Problema E

Null, First e Follow de gramáticas algébricas

Problema

Considere uma gramática algébrica $G = (\Sigma, N, P, S)$ com possivelmente produções *epsilon*.

Escreva um programa que lê G e que calcule o predicado $NULL$ e os conjuntos $FIRST$ e $FOLLOW$ para os não terminais da gramática G.

Considere um não terminal A e as suas produções $A \to \alpha_1 \mid \ldots \mid \alpha_n$

NULL. O predicado $NULL$ sobre não-terminais está definido da seguinte forma:

$$NULL(A) = NULL(\alpha_1) \vee \ldots \vee NULL(\alpha_n)$$

e sobre uma palavraα de $(N \cup \Sigma)^*$ da seguinte forma:

$$NULL(\alpha) = \begin{cases} \top & se\ \alpha = \epsilon \\ NULL(X_1) \wedge \ldots \wedge NULL(X_m) & se\ \alpha = X_1 \ldots\ X_m\ e\ \forall i \in \{1..m\}, X_i \in N \\ \bot & \text{senão} \end{cases}$$

FIRST. O conjunto $FIRST$ sobre não-terminais está definido da seguinte forma:

$$FIRST(A) = FIRST(\alpha_1) \cup \ldots \cup FIRST(\alpha_n)$$

e sobre uma palavraα de $(N \cup \Sigma)^*$ da seguinte forma:

$$FIRST(\alpha) = \begin{cases} \emptyset & se\ \alpha = \epsilon \\ a_1 & se\ \alpha \in \Sigma^+\ e\ \alpha = a_1 \ldots a_p \\ FIRST(\alpha_1) & se\ \alpha = \alpha_1 X \alpha_2\ com\ \alpha_1 \in \Sigma^+,\ X \in N\ e\ \alpha_2 \in (N \cup \Sigma)^* \\ FIRST(X) & se\ \alpha = X\alpha_2\ com\ X \in N,\ \ NULL(X) = \bot e\ \alpha_2 \in (N \cup \Sigma)^* \\ FIRST(X) \cup FIRST(\alpha_2) & se\ \alpha = X\alpha_2\ com\ X \in N,\ \ NULL(X) = \top e\ \alpha_2 \in (N \cup \Sigma)^* \end{cases}$$

FOLLOW. O conjunto $FOLLOW$
sobre não-terminais está definido da seguinte forma:

$$FOLLOW(A) = (\bigcup_{Y \to\ \alpha X \beta} (FIRST(\beta))) \cup (\bigcup_{Y \to\ \alpha X \beta\ \wedge\ Null(\beta)} FOLLOW(Y))$$

Para o caso do não-terminal S, juntamos ao conjunto calculado pela fórmula anterior o símbolo $\sharp$ que representa o carácter "*fim de entrada*"

Entrada

Para simplificar o formatos dos dados em entrada admitiremos aqui que os símbolos não terminais são representadas por nomes (string) começados por maiúsculas e os símbolos terminais são constituídos exclusivamente por minúsculas. Em particular o símbolo inicial será sempre o não-terminal S. Uma produção terá sempre o formato `N ->` α em que α é uma sequência não vazia de símbolos (terminais ou não-terminais separados por um espaço). Em particular o símbolo ϵ é representado pelo carácter `_` (underscore).

O formato dos dados em entrada é então o seguinte.

Na primeira linha consta um inteiro n que indica quantas produções tem a gramática.

As restantes n linhas introduzem as produções da gramática (uma por linha).

Saída

Imaginemos que a gramática em entrada tenha m símbolos não terminais. a saída é constituída por $3 \times m$ linhas.

As m primeiras linhas contêm a informação calculada para $NULL$ no formato

`NULL(X) = True`

ou

`NULL(X) = False`

listadas por ordem alfabética dos não-terminais X, com a excepção do não terminal S que é apresentado na primeira posição.

As m linhas seguintes contêm a informação calculada para $FIRST$ no formato

`FIRST(X) = a1 a2 a3 ...  ap`

se, para o não-terminal X, $FIRST(X) = \{a_1 \dots a_p\}$, ordenado pela ordem alfabética. A ordem de apresentação seguida para os não-terminais é a mesma que a dos $FIRST$.

As m linhas finais apresentam a informação calculada para $FOLLOW$ da mesma forma que se apresenta a informação para $FIRST$. O caso particular do símbolo $\sharp$ de $FOLLOW(S)$ segue a regra da ordem alfabética e é apresentado na primeira posição.

Exemplo de entrada

```
11
S -> A
S -> B D e
A -> A e
A -> e
B -> d
B -> C C
C -> e C
C -> e
C -> _
D -> a D
D -> a
```

Exemplo de Saída

```
NULL(S) = False
NULL(A) = False
NULL(B) = True
NULL(C) = True
NULL(D) = False
FIRST(S) = a d e
FIRST(A) = e
```

```
FIRST(B) = d e
FIRST(C) = e
FIRST(D) = a
FOLLOW(S) = #
FOLLOW(A) = # e
FOLLOW(B) = a
FOLLOW(C) = a e
FOLLOW(D) = e
```