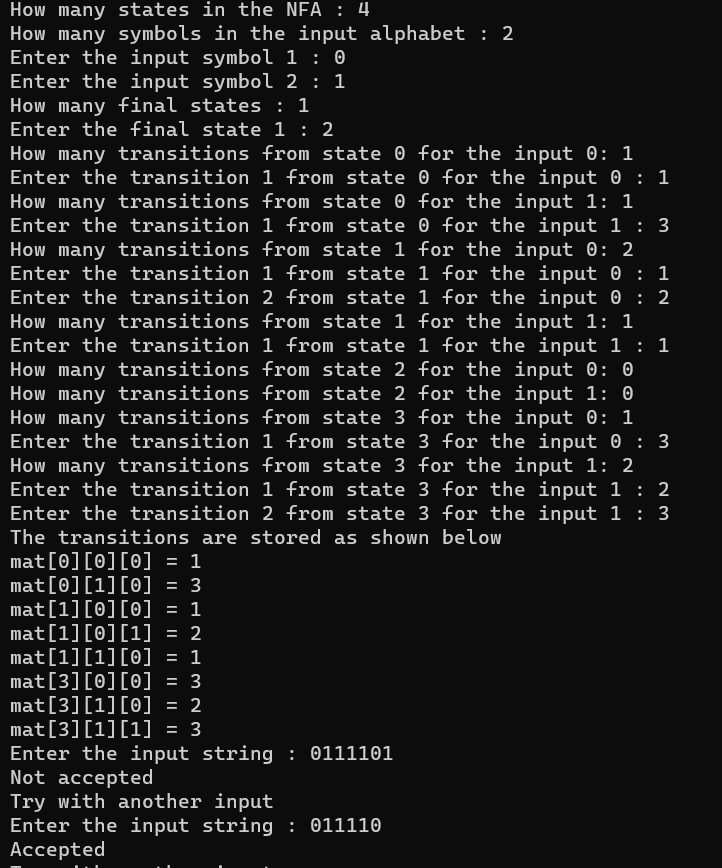
printf("Try with another input\n");

}

return 0;

}

Output:



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];

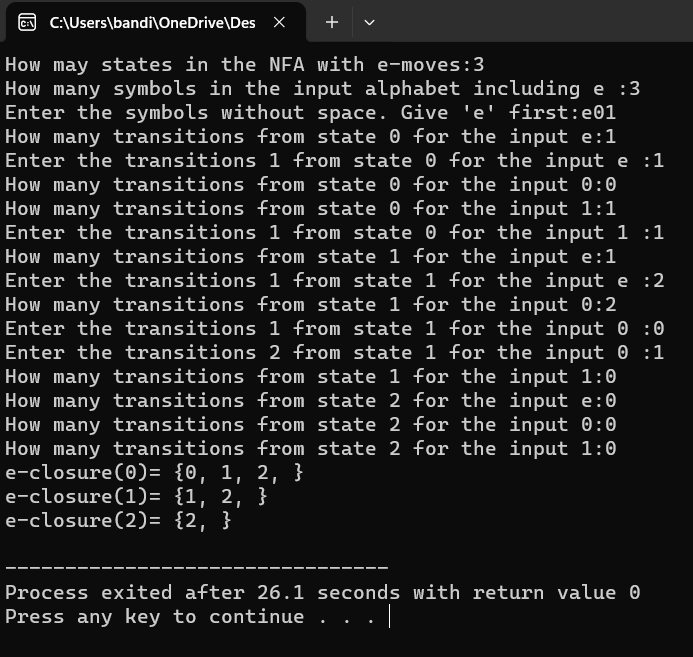
ptr++;

find\_e\_closure(y[j]);

}

}

Output:



c-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:

