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

int nP,nV,fi[10],fo[10];
char p[10][10],first[10][10],follow[10][10],V[10]; // V: non-terminal

int search(int, char *, int);
bool add(int *, char *, char);
int FIRST(char, int);
int FOLLOW(char, int, int);
bool epsilon(int, int, int *, char*, int);

int main()
{
    int i,j;
    printf("Enter no. of productions: ");
    scanf("%d",&nP);
    printf("Enter the productions:\tV->(V+T)*\t");
    printf("(Use ^ for null string):\n");
    for (i = 0; i < nP; i++)
    {
        scanf("%s",p[i]);
    }

    // extract non-terminals
    nV = 0;
    for (i = 0; i < nP; i++)
    {
        if (search(nV,V,p[i][0])==-1)
        {
            V[nV++] = p[i][0];
        }
    }

    for (i = 0; i < nV; i += 1)
    {
        FIRST(V[i],0);
    }

    follow[0][0] = '$';
    fo[0]++;
    for (i = 0; i < nV; i += 1)
    {
        FOLLOW(V[i],0,3);
    }
    printf("\n");

    for (i = 0; i < nV; i += 1)
    {
        printf("FIRST(%c) = { ",V[i]);
        for (j = 0; j < fi[i]; j += 1)
        {
            printf("%c ",first[i][j]);
        }
        printf("}\n");
    }
    for (i = 0; i < nV; i += 1)
    {
        printf("FOLLOW(%c) = { ",V[i]);
        for (j = 0; j < fo[i]; j += 1)
        {
            printf("%c ",follow[i][j]);
        }
        printf("}\n");
    }
}
```

```
int FIRST(char v, int start)
{
    int index = search(nV,V,v);
    for (int i = start; i < nP; i += 1)
    {
        if (v==p[i][0])
        {
            if (search(nV,V,p[i][3]) == -1) //check if non-terminal: if not non-terminal
            {
                add(&fi[index],first[index],p[i][3]);
            }
            else
            {
                bool eps = epsilon(i,3,&fi[index],first[index],0);
                if (eps)
                {
                    add(&fi[index],first[index],'^');
                }
            }
        }
    }
    return index;
}

int FOLLOW(char v, int r, int c)
{
    int index = search(nV,V,v);
    int temp = c;
    for (int i = r; i < nP; i += 1)
    {
        for (int j = c; j < strlen(p[i]); j += 1)
        {
            if (p[i][j]==v)
            {
                if (p[i][j+1]!='\0')
                {
                    bool eps = epsilon(i,j+1,&fo[index],follow[index],1);
                    if (eps)
                    {
                        int index2 = FOLLOW(p[i][0],r,temp+1);
                        for (int k = 0; k < fo[index2]; k += 1)
                        {
                            add(&fo[index],follow[index],follow[index2][k]);
                        }
                    }
                }
                else
                {
                    int index2 = FOLLOW(p[i][0],r,temp+1);
                    for (int k = 0; k < fo[index2]; k += 1)
                    {
                        add(&fo[index],follow[index],follow[index2][k]);
                    }
                }
            }
            c = 3;
        }
    }
    return index;
}

int search(int n, char a[], int key)
{
    for (int i = 0; i < n; i += 1)
        if (key==a[i])
            return i;
    return -1;
}
```

```
bool add(int *count, char set[], char t)
{
    if (search(*count, set, t) == -1) //check if terminal already added
    {
        set[(*count)++] = t;
    }
}

bool epsilon(int r, int c,int *count, char set[], int f)
{
    int index, eps;
    do
    {
        eps = 0;
        if (search(nV,V,p[r][c])==-1)
        {
            add(count,set,p[r][c]);
            return eps;
        }
        index = FIRST(p[r][c],r+1);
        for (int i = 0; i < fi[index]; i += 1)
        {
            if (first[index][i]=='^')
            {
                eps = 1;
                continue;
            }
            add(count,set,first[index][i]);
        }
    } while (eps && p[r][++c]!='\0');
    return eps;
}
```