2CS701- Compiler Construction

Practical 3

Aim: **Write a program to find first( ), and follow() set for each non-terminal of given grammar.**

Author: Darshil Maru 20BCE514

Date: September 27, 2022

Guide: Prof. Deepti Saraswat

Code:

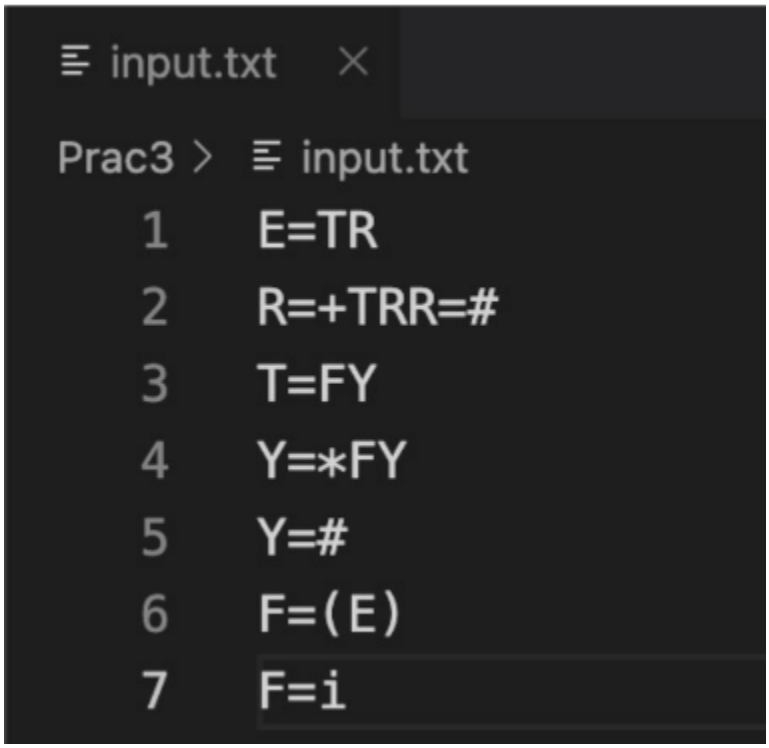
```c
#include <stdio.h>
#include <ctype.h>
#include <string.h>
#define OJ                              \
    freopen("input.txt", "r", stdin); \
    freopen("output.txt", "w", stdout);
void followfirst(char, int, int);
void follow(char c);
void findfirst(char, int, int);
int count, n = 0;
char calc_first[10][100];
char calc_follow[10][100];
int m = 0;
char production[10][10];
char f[10], first[10];
int k;
char ck;
int e;
int main(int argc, char **argv)
{
    OJ;

    int jm = 0;
    int km = 0;
    int i, choice;
    char c, ch;
    count = 8;
    for (int i = 0; i < 8; i++)
    {
        char s[10];
        scanf("%s", &s);
        strcpy(production[i], s);
    }
    int kay;
    char done[count];
    int ptr = -1;
    for (k = 0; k < count; k++)
    {
```

```
        for (kay = 0; kay < 100; kay++)
        {
            calc_first[k][kay] = '!';
        }
    }
    int point1 = 0, point2, xxx;
    for (k = 0; k < count; k++)
    {
        c = production[k][0];
        point2 = 0;
        xxx = 0;
        for (kay = 0; kay <= ptr; kay++)
            if (c == done[kay])
                xxx = 1;
        if (xxx == 1)
            continue;
        findfirst(c, 0, 0);
        ptr += 1;
        done[ptr] = c;

        printf("\n First(%c) = { ", c);
        calc_first[point1][point2++] = c;
        for (i = 0 + jm; i < n; i++)
        {
            int lark = 0, chk = 0;
            for (lark = 0; lark < point2; lark++)
            {
                if (first[i] == calc_first[point1][lark])
                {
                    chk = 1;
                    break;
                }
            }
            if (chk == 0)
            {
                printf("%c, ", first[i]);
                calc_first[point1][point2++] = first[i];
            }
        }
        printf("}\n");
```

```
        jm = n;
        point1++;
    }
    printf("\n");
printf("
\n\n");
char donee[count];
ptr = -1;
for(k = 0; k < count; k++) {
        for (kay = 0; kay < 100; kay++)
        {
            calc_follow[k][kay] = '!';
        }
}
point1 = 0;
int land = 0;
for(e = 0; e < count; e++)
{
        ck = production[e][0];

        point2 = 0;
        xxx = 0;
        for (kay = 0; kay <= ptr; kay++)
            if (ck == donee[kay])
                xxx = 1;
        if (xxx == 1)
            continue;
        land += 1;
        follow(ck);
        ptr += 1;
        donee[ptr] = ck;
        printf(" Follow(%c) = { ", ck);
        calc_follow[point1][point2++] = ck;
        for (i = 0 + km; i < m; i++)
        {
            int lark = 0, chk = 0;
            for (lark = 0; lark < point2; lark++)
            {
                if (f[i] == calc_follow[point1][lark])
                {
```

```
                    chk = 1;
                    break;
                }
            }
            if (chk == 0)
            {
                printf("%c, ", f[i]);
                calc_follow[point1][point2++] = f[i];
            }
        }
        printf(" }\n\n");
        km = m;
        point1++;
}
}
```

Input:



Output:

output.txt

Prac3 > output.txt

```
1
2   First(E) = { (, i, }
3
4   First(R) = { +, }
5
6   First(T) = { (, i, }
7
8   First(Y) = { *, #, }
9
10  First(F) = { (, i, }
11
12  ------------------------------------------
13
14  Follow(E) = { $, ),  }
15
16  Follow(R) = { $, ), +, =,  }
17
18  Follow(T) = { +,  }
19
20  Follow(Y) = { +,  }
21
22  Follow(F) = { *, +,  }
23
24
```