**EXP 1:**

**Program:**

#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)

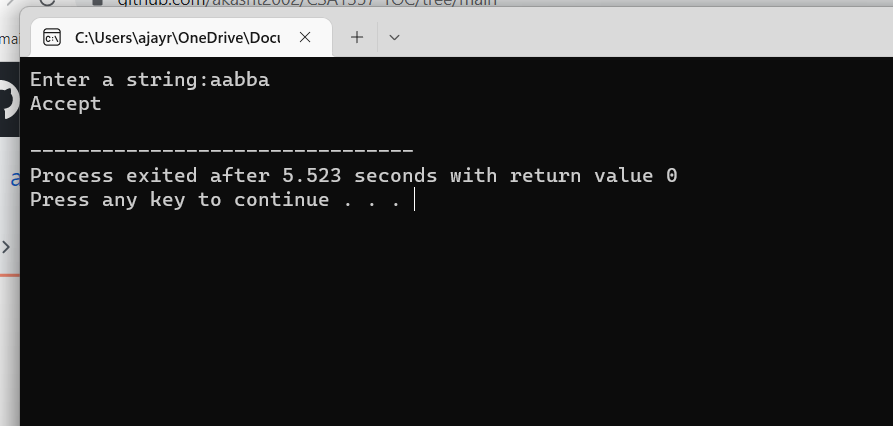
printf("dontAccept\n");

else

printf("Accept\n");

}

**Output:**

****

**Exp2:**

**Program:**

#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]=='0')

next\_state=trans\_table[present\_state][0];

else if(input\_string[i]=='1')

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)

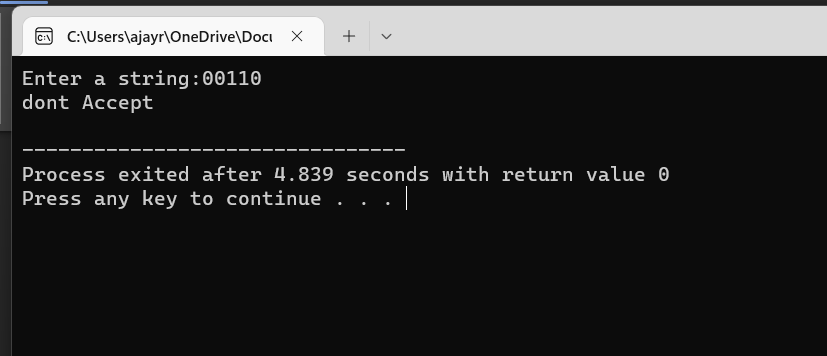
printf("Accept\n");

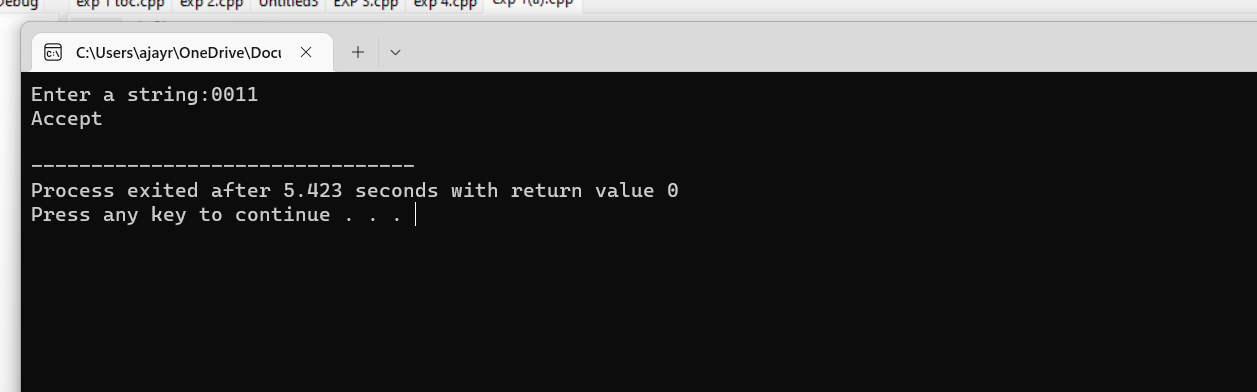
else

printf("dont Accept\n");

}

**Output:**

****

****

**Exp 3;**

**Program:**

#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");

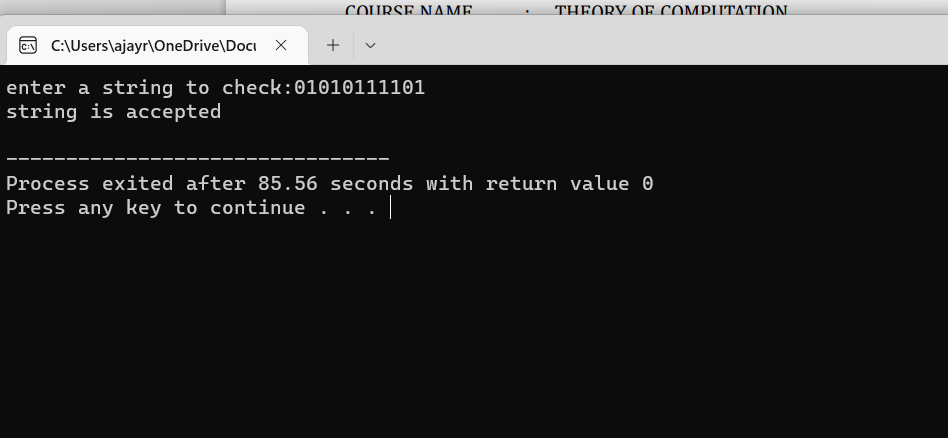
else

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

}

}

**Output:**



**Exp 4:**

**Program:**

#include<stdio.h>

#include<string.h>

int main()

{

char s[100];

int i,flag,flag1,a,b;

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)

{

flag1=1;

a=0;b=l-1;

while(a!=(l/2))

{

if(s[a]!=s[b])

{

flag1=0;

}

a=a+1;

b=b-1;

}

if (flag1==1)

{

printf("The string is a palindrome\n");

printf("string is accepted\n");

}

else

{

printf("The string is not a palindrome\n");

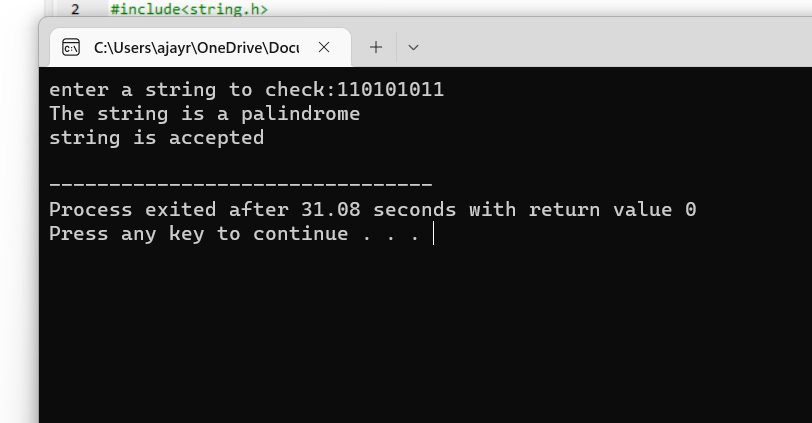
printf("string is Not accepted\n");

}

}

}

Output:

****

**EXP 5**

**Program:**

#include<stdio.h>

#include<string.h>

int main()

{

char s[100];

int i,flag,flag1,a,b;

int l,count1,count2;

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)

{

i=0;count1=0;

while(s[i]=='0') // Count the no of 0s in the front

{

count1++;

i++;

}

while(s[i]=='1')

{

i++; // Skip all 1s

}

flag1=1;

count2=0;

while(i<l)

{

if(s[i]=='0')// Count the no of 0s at the end

{

count2++;

}

else

{

flag1=0;

}

i++;

}

if(flag1==1)

{

if(count1==count2)

{

printf("The string satisfies the condition 0n1m0n\n");

printf("String Accepted\n");

}

else

{

printf("The string does not satisfy the condition 0n1m0n\n");

printf("String Not Accepted\n");

}

}

else

{

printf("The string does not satisfy the condition 0n1m0n\n");

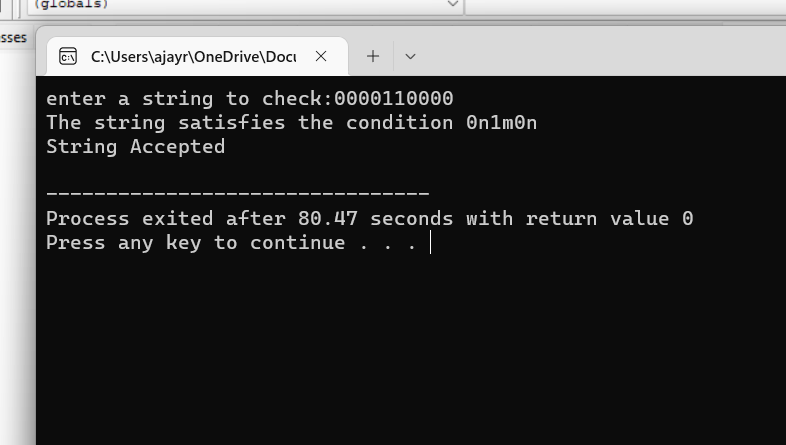
printf("String Not Accepted\n");

}

}

}

**Output:**

****

**Exp 6;**

**Program;**

#include<stdio.h>

#include<string.h>

int main()

{

char s[100];

int i,flag,flag1,flag2;

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(l%2!=0) // If string length is odd

{

printf("The string does not satisfy the condition 0n1n\n");

printf("String Not Accepted\n");

}

else

{

// To check first half contains 0s

flag1=1;

for(i=0;i<(l/2);i++)

{

if(s[i]!='0')

{

flag1=0;

}

}

// To check second half contains 1s

flag2=1;

for(i=l/2;i<l;i++)

{

if(s[i]!='1')

{

flag2=0;

}

}

if(flag1==1 && flag2==1)

{

printf("The string satisfies the condition 0n1n\n");

printf("String Accepted\n");

}

else

{

printf("The string does not satisfy the condition 0n1n\n");

printf("String Not Accepted\n");

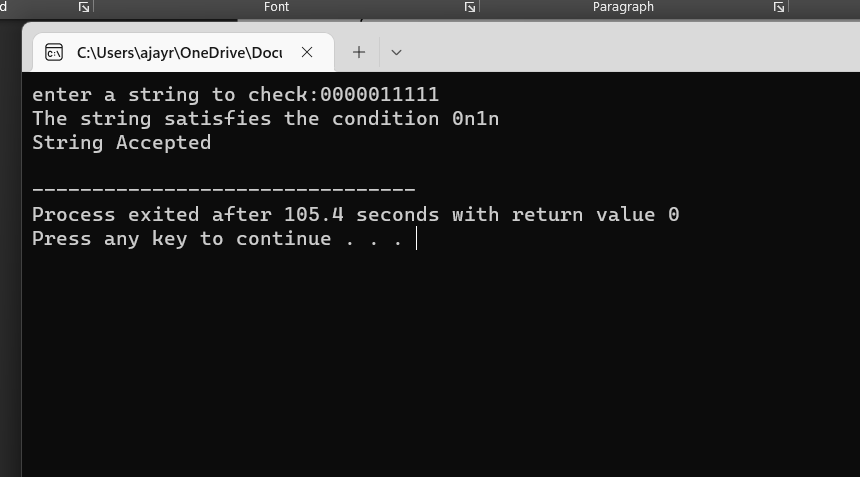
}

}

}

}

**Output:**

****

**Exp 7:**

**Program;**

#include<stdio.h>

#include<string.h>

int main()

{

char s[100];

int i,flag,flag1;

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 Valid\n");

else

printf("string is Not Valid\n");

if(flag==1)

{

flag1=0;

for(i=0;i<l-2;i++)

{

if(s[i]=='1')

{

if(s[i+1]=='0' && s[i+2]=='1')

{

flag1=1;

printf("Substring 101 exists. String accepted\n");

break;

}

}

}

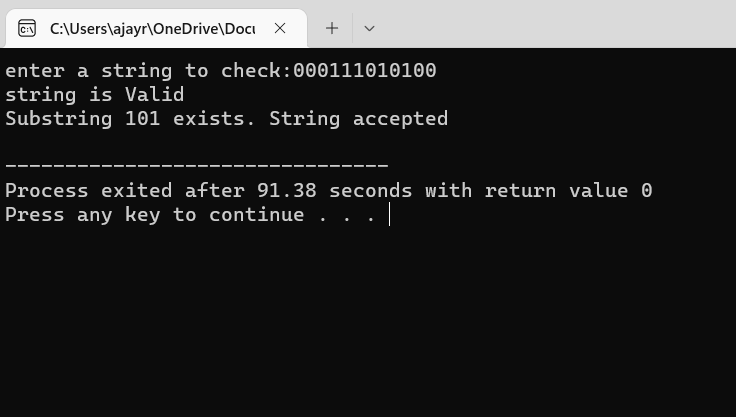
if(flag1==0)

printf("Substring 101 does not exist. String not accepted\n");

}

}

**Output:**

****

**Exp 8:**

**Program:**

#include <stdio.h>

#include <string.h>

#define MAX\_STRING\_LENGTH 100

int current\_state = 0;

int simulate\_NFA(char\* input\_string) {

int length = strlen(input\_string);

for (int i = 0; i < length; i++) {

if (current\_state == 0 && input\_string[i] == 'b') {

current\_state = 1;

} else if (current\_state == 1 && input\_string[i] == 'a') {

current\_state = 2;

return 1;

} else {

current\_state = 0;

return 0;

}

}

return 0;

}

int main() {

char input\_string[MAX\_STRING\_LENGTH];

printf("Enter a string to check: ");

scanf("%s", input\_string);

if (simulate\_NFA(input\_string)) {

printf("Accepted\n");

} else {

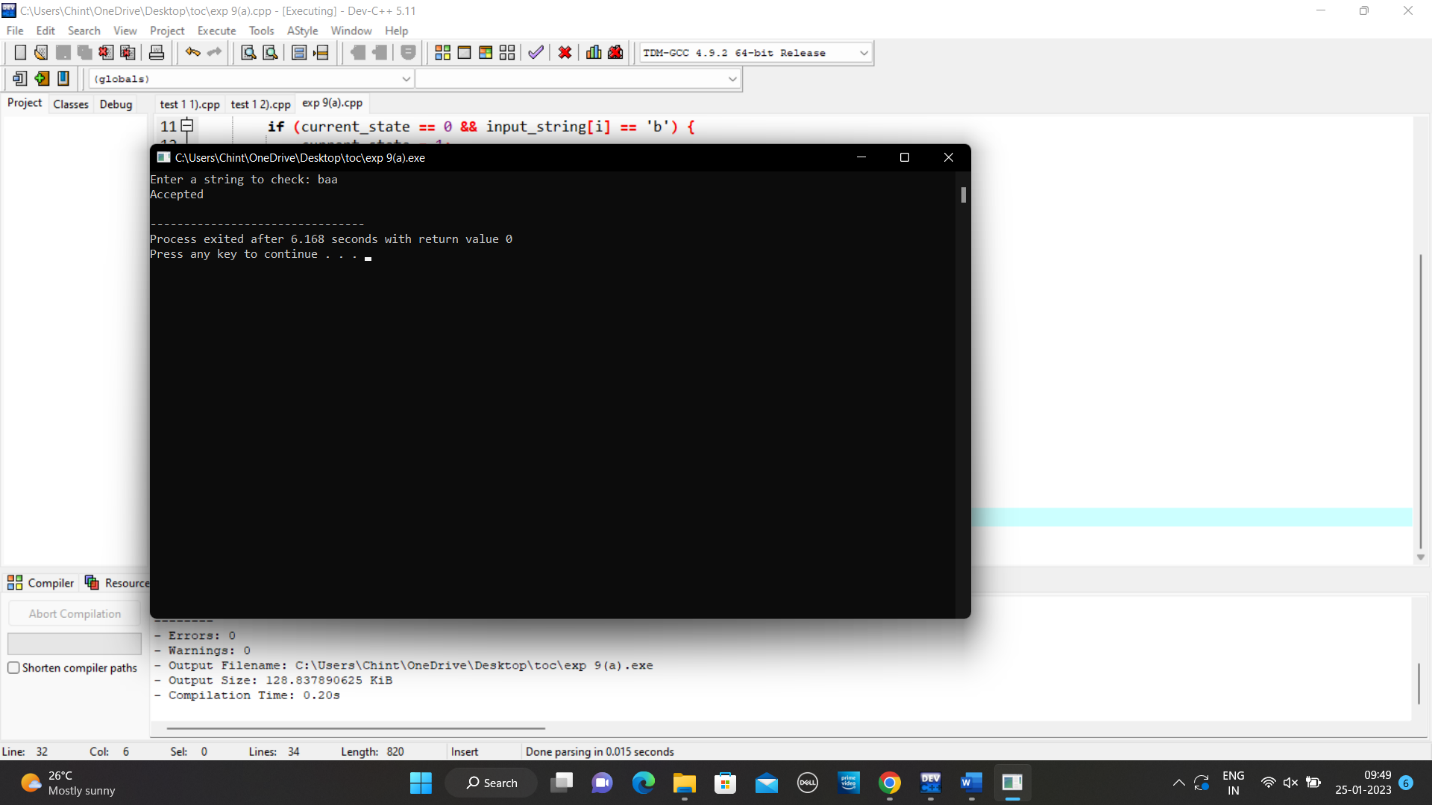
printf("Rejected\n");

}

return 0;

}

**Output:**



**Exp 9:**

**Program:**

#include <stdio.h>

#include <string.h>

#define MAX\_STRING\_LENGTH 100

int current\_state = 0;

int simulate\_NFA(char\* input\_string) {

int length = strlen(input\_string);

for (int i = 0; i < length; i++) {

if (current\_state == 0 && input\_string[i] == '0') {

current\_state = 1;

} else if (current\_state == 1 && input\_string[i] == '1') {

current\_state = 2;

return 1;

} else {

current\_state = 0;

return 0;

}

}

return 0;

}

int main() {

char input\_string[MAX\_STRING\_LENGTH];

printf("Enter a string to check: ");

scanf("%s", input\_string);

if (simulate\_NFA(input\_string)) {

printf("Accepted\n");

} else {

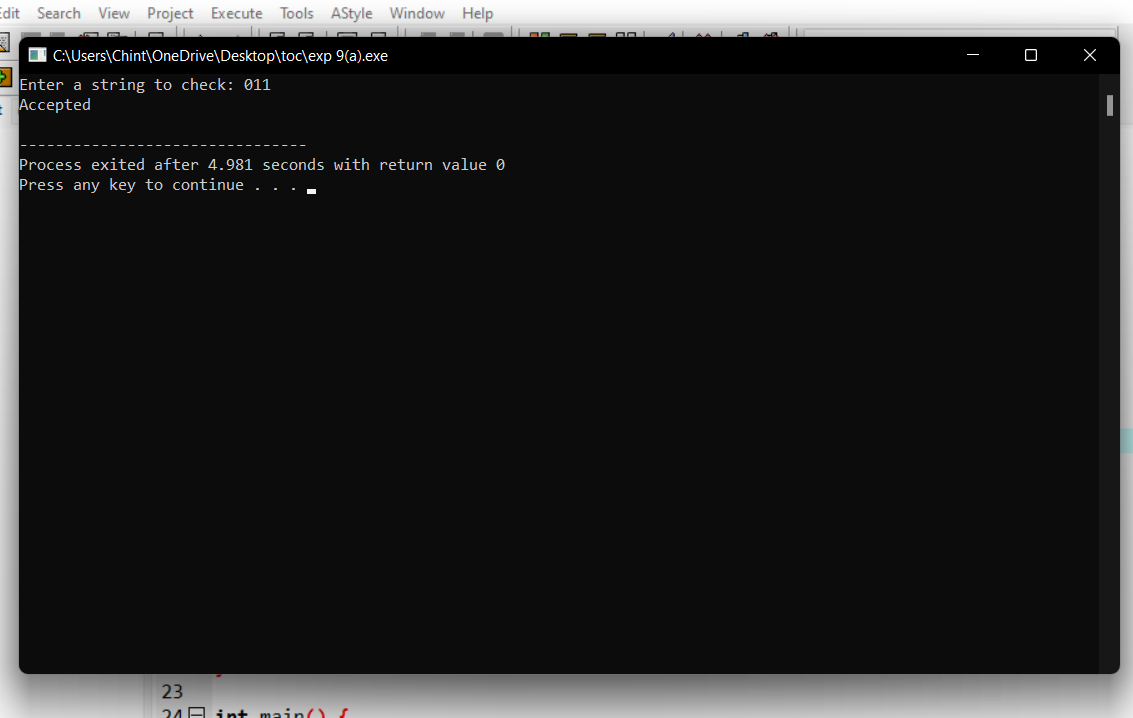
printf("Rejected\n");

**}**

return 0;

}

**Output:**



**Exp 10:**

**Program:**

#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:**

