Lab program 6:

6. Write a program to compute FOLLOW set.

Aim: Program to compute FOLLOW set for a given grammar of non-terminals.

Algorithm:
Rules to compute FOLLOW set:

1. FOLLOW(S) = { $ }   // where S is the starting Non-Terminal
2. If A -> pBq is a production, where p, B and q are any grammar symbols,then everything in FIRST(q) except Є is in FOLLOW(B.
3. If A->pB is a production, then everything in FOLLOW(A) is in FOLLOW(B).
4. If A->pBq is a production and FIRST(q) contains Є, then FOLLOW(B) contains { FIRST(q) – Є } U FOLLOW(A)

Program:

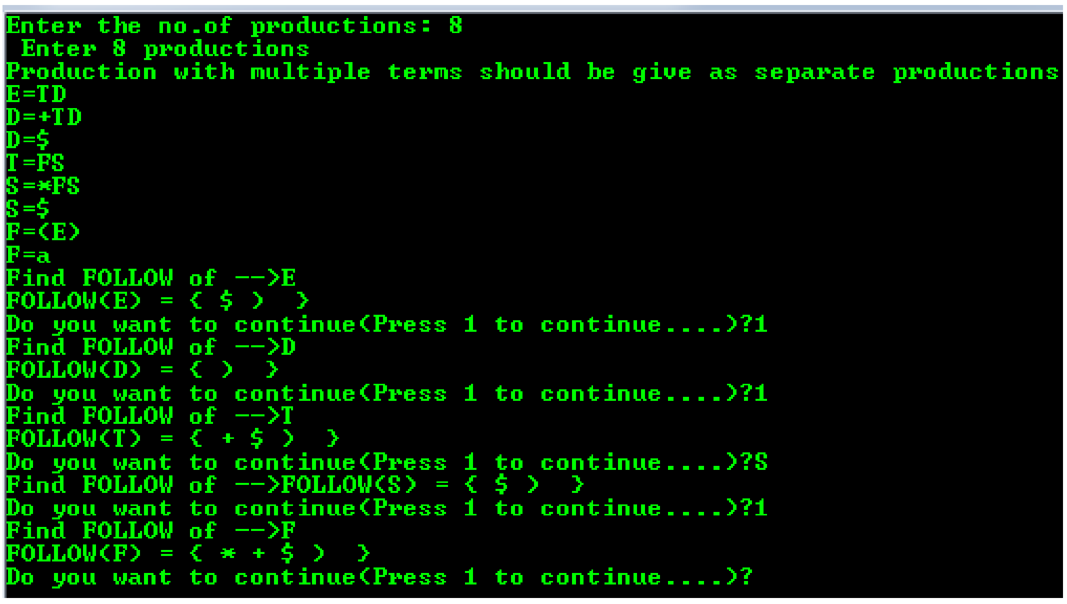
```
#include<stdio.h>
#include<string.h>
int n,m=0,p,i=0,j=0;
char a[10][10],followResult[10];
void follow(char c);
void first(char c);
void addToResult(char);
int main()
{
      int i,choice;
      char c,ch;
      printf("Enter the no.of productions: ");
      scanf("%d", &n);
      printf(" Enter %d productions\nProduction with multiple terms should be give as separate
                                                                          productions \n", n);
      for(i=0;i<n;i++)
          scanf("%s%c",a[i],&ch);    // gets(a[i]);
      do
     {
            m=0;
            printf("Find FOLLOW of -->");
            scanf(" %c",&c);
            follow(c);
            printf("FOLLOW(%c) = { ",c);
            for(i=0;i<m;i++)
                  printf("%c ",followResult[i]);
            printf(" }\n");
            printf("Do you want to continue(Press 1 to continue....)?");
            scanf("%d%c",&choice,&ch);
       }
       while(choice==1);
}
```

```
void follow(char c)
{
      if(a[0][0]==c)addToResult('$');
      for(i=0;i<n;i++)
      {
            for(j=2;j<strlen(a[i]);j++)
            {
                  if(a[i][j]==c)
                  {
                       if(a[i][j+1]!='\0')first(a[i][j+1]);
                       if(a[i][j+1]=='\0'&&c!=a[i][0])
                            follow(a[i][0]);
                  }
            }
       }
}

void first(char c)
{
        int k;
        if(!(isupper(c)))       //f[m++]=c;
              addToResult(c);
              for(k=0;k<n;k++)
              {
                     if(a[k][0]==c)
                    {
                         if(a[k][2]=='$') follow(a[i][0]);
                         else if(islower(a[k][2]))
                              //f[m++]=a[k][2];
                             addToResult(a[k][2]);
                         else
                             first(a[k][2]);
                     }
              }
}

void  addToResult(char c)
{
   int i;
   for( i=0;i<=m;i++)
       if(followResult[i]==c)
           return;
  followResult[m++]=c;
}
```

Output:



```
Enter the no.of productions: 8
 Enter 8 productions
Production with multiple terms should be give as separate productions
E=TD
D=+TD
D=$
T=FS
S=*FS
S=$
F=(E)
F=a
Find FOLLOW of -->E
FOLLOW(E) = { $ )  }
Do you want to continue(Press 1 to continue....)?1
Find FOLLOW of -->D
FOLLOW(D) = { )  }
Do you want to continue(Press 1 to continue....)?1
Find FOLLOW of -->T
FOLLOW(T) = { + $ )  }
Do you want to continue(Press 1 to continue....)?S
Find FOLLOW of -->FOLLOW(S) = { $ )  }
Do you want to continue(Press 1 to continue....)?1
Find FOLLOW of -->F
FOLLOW(F) = { * + $ )  }
Do you want to continue(Press 1 to continue....)?
```