Week - 7

Aim: Write a program to identify the FIRST and FOLLOW of every variable of the given grammar. The program takes the number of productions in the given grammar and later takes the productions.

```
#include <bits/stdc++.h>

void FIRST(char*, char);
void addToArray(char*, char);
void printArray(char*);
void FOLLOW(char *result, char c);

int n;
char production[20][20];

main()
{
    int i, j = 0, k, foundNt = 0;
    char c, result[20], nt[20];
    nt[0] = '\0';
    cout << "Enter no. of productions: ";
    cin >> n;
    for(i=0; i<n; i++)
    {
        cout << "Enter production number +ve: ", i+1);
        cin >> production[i];
        addToArray(nt, production[i][0]);
    }
    for(k=0; nt[k] != '\0'; k++)
    {
        c = nt[k];
        FIRST(result, c);
        cout << "FIRST(" << c << ") = { ", c);
        printArray(result);
        printf("} \n");
        getchar();
    }
```

AP21110011494

```
for (k=0; nt[k] != '\0'; k++)
{
    c = nt[k];
    FOLLOW (result, c);
    printf ("\n FOLLOW(%c) = { ", c);
    printArray (result);
    printf ("}\n");

}
}

void FIRST (char *result, char c)
{ int i,j,k;
  char subResult [20];
  int foundEpsilon;
  subResult [0] = '\0';
  Result [0] = '\0';

  if (!(isupper (c)))
  {
     addToArray (Result, c);

  }
  else
  {
     for (i=0; i<n; i++)
     {  if (production [i] [0] == c)
        {
           if (production [i][2] == '^')
           {
              addToArray (Result, '^');

           }
           else
           {
              for (j=2; production [i][j] != '\0'; j++)
              {
                 foundEpsilon = 0;
```

```
FIRST(subResult, production[i][j]);
for(k=0; subResult[k]!='\0'; k++)
{
    if(subResult[k]=='^')
        foundEpsilon=1;
    else
        addToArray(Result, subResult[k]);
}
if(!foundEpsilon)
    break;
}
}
}
}

void FOLLOW(char *result, char c)
{
    int i, j, k, t, foundEpsilon=0;
    char subResult[20];
    result[0]='\0';

    if(c==production[0][0])
        addToArray(result, '$');
    for(i=0; i<n; i++)
    {
        int l=strlen(production[i]);
        for(j=2; j<l; j++)
            foundEpsilon=0;
            if(production[i][j]==c)
            {
                if(j==l-1)
                {
                    FOLLOW(result, production[i][0]);
                }
                else
                {
                    if(!isupper(production[i][j+1]))
```

```
{
    addToArray(result, production[i][j+1]);
}
else
{
    for (k = j+1; k<l; k++)
    {
        subResult[0] = '!0';
        foundEpsilon = 0;
        FIRST(subResult, production[i][k]);
        for (t=0; subResult[t] != '0'; t++)
        {
            if (subResult[t] == '^')
            {
                foundEpsilon = 1;
            }
            else
            {
                addToArray(result, subResult[t]);
            }
        }
        if (!foundEpsilon)
        {
            break;
        }
    }
    if (foundEpsilon)
    {
        FOLLOW(result, production[i][0]);
    }
}}}}
}
```

AP21110011494

Input:

3
S > aSa
S > bSb
S > c

Output:

FIRST(S) = { a, b, c }

FOLLOW(S) = { $, a, b }