**1.Write a C program to simulate a Deterministic Finite Automata (DFA) for the given language representing strings that start with a and end with a.**

#include<stdio.h>

#include<string.h>

int main()

{

char str[100];

printf("enter the string to be checked: ");

scanf("%s",&str);

int length=strlen(str);

if(str[0] == 'a' && str[length-1] == 'a')

{

printf("Accepted");

}

else

{

printf("Rejected");

return 0;

}

}

**2.Write a C program to simulate a Deterministic Finite Automata (DFA) for the given language representing strings that start with 0 and end with 1**

#include<stdio.h>

#include<string.h>

int main()

{

char str[100];

printf("enter the string to be checked: ");

scanf("%s",str);

int lenght = strlen(str);

if(str[0] == '0' && str[lenght-1] == '1')

{

printf("Accepted");

}

else

{

printf("Rejected");

return 0;

}

}

**3. Write a C program to check whether a given string belongs to the language defined by a Context Free Grammar (CFG)**

**S → 0A1 A → 0A | 1A | ε**

#include<stdio.h>

#include<string.h>

int main()

{

char str[100];

printf("enter the string to be checked: ");

scanf("%s",str);

int length=strlen(str);

if (str[0]=='0'&&str[length-1]=='1')

{

printf("accepted");

}

else

{

printf("rejected");

}

return 0;

}

**4. Write a C program to check whether a given string belongs to the language defined by a Context Free Grammar (CFG)**

**S → 0S0 | 1S1 | 0 | 1 | ε**

#include<stdio.h>

#include<string.h>

int main()

{

char str[100];

printf("enter the string to be checked: ");

scanf("%s",str);

int length= strlen(str);

if (str[0]==str[length-1]&&str[1]==str[length-2])

{

printf("Accepted");

}

else

{

printf("Rejected");

return 0;

}

}

**5. Write a C program to check whether a given string belongs to the language defined by a Context Free Grammar (CFG)**

**S → 0S0 | A A → 1A | ε**

#include<stdio.h>

#include<string.h>

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')

{

count1++;

i++;

}

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

{

i++;

}

flag1=1;

count2=0;

while(i<l)

{

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

{

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");

}

}

}

**6. Write a C program to check whether a given string belongs to the language defined by a Context Free Grammar (CFG)**

**S → 0S1 | ε**

#include<stdio.h>

#include<string.h>

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)

{

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

printf("String Not Accepted\n");

}

else

{

flag1=1;

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

{

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

{

flag1=0;

}

}

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");

}

}

}

}

**7. Write a C program to check whether a given string belongs to the language defined by a Context Free Grammar (CFG)**

**S → A101A, A → 0A | 1A | ε**

#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");

}

}

**8. Write a C program to simulate a Non-Deterministic Finite Automata (NFA) for the given languagerepresenting strings that start with b and end with a**

#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;

}

**9. Write a C program to simulate a Non-Deterministic Finite Automata (NFA) for the given languagerepresenting strings that start with o and end with 1**

#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;

}

**10. Write a C program to find ε -closure for all the states in a Non-Deterministic Finite Automata (NFA) with ε -moves**

#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]);

}

}