Compiladores

Analise Sintática

First Follow.

Na aula de hoje...

- **Revisão: gramáticas**
- **Relações em uma gramática: Cabeça, Último, Primeiro (*First*) e Seguinte (*Follow*)**
  - Capítulo 4 (seção 4.4.2) do livro ***Compiladores : Princípios, técnicas e ferramentas***

Definições

- **Alfabeto ou vocabulário:** Conjunto finito não vazio de símbolos. Símbolo é um elemento qualquer de um alfabeto.
  - Ex:
    - {a,b}
    - {0,1,2,3,4,5,6,7,8,9}

- **Cadeia:** Concatenação de símbolos de um alfabeto. Define-se como cadeia vazia ou nula uma cadeia que não contém nenhum símbolo.
  - Ex:
    - aab
    - 123094
    - λ – cadeia nula

Definições

- **Comprimento de cadeia:** Número de símbolos de uma cadeia. Ex:
  - |aab| = 3
  - |123094|=6
  - |λ|=0

Definições

- **Linguagem** é uma coleção de **cadeias** de símbolos, de comprimento finito. Estas cadeias são denominadas sentenças da linguagem, e são **formadas pela justaposição de elementos individuais, que são os símbolos ou átomos da linguagem**.

Definições

❑ Métodos de Representação de Linguagens

1) **Enumeração** (especificação) das cadeias de símbolos que formam as suas sentenças (somente linguagens finitas podem ser representadas por este método)

2) Através de um conjunto de **leis de formação** das cadeias (**Gramática**)

3) Através de regras de aceitação de cadeias (às regras de aceitação dá-se o nome de **Reconhecedor** - autômatos)

Gramáticas

2) Leis de Formação

- Através de um conjunto de leis de formação das cadeias (ao conjunto de leis de formação dá-se o nome de **Gramática**)
  - dada uma cadeia de símbolos, só é possível afirmar que tal cadeia pertence à linguagem se for possível, aplicando-se as leis de formação que compõem a gramática da linguagem, derivar (sintetizar) a cadeia em questão.
  - Ao processo de obtenção de uma sentença a partir da gramática dá-se o nome de **derivação da sentença**.

Gramáticas

❑ Gramáticas

- Formalmente as gramáticas, são caracterizadas como quádruplas ordenadas

$$G = (V_n, V_t, P, S)$$

- onde:
  - ✓ **Vn representa o vocabulário não terminal (variáveis)** da gramática. Este vocabulário corresponde ao conjunto de todos os símbolos dos quais a gramática se vale para definir as leis de formação das sentenças da linguagem.

Gramáticas

Gramáticas

- Formalmente as gramáticas, são caracterizadas como quádruplas ordenadas

G = (Vn, Vt, P, S)

- onde:
  - ✓ **Vt** é o **vocabulário terminal**, contendo os símbolos que constituem as sentenças da linguagem. Dá-se o nome de terminais aos elementos de **Vt**.

Gramáticas

❑ Gramáticas

- Formalmente as gramáticas, são caracterizadas como quádruplas ordenadas

G = (Vn, Vt, P, S)

- onde:
  - ✓ **P** são as **regras de produção**, que definem o conjunto de **todas as leis de formação** utilizadas pela gramática para **definir a linguagem**. Para tanto, cada construção parcial, representada por um não-terminal, é definida como um conjunto de regras de formação relativas à definição do não-terminal a ela referente. A cada uma destas regras de formação que compõem o conjunto P dá-se o nome de produção da gramática

Gramáticas

❑ Gramáticas

Símbolos que derivam

Símbolos terminais

✓ Exemplo: **G = (Vn, Vt, P, S)**

G = ({S, A, B}, {a, b}, P, S)

Início da derivação

Regras de produção

➢ P:

S → AB
A → a
B → bB|b

1. Qual a linguagem produzida por esta gramática?
2. Faça a árvore de derivação para a cadeia **abbbbb**

Gramáticas

❑ Gramáticas

Símbolos que derivam

Símbolos terminais

✓ Exemplo: **G = (Vn, Vt, P, S)**

G = ({S, A, B}, {a, b}, P, S)

Início da derivação

Regras de produção

➢P:

S → AB
A → a
B → bB|b

1. Qual a linguagem produzida por esta gramática?

   Linguagem que começa com um símbolo ***a*** seguido de n símbolos **b's** (n≥1).

   Expressão regular que representa a linguagem → **abb***

2. Faça a árvore de derivação para a cadeia **abbbbb**

Gramáticas

❑ Árvore de derivação

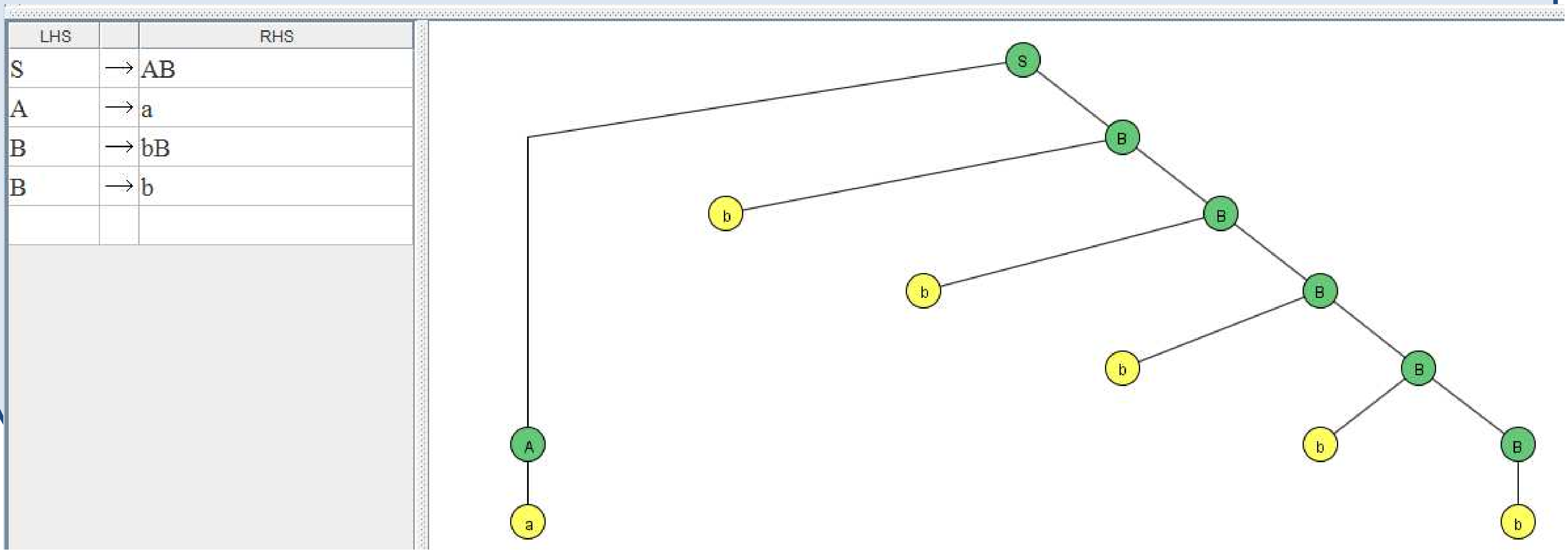
- a raiz tem como rótulo o símbolo inicial S da gramática.
- a cada nó rotulado por um **não-terminal** A corresponde uma regra de A. Se a regra for $A \rightarrow X_1X_2 \ldots X_m$, os filhos do nó são rotulados, da esquerda para a direita, por $X_1, X_2, \ldots, X_m$. (cada um dos X_i pode ser um terminal ou um não-terminal.)
- um nó rotulado por um terminal é sempre uma folha da árvore, e não tem filhos.

Gramáticas

- **Árvore de derivação para a gramática e cadeia** *abbbbb*

G = ({S, A, B}, {a, b}, P, S)

P:{
S → AB
A → a
B → bB|b
}



Gramáticas

1. Duas gramáticas G1 e G2 são **equivalentes** se produzem a mesma linguagem

L(G1) = L(G2)

2. Uma sentença é **ambígua** se existem duas ou mais sequências de derivação que a define.

3. Uma gramática é **ambígua** se possui alguma sentença ambígua.

G: $S \rightarrow AB$
$A \rightarrow AA \mid B \mid a$
$B \rightarrow Bcd \mid A$

– Verifique se x= aaacd é ambígua.

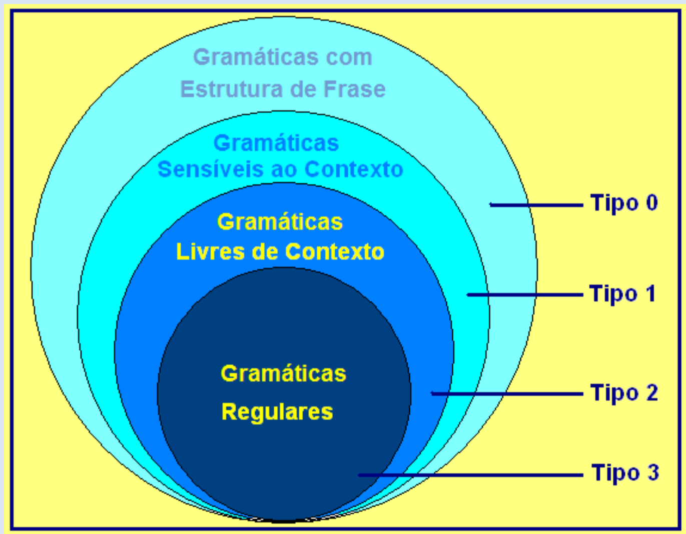
É ambígua → permite derivação mais à direita e mais à esquerda

Classes Gramaticais

- Conforme as restrições impostas ao formato das produções de uma gramática, a classe de linguagens que tal gramática gera varia correspondentemente.

- A teoria mostra que há quatro classes de gramáticas capazes de gerar quatro classes correspondentes de linguagens, de acordo com a denominada ***Hierarquia de Chomsky***.

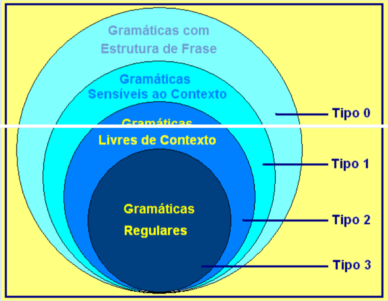
  a. **Gramáticas com Estrutura de Frase ou Tipo 0**
  b. **Gramáticas Sensíveis ao Contexto ou Tipo 1**
  c. **Gramáticas Livres de Contexto ou Tipo 2**
  d. **Gramáticas Regulares ou Tipo 3**



Classes Gramaticais

a. Gramáticas com Estrutura de Frase ou Tipo 0

- São aquelas às quais nenhuma restrição é imposta.
  - Exemplo de reconhecedor: Máquina de Turing com fita de entrada infinita



- Produções da forma

  $\alpha \rightarrow \beta$

  Onde: $\alpha \in \mathbf{(Vn \cup Vt)^{+}}$

  $\beta \in \mathbf{(Vn \cup Vt)^{*}}$

G = (Vn, Vt, P, S)

  - Lado esquerdo da regra de produção pode conter N símbolos (terminais ou não terminais);
  - Lado direito da regra de produção pode conter N símbolos (terminais ou não terminais ou **vazio**);

Classes Gramaticais

a. Gramáticas com Estrutura de Frase ou Tipo 0

- Exemplo de GEF:

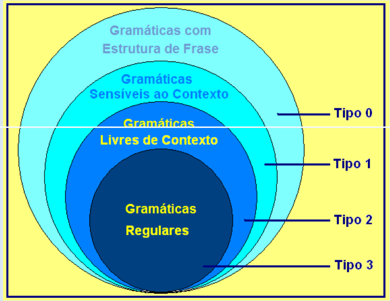
  G = ({A, B, C}, {a, b}, P, A)

  P: A → BC

  BC → CB

  B → b

  C → a



- Qual a linguagem gerada?
  - L(G) = **?**

Classes Gramaticais

a. Gramáticas com Estrutura de Frase ou Tipo 0

- Exemplo de GEF:

  $G = (\{A, B, C\}, \{a, b\}, P, A)$

  P: $A \rightarrow BC$

  $BC \rightarrow CB$

  $B \rightarrow b$

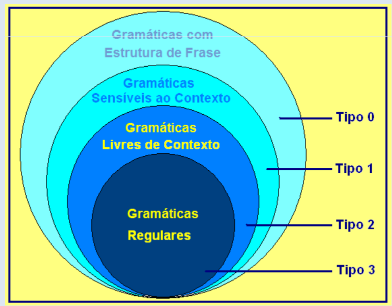
  $C \rightarrow a$

- Qual a linguagem gerada?
  - $L(G) = \{ba, ab\}$

Classes Gramaticais

b. Gramáticas Sensíveis ao Contexto ou Tipo 1

- Restrição: nenhuma substituição pode reduzir o comprimento da forma sentencial à qual a substituição é aplicada.



- Produções da forma

$\alpha \rightarrow \beta$

Onde: $\alpha \in \mathbf{(Vn \cup Vt)^+}$

$\beta \in \mathbf{(Vn \cup Vt)^+}$

$|\alpha| \leq |\beta|$

Classes Gramaticais

b. Gramáticas Sensíveis ao Contexto ou Tipo 1

$\alpha \rightarrow \beta$

Onde: $\alpha \in \mathbf{(Vn \cup Vt)^+}$

$\beta \in \mathbf{(Vn \cup Vt)^+}$

$|\alpha| \leq |\beta|$

- Exemplo de GSC:

$G = (\{S, B, C\}, \{a, b, c\}, P, S)$

$P: S \rightarrow aSBC \mid aBC$

$CB \rightarrow HB$

$HB \rightarrow HC$

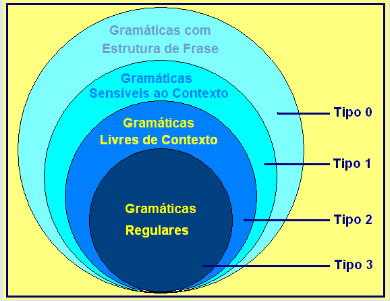
$HC \rightarrow BC$

$aB \rightarrow ab$

$bB \rightarrow bb$

$bC \rightarrow bc$

$cC \rightarrow cc$



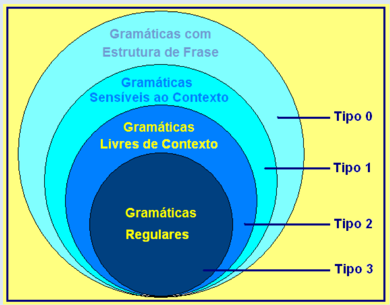
Faça a derivação (mais à esquerda ou mais à direita)

- Qual a linguagem gerada?
  - $L(G) = \{a^n b^n c^n \mid n \geq 1\}$

Classes Gramaticais

c. Gramáticas Livres de Contexto ou Tipo 2

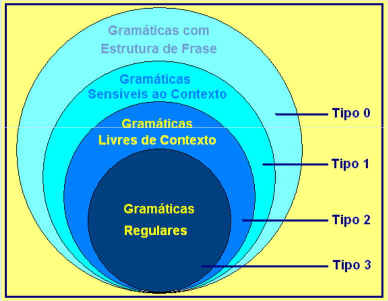
- As Gramáticas Livres de Contexto (GLC) ou do Tipo 2 são aquelas que no lado esquerdo da regra há apenas um símbolo não-terminal.



Classes Gramaticais

c. Gramáticas Livres de Contexto ou Tipo 2

- Qual a linguagem gerada para:



$$G = (\{S,A,B\},\{a,b\},P,S)$$

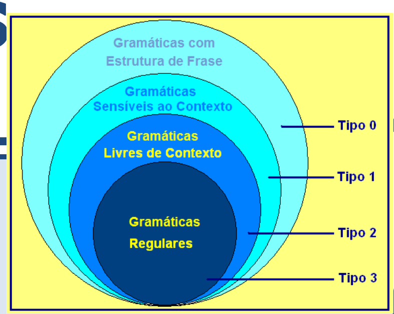
$$P: S \rightarrow AB$$

$$A \rightarrow aA \mid a$$

$$B \rightarrow bB \mid b$$

$$L(G)=\{a^n b^m \mid n, m \geq 1\}$$

Classes Gramaticais



d. Gramáticas Regulares ou Tipo 3

- Aplicando-se mais uma restrição sobre a forma das produções, pode-se criar uma nova classe de gramáticas, as **Gramáticas Regulares** (GR), de grande importância no estudo dos **compiladores** por possuírem propriedades adequadas para a obtenção de reconhecedores simples. Nas GRs, as produções são restritas às formas seguintes:

$A \rightarrow aB$ **ou** $A \rightarrow Ba$

$A \rightarrow a$

$A \rightarrow \varepsilon$

onde $A,B \in Vn$ e $a \in Vt$

no lado esquerdo da regra há apenas um símbolo não-terminal

Classes Gramaticais

d. Gramáticas Regulares ou Tipo 3

- Exemplo GR em EBNF:

  G = ({<Dig>, <Int>}, {+, -, 0, ..., 9}, P, <Int>)

  P: <Int> ::= +<Dig> | -<Dig>

  <Dig> ::= 0<Dig> | 1<Dig>|...| 9<Dig> | 0 | 1 | 2 |...|9

- Qual linguagem gerada?
  - L(G) = conj. números inteiros com sinal ±[0..9]

Na aula de hoje...

- Revisão: gramáticas
- **Relações em uma gramática: Cabeça, Último, Primeiro (*First*) e Seguinte (*Follow*)**
  - Capítulo 4 (seção 4.4.2) do livro ***Compiladores : Princípios, técnicas e ferramentas***

Relações em uma Gramática

- A construção de analisadores sintáticos é facilitada através de algumas funções associadas a gramática: CABEÇA, ÚLTIMO, PRIMEIRO*(First)* e Seguinte *(Follow)*.
  - Ideia: permitem escolher qual produção aplicar com base no próximo símbolo lido.

Cabeça

- É uma das mais simples de identificar
  - um de seus elementos é a cabeça do lado direito de uma regra
- Dada uma produção na forma

  $\alpha \rightarrow \beta\gamma$

  Onde: $\alpha \in$ **Vn**

  $\beta \in$ **(Vn $\cup$ Vt)**

  $\gamma \in$ **(Vn $\cup$ Vt)***
- **Cabeça (α) = β**

Exemplo

P: $S \rightarrow AB$
$A \rightarrow aA \mid a$
$B \rightarrow bB \mid b$

Cabeça (S) = ?
Cabeça (A) = ?
Cabeça (B) = ?

Cabeça

- É uma das mais simples de identificar
  - um de seus elementos (terminal ou não-terminal) é a cabeça do lado direito de uma regra
- Dada uma produção na forma

  $\alpha \rightarrow \beta\gamma$

  Onde: $\alpha \in$ **Vn**

  $\beta \in$ **(Vn $\cup$ Vt)**

  $\gamma \in$ **(Vn $\cup$ Vt)***
- **Cabeça (α) = β**

Exemplo

P: S → AB
A → aA | a
B → bB | b

Cabeça (S) = {A}
Cabeça (A) = {a}
Cabeça (B) = {b}

Último

- Relaciona um dado não terminal, existente do lado esquerdo de uma certa regra, com o último elemento que aparece do lado direito desta regra

- Dada uma produção na forma

  $\alpha \rightarrow \gamma\beta$

  Onde: $\alpha \in$ **Vn**

  $\beta \in$ **(Vn $\cup$ Vt)**

  $\gamma \in$ **(Vn $\cup$ Vt)***

- **Último (α) = β**

Exemplo

P: S → AB
A → aA | a
B → bB | b

Último (S) = ?
Último (A) = ?
Último (B) = ?

Último

- Relaciona um dado não terminal, existente do lado esquerdo de uma certa regra, com o último elemento (terminal ou não-terminal) que aparece do lado direito desta regra

- Dada uma produção na forma

  $\alpha \rightarrow \gamma\beta$

  Onde: $\alpha \in$ **Vn**

  $\beta \in$ **(Vn $\cup$ Vt)**

  $\gamma \in$ **(Vn $\cup$ Vt)***

- **Último (α) = β**

Exemplo

P: S → AB
A → aA | a
B → bB | b

Último (S) = {B}
Último (A) = {A,a}
Último (B) = {B,b}

Primeiro (*First*)

- Relação próxima a relação cabeça; entretanto, deve conter **somente terminais**
- Primeiro(A) = x, onde A produz x como seu símbolo mais à esquerda com n derivações, sendo A ϵ Vn e x ϵ Vt*
  - x pode ser a cadeia vazia

Exemplo

P: S → AB
A → aA | a
B → bB | b

Primeiro (S) = {a}
Primeiro (A) = {a}
Primeiro (B) = {b}

Primeiro (*First*)

- Para determinar PRIMEIRO(X):

1. Se **x** é um terminal, então PRIMEIRO (**x**) = {**x**}
2. Se **X** é não-terminal e X →aα é uma produção, então se acrescenta **a** ao conjunto PRIMEIRO de x
3. Se **X** → ε é uma produção ε deve ser adicionado ao conjunto PRIMEIRO de x
4. Se X→$Y_1Y_2...Y_k$ é uma produção, então todo i tal que todos $Y_1...Y_{i-1}$ são não-terminais e PRIMEIRO(Y_j) contém ε, onde j=1,2...i-1. acrescente todo símbolo diferente de ε de PRIMEIRO(Y_j) a PRIMEIRO(X). Se $\varepsilon \in$ PRIMEIRO(X), para todo i=1,2..k. então acrescente ε a PRIMEIRO(X).

Primeiro (*First*)

- Exemplo

E → TE′
E′ → +TE′ | ε
T → FT′
T′ → *FT′ | ε
F → (E) | id

Primeiro (E) = {?}
Primeiro (E′) = {?}
Primeiro (T) = {?}
Primeiro (T′) = {?}
Primeiro (F) = {?}

Primeiro (*First*)

- Exemplo

E → TE'
E' → +TE' | ε
T → FT'
T' → *FT' | ε
F → (E) | id

Primeiro (E) = Primeiro(T) = Primeiro(F) ={(, id}
Primeiro (E') = {+, ε}
Primeiro (T) = Primeiro(F) = {(, id}
Primeiro (T') = {*, ε}
Primeiro (F) = {(, id}

F deriva em ε?
R: Não, então primeiro(T) = primeiro(F) = {**(**, **id**}
Se F derivasse em ε era preciso incluir o primeiro(T') em primeiro(T)

Primeiro (*First*)

- Exemplo 2

E → TE′
E′ → +TE′ | ε
T → FT′
H → E′T
T′ → *FT′ | ε
F → (E) | id

Primeiro (H) = {?}

Se fosse incluída esta regra na gramática como ficaria o primeiro(H)?

Primeiro (*First*)

- Exemplo 2

$E \rightarrow TE'$
$E' \rightarrow +TE' \mid \varepsilon$
$T \rightarrow FT'$
$H \rightarrow E'T$
$T' \rightarrow *FT' \mid \varepsilon$
$F \rightarrow (E) \mid id$

Primeiro (H) = Primeiro(E')∪Primeiro(T)
= { +, (, id }

E' deriva em ε?

R: Sim, então incluir o primeiro(T) em primeiro(H). Se primeiro(T) e primeiro(E') contem ε então incluir ε em primeiro(H), caso contrário remover ε do primeiro(H)

Regra 4
slide 33

Seguinte (*Follow*)

- Se A é um *não-terminal*, Seguinte(A) é o conjunto de terminais que podem figurar imediatamente à direita de A em alguma forma sentencial

- Seguinte(A)=**x** para a regra $S \rightarrow \alpha A \beta$ e primeiro(β)=**x**

  Onde: $A \in \mathbf{Vn}$

  $x \in \mathbf{Vt}^*$

  α e $\beta \in (\mathbf{Vn} \cup \mathbf{Vt})^*$

Seguinte (*Follow*)

- Para determinar Seguinte(A):

1. Colocar $\$$ em Seguinte(S) se S é o símbolo de partida - $\$$ é o marcador de fim de entrada durante a análise sintática

2. Se existe uma produção $A \rightarrow \alpha B \beta$ e $\beta \not\Rightarrow \varepsilon$ então tudo que estiver em PRIMEIRO(β), exceto ε, deve ser adicionado em Seguinte(B)

3. Se existe uma produção $A \rightarrow \alpha B$ ou $A \rightarrow \alpha B \beta$ onde PRIMEIRO(β) contem ε ($\beta \rightarrow \varepsilon$), então tudo que está em Seguinte(A) está em Seguinte(B)

Seguinte (*Follow*)

E → TE'
E' → +TE' | ε
T → FT'
T' → *FT' | ε
F → (E) | id

Primeiro (E) = Primeiro(T) = Primeiro(F) = {(, id}
Primeiro (E') = {+, ε}
Primeiro (T) = Primeiro(F) = {(, id}
Primeiro (T') = {*, ε}
Primeiro (F) = {(, id}

Seguinte (*Follow*)

E → TE′
E′ → +TE′ | ε
T → FT′
T′ → *FT′ | ε
F → (E) | id

Primeiro (E) = Primeiro(T) = Primeiro(F) = {(, id}
Primeiro (E’) = {+, ε}
Primeiro (T) = Primeiro(F) = {(, id}
Primeiro (T’) = {*, ε}
Primeiro (F) = {(, id}

Seguinte (*Follow*)

$E \rightarrow TE'$

$E' \rightarrow +TE' \mid \varepsilon$

$T \rightarrow FT'$

$T' \rightarrow *FT' \mid \varepsilon$

$F \rightarrow (E) \mid id$

Primeiro (E) = Primeiro(T) = Primeiro(F) = {(, id}

Primeiro (E’) = {+, ε}

Primeiro (T) = Primeiro(F) = {(, id}

Primeiro (T’) = {*, ε}

Primeiro (F) = {(, id}

Seguinte (*Follow*)

E → TE'
E' → +TE' | ε
T → FT'
T' → *FT' | ε
F → (E) | id

Primeiro (E) = Primeiro(T) = Primeiro(F) = {(, id}
Primeiro (E') = {+, ε}
Primeiro (T) = Primeiro(F) = {(, id}
Primeiro (T') = {*, ε}
Primeiro (F) = {(, id}

Seguinte (*Follow*)

E $\rightarrow$ TE'
E' $\rightarrow$ +TE' | ε
T $\rightarrow$ FT'
T' $\rightarrow$ *FT' | ε
F $\rightarrow$ (E) | id

Primeiro (E) = Primeiro(T) = Primeiro(F) = {(, id}
Primeiro (E') = {+, ε}
Primeiro (T) = Primeiro(F) = {(, id}
Primeiro (T') = {*, ε}
Primeiro (F) = {(, id}

Exercícios

Encontre os conjuntos Primeiro(*First)* para as gramáticas abaixo

Exercícios

Encontre os conjuntos Primeiro(*First)* para as gramáticas abaixo

Exercícios

Encontre os conjuntos Primeiro(*First*) para as gramáticas abaixo

Exercícios

Encontre os conjuntos Primeiro(*First)* para as gramáticas abaixo

Exercícios

Encontre os conjuntos Primeiro(*First)* para as gramáticas abaixo

Exercícios

Encontre os conjuntos Primeiro(*First)* para as gramáticas abaixo

Exercícios

Encontre os conjuntos Primeiro(*First)* para as gramáticas abaixo

Exercícios

Encontre os conjuntos Primeiro(*First)* para as gramáticas abaixo

Primeiro(S)= {}
Primeiro(A)= {}
Primeiro(B)= {}
Primeiro(C)= {}

b) $S \rightarrow ABd$
$A \rightarrow aA \mid \varepsilon$
$B \rightarrow bB \mid cA \mid AC$
$C \rightarrow cB \mid \varepsilon$

Quando tem uma regra do tipo A → BCD, o ε só entra no Primeiro(A) se ele puder ser gerado por B, C e D também.

Exercícios

Encontre os conjuntos Primeiro(*First*) para as gramáticas abaixo

b) $S \rightarrow ABd$
$A \rightarrow aA \mid \varepsilon$
$B \rightarrow bB \mid cA \mid AC$
$C \rightarrow cB \mid \varepsilon$

Primeiro(S)= {}
Primeiro(A)= {}
Primeiro(B)= {}
Primeiro(C)= {c, ε}

Quando tem uma regra do tipo A → BCD, o ε só entra no Primeiro(A) se ele puder ser gerado por B, C e D também.

Exercícios

Encontre os conjuntos Primeiro(*First*) para as gramáticas abaixo

Primeiro(S)= {}
Primeiro(A)= {}
Primeiro(B)= {b, c, **U** Primeiro(A)}
Primeiro(C)= {c, ε}

b) $S \to ABd$
$A \to aA \mid \varepsilon$
$B \to bB \mid cA \mid AC$
$C \to cB \mid \varepsilon$

Quando tem uma regra do tipo A → BCD, o ε só entra no Primeiro(A) se ele puder ser gerado por B, C e D também.

Exercícios

Encontre os conjuntos Primeiro(*First)* para as gramáticas abaixo

b) $S \rightarrow ABd$
$A \rightarrow aA \mid \varepsilon$
$B \rightarrow bB \mid cA \mid AC$
$C \rightarrow cB \mid \varepsilon$

Primeiro(S)= {}
Primeiro(A)= {a, ε}
Primeiro(B)= {b, c, **U** Primeiro(A)}
Primeiro(C)= {c, ε}

Quando tem uma regra do tipo A → BCD, o ε só entra no Primeiro(A) se ele puder ser gerado por B, C e D também.

Exercícios

Encontre os conjuntos Primeiro(*First*) para as gramáticas abaixo

Primeiro(S)= {}
Primeiro(A)= {a, ε}
Primeiro(B)= {b, c, **U** Primeiro(A)} = {b, c, a, ε}
Primeiro(C)= {c, ε}

b) $S \rightarrow ABd$
$A \rightarrow aA \mid \varepsilon$
$B \rightarrow bB \mid cA \mid AC$
$C \rightarrow cB \mid \varepsilon$

Quando tem uma regra do tipo A → BCD, o ε só entra no Primeiro(A) se ele puder ser gerado por B, C e D também.

Exercícios

Encontre os conjuntos Primeiro(*First)* para as gramáticas abaixo

b) $S \rightarrow ABd$
$A \rightarrow aA \mid \varepsilon$
$B \rightarrow bB \mid cA \mid AC$
$C \rightarrow cB \mid \varepsilon$

Primeiro(S)= {Primeiro(A) **U** Primeiro(B) **U** d}
Primeiro(A)= {a, ε}
Primeiro(B)= {b, c, **U** Primeiro(A)} = {b, c, a, ε}
Primeiro(C)= {c, ε}

Primeiro(A) gera **ε,** então Primeiro(B) entra em Primeiro(S) e Primeiro(B) gera **ε,** então **d** também entra em Primeiro(S)

Quando tem uma regra do tipo A → BCD, o **ε** só entra no Primeiro(A) se ele puder ser gerado por B, C e D também.

Exercícios

Encontre os conjuntos Primeiro(*First*) para as gramáticas abaixo

b) $S \rightarrow ABd$
$A \rightarrow aA \mid \varepsilon$
$B \rightarrow bB \mid cA \mid AC$
$C \rightarrow cB \mid \varepsilon$

Primeiro(S)= {a, b, c, d, ε}
Primeiro(A)= {a, ε}
Primeiro(B)= {b, c, a, ε }
Primeiro(C)= {c, ε}

Quando tem uma regra do tipo A → BCD, o ε só entra no Primeiro(A) se ele puder ser gerado por B, C e D também.

Exercícios

Encontre os conjuntos Primeiro(*First)* para as gramáticas abaixo

Primeiro(S)= {}
Primeiro(A)= {}
Primeiro(B)= {}
Primeiro(C)= {}
Primeiro(D)= {}

Exercícios

Encontre os conjuntos Primeiro(*First)* para as gramáticas abaixo

Primeiro(S)= {a, g, c, b, f, ε}
Primeiro(A)= {a, g, c, ε}
Primeiro(B)= {b, f, ε}
Primeiro(C)= {c}
Primeiro(D)= {g, c, ε}

Exercícios

Encontre os conjuntos Primeiro(*First*) para as gramáticas abaixo

d) $S \rightarrow aA \mid bB$
$A \rightarrow aAS \mid BD$
$B \rightarrow bB \mid fAC \mid \varepsilon$
$C \rightarrow cC \mid Dd$
$D \rightarrow gD \mid C \mid \varepsilon$

Primeiro(S)= {}
Primeiro(A)= {}
Primeiro(B)= {}
Primeiro(C)= {}
Primeiro(D)= {}

Quando tem uma regra do tipo A → BCD, o ε só entra no Primeiro(A) se ele puder ser gerado por B, C e D também.

Exercícios

Encontre os conjuntos Primeiro(*First*) para as gramáticas abaixo

d) $S \rightarrow aA \mid bB$
$A \rightarrow aAS \mid BD$
$B \rightarrow bB \mid fAC \mid \varepsilon$
$C \rightarrow cC \mid Dd$
$D \rightarrow gD \mid C \mid \varepsilon$

Primeiro(S)= {a,b}
Primeiro(A)= {a, b, f, g, c, d, ε}
Primeiro(B)= {b, f, ε}
Primeiro(C)= {g, c, d, ε}
Primeiro(D)= {c, g, d, ε }

Quando tem uma regra do tipo A → BCD, o ε só entra no Primeiro(A) se ele puder ser gerado por B, C e D também.

Exercícios

Encontre os conjuntos Seguinte(*Follow)* para as gramáticas abaixo

Exercícios

Encontre os conjuntos Seguinte(*Follow)* para as gramáticas abaixo

Exercícios

Encontre os conjuntos Seguinte(*Follow)* para as gramáticas abaixo

Exercícios

Encontre os conjuntos Seguinte(*Follow)* para as gramáticas abaixo

Exercícios

Encontre os conjuntos Seguinte(*Follow)* para as gramáticas abaixo

Exercícios

- Encontre os conjuntos *First* e *Follow* para a gramática LALG.