Program Code

```
#include<stdio.h>

#include<stdlib.h>

struct node {
  int st;
  struct node *link;
};

int state, alpha, s, no_transition,c,r, buffer[20];
char alphabet[20];

int e_closure[20][20] = { 0 };
struct node *transition[20][20] = { NULL };

void findclosure(int x, int sta)
{
  struct node * temp;
  int i;
  if (buffer[x])
    return;
  e_closure[sta][c++] = x;
  buffer[x] = 1;
  if (alphabet[alpha - 1] == 'e' && transition[x][alpha - 1] !=
NULL)
  {
    temp = transition[x][alpha - 1];
    while (temp != NULL)
    {
      findclosure(temp->st, sta);
      temp = temp -> link;
    }
  }
}


int findalpha(char y)
{
  for (int i = 0; i < alpha; i++)
  {
    if (alphabet[i] == y)
    {
      return i;
    }
  }
  return (999);
}

void insert_trantbl(int r, char y, int s)
{
  int j;
  struct node * temp;
  j = findalpha(y);
  if (j == 999)
  {
    printf("error\n");
    exit(0);
  }
```

```
  temp = (struct node * ) malloc(sizeof(struct node));
  temp->st = s;
  temp->link = transition[r][j];
  transition[r][j] = temp;
}

void print_e_closure(int i)
{
  int j;
  printf("{");
    for (j = 0; e_closure[i][j] != -1; j++)
      printf("q%d, ",e_closure[i][j]);
     printf("}");
}

void main() {
  int i, j, k, m, t, n;
  char y;

  struct node *temp;
  printf("Enter the number of alphabets ? \n");
  scanf(" %d", &alpha);
  printf("\nEnter alphabets ? \n");
  for (i = 0; i < alpha; i++)
  {
    scanf(" %c",&alphabet[i]);
  }
  printf("\nEnter the number of states ? \n");
  scanf("%d", &state);
  printf("\nEnter no of transition ? \n");
  scanf("%d", &no_transition);
  printf("\nEnter transition ? \n");
  for (i = 0; i < no_transition; i++)
  {
    scanf("%d %c%d",&r,&y,&s);
    insert_trantbl(r,y,s);
  }
  printf("\n");
  printf("e - closure of states……\n");
  printf("————————-\n");

  for (i = 0; i<state; i++)
  {
    c = 0;
    for (j = 0; j < 20; j++)
    {
      buffer[j] = 0;
      e_closure[i][j] = -1;
    }
    findclosure(i, i);
    printf("\ne - closure(q%d): ", i);
    print_e_closure(i);

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

Output

```
students@pgcse-HP-280-G1-MT:~/Desktop/R7_66/R7_66/1/2$ ./epsilon
Enter the number of alphabets ?
4

Enter alphabets ?
0 1 2 e

Enter the number of states ?
3

Enter no of transition ?
5

Enter transition ?
0 1 0
0 e 1
1 1 1
1 e 2
2 2 2

e - closure of states.....
----------

e - closure(q0): {q0, q1, q2, }
e - closure(q1): {q1, q2, }
e - closure(q2): {q2, }
```