Problem 3: First and Follow Function

Introduction:

The construction of a predictive parser is aided by two functions associated with a grammar G. These functions, **FIRST** and **FOLLOW**, allow us to fill in the entries of a predictive parsing table for G, whenever possible. First and Follow sets are needed so that the parser can properly apply the needed production rule at the correct position.

FIRST(α)

First(α) is a set of terminal symbols that begin in strings derived from α.

Rule 1:

If FIRST (α) is the set of terminals that begin the strings derived from **α**.

Example 1:

Consider the production rule-

A → abc | def | ghi

Then, we have-

First (A) = {a, d, g}

Rule 2:

If **α** → ∈, then ∈, is also in FIRST (**α**).

Example 2:

For a production rule X → ∈,

First(X) = {∈}

Rule 3:

If FIRST (α) is the set of non-terminals:

```
if X → Y1 Y2 Y3 ... YK is a rule then
   if a ∈ FIRST(Y1) then
      add a to FIRST(X)
   if ε ∈ FIRST(Y1) and a ∈ FIRST(Y2) then
      add a to FIRST(X)
   if ε ∈ FIRST(Y1) and ε ∈ FIRST(Y2) and a ∈ FIRST(Y3) then
      add a to FIRST(X)
   ...
   if ε ∈ FIRST(Yi) for all Yi then
      add ε to FIRST(X)
```

Example 3:

First (Y1)→ {a, b, ∈}

First (Y2)→ {c, d, ∈}

First (Y3) → {e, f, ∈}

Then First (X)→ {a, b, c, d, e, f, ∈}

Example 4:

First (Y1)→ {a, b, ∈}

First (Y2)→ {c, d, ∈}
First (Y3) → {e, f}
Then First (X)→ {a, b, c, d, e, f}

Example 5:
First (Y1)→ {a, b, ∈}
First (Y2)→ {c, d}
First (Y3) → {e, f, ∈}
Then First (X)→ {a, b, c, d}

Example 6:
First (Y1)→ {a, b}
First (Y2)→ {c, d, ∈}
First (Y3) → {e, f, ∈}
Then First (X)→ {a, b}

FOLLOW (A):
FOLLOW (A), for nonterminal A, to be the set of terminals that can appear immediately to the right of A in some sentential form. That is, the set of terminals such that there exists a derivation of the form S→ aAαβ for some α and β.

To compute FOLLOW (A) for all nonterminal A, apply the following rules until nothing can be added to any FOLLOW set:
Rule 1:
Place $ in FOLLOW(S), where S is the start symbol and $ is the input right end marker.
Rule 2:
If there is a production A→ αBβ, then everything in FIRST (β), except for ∈, is placed in FOLLOW (B).
Rule 3:
If there is a production A →αB, or a production A → aBβ where FIRST (β) contains e (i.e., β→∈), Then everything in FOLLOW (A) is in FOLLOW (B).
Example 1:

S → aSe | B	FIRST(S) = {a, b, c, d, ε}
B → bBCf | C	FIRST (B) = {b, c, d, ε}
C → cCg | d | ε	FIRST(C) = {c, d, ε}

According to Rule 1:
FOLLOW (S) = {$}
According to Rule 2:

FOLLOW(C) = {f, g}
FOLLOW (B) = {c, d, f}
FOLLOW(S) = {$, e}

According to Rule 3:

FOLLOW (C) = {f, g} ∪ FOLLOW (B) = {c, d, e, f, g, $}
FOLLOW (B) ={c, d, f} ∪ FOLLOW(S) = {c, d, e, f, $}
FOLLOW (S) = {$, e}

Example 2:

Consider the expression grammar stated below:

E→ T E'
E'→ + T E' | e
T→ F T'
T'→ * F T' | e
F → (E) | id

Then:

FIRST (E) = FIRST (T) = FIRST (F) = {(, id}
FIRST (E') = {+, e}
FIRST (T') = {*, e}
FOLLOW (E) = FOLLOW (E') = {), $}
FOLLOW (T) = FOLLOW (T') = {+,), $}
FOLLOW (F) = {+, *,), $}

Example 3:

Consider the expression grammar stated below:

S → aBDh
B → cC
C → bC | ∈
D → EF
E → g | ∈
F → f | ∈

The first and follow functions are as follows-

First(S) = {a}
First (B) = {c}
First (C) = {b, ∈}
First (D) = {First (E) – ∈} ∪ First (F) = {g, f, ∈}
First (E) = {g, ∈}
First (F) = {f, ∈}

Follow (S) = {$}
Follow (B) = {First (D) – ∈} ∪ First (h) = {g, f, h}
Follow(C) = Follow (B) = {g, f, h}
Follow (D) = First (h) = {h}
Follow (E) = {First (F) – ∈} ∪ Follow (D) = {f, h}
Follow (F) = Follow (D) = {h}

Sample Input:
You can take input from a **text file/console.** Instead of **epsilon (**∈) use **hash (#)** symbol.

```
Input:
E -> TR
R -> +T R| #
T -> F Y
Y -> *F Y | #
F -> (E) | i

Output:
 First (E) = {(, i,}
 First(R) = {+, #,}
 First (T) = {(, i,}
 First(Y) = {*, #,}
 First (F) = {(, i,}

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

 Follow (E) = {$,),}
 Follow(R) = {$,),}
 Follow (T) = {+, $,),}
 Follow(Y) = {+, $,),}
 Follow (F) = {*, +, $,),}
```