**COMPILER DESIGN LAB**

Name:-Saket Kumar Baranwal

RA1911003010414

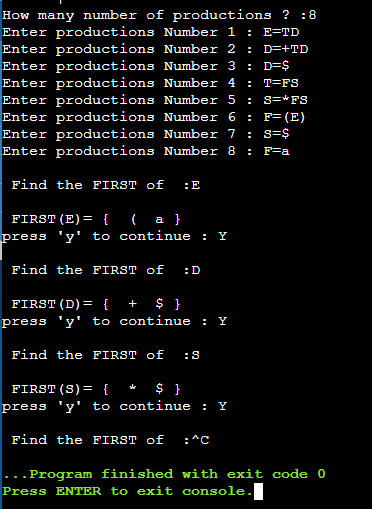
# Computation of FIRST and Follow

# Aim: - To study the c program to implement first of a given grammar.

# Code:-

# #include<stdio.h> #include<ctype.h> void FIRST(char[],char ); void addToResultSet(char[],char); int numOfProductions; char productionSet[10][10]; main() {     int i;     char choice;     char c;     char result[20];     printf("How many number of productions ? :");     scanf(" %d",&numOfProductions);     for(i=0;i<numOfProductions;i++)//read production string eg: E=E+T     {         printf("Enter productions Number %d : ",i+1);         scanf(" %s",productionSet[i]);     }     do     {         printf("\n Find the FIRST of  :");         scanf(" %c",&c);         FIRST(result,c); //Compute FIRST; Get Answer in 'result' array         printf("\n FIRST(%c)= { ",c);         for(i=0;result[i]!='\0';i++)         printf(" %c ",result[i]);       //Display result         printf("}\n");          printf("press 'y' to continue : ");         scanf(" %c",&choice);     }     while(choice=='y'||choice =='Y'); } /\*  \*Function FIRST:  \*Compute the elements in FIRST(c) and write them  \*in Result Array.  \*/ void FIRST(char\* Result,char c) {     int i,j,k;     char subResult[20];     int foundEpsilon;     subResult[0]='\0';     Result[0]='\0';     //If X is terminal, FIRST(X) = {X}.     if(!(isupper(c)))     {         addToResultSet(Result,c);                return ;     }     //If X is non terminal     //Read each production     for(i=0;i<numOfProductions;i++)     { //Find production with X as LHS         if(productionSet[i][0]==c)         { //If X → ε is a production, then add ε to FIRST(X).  if(productionSet[i][2]=='$') addToResultSet(Result,'$');             //If X is a non-terminal, and X → Y1 Y2 … Yk             //is a production, then add a to FIRST(X)             //if for some i, a is in FIRST(Yi),             //and ε is in all of FIRST(Y1), …, FIRST(Yi-1).       else             {                 j=2;                 while(productionSet[i][j]!='\0')                 {                 foundEpsilon=0;                 FIRST(subResult,productionSet[i][j]);                 for(k=0;subResult[k]!='\0';k++)                     addToResultSet(Result,subResult[k]);                  for(k=0;subResult[k]!='\0';k++)                      if(subResult[k]=='$')                      {                          foundEpsilon=1;                          break;                      }                  //No ε found, no need to check next element                  if(!foundEpsilon)                      break;                  j++;                 }             }     } }     return ; } /\* addToResultSet adds the computed  \*element to result set.  \*This code avoids multiple inclusion of elements   \*/ void addToResultSet(char Result[],char val) {     int k;     for(k=0 ;Result[k]!='\0';k++)         if(Result[k]==val)             return;     Result[k]=val;     Result[k+1]='\0'; }

# Output:-



# Aim:- To study the c program to implement follow of a given grammar.

# Code:-

# #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;  int 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:-

# 