Date
02/02/23

# EXPERIMENT- 3

## CONVERTING NFA To DFA

Hour: A218 DO: 2 faculty:

AIM: To convert the given NFA - Non-deterministi Automata into DFA ie Deterministic finite automata using c programming language.

ALGORITHM:

1. Start
2. Get input from user.
3. Set only state in SDFA to "unmarked".
4. While unmarked state do:
5. Let T be unmarked state.
6. for each a in %. do S = e - Closure (NFA (T,a))
   if S not in SDFA, add S to SDFA
   Set Move DFA (T,a) to S.
7. For each S in SDFA if any S & S in final in DFA then mark S as final state
8. Print result & stop.

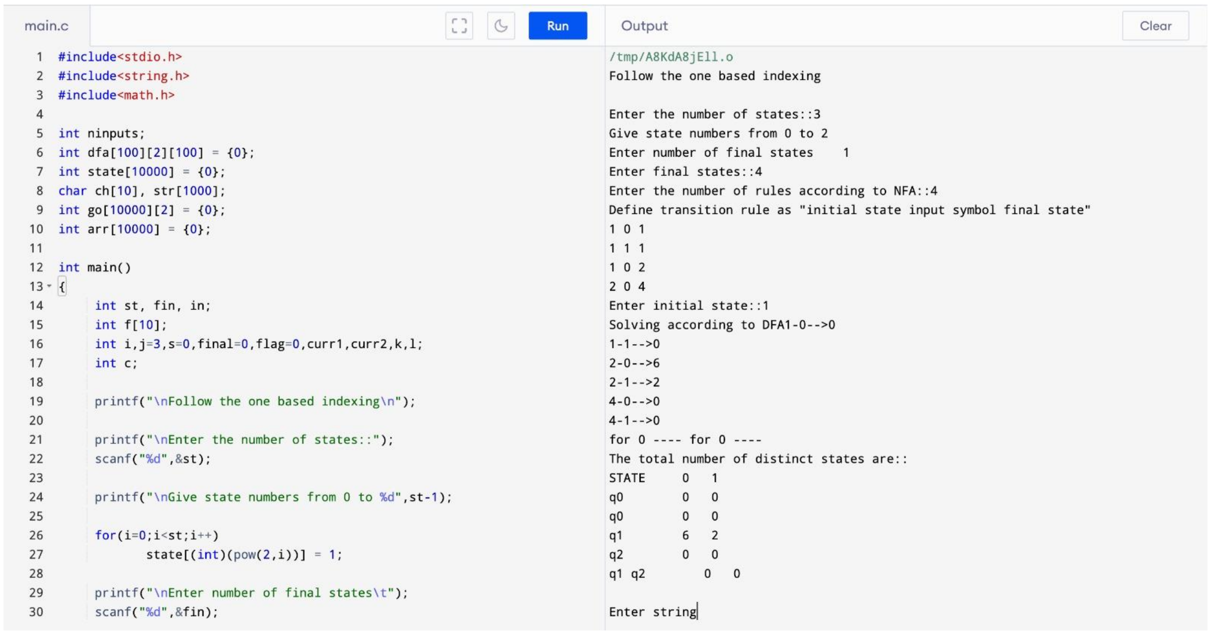
PROGRAM:

```
#include<stdio.h>
#include<String.h>
#include<math.h>
int ninputs;
int dfa[100][2][100] = {0};
int state [1000] = {0};
char ch[10], str[1000];
 int go [10000] [2] = {0};
 int arr [10000] = {0};
int main()
{ int st, fin, in;
   int f[10];
   int i, j=3, s=0, final = 0, flag = 0, curr1, curr2,k,
int c;
printf ("\n Follow the one based indexing \n");
printf ("\n Enter number of states :: ");
scanf ("%d", &st);
printf ("\n Give state numbers from 0 to %d", st-1);
for (i=0; i<st ; i++)
  State [(int)(pow (2,i))] = 1;
printf ("\n Enter number of final states \t");
Scanf ("%d", &fin);
printf ("\n Enter final states :: ");
for (i=0; i<fin ; i++)
{  Scanf ("%d", &f[i]); }
int p,q,r,rel;
printf ("\n Enter number of rules acc to NFA :: ");
Scanf ("%d", &rel);
printf ("\n\nDefine transition rule as \"initial state
   input symbol final state \"\n");
for (i=0 ; i <rel ; i++)
{ scanf ("%d %d %d", &p, &q, &r);
  if (q==0)
    dfa [p] [0] [r] =1;
    else dfa [p][1] [r]=1; }
printf ("\n Enter initial state :: ");
  scanf ("%d", &in);
  in =pow ( 2,in);
 i=0;
printf ("\n solving acc to DFA");
int x=0
for (i=0; i<st ; i++)
```

Input and Output

```
Enter number of States : 3
Give state numbers from 0 to 2
Enter number of final states : 1
    Enter final states : 4

Enter number of rules according to NFA :
Define transition rule as "initial st

101
111
102
204

Enter initial state : 1
Solving according to DFA 0-1 0

1-1 → 0
2-0 → 6
 2-1 → 2
 4-0 → 0
 4-1 → 0

  [illegible] 0 . . . for 0 . . .

Total number of distict states are

state 0 1
q0  0 0
q0  0 0
q1  6 2
q2  0 0
q1 q2 0 0
```

O/P [illegible]
2/2/23



```c
#include<stdio.h>
#include<string.h>
#include<math.h>

int ninputs;
int dfa[100][2][100] = {0};
int state[10000] = {0};
char ch[10], str[1000];
int go[10000][2] = {0};
int arr[10000] = {0};

int main()
{
    int st, fin, in;
    int f[10];
    int i,j=3,s=0,final=0,flag=0,curr1,curr2,k,l;
    int c;

    printf("\nFollow the one based indexing\n");

    printf("\nEnter the number of states::");
    scanf("%d",&st);

    printf("\nGive state numbers from 0 to %d",st-1);

    for(i=0;i<st;i++)
        state[(int)(pow(2,i))] = 1;

    printf("\nEnter number of final states\t");
    scanf("%d",&fin);
```

```
/tmp/A8KdA8jEll.o
Follow the one based indexing

Enter the number of states::3
Give state numbers from 0 to 2
Enter number of final states    1
Enter final states::4
Enter the number of rules according to NFA::4
Define transition rule as "initial state input symbol final state"
1 0 1
1 1 1
1 0 2
2 0 4
Enter initial state::1
Solving according to DFA1-0-->0
1-1-->0
2-0-->6
2-1-->2
4-0-->0
4-1-->0
for 0 ---- for 0 ----
The total number of distinct states are::
STATE     0   1
q0        0   0
q0        0   0
q1        6   2
q2        0   0
q1 q2        0   0

Enter string
```

```
for (i=0; i<st; i++)
{ for (j=0; j<2; j++)
   { int stf=0;
      for (k=0; k<st; k++)
   { if (dfa [i][j][k]==1)
      stf =stf +pow (2,k); }
go [(int)(pow(2,i)) [j] = stf;
 printf ("%d-%d -->%d \n", (int) (pow(2,i)), j, stf);
 if (State [stf] == 0)
        arr[x++]=stf;
   state[stf] = 1; }}
//for new states
for (i=0; i<x; i++)
{ printf ("for %d ---", arr[x]);
     for (j=0; j<2; j++)
      { int new=0;
         for (k=0; k<st; k++)
        { if (arr[i] & (1<<k))
           { int h = pow (2,k);
              if (new ==0)
                  new = go[h][j];
                new = new | go [h][j]);
     }
  }
 if (flag)
 printf ("\n String Accepted");
 else printf ("\n String Rejected");
} return0;
}
```

RESULT: The implimentation of Converting NFA to DFA in C was complied and executed.