

```

1  #include<stdio.h>
2  #include<stdlib.h>
3  #include<string.h>
4  #include<ctype.h>
5
6  int statecount=0,ipsymbolcount=0;
7  int ipsymbols[10];
8  int transitions[10][10][10];
9  char nfa_table[10][10][10];
10 char final_dfa[10][10][10];
11
12 void main()
13 {
14     printf("Kindly enter number of states: ");
15     scanf("%d",&statecount);
16     printf("\n");
17     printf("Kindly enter number of input symbols: ");
18     scanf("%d",&ipsymbolcount);
19     for(int i=0;i<ipsymbolcount;i++)
20     {
21         printf("Kindly enter i/p symbol %d: ",i+1);
22         scanf("%d",&ipsymbols[i]);
23     }
24     printf("\n");
25     for(int i=0;i<ipsymbolcount;i++)
26     {
27         printf("Kindly enter NFA Matrix for i/p symbol %d: \n",ipsymbols[i]);
28         for(int j=0;j<statecount;j++)
29         {
30             for(int k=0;k<statecount;k++)
31             {
32                 scanf("%d",&transitions[i][j][k]);
33             }
34         }
35     }
36     char str[10];
37     for (int i = 0; i < statecount; i++)
38     {
39         for (int j = 0; j < statecount; j++)
40         {
41             for (int k = 0; k < ipsymbolcount; k++)
42             {
43                 if (transitions[k][i][j] == 1)
44                 {
45                     sprintf(str, "q%d", j);
46                     if (strcmp(nfa_table[i][k], str) != 0)
47                     {
48                         strcat(nfa_table[i][k], str);
49                     }
50                 }
51             }
52         }
53     }
54     printf("\n");
55     printf("The NFA table is as follows: \n");
56     for(int i=0;i<ipsymbolcount;i++)
57     {
58         printf("\t%d",ipsymbols[i]);
59     }
60     printf("\n");
61     printf("_____ \n");
62     for(int i=0;i<statecount;i++)
63     {
64         printf("q%d |",i);
65         for(int j=0;j<ipsymbolcount;j++)
66         {
67             printf("\t%s",nfa_table[i][j]);
68         }
69         printf("\n");
70     }
71     printf("\n");
72     char queue[20][10];
73     int front = 0;
74     int rear = 0;
75     int rows = 0;
76     for (int i = 0; i < 20; i++)
77         strcpy(queue[i], "");
78     strcpy(queue[rear], "q0");

```

```

79 rear++;
80 strcpy(final_dfa[rows][0], "q0");
81 while (strcmp(queue[front], "") != 0)
82 {
83     int temp_rows = rows;
84     char new_states[20];
85     for (int i = 0; i < 20; i++)
86         strcpy(new_states, "");
87     for (int j = 0; j < ipsymbolcount; j++)
88     {
89         for (int i = 0; i < 20; i++)
90             strcpy(new_states, "");
91         for (int i = 1; i < strlen(queue[front]); i += 2)
92         {
93             if (isdigit(queue[front][i]))
94             {
95                 int n = queue[front][i] - '0';
96                 for (int l = 1; l < strlen(nfa_table[n][j]); l += 2)
97                 {
98                     int num1;
99                     if (isdigit(nfa_table[n][j][l]))
100                     {
101                         num1 = nfa_table[n][j][l] - '0';
102                         int flag2 = 0;
103                         int num2;
104                         for (int m=1;m< strlen(new_states);mp+=2)
105                         {
106                             if (isdigit(new_states[m]))
107                             {
108                                 num2 = new_states[m] - '0';
109                                 if (num1 == num2)
110                                     flag2 = 1;
111                             }
112                         }
113                         if (flag2 == 0)
114                         {
115                             char temp[20];
116                             sprintf(temp, "q%d", num1);
117                             strcat(new_states, temp);
118                         }
119                     }
120                 }
121             }
122             int temp_states[20];
123             int temp_index = 0;
124             for (int d = 0; d < strlen(new_states); d++)
125             {
126                 if (isdigit(new_states[d]))
127                 {
128                     temp_states[temp_index++] = new_states[d] - '0';
129                 }
130             }
131             for (int q = 0; q < temp_index; q++)
132             {
133                 for (int r = 0; r < temp_index - q - 1; r++)
134                 {
135                     if (temp_states[r] > temp_states[r + 1])
136                     {
137                         int swap = temp_states[r];
138                         temp_states[r] = temp_states[r + 1];
139                         temp_states[r + 1] = swap;
140                     }
141                 }
142             }
143             char tempstr[20];
144             strcpy(new_states, "");
145             for (int q = 0; q < temp_index; q++)
146             {
147                 sprintf(tempstr, "q%d", temp_states[q]);
148                 strcat(new_states, tempstr);
149             }
150             int flag = 0;
151             for (int a = 0; a < rear; a++)
152             {
153                 if (strcmp(queue[a], new_states) == 0)
154                 {
155                     flag = 1;
156                 }

```

```

157     }
158     if (flag == 0)
159     {
160         strcpy(queue[rear], new_states);
161         rear++;
162         strcpy(final_dfa[++temp_rows][0], new_states);
163     }
164     strcpy(final_dfa[rows][j + 1], new_states);
165 }
166 rows++;
167 front++;
168 }
169 printf("\nThe DFA table is as follows:\n");
170 printf("%-10s|", " ");
171 for (int i = 0; i < ipsymbolcount; i++)
172     printf("Input %-4d|", ipsymbols[i]);
173 printf("\n");
174 for (int i = 0; i < 11 * (ipsymbolcount + 1); i++)
175     printf("%s", "=");
176 printf("\n");
177 for (int i = 0; i < rows; i++)
178 {
179     for (int j = 0; j < ipsymbolcount + 1; j++)
180     {
181         printf("%-10s|", final_dfa[i][j]);
182     }
183     printf("\n");
184 }
185 }

```