COMPILER LAB **CEN-792**

**SUBMITTED TO** MR.SARFARAZ MASOOD

**SUBMITTED BY:**

MD.SHARIQUE SHAHAB  
13-BCS-0035

**DFA (String Acceptance)**

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

FILE \*fp;

char s[50],ch;

int n,i=0,k=0,j=0,init,final[10],n\_final,tab[100][100],newstate,count=0,row=-1,col=0;

int column\_no(char c)

{

switch(c)

{

case 'a':

return 0;

case 'b':

return 1;

case 'c':

return 2;

case 'd':

return 3;

case 'e':

return 4;

case 'f':

return 5;

case 'g':

return 6;

}

}

void Read\_Automaton()

{

fp=fopen("dfa.txt","r");

if(!fp)

{

printf("Cannot open the file\n");

return ;

}

ch=fgetc(fp);

init=ch-'0';

while(!feof(fp))

{

ch=fgetc(fp);

if(ch=='\n')

{

count++;

}

else if(count == 1 && (ch>='0' && ch<='9') )

{

final[i]=ch-'0';

i++;

}

if(count >= 2)

{

if(ch>='0' && ch<='9')

{

tab[row][col]=ch-'0';

col++;

}

else if(ch=='\n')

{

col=0;

row++;

}

}

}

n\_final=i;

//////////////////////////////////////////////////////////////////

//Print

printf("Initia state :%d\n",init);

printf("Final State:");

for(j=0;j<i;j++)

printf("%d ",final[j]);

printf("\nAutomaton in Tabular Form:");

for(i=0;i<=row;i++)

{

printf("\n");

for(j=0;j<col;j++)

{

printf("%d\t",tab[i][j]);

}

}

}

int main()

{

Read\_Automaton();

while(1)

{

a20:

printf("\nEnter the string");

scanf("%s",s);

if(strlen(s)>49)

{

printf("string size should be less than 50\n");

return 0;

}

newstate=init;

j=0;

while(s[j]!='\0')

{

if(s[j]>='a' && s[j]<='z')

{

i=column\_no(s[j]);

if(i>col-1)

{

printf("Not accepted\n");

goto a20;

}

newstate=tab[newstate][i];

if(newstate==-1)

{

printf("Not accepted\n");

goto a20;

}

}

else

{

printf("Enter string in the form of letters\n");

goto a20;

}

j++;

}

for(i=0;i<n\_final;i++)

{

if(newstate == final[i])

{

printf("Accepted\n");

goto a20;

}

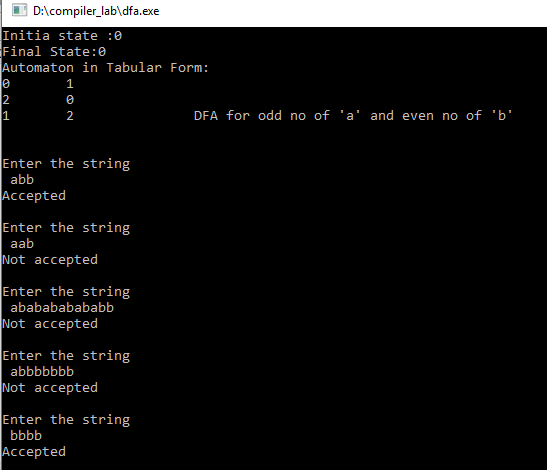
}

printf("Not accepted\n");

goto a20;

}

}



**2. NFA (String acceptance or not using nfa)**

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

int main()

{

char ch,str[100];

int count,initial,final[5],k,curstate,flag,col=0;

int i,j,l;

int nfa[10][10][10];

FILE \*f;

f=fopen("nfa.txt","r");

if(f == NULL)

{

printf("Empty File ");

}

for(i=0;i<10;i++)

for(j=0;j<10;j++)

for(k=0;k<10;k++)

nfa[i][j][k]=-2;

ch=getc(f);

count=0;

k=0;

while(ch !=EOF )

{ if(count==0 && ch!='\n')

{

initial=ch-48; printf("Initial State : %d\n",initial);

}

if(ch=='\n')

count++;

if(count==1)

{ ch=getc(f);

if(ch!=',')

{

final[k]=ch-48;

printf("Final State %d: q%d\n",k+1,final[k]);

k++; col++;}

}

if(count==2)

break;

ch=getc(f);

}

ch=getc(f);

i=0; j=0; k=0;

while(ch!=EOF)

{

if(ch==',')

{

k++; ch=getc(f); nfa[i][j][k]=ch-48;

}

if(ch==' ')

{ j++; k=0;

}

if(ch=='\n')

{

j=0; k=0; i++;

}

if(ch=='-')

{

ch=getc(f);

nfa[i][j][k]=ch-48;

nfa[i][j][k]= (-1)\*nfa[i][j][k];

}

if(nfa[i][j][k]!=-1)

{

nfa[i][j][k]=ch-48;

}

ch=getc(f);

}

fclose(f);

while(1)

{ curstate=initial; flag=0; printf("\nEnter String :");

gets(str);

for(i=0;i<strlen(str);i++)

{

if(str[i]-97 > 1)

{

printf("Invalid String\n");

break;

}

if(nfa[curstate][str[i]-97][0]== -1 || nfa[curstate][str[i]-97][1]==-1)

{

printf("REJECTED.\n");

flag=-1;

break;

}

if(nfa[curstate][str[i]-97][0] >0 && nfa[curstate][str[i]-97][1] > 0 )

{

curstate=nfa[curstate][str[i]-97][1] || nfa[curstate][str[i]-97][0] ;

}}

if(flag!=-1)

{

if(curstate==final[i])

{ printf("ACCEPTED\n");

flag=1;

// break; }

}

if(flag==0)

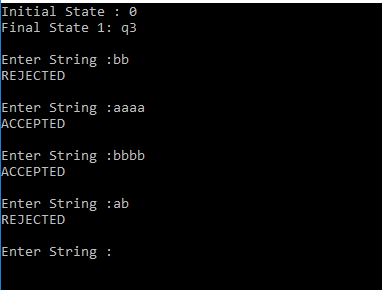
{

printf("REJECTED\n");

}

}

}



**3. NFA to DFA (Conversion)**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

char MY\_NFA[10][10][10], MY\_array[10][10],curstate,end[10];

int rm,cm,tm=0;

void \_ad(int j)

{

int l; int pt,k,t=0;

int u[10];int fu[10];

for(l=0;l<=rm-1;l++)

{

if(MY\_array[tm-1][l]>='0'&&MY\_array[tm-1][l]<='9')

{

pt=MY\_array[tm-1][l]-48;

for(k=0;k<=rm-1;k++)

{

if(MY\_NFA[pt][j][k]>='0'&&MY\_NFA[pt][j][k]<='9')

{

u[t]=MY\_NFA[pt][j][k];

t++;

}}}}

if(t==0)

MY\_NFA[rm][j][0]='$';

else if(t==1)

MY\_NFA[rm][j][0]=u[0];

else

{

for(l=0;l<t;l++)

for(pt=0;pt<t-1;pt++)

{

if(u[pt]>u[pt+1])

{

k=u[pt];

u[pt]=u[pt+1];

u[pt+1]=k;

} }

pt=0;

for(l=0;l<t;l++)

{

if(u[l+1]==u[l])

continue;

fu[pt]=u[l];

pt++;

}

for(l=0;l<pt;l++)

MY\_NFA[rm][j][l]=fu[l];

} }

int \_chak(int i, int j)

{

int l,k,flag=0;

for(l=0;l<tm;l++)

{ flag=0;

for(k=0;k<=rm;k++)

{

if(MY\_NFA[i][j][k]>='0'&&MY\_NFA[i][j][k]<='9')

if(MY\_array[l][k]==MY\_NFA[i][j][k])

flag=flag+1;

}

if(flag==countk(i,j))

break;

}

if(flag==0)

return 1;

return 0; }

void MY\_arrayentry(int i,int j)

{ int l=0,k=0,p=0; char t;

for(k=0;k<=rm;k++)

if(MY\_NFA[i][j][k]>='0'&&MY\_NFA[i][j][k]<='9')

{ MY\_array[tm][l]=MY\_NFA[i][j][k]; l++;

}

tm=tm+1; }

int countk(int i ,int j)

{ int k=0,count=0;

for(k=0;k<=rm;k++)

{

if(MY\_NFA[i][j][k]>='0'&&MY\_NFA[i][j][k]<='9')

count++; }

return count;}

int main()

{

FILE \*f;

int i=0,j=0,k=0; int l; int orgcount=0,col, flag=0; int fcol=0; char ch;

f=fopen("mynfa.txt","r");

if(f==NULL)

{

printf("FILE NOT PRESENT\n");

exit(0);

}

ch=fgetc(f);

while(ch!=EOF)

{

if (flag==0)

{ curstate=ch; flag=2;}

else if(flag==2)

{ ch=fgetc(f); end[i]=ch; i++; ch=fgetc(f);

while(ch!='\n')

{ if(ch!=',' && ch!=(char)13)

{

end[i]=ch; i++; }

ch=fgetc(f); }

flag=3; fcol=i-1; i=0;

}

else if(flag==3)

{

if(ch=='\n')

{

i++; col=j; j=0; k=0; }

else if(ch==',')

{ ch=getc(f); MY\_NFA[i][j][k]=ch; k++;

}

else if(ch==' ')

{

j++; k=0; }

else if(ch!=',' && ch!=' ')

{

if(ch=='$')

{

MY\_NFA[i][j][0]=ch; }

else

{

MY\_NFA[i][j][0]=ch; k++;

} } }

ch=fgetc(f); }

rm=i; orgcount=i; cm=col; flag=0;

for(i=0;i<=rm;i++)

{ for(j=0;j<=col;j++)

{ flag=0;

if(countk(i,j)>1)

{

if(tm==0)

{ MY\_arrayentry(i,j); rm=rm+1;

for(l=0;l<=col;l++)

\_ad(l); }

else if(\_chak(i,j))

{ MY\_arrayentry(i,j); rm=rm+1;

for(l=0;l<=col;l++)

\_ad(l); }

else

{ flag=1; break;

} } }

if(flag==1)

break;

}

for(i=0;i<=rm;i++)

{

if(i==orgcount+1)

printf("New States\n");

for(j=0;j<=col;j++)

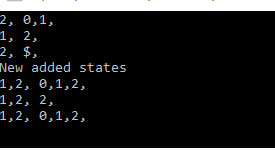
{ for(l=0;l<=rm;l++)

if(MY\_NFA[i][j][l]!='\0')

printf("%c,",MY\_NFA[i][j][l]);

printf(" "); } printf("\n"); } }

return 0;



4. **Mealy machine**

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

int main()

{

FILE \*f;

char ch,path[20];

int i,j,\_st[5][5],\_o[5][5],initial\_state,len,matrix[10][10],k,curstate;

f=fopen("mealy.txt","r");

if(f==NULL)

{

printf("Not a file ");

}

ch=getc(f); initial\_state=ch-48;

printf("Initial state : %d\n",initial\_state);

ch=getc(f);

//printf("%c",ch);

i=0; j=0;

while(ch!=EOF)

{

if(ch==' ') j++;

if(ch=='\n')

{ j=0; i++;

}

if(j%2==0)

{

if(ch=='-')

{ ch=getc(f); \_st[i][j]=ch-48; \_st[i][j]= (-1)\* \_st[i][j];

}

if(\_st[i][j]!=-1)

\_st[i][j]=ch-48; }

else

{

if(ch=='-')

{ ch=getc(f); \_o[i][j]=ch-48; \_o[i][j]= (-1)\* \_o[i][j];

}

if(\_o[i][j]!=-1)

\_o[i][j]=ch-48; }

ch=getc(f);

}

fclose(f);

while(1)

{

scanf("%s",&path); len=strlen(path); curstate=initial\_state;

for(i=0;i<len;i++)

{

if(\_st[curstate][path[i]-48]==-1)

{ printf("Error"); break;

}

if(\_st[curstate][path[i]-48]!=-1)

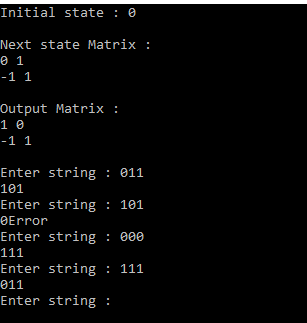
{

printf("%d",\_o[curstate][path[i]-48]);

curstate=\_st[curstate][path[i]-48];

} }}}

OUTPUT :



5. **Moore Machine**

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

int main()

{

FILE \*f;

char ch,str[20];

int i,j,\_r[5],\_o[5][5],initial\_state,len,MY\_MOORE[10][10],k,curstate,l;

f=fopen("moore.txt","r");

if(f==NULL)

{

printf("Empty File ");

}

ch=getc(f); initial\_state=ch-48;

printf("Initial State : %d\n",initial\_state);

ch=getc(f);

for(i=0;i<4;i++)

{ for(j=0;j<3;j++)

{

fscanf(f,"%d",&MY\_MOORE[i][j]);

}

// printf("\n");

}

k=0;

for(i=0;i<4;i++)

{ for(j=0;j<3;j++)

{ if(j==0)

{

\_r[k++]=MY\_MOORE[i][j];

}

else

{ l=j-1; \_o[i][l]=MY\_MOORE[i][j];

}}}

fclose(f);

while(1)

{ curstate=initial\_state;

scanf("%s",&str); len=strlen(str);

for(i=0;i<len;i++)

{

curstate=\_o[curstate][str[i]-48];

if(\_r[curstate]== -1 )

{ printf("Error");

break;}

if(\_r[curstate] != -1)

{

printf("%d",\_r[curstate]);

}}}}

OUTPUT :

