Exp 6 - FIRST AND FOLLOW IMPLEMENTATION

Aim: To write a program for implementation of First and Follow

FIRST:

Algorithm:

Rule-01:
For a production rule X → ∈,
First(X) = { ∈ }

Rule-02:
For any terminal symbol ‘a’,
First(a) = { a }

Rule-03:
For a production rule X → Y1Y2Y3,

Calculating First(X)
• If ∈ ∉ First(Y1), then First(X) = First(Y1) • If ∈ ∈ First(Y1), then First(X) = { First(Y1) – ∈ } ∪ First(Y2Y3)
Calculating First(Y2Y3) • If ∈ ∉ First(Y2), then First(Y2Y3) = First(Y2) • If ∈ ∈ First(Y2), then First(Y2Y3) = { First(Y2) – ∈ } ∪ First(Y3)
Similarly, we can make expansion for any production rule X → Y1Y2Y3…..Yn.

Exp 6 - FIRST AND FOLLOW IMPLEMENTATION
RA1911003010143
Abhishek Kumar

Program:

```c
#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';
```

Exp 6 - FIRST AND FOLLOW IMPLEMENTATION
RA1911003010143
Abhishek Kumar

```
    // 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++)
```

Exp 6 - FIRST AND FOLLOW IMPLEMENTATION
RA1911003010143
Abhishek Kumar

```
        if (Result[k] == val)
            return;
    Result[k] = val;
    Result[k + 1] = '\0';
}
```

Output:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

PS C:\Users\abhis\Desktop\Study\6th Semester\Compiler Des
ign\Lab\Exp 6> cd "c:\Users\abhis\Desktop\Study\6th Semes
ter\Compiler Design\Lab\Exp 6\" ; if ($?) { g++ Exp6.cpp
-o Exp6 } ; if ($?) { .\Exp6 }
How many number of productions ? :8
Enter productions Number 1 : E=TD
Enter productions Number 2 : D=+RD
Enter productions Number 3 : D=$
Enter productions Number 4 : T=FS
Enter productions Number 5 : S=*FS
Enter productions Number 6 : S=$
Enter productions Number 7 : F=(E)
Enter productions Number 8 : F=a

 Find the FIRST of :E

 FIRST(E)= {  (  a }
press 'y' to continue : y

 Find the FIRST of :D

 FIRST(D)= {  +  $ }
press 'y' to continue : Y

 Find the FIRST of :S

 FIRST(S)= {  *  $ }
press 'y' to continue : Y

 Find the FIRST of :a

 FIRST(a)= {  a }
press 'y' to continue :
```

Exp 6 - FIRST AND FOLLOW IMPLEMENTATION
RA1911003010143
Abhishek Kumar

FOLLOW:

Algorithm:

Rule-01:
For the start symbol S, place $ in Follow(S).

Rule-02:
For any production rule A → αB,
Follow(B) = Follow(A)

Rule-03:
For any production rule A → αBβ,
- If ∈ ∉ First(β), then Follow(B) = First(β)
- If ∈ ∈ First(β), then Follow(B) = { First(β) – ∈ } ∪ Follow(A)

Program:

```c
#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);
```

Exp 6 - FIRST AND FOLLOW IMPLEMENTATION
RA1911003010143
Abhishek Kumar

```
        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;
```

Exp 6 - FIRST AND FOLLOW IMPLEMENTATION
RA1911003010143
Abhishek Kumar

```
    followResult[m++] = c;
}
```

Output:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

Enter the no.of productions: 8
 Enter 8 productions
Production with multiple terms should be give as separate p
roductions
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 -->S
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....)?
```

Result: Implementation of First and Follow was successfully performed using C.