**Name and Roll No :** Yash Ginoya (20BCE075)

**Course and Course code :** Compiler Construction (2CS701)

**Practical 3 :**

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

----------------------------------------------------------------------------------------------------------------------------

**Code:**

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

**#define max 20**

**using namespace std;**

**char prod[max][10];**

**char ter[10], nt[10];**

**char first[10][10], follow[10][10];**

**int eps[10];**

**int count\_var = 0;**

**int findpos(char ch)**

**{**

**int n;**

**for (n = 0; nt[n] != '\0'; n++)**

**if (nt[n] == ch)**

**break;**

**if (nt[n] == '\0')**

**return 1;**

**return n;**

**}**

**int IsCap(char c)**

**{**

**if (c >= 'A' && c <= 'Z')**

**return 1;**

**return 0;**

**}**

**void add(char \*arr, char c)**

**{**

**int i, flag = 0;**

**for (i = 0; arr[i] != '\0'; i++)**

**{**

**if (arr[i] == c)**

**{**

**flag = 1;**

**break;**

**}**

**}**

**if (flag != 1)**

**arr[strlen(arr)] = c;**

**}**

**void addarr(char \*s1, char \*s2)**

**{**

**int i, j, flag = 99;**

**for (i = 0; s2[i] != '\0'; i++)**

**{**

**flag = 0;**

**for (j = 0;; j++)**

**{**

**if (s2[i] == s1[j])**

**{**

**flag = 1;**

**break;**

**}**

**if (j == strlen(s1) && flag != 1)**

**{**

**s1[strlen(s1)] = s2[i];**

**break;**

**}**

**}**

**}**

**}**

**void addprod(char \*s)**

**{**

**int i;**

**prod[count\_var][0] = s[0];**

**for (i = 3; s[i] != '\0'; i++)**

**{**

**if (!IsCap(s[i]))**

**add(ter, s[i]);**

**prod[count\_var][i - 2] = s[i];**

**}**

**prod[count\_var][i - 2] = '\0';**

**add(nt, s[0]);**

**count\_var++;**

**}**

**void findfirst()**

**{**

**int i, j, n, k, e, n1;**

**for (i = 0; i < count\_var; i++)**

**{**

**for (j = 0; j < count\_var; j++)**

**{**

**n = findpos(prod[j][0]);**

**if (prod[j][1] == (char)238)**

**eps[n] = 1;**

**else**

**{**

**for (k = 1, e = 1; prod[j][k] != '\0' && e == 1; k++)**

**{**

**if (!IsCap(prod[j][k]))**

**{**

**e = 0;**

**add(first[n], prod[j][k]);**

**}**

**else**

**{**

**n1 = findpos(prod[j][k]);**

**addarr(first[n], first[n1]);**

**if (eps[n1] == 0)**

**e = 0;**

**}**

**}**

**if (e == 1)**

**eps[n] = 1;**

**}**

**}**

**}**

**}**

**void findfollow()**

**{**

**int i, j, k, n, e, n1;**

**n = findpos(prod[0][0]);**

**add(follow[n], '#');**

**for (i = 0; i < count\_var; i++)**

**{**

**for (j = 0; j < count\_var; j++)**

**{**

**k = strlen(prod[j]) - 1;**

**for (; k > 0; k--)**

**{**

**if (IsCap(prod[j][k]))**

**{**

**n = findpos(prod[j][k]);**

**if (prod[j][k + 1] == '\0')**

**{**

**n1 = findpos(prod[j][0]);**

**addarr(follow[n], follow[n1]);**

**}**

**if (IsCap(prod[j][k + 1]))**

**{**

**n1 = findpos(prod[j][k + 1]);**

**addarr(follow[n], first[n1]);**

**if (eps[n1] == 1)**

**{**

**n1 = findpos(prod[j][0]);**

**addarr(follow[n], follow[n1]);**

**}**

**}**

**else if (prod[j][k + 1] != '\0')**

**add(follow[n], prod[j][k + 1]);**

**}**

**}**

**}**

**}**

**}**

**int main()**

**{**

**char s[max], i;**

**cout << "Enter the productions\n";**

**cin >> s;**

**while (strcmp("end", s))**

**{**

**addprod(s);**

**cin >> s;**

**}**

**findfirst();**

**findfollow();**

**for (i = 0; i < strlen(nt); i++)**

**{**

**cout << nt[i] << "\t";**

**cout << first[i];**

**if (eps[i] == 1)**

**cout << ((char)238) << "\t";**

**else**

**cout << "\t";**

**cout << follow[i] << "\n";**

**}**

**return 0;**

**;**

**}**

**OutPut :**

**E->TE'**

**E->+TE'**

**E'->**

**T->FT'**

**T'->\*FT'**

**T'->**

**F->ID**

**end**

**E +> #'**

**T > +>'**

**F > >**

**Conclusion :**

In this practical, i learn how to write a program to find first( ), and follow() set for each non-terminal of given grammar.