NAME : EBIN MATHEW
CLASS : S7R
ROLL NO : 19

AIM : EPSILON TRANSITION

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define MAX_LEN 100
char buffer[MAX_LEN];
int symbols;

void reset(int ar[], int size)
{
	int i;
	for (i = 0; i < size; i++)
	{
		ar[i] = 0;
	}
}
void check(int ar[], char S[])
{
	int i, j;
	int len = strlen(S);
	for (i = 0; i < len; i++)
	{
		j = ((int)(S[i]) - 65);
		ar[j]++;
	}
}
void state(int ar[], int size, char S[])
```

```
{
	int j, k = 0;
	for (j = 0; j < size; j++)
	{
		if (ar[j] != 0)
		S[k++] = (char)(65 + j);
	}
	S[k] = '\0';
}
int closure(int ar[], int size)
{
	int i;
	for (i = 0; i < size; i++)
	{
		if (ar[i] == 1)
		return i;
	}
	return (100);
}
void Display_closure(int states, int closure_ar[],char
*closure_table[],char *NFA_TABLE[][symbols + 1],char
*DFA_TABLE[][symbols])
{
	int i;
	for (i = 0; i < states; i++)
	{
		reset(closure_ar, states);
		closure_ar[i] = 2;
		if (strcmp(&NFA_TABLE[i][symbols], "-") != 0)
		{
			strcpy(buffer, &NFA_TABLE[i][symbols]);
			check(closure_ar, buffer);
			int z = closure(closure_ar, states);
			while (z != 100)
			{
```

```
                        if (strcmp(&NFA_TABLE[z][symbols], "-") !
= 0)
                        {
                                strcpy(buffer, &NFA_TABLE[z]
[symbols]);
                                check(closure_ar, buffer);
                        }
                        closure_ar[z]++;
                        z = closure(closure_ar, states);
                }
        }
        printf("\n e-Closure (%c) :\t", (char)(65 + i));
        state(closure_ar, states, buffer);
        strcpy(&closure_table[i], buffer);
        printf("%s\n", &closure_table[i]);
    }
}
int main()
{
    int i, j, states;
    char buf[10];
    printf("Enter the number of states :");
    scanf("%d",&states);
    printf("Enter the number of symbols :");
    scanf("%d",&symbols);
    printf("\n STATES OF NFA :\t\t");
    for (i = 0; i < states; i++)
        printf("%c, ", (char)(65 + i));
    printf("\n");
    printf("\n GIVEN SYMBOLS FOR NFA: \t");
    for (i = 0; i < symbols; i++)
        printf("%d, ", i);
    printf("eps");
    printf("\n\n");
    char *NFA_TABLE[states][symbols + 1];
```

```
	char *DFA_TABLE[MAX_LEN][symbols];
	for (i = 0; i < states; i++)
	{
		for (j = 0; j < symbols+1; j++)
		{
			if(j==symbols)
			{
				printf("Transition from state %c with symbol
eps = ", (char)(65 + i));
				scanf("%s",&buf);
				strcpy(&NFA_TABLE[i][j], buf);
			}
			else
			{
				printf("Transition from state %c with symbol
%d = ", (char)(65 + i), j );
				scanf("%s",&buf);
				strcpy(&NFA_TABLE[i][j], buf);
			}
		}

	}
	printf("\n NFA STATE TRANSITION TABLE \n\n\n");
	printf("STATES\t");
	for (i = 0; i < symbols; i++)
		printf("|%d\t", i);
	printf("eps\n");
	printf("--------+----------------------------------\n");
	for (i = 0; i < states; i++)
	{
		printf("%c\t", (char)(65 + i));
		for (j = 0; j <= symbols; j++)
		{
			printf("|%s \t", &NFA_TABLE[i][j]);
		}
```

```
            printf("\n");
        }
        int closure_ar[states];
        char *closure_table[states];
        Display_closure(states, closure_ar, closure_table,
NFA_TABLE,DFA_TABLE);
        return 0;
}
```