```c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<ctype.h>

int statecount=0,ipsymbolcount=0;
int ipsymbols[10];
int transitions[10][10][10];
char nfa_table[10][10][10];
char final_dfa[10][10][10];

void main()
{
        printf("Kindly enter number of states: ");
        scanf("%d",&statecount);
        printf("\n");
        printf("Kindly enter number of input symbols: ");
        scanf("%d",&ipsymbolcount);
        for(int i=0;i<ipsymbolcount;i++)
        {
                printf("Kindly enter i/p symbol %d: ",i+1);
                scanf("%d",&ipsymbols[i]);
        }
        printf("\n");
        for(int i=0;i<ipsymbolcount;i++)
        {
                printf("Kindly enter NFA Matrix for i/p symbol %d: \n",ipsymbols[i]);
                for(int j=0;j<statecount;j++)
                {
                        for(int k=0;k<statecount;k++)
                        {
                                scanf("%d",&transitions[i][j][k]);
                        }
                }
        }
        char str[10];
        for (int i = 0; i < statecount; i++)
        {
                for (int j = 0; j < statecount; j++)
                {
                        for (int k = 0; k < ipsymbolcount; k++)
                        {
                                if (transitions[k][i][j] == 1)
                                {
                                        sprintf(str, "q%d", j);
                                        if (strcmp(nfa_table[i][k], str) != 0)
                                        {
                                                strcat(nfa_table[i][k], str);
                                        }
                                }
                        }
                }
        }
        printf("\n");
        printf("The NFA table is as follows: \n");
        for(int i=0;i<ipsymbolcount;i++)
        {
                printf("\t%d",ipsymbols[i]);
        }
        printf("\n");
        printf("   _____________________\n");
        for(int i=0;i<statecount;i++)
        {
                printf("q%d |",i);
                for(int j=0;j<ipsymbolcount;j++)
                {
                        printf("\t%s",nfa_table[i][j]);
                }
                printf("\n");
        }
        printf("\n");
        char queue[20][10];
        int front = 0;
        int rear = 0;
        int rows = 0;
        for (int i = 0; i < 20; i++)
                strcpy(queue[i], "");
        strcpy(queue[rear], "q0");
        rear++;
        strcpy(final_dfa[rows][0], "q0");
        while (strcmp(queue[front], "") != 0)
        {
                int temp_rows = rows;
                char new_states[20];
                for (int i = 0; i < 20; i++)
                    strcpy(new_states, "");
                for (int j = 0; j < ipsymbolcount; j++)
                {
                    for (int i = 0; i < 20; i++)
                        strcpy(new_states, "");
                    for (int i = 1; i < strlen(queue[front]); i += 2)
                    {
                        if (isdigit(queue[front][i]))
                        {
                            int n = queue[front][i] - '0';
                            for (int l = 1; l < strlen(nfa_table[n][j]); l += 2)
                            {
                                int num1;
                                if (isdigit(nfa_table[n][j][l]))
                                {
                                        num1 = nfa_table[n][j][l] - '0';
                                        int flag2 = 0;
                                        int num2;
                                        for (int m=1;m< strlen(new_states);mp+=2)
                                        {
                                            if (isdigit(new_states[m]))
                                            {
                                                num2 = new_states[m] - '0';
                                                if (num1 == num2)
                                                    flag2 = 1;
                                            }
                                        }
                                        if (flag2 == 0)
                                        {
                                            char temp[20];
                                            sprintf(temp, "q%d", num1);
                                            strcat(new_states, temp);
                                        }
                            }
                        }
                    }
                    int temp_states[20];
                    int temp_index = 0;
                    for (int d = 0; d < strlen(new_states); d++)
                    {
                        if (isdigit(new_states[d]))
                        {
                            temp_states[temp_index++] = new_states[d] - '0';
                        }
                    }
                    for (int q = 0; q < temp_index; q++)
                    {
                        for (int r = 0; r < temp_index - q - 1; r++)
                        {
                            if (temp_states[r] > temp_states[r + 1])
                            {
                                int swap = temp_states[r];
                                temp_states[r] = temp_states[r + 1];
                                temp_states[r + 1] = swap;
                            }
                        }
                    }
                    char tempstr[20];
                    strcpy(new_states, "");
                    for (int q = 0; q < temp_index; q++)
                    {
                        sprintf(tempstr, "q%d", temp_states[q]);
                        strcat(new_states, tempstr);
                    }
                    int flag = 0;
                    for (int a = 0; a < rear; a++)
                    {
                        if (strcmp(queue[a], new_states) == 0)
                        {
                            flag = 1;
                        }
                    }
                    if (flag == 0)
                    {
                        strcpy(queue[rear], new_states);
                        rear++;
                        strcpy(final_dfa[++temp_rows][0], new_states);
                    }
                    strcpy(final_dfa[rows][j + 1], new_states);
                }
                rows++;
                front++;
        }
        printf("\nThe DFA table is as follows:\n");
        printf("%-10s|", " ");
        for (int i = 0; i < ipsymbolcount; i++)
                printf("Input %-4d|", ipsymbols[i]);
        printf("\n");
        for (int i = 0; i < 11 * (ipsymbolcount + 1); i++)
                printf("%s", "=");
        printf("\n");
        for (int i = 0; i < rows; i++)
        {
                for (int j = 0; j < ipsymbolcount + 1; j++)
                {
                        printf("%-10s|", final_dfa[i][j]);
                }
        printf("\n");
        }
}
```