

Příklad:

$\langle \text{STMT} \rangle ::= \langle \text{IF-STMT} \rangle \mid \langle \text{WHILE-STMT} \rangle \mid \langle \text{BEGIN-STMT} \rangle \mid \langle \text{ASSG-STMT} \rangle$

$\langle \text{IF-STMT} \rangle ::= \mathbf{if} \langle \text{BOOL-EXPR} \rangle \mathbf{then} \langle \text{STMT} \rangle \mathbf{else} \langle \text{STMT} \rangle$

$\langle \text{WHILE-STMT} \rangle ::= \mathbf{while} \langle \text{BOOL-EXPR} \rangle \mathbf{do} \langle \text{STMT} \rangle$

$\langle \text{BEGIN-STMT} \rangle ::= \mathbf{begin} \langle \text{STMT-LIST} \rangle \mathbf{end}$

$\langle \text{STMT-LIST} \rangle ::= \langle \text{STMT} \rangle \mid \langle \text{STMT} \rangle ; \langle \text{STMT-LIST} \rangle$

$\langle \text{ASSG-STMT} \rangle ::= \langle \text{VAR} \rangle := \langle \text{ARITH-EXPR} \rangle$

$\langle \text{BOOL-EXPR} \rangle ::= \langle \text{ARITH-EXPR} \rangle \langle \text{COMPARE-OP} \rangle \langle \text{ARITH-EXPR} \rangle$

$\langle \text{COMPARE-OP} \rangle ::= < \mid > \mid \leq \mid \geq \mid = \mid \neq$

$\langle \text{ARITH-EXPR} \rangle ::= \langle \text{VAR} \rangle \mid \langle \text{CONST} \rangle \mid (\langle \text{ARITH-EXPR} \rangle \langle \text{ARITH-OP} \rangle \langle \text{ARITH-EXPR} \rangle)$

$\langle \text{ARITH-OP} \rangle ::= + \mid - \mid * \mid /$

$\langle \text{CONST} \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

$\langle \text{VAR} \rangle ::= a \mid b \mid c \mid \dots \mid x \mid y \mid z$

Symbolům, které mají v předchozím příkladě tvar $\langle xxx \rangle$, se říká **neterminální symboly (neterminály)**.

Pravidla popisují, jaké řetězce může daný neterminál reprezentovat.

Z neterminálu $\langle STMT \rangle$ můžeme například dostat text

```
while  $x \leq y$  do begin  $x := (x + 1)$ ;  $y := (y - 1)$  end
```

Poznámka: Výše použité notaci se říká Backus-Naurova forma (BNF).

while $x \leq y$ **do begin** $x := (x + 1); y := (y - 1)$ **end**

while $x \leq y$ **do begin** $x := (x + 1); y := (y - 1)$ **end**

$\langle \text{STMT} \rangle$

while $x \leq y$ **do begin** $x := (x + 1); y := (y - 1)$ **end**

$\langle \text{STMT} \rangle$

$\langle \text{WHILE-STMT} \rangle$

while $x \leq y$ **do begin** $x := (x + 1); y := (y - 1)$ **end**

$\langle \text{STMT} \rangle$

$\langle \text{WHILE-STMT} \rangle$

while $\langle \text{BOOL-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $x \leq y$ **do begin** $x := (x + 1); y := (y - 1)$ **end**

$\langle \text{STMT} \rangle$

$\langle \text{WHILE-STMT} \rangle$

while $\langle \text{BOOL-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{ARITH-EXPR} \rangle \langle \text{COMPARE-OP} \rangle \langle \text{ARITH-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $x \leq y$ **do begin** $x := (x + 1); y := (y - 1)$ **end**

⟨STMT⟩

⟨WHILE-STMT⟩

while ⟨BOOL-EXPR⟩ **do** ⟨STMT⟩

while ⟨ARITH-EXPR⟩⟨COMPARE-OP⟩⟨ARITH-EXPR⟩ **do** ⟨STMT⟩

while ⟨VAR⟩⟨COMPARE-OP⟩⟨ARITH-EXPR⟩ **do** ⟨STMT⟩

while $x \leq y$ **do begin** $x := (x + 1); y := (y - 1)$ **end**

$\langle \text{STMT} \rangle$

$\langle \text{WHILE-STMT} \rangle$

while $\langle \text{BOOL-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{ARITH-EXPR} \rangle \langle \text{COMPARE-OP} \rangle \langle \text{ARITH-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{VAR} \rangle \langle \text{COMPARE-OP} \rangle \langle \text{ARITH-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{VAR} \rangle \leq \langle \text{ARITH-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $x \leq y$ **do begin** $x := (x + 1); y := (y - 1)$ **end**

$\langle \text{STMT} \rangle$

$\langle \text{WHILE-STMT} \rangle$

while $\langle \text{BOOL-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{ARITH-EXPR} \rangle \langle \text{COMPARE-OP} \rangle \langle \text{ARITH-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{VAR} \rangle \langle \text{COMPARE-OP} \rangle \langle \text{ARITH-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{VAR} \rangle \leq \langle \text{ARITH-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{VAR} \rangle \leq \langle \text{VAR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $x \leq y$ **do begin** $x := (x + 1); y := (y - 1)$ **end**

$\langle \text{STMT} \rangle$

$\langle \text{WHILE-STMT} \rangle$

while $\langle \text{BOOL-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{ARITH-EXPR} \rangle \langle \text{COMPARE-OP} \rangle \langle \text{ARITH-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{VAR} \rangle \langle \text{COMPARE-OP} \rangle \langle \text{ARITH-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{VAR} \rangle \leq \langle \text{ARITH-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{VAR} \rangle \leq \langle \text{VAR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $x \leq \langle \text{VAR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $x \leq y$ **do begin** $x := (x + 1); y := (y - 1)$ **end**

$\langle \text{STMT} \rangle$

$\langle \text{WHILE-STMT} \rangle$

while $\langle \text{BOOL-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{ARITH-EXPR} \rangle \langle \text{COMPARE-OP} \rangle \langle \text{ARITH-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{VAR} \rangle \langle \text{COMPARE-OP} \rangle \langle \text{ARITH-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{VAR} \rangle \leq \langle \text{ARITH-EXPR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $\langle \text{VAR} \rangle \leq \langle \text{VAR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $x \leq \langle \text{VAR} \rangle$ **do** $\langle \text{STMT} \rangle$

while $x \leq y$ **do** $\langle \text{STMT} \rangle$

while $x \leq y$ **do begin** $x := (x + 1); y := (y - 1)$ **end**

\langle STMT \rangle

\langle WHILE-STMT \rangle

while \langle BOOL-EXPR \rangle **do** \langle STMT \rangle

while \langle ARITH-EXPR \rangle \langle COMPARE-OP \rangle \langle ARITH-EXPR \rangle **do** \langle STMT \rangle

while \langle VAR \rangle \langle COMPARE-OP \rangle \langle ARITH-EXPR \rangle **do** \langle STMT \rangle

while \langle VAR $\rangle \leq \langle$ ARITH-EXPR \rangle **do** \langle STMT \rangle

while \langle VAR $\rangle \leq \langle$ VAR \rangle **do** \langle STMT \rangle

while $x \leq \langle$ VAR \rangle **do** \langle STMT \rangle

while $x \leq y$ **do** \langle STMT \rangle

while $x \leq y$ **do** \langle BEGIN-STMT \rangle

while $x \leq y$ **do begin** $x := (x + 1); y := (y - 1)$ **end**

\langle STMT \rangle

\langle WHILE-STMT \rangle

while \langle BOOL-EXPR \rangle **do** \langle STMT \rangle

while \langle ARITH-EXPR \rangle \langle COMPARE-OP \rangle \langle ARITH-EXPR \rangle **do** \langle STMT \rangle

while \langle VAR \rangle \langle COMPARE-OP \rangle \langle ARITH-EXPR \rangle **do** \langle STMT \rangle

while \langle VAR $\rangle \leq \langle$ ARITH-EXPR \rangle **do** \langle STMT \rangle

while \langle VAR $\rangle \leq \langle$ VAR \rangle **do** \langle STMT \rangle

while $x \leq \langle$ VAR \rangle **do** \langle STMT \rangle

while $x \leq y$ **do** \langle STMT \rangle

while $x \leq y$ **do** \langle BEGIN-STMT \rangle

while $x \leq y$ **do begin** \langle STMT-LIST \rangle **end**

while $x \leq y$ **do begin** $x := (x + 1); y := (y - 1)$ **end**

\langle STMT \rangle

\langle WHILE-STMT \rangle

while \langle BOOL-EXPR \rangle **do** \langle STMT \rangle

while \langle ARITH-EXPR \rangle \langle COMPARE-OP \rangle \langle ARITH-EXPR \rangle **do** \langle STMT \rangle

while \langle VAR \rangle \langle COMPARE-OP \rangle \langle ARITH-EXPR \rangle **do** \langle STMT \rangle

while \langle VAR $\rangle \leq \langle$ ARITH-EXPR \rangle **do** \langle STMT \rangle

while \langle VAR $\rangle \leq \langle$ VAR \rangle **do** \langle STMT \rangle

while $x \leq \langle$ VAR \rangle **do** \langle STMT \rangle

while $x \leq y$ **do** \langle STMT \rangle

while $x \leq y$ **do** \langle BEGIN-STMT \rangle

while $x \leq y$ **do begin** \langle STMT-LIST \rangle **end**

...

Formálně je **bezkontextová gramatika** definována jako čtveřice

$$G = (\Pi, \Sigma, S, P)$$

kde:

- Π je konečná množina **neterminálních symbolů (neterminálů)**
- Σ je konečná množina **terminálních symbolů (terminálů)**,
přičemž $\Pi \cap \Sigma = \emptyset$
- $S \in \Pi$ je **počáteční neterminál**
- $P \subseteq \Pi \times (\Pi \cup \Sigma)^*$ je konečná množina **přepisovacích pravidel**

Poznámky:

- Pro označení neterminálních symbolů budeme používat velká písmena A, B, C, \dots
- Pro označení terminálních symbolů budeme používat malá písmena a, b, c, \dots nebo číslice $0, 1, 2, \dots$
- Pro označení řetězců z $(\Pi \cup \Sigma)^*$ budeme používat malá písmena řecké abecedy $\alpha, \beta, \gamma, \dots$
- Místo zápisu (A, α) budeme pro pravidla používat zápis

$$A \rightarrow \alpha$$

A – levá strana pravidla

α – pravá strana pravidla

Příklad: Gramatika $G = (\Pi, \Sigma, S, P)$, kde

- $\Pi = \{A, B, C\}$
- $\Sigma = \{a, b\}$
- $S = A$
- P obsahuje pravidla

$$A \rightarrow aBBb$$

$$A \rightarrow AaA$$

$$B \rightarrow \varepsilon$$

$$B \rightarrow bCA$$

$$C \rightarrow AB$$

$$C \rightarrow a$$

$$C \rightarrow b$$

Poznámka: Pokud máme více pravidel se stejnou levou stranou, jako třeba

$$A \rightarrow \alpha_1 \qquad A \rightarrow \alpha_2 \qquad A \rightarrow \alpha_3$$

můžeme je stručněji zapsat jako

$$A \rightarrow \alpha_1 \mid \alpha_2 \mid \alpha_3$$

Například pravidla dříve uvedené gramatiky můžeme zapsat jako

$$\begin{aligned} A &\rightarrow aBBb \mid AaA \\ B &\rightarrow \varepsilon \mid bCA \\ C &\rightarrow AB \mid a \mid b \end{aligned}$$

Řetězce z $(\Pi \cup \Sigma)^*$ nazýváme **větné formy**.

Na větných formách definujeme relaci $\Rightarrow \subseteq (\Pi \cup \Sigma)^* \times (\Pi \cup \Sigma)^*$ takovou, že

$$\alpha \Rightarrow \alpha'$$

právě když $\alpha = \beta_1 A \beta_2$ a $\alpha' = \beta_1 \gamma \beta_2$ pro nějaká $\beta_1, \beta_2, \gamma \in (\Pi \cup \Sigma)^*$ a $A \in \Pi$, kde $(A \rightarrow \gamma) \in P$.

Příklad: Jestliže $(B \rightarrow bCA) \in P$, pak

$$aCBbA \Rightarrow aCbCABA$$

Poznámka: Neformálně řečeno zápis $\alpha \Rightarrow \alpha'$ znamená, že z větné formy α je možné jedním krokem odvodit větnou formu α' , a to tak, že výskyt nějakého neterminálu A v α nahradíme pravou stranou nějakého pravidla $A \rightarrow \alpha$, kde se A vyskytuje na levé straně.

Řetězce z $(\Pi \cup \Sigma)^*$ nazýváme **větné formy**.

Na větných formách definujeme relaci $\Rightarrow \subseteq (\Pi \cup \Sigma)^* \times (\Pi \cup \Sigma)^*$ takovou, že

$$\alpha \Rightarrow \alpha'$$

právě když $\alpha = \beta_1 A \beta_2$ a $\alpha' = \beta_1 \gamma \beta_2$ pro nějaká $\beta_1, \beta_2, \gamma \in (\Pi \cup \Sigma)^*$ a $A \in \Pi$, kde $(A \rightarrow \gamma) \in P$.

Příklad: Jestliže $(B \rightarrow bCA) \in P$, pak

$$aC\underline{B}bA \Rightarrow aC\underline{bCA}bA$$

Poznámka: Neformálně řečeno zápis $\alpha \Rightarrow \alpha'$ znamená, že z větné formy α je možné jedním krokem odvodit větnou formu α' , a to tak, že výskyt nějakého neterminálu A v α nahradíme pravou stranou nějakého pravidla $A \rightarrow \alpha$, kde se A vyskytuje na levé straně.

Derivace délky n větné formy α' z větné formy α je posloupnost větných forem

$$\beta_0, \beta_1, \beta_2, \dots, \beta_n$$

takových, že

- $\alpha = \beta_0$
- $\beta_{i-1} \Rightarrow \beta_i$ pro všechna $i \in \{1, 2, \dots, n\}$
- $\alpha' = \beta_n$

což můžeme stručněji zapsat jako

$$\alpha = \beta_0 \Rightarrow \beta_1 \Rightarrow \beta_2 \Rightarrow \dots \Rightarrow \beta_{n-1} \Rightarrow \beta_n = \alpha'$$

Skutečnost, že pro dané n existuje nějaká derivace délky n větné formy α' z větné formy α , označujeme zápisem

$$\alpha \Rightarrow^n \alpha'$$

Skutečnost, že existuje nějaká derivace (nějaké délky n , kde $n \geq 0$) větné formy α' z větné formy α , označujeme zápisem

$$\alpha \Rightarrow^* \alpha'$$

Poznámka: Relace \Rightarrow^* je reflexivním a tranzitivním uzávěrem relace \Rightarrow (tj. nejmenší reflexivní a tranzitivní relací obsahující relaci \Rightarrow).

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$B \rightarrow \varepsilon \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$B \rightarrow \varepsilon \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

A

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$\underline{A} \rightarrow aBBb \mid AaA$$

$$B \rightarrow \varepsilon \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

A

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$\begin{aligned} A &\rightarrow aBBb \mid AaA \\ B &\rightarrow \varepsilon \mid bCA \\ C &\rightarrow AB \mid a \mid b \end{aligned}$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb$$

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$B \rightarrow \varepsilon \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb$$

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$\underline{B} \rightarrow \varepsilon \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow a\underline{B}Bb$$

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$\underline{B} \rightarrow \varepsilon \mid \underline{bCA}$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow a\underline{B}Bb \Rightarrow a\underline{bCA}Bb$$

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$B \rightarrow \varepsilon \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abCABb$$

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$\begin{aligned} A &\rightarrow aBBb \mid AaA \\ B &\rightarrow \varepsilon \mid bCA \\ C &\rightarrow AB \mid a \mid b \end{aligned}$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abC\underline{A}Bb$$

Bezkontextové gramatiky

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$\begin{aligned} A &\rightarrow \underline{aBBb} \mid AaA \\ B &\rightarrow \varepsilon \mid bCA \\ C &\rightarrow AB \mid a \mid b \end{aligned}$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abC\underline{A}Bb \Rightarrow abC\underline{aBBb}Bb$$

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$B \rightarrow \varepsilon \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb$$

Bezkontextové gramatiky

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$\underline{B} \rightarrow \varepsilon \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaB\underline{B}bBb$$

Bezkontextové gramatiky

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$\underline{B} \rightarrow \underline{\epsilon} \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaB\underline{B}bBb \Rightarrow abCaBbBb$$

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$B \rightarrow \varepsilon \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb$$

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$B \rightarrow \varepsilon \mid bCA$$

$$\underline{C} \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow ab\underline{C}aBbBb$$

Bezkontextové gramatiky

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$B \rightarrow \varepsilon \mid bCA$$

$$\underline{C} \rightarrow AB \mid a \mid \underline{b}$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow ab\underline{C}aBbBb \Rightarrow ab\underline{b}aBbBb$$

Bezkontextové gramatiky

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$B \rightarrow \varepsilon \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb \Rightarrow abbaBbBb$$

Bezkontextové gramatiky

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$\underline{B} \rightarrow \varepsilon \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb \Rightarrow abbaBb\underline{B}b$$

Bezkontextové gramatiky

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$\underline{B} \rightarrow \underline{\epsilon} \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb \Rightarrow abbaBb\underline{B}b \Rightarrow abbaBbb$$

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$B \rightarrow \varepsilon \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb \Rightarrow abbaBbBb \Rightarrow abbaBbb$$

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$\underline{B} \rightarrow \varepsilon \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb \Rightarrow abbaBbBb \Rightarrow abba\underline{B}bb$$

Bezkontextové gramatiky

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$\underline{B} \rightarrow \underline{\epsilon} \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb \Rightarrow abbaBbBb \Rightarrow abba\underline{B}bb \Rightarrow abbabb$$

Bezkontextové gramatiky

Jazyk $L(G)$ generovaný gramatikou $G = (\Pi, \Sigma, S, P)$ je množina všech slov v abecedě Σ , která lze odvodit nějakou derivací z počátečního neterminálu S pomocí pravidel z P , tj.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

Příklad: $G = (\Pi, \Sigma, A, P)$, kde $\Pi = \{A, B, C\}$, $\Sigma = \{a, b\}$ a P obsahuje pravidla

$$A \rightarrow aBBb \mid AaA$$

$$B \rightarrow \varepsilon \mid bCA$$

$$C \rightarrow AB \mid a \mid b$$

$abbabb \in L(G)$ protože:

$$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb \Rightarrow abbaBbBb \Rightarrow abbaBbb \Rightarrow abbabb$$

Příklad: Chceme vytvořit gramatiku generující jazyk

$$L = \{a^n b^n \mid n \geq 0\}$$

Příklad: Chceme vytvořit gramatiku generující jazyk

$$L = \{a^n b^n \mid n \geq 0\}$$

Gramatika $G = (\Pi, \Sigma, S, P)$, kde $\Pi = \{S\}$, $\Sigma = \{a, b\}$ a P obsahuje

$$S \rightarrow aSb \mid \varepsilon$$

Příklad: Chceme vytvořit gramatiku generující jazyk

$$L = \{a^n b^n \mid n \geq 0\}$$

Gramatika $G = (\Pi, \Sigma, S, P)$, kde $\Pi = \{S\}$, $\Sigma = \{a, b\}$ a P obsahuje

$$S \rightarrow aSb \mid \varepsilon$$

$$S \Rightarrow \varepsilon$$

$$S \Rightarrow aSb \Rightarrow ab$$

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$$

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaabbbb$$

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaaaSbbbb \Rightarrow aaaabbbbb$$

...

Příklad: Chceme vytvořit gramatiku generující jazyk tvořený všemi palindromy nad abecedou $\{a, b\}$, tj.

$$L = \{w \in \{a, b\}^* \mid w = w^R\}$$

Poznámka: w^R označuje tzv. **zrcadlový obraz** slova w , tj. slovo w zapsané pozpátku.

Příklad: Chceme vytvořit gramatiku generující jazyk tvořený všemi palindromy nad abecedou $\{a, b\}$, tj.

$$L = \{w \in \{a, b\}^* \mid w = w^R\}$$

Poznámka: w^R označuje tzv. **zrcadlový obraz** slova w , tj. slovo w zapsané pozpátku.

Řešení:

$$S \rightarrow aSa \mid bSb \mid a \mid b \mid \varepsilon$$

Příklad: Chceme vytvořit gramatiku generující jazyk tvořený všemi palindromy nad abecedou $\{a, b\}$, tj.

$$L = \{w \in \{a, b\}^* \mid w = w^R\}$$

Poznámka: w^R označuje tzv. **zrcadlový obraz** slova w , tj. slovo w zapsané pozpátku.

Řešení:

$$S \rightarrow aSa \mid bSb \mid a \mid b \mid \varepsilon$$

$$S \Rightarrow aSa \Rightarrow abSba \Rightarrow abaSaba \Rightarrow abaaaba$$

Příklad: Chceme vytvořit gramatiku generující jazyk L tvořený všemi dobře uzávorkovanými sekvencemi symbolů '(' a ')'.
Například $((()())()) \in L$, ale $)() \notin L$.

Příklad: Chceme vytvořit gramatiku generující jazyk L tvořený všemi dobře uzávorkovanými sekvencemi symbolů '(' a ')'.
Například $((()())()) \in L$, ale $)() \notin L$.

Řešení:

$$S \rightarrow \varepsilon \mid (S) \mid SS$$

Příklad: Chceme vytvořit gramatiku generující jazyk L tvořený všemi dobře uzávorkovanými sekvencemi symbolů '(' a ')'.
Například $((()())()) \in L$, ale $)() \notin L$.

Řešení:

$$S \rightarrow \varepsilon \mid (S) \mid SS$$

$$\begin{aligned} S &\Rightarrow SS \Rightarrow (S)S \Rightarrow (S)(S) \Rightarrow (SS)(S) \Rightarrow ((S)S)(S) \Rightarrow \\ &((()S)(S) \Rightarrow (() (S))(S) \Rightarrow (() ()) (S) \Rightarrow (() ()) ((S)) \Rightarrow \\ &(() ()) (()) \end{aligned}$$

Příklad: Chceme vytvořit gramatiku generující jazyk L tvořený všemi dobře vytvořenými aritmetickými výrazy, kde operandy jsou vždy tvaru 'a', a kde jako operátory můžeme používat symboly $+$ a $*$.

Například $(a + a) * a + (a * a) \in L$.

Příklad: Chceme vytvořit gramatiku generující jazyk L tvořený všemi dobře vytvořenými aritmetickými výrazy, kde operandy jsou vždy tvaru 'a', a kde jako operátory můžeme používat symboly $+$ a $*$.

Například $(a + a) * a + (a * a) \in L$.

Řešení:

$$E \rightarrow a \mid E + E \mid E * E \mid (E)$$

Příklad: Chceme vytvořit gramatiku generující jazyk L tvořený všemi dobře vytvořenými aritmetickými výrazy, kde operandy jsou vždy tvaru 'a', a kde jako operátory můžeme používat symboly $+$ a $*$.

Například $(a + a) * a + (a * a) \in L$.

Řešení:

$$E \rightarrow a \mid E + E \mid E * E \mid (E)$$

$$E \Rightarrow E + E \Rightarrow E * E + E \Rightarrow (E) * E + E \Rightarrow (a + a) * E + E \Rightarrow (a + a) * a + E \Rightarrow (a + a) * a + (E) \Rightarrow (a + a) * a + (E * E) \Rightarrow (a + a) * a + (a * E) \Rightarrow (a + a) * a + (a * a)$$

$$A \rightarrow aBBb \mid AaA$$
$$B \rightarrow \varepsilon \mid bCA$$
$$C \rightarrow AB \mid a \mid b$$

A

$A \rightarrow aBBb \mid AaA$

$B \rightarrow \varepsilon \mid bCA$

$C \rightarrow AB \mid a \mid b$

A

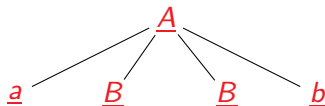
A

A \rightarrow aBBb | AaA

B \rightarrow ε | bCA

C \rightarrow AB | a | b

A

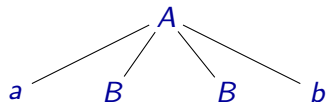


A \rightarrow aBBb | AaA

B \rightarrow ε | bCA

C \rightarrow AB | a | b

A \Rightarrow aBBb

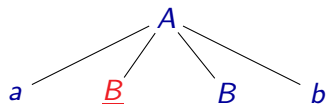


$A \rightarrow aBBb \mid AaA$

$B \rightarrow \varepsilon \mid bCA$

$C \rightarrow AB \mid a \mid b$

$A \Rightarrow aBBb$



$A \rightarrow aBBb \mid AaA$

$\underline{B} \rightarrow \varepsilon \mid bCA$

$C \rightarrow AB \mid a \mid b$

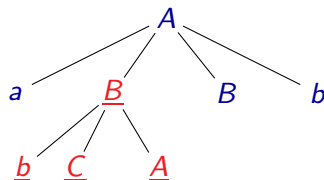
$A \Rightarrow a\underline{B}Bb$

Derivační strom

$A \rightarrow aBBb \mid AaA$

$B \rightarrow \varepsilon \mid \underline{bCA}$

$C \rightarrow AB \mid a \mid b$



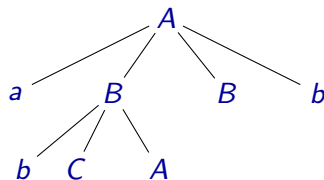
$A \Rightarrow a\underline{B}Bb \Rightarrow a\underline{bCA}Bb$

Derivační strom

$A \rightarrow aBBb \mid AaA$

$B \rightarrow \varepsilon \mid bCA$

$C \rightarrow AB \mid a \mid b$



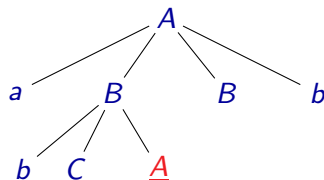
$A \Rightarrow aBBb \Rightarrow abCABb$

Derivační strom

$\underline{A} \rightarrow aBBb \mid AaA$

$B \rightarrow \varepsilon \mid bCA$

$C \rightarrow AB \mid a \mid b$



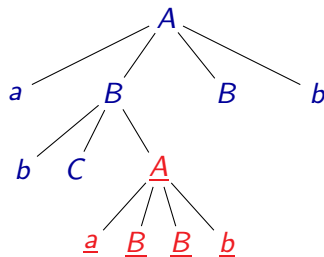
$A \Rightarrow aBBb \Rightarrow abC\underline{A}Bb$

Derivační strom

$\underline{A} \rightarrow \underline{aBBb} \mid AaA$

$B \rightarrow \varepsilon \mid bCA$

$C \rightarrow AB \mid a \mid b$



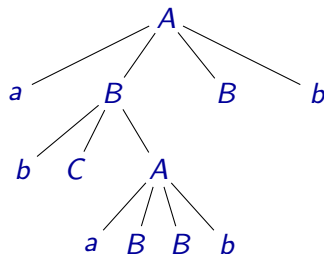
$A \Rightarrow aBBb \Rightarrow abC\underline{A}Bb \Rightarrow abC\underline{aBBb}Bb$

Derivační strom

$A \rightarrow aBBb \mid AaA$

$B \rightarrow \varepsilon \mid bCA$

$C \rightarrow AB \mid a \mid b$



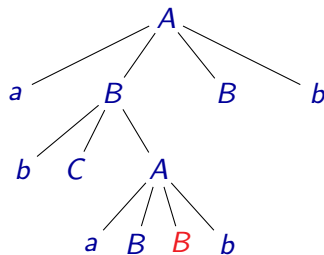
$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb$

Derivační strom

$A \rightarrow aBBb \mid AaA$

$\underline{B} \rightarrow \varepsilon \mid bCA$

$C \rightarrow AB \mid a \mid b$



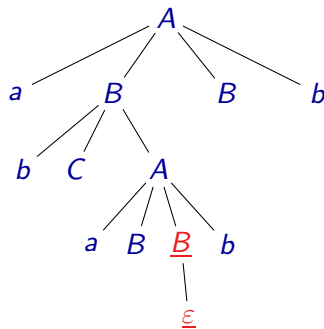
$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaB\underline{B}bBb$

Derivační strom

$A \rightarrow aBBb \mid AaA$

$\underline{B} \rightarrow \underline{\epsilon} \mid bCA$

$C \rightarrow AB \mid a \mid b$



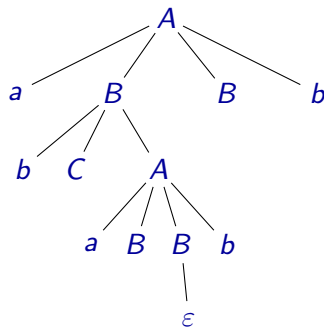
$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaB\underline{B}bBb \Rightarrow abCaBbBb$

Derivační strom

$A \rightarrow aBBb \mid AaA$

$B \rightarrow \varepsilon \mid bCA$

$C \rightarrow AB \mid a \mid b$



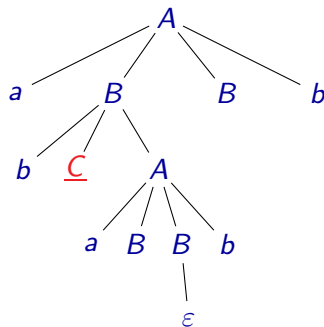
$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb$

Derivační strom

$A \rightarrow aBBb \mid AaA$

$B \rightarrow \varepsilon \mid bCA$

$\underline{C} \rightarrow AB \mid a \mid b$



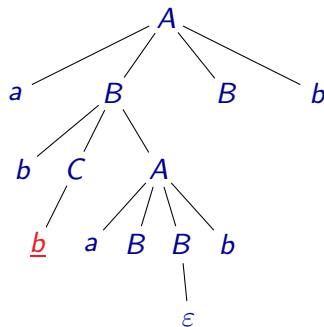
$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow ab\underline{C}aBbBb$

Derivační strom

$A \rightarrow aBBb \mid AaA$

$B \rightarrow \varepsilon \mid bCA$

$\underline{C} \rightarrow AB \mid a \mid \underline{b}$



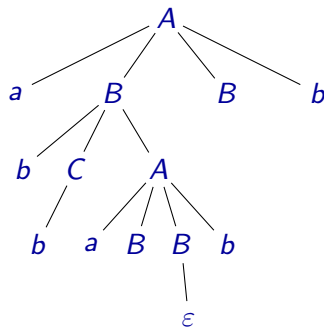
$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow ab\underline{C}aBbBb \Rightarrow ab\underline{b}aBbBb$

Derivační strom

$A \rightarrow aBBb \mid AaA$

$B \rightarrow \varepsilon \mid bCA$

$C \rightarrow AB \mid a \mid b$



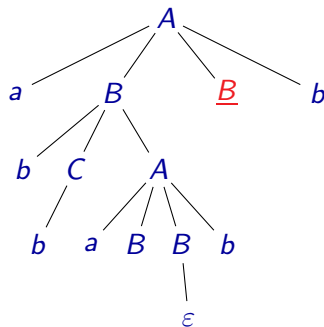
$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb \Rightarrow abbaBbBb$

Derivační strom

$A \rightarrow aBBb \mid AaA$

$\underline{B} \rightarrow \varepsilon \mid bCA$

$C \rightarrow AB \mid a \mid b$



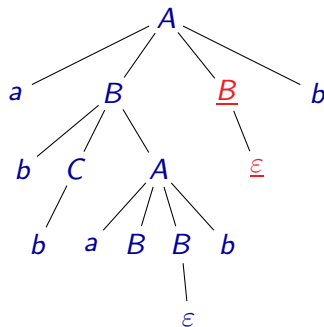
$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb \Rightarrow abbaBb\underline{B}b$

Derivační strom

$A \rightarrow aBBb \mid AaA$

$\underline{B} \rightarrow \underline{\epsilon} \mid bCA$

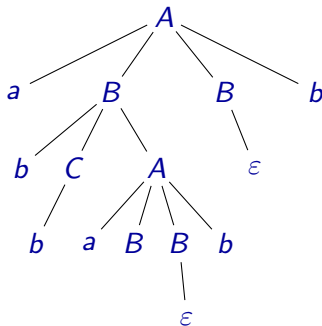
$C \rightarrow AB \mid a \mid b$



$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb \Rightarrow abbaBb\underline{B}b \Rightarrow abbaBbb$

Derivační strom

$A \rightarrow aBBb \mid AaA$
 $B \rightarrow \varepsilon \mid bCA$
 $C \rightarrow AB \mid a \mid b$



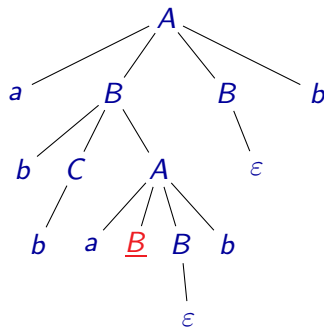
$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb \Rightarrow abbaBbBb \Rightarrow abbaBbb$

Derivační strom

$A \rightarrow aBBb \mid AaA$

$\underline{B} \rightarrow \varepsilon \mid bCA$

$C \rightarrow AB \mid a \mid b$



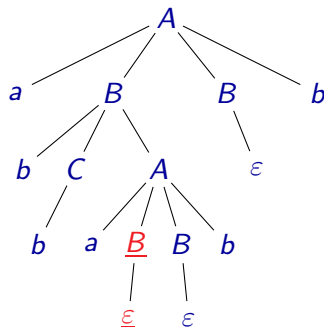
$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb \Rightarrow abbaBbBb \Rightarrow abba\underline{B}bb$

Derivační strom

$A \rightarrow aBBb \mid AaA$

$\underline{B} \rightarrow \underline{\epsilon} \mid bCA$

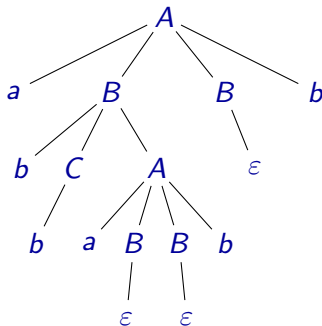
$C \rightarrow AB \mid a \mid b$



$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb \Rightarrow abbaBbBb \Rightarrow abba\underline{B}bb \Rightarrow abbabb$

Derivační strom

$A \rightarrow aBBb \mid AaA$
 $B \rightarrow \varepsilon \mid bCA$
 $C \rightarrow AB \mid a \mid b$



$A \Rightarrow aBBb \Rightarrow abCABb \Rightarrow abCaBBbBb \Rightarrow abCaBbBb \Rightarrow abbaBbBb \Rightarrow abbaBbb \Rightarrow abbabb$

Každé derivaci odpovídá nějaký **derivační strom**:

- Vrcholy stromu jsou ohodnoceny terminály a neterminály.
- Kořen stromu je ohodnocen počátečním neterminálem.
- Listy stromu jsou ohodnoceny terminály nebo symboly ε .
- Ostatní vrcholy stromu jsou ohodnoceny neterminály.
- Pokud je vrchol ohodnocen neterminálem A , pak jeho potomci jsou ohodnoceni symboly pravé strany nějakého přepisovacího pravidla $A \rightarrow \alpha$.

$$E \rightarrow E + E \mid E * E \mid (E) \mid a$$

Levá derivace je derivace, ve které v každém kroku nahrazujeme vždy nejlevější neterminál.

$$\underline{E} \Rightarrow \underline{E} + E \Rightarrow \underline{E} * E + E \Rightarrow a * \underline{E} + E \Rightarrow a * a + \underline{E} \Rightarrow a * a + a$$

Pravá derivace je derivace, ve které v každém kroku nahrazujeme vždy nejpravější neterminál.

$$\underline{E} \Rightarrow E + \underline{E} \Rightarrow \underline{E} + a \Rightarrow E * \underline{E} + a \Rightarrow \underline{E} * a + a \Rightarrow a * a + a$$

Derivace však nemusí být ani levá ani pravá:

$$\underline{E} \Rightarrow \underline{E} + E \Rightarrow E * \underline{E} + E \Rightarrow E * a + \underline{E} \Rightarrow \underline{E} * a + a \Rightarrow a * a + a$$

- Jednomu derivačnímu stromu může odpovídat více různých derivací.
- Každému derivačnímu stromu odpovídá právě jedna levá a právě jedna pravá derivace.

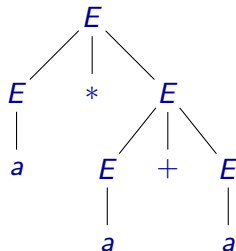
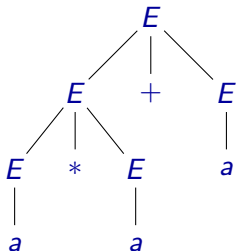
Nejednoznačné gramatiky

Gramatika G je **nejednoznačná**, jestliže existuje nějaké slovo $w \in L(G)$, kterému přísluší dva různé derivační stromy, resp. dvě různé levé či dvě různé pravé derivace.

Příklad:

$E \Rightarrow E + E \Rightarrow E * E + E \Rightarrow a * E + E \Rightarrow a * a + E \Rightarrow a * a + a$

$E \Rightarrow E * E \Rightarrow E * E + E \Rightarrow a * E + E \Rightarrow a * a + E \Rightarrow a * a + a$



Nejednoznačné gramatiky

Někdy je možné nejednoznačnou gramatiku nahradit gramatikou, která generuje tentýž jazyk, ale není nejednoznačná.

Příklad: Gramatiku

$$E \rightarrow E + E \mid E * E \mid (E) \mid a$$

můžeme nahradit ekvivalentní gramatikou

$$\begin{aligned} E &\rightarrow T \mid T + E \\ T &\rightarrow F \mid F * T \\ F &\rightarrow a \mid (E) \end{aligned}$$

Poznámka: Pokud se nejednoznačná gramatika žádnou ekvivalentní jednoznačnou gramatikou nahradit nedá, říkáme, že je **podstatně nejednoznačná**.

Gramatiky G_1 a G_2 jsou **ekvivalentní**, jestliže generují tentýž jazyk, tj. jestliže $L(G_1) = L(G_2)$.

Poznámka: Problém ekvivalence bezkontextových gramatik je algoritmicky nerozhodnutelný. Dá se dokázat, že není možné vytvořit algoritmus, který by pro libovolné dvě bezkontextové gramatiky rozhodl, zda jsou ekvivalentní či ne.

Dokonce je algoritmicky nerozhodnutelný i problém, zda gramatika generuje jazyk Σ^* .

Definice

Jazyk L je **bezkontextový**, jestliže existuje bezkontextová gramatika G taková, že $L = L(G)$.

Třída bezkontextových jazyků je uzavřená vůči:

- zřetězení
- sjednocení
- iteraci

Třída bezkontextových jazyků však není uzavřená vůči:

- doplňku
- průniku

Bezkontextové jazyky

Máme dány gramatiky $G_1 = (\Pi_1, \Sigma, S_1, P_1)$ a $G_2 = (\Pi_2, \Sigma, S_2, P_2)$, přičemž můžeme předpokládat, že $\Pi_1 \cap \Pi_2 = \emptyset$ a $S \notin \Pi_1 \cup \Pi_2$.

- Gramatika G taková, že $L(G) = L(G_1)L(G_2)$:

$$G = (\Pi_1 \cup \Pi_2 \cup \{S\}, \Sigma, S, P_1 \cup P_2 \cup \{S \rightarrow S_1S_2\})$$

- Gramatika G taková, že $L(G) = L(G_1) \cup L(G_2)$:

$$G = (\Pi_1 \cup \Pi_2 \cup \{S\}, \Sigma, S, P_1 \cup P_2 \cup \{S \rightarrow S_1, S \rightarrow S_2\})$$

- Gramatika G taková, že $L(G) = L(G_1)^*$:

$$G = (\Pi_1 \cup \{S\}, \Sigma, S, P_1 \cup \{S \rightarrow \varepsilon, S \rightarrow S_1S\})$$

Definice

Bezkontextová gramatika $G = (\Pi, \Sigma, S, P)$ je **redukovaná**, jestliže:

- Každý neterminál $X \in \Pi$ je možné přepsat na sekvenci terminálů, tj. existuje $w \in \Sigma^*$ takové, že $X \Rightarrow^* w$.
- Každý neterminál $X \in \Pi$ je dosažitelný z počátečního neterminálu S , tj. existují $\alpha, \beta \in (\Pi \cup \Sigma)^*$ takové, že $S \Rightarrow^* \alpha X \beta$.

Ke každé bezkontextové gramatice je možné sestrojít ekvivalentní redukovanou gramatiku.

Redukované gramatiky

Algoritmus, který zkonstruuje množinu

$$\mathcal{T} = \{X \in \Pi \mid \exists w \in \Sigma^* : X \Rightarrow^* w\}$$

```
1   $\mathcal{T} \leftarrow \emptyset$ 
2  change  $\leftarrow$  TRUE
3  while change
4      do change  $\leftarrow$  FALSE
5          for each  $(X \rightarrow \alpha) \in P$ 
6              do if  $X \notin \mathcal{T}$  and  $\alpha \in (\mathcal{T} \cup \Sigma)^*$ 
7                  then  $\mathcal{T} \leftarrow \mathcal{T} \cup \{X\}$ 
8                      change  $\leftarrow$  TRUE
9  return  $\mathcal{T}$ 
```

Z gramatiky G pak vytvoříme gramatiku G' tak, že z G odstraníme všechny neterminály nepatřící do \mathcal{T} a pravidla, kde se vyskytují.

Příklad: Gramatika $G = (\{S, A, B, C\}, \{a, b\}, S, P)$

$$S \rightarrow A \mid B$$

$$A \rightarrow aB \mid bS \mid b$$

$$B \rightarrow AB \mid Ba$$

$$C \rightarrow AS \mid b$$

Příklad: Gramatika $G = (\{S, A, B, C\}, \{a, b\}, S, P)$

$$S \rightarrow A \mid B$$

$$A \rightarrow aB \mid bS \mid b$$

$$B \rightarrow AB \mid Ba$$

$$C \rightarrow AS \mid b$$

$$\mathcal{T} = \{$$

Příklad: Gramatika $G = (\{S, A, B, C\}, \{a, b\}, S, P)$

$$S \rightarrow A \mid B$$

$$A \rightarrow aB \mid bS \mid b$$

$$B \rightarrow AB \mid Ba$$

$$C \rightarrow AS \mid b$$

$$\mathcal{T} = \{A,$$

Příklad: Gramatika $G = (\{S, A, B, C\}, \{a, b\}, S, P)$

$$S \rightarrow A \mid B$$

$$A \rightarrow aB \mid bS \mid b$$

$$B \rightarrow AB \mid Ba$$

$$C \rightarrow AS \mid b$$

$$\mathcal{T} = \{A, C,$$

Příklad: Gramatika $G = (\{S, A, B, C\}, \{a, b\}, S, P)$

$$S \rightarrow A \mid B$$

$$A \rightarrow aB \mid bS \mid b$$

$$B \rightarrow AB \mid Ba$$

$$C \rightarrow AS \mid b$$

$$\mathcal{T} = \{A, C, S\}$$

Příklad: Gramatika $G = (\{S, A, B, C\}, \{a, b\}, S, P)$

$$S \rightarrow A \mid B$$

$$A \rightarrow aB \mid bS \mid b$$

$$B \rightarrow AB \mid Ba$$

$$C \rightarrow AS \mid b$$

$$\mathcal{T} = \{A, C, S\}$$

Příklad: Gramatika $G = (\{S, A, B, C\}, \{a, b\}, S, P)$

$$S \rightarrow A \mid B$$

$$A \rightarrow aB \mid bS \mid b$$

$$B \rightarrow AB \mid Ba$$

$$C \rightarrow AS \mid b$$

$$\mathcal{T} = \{A, C, S\}$$

Gramatika $G' = (\{S, A, C\}, \{a, b\}, S, P')$

$$S \rightarrow A$$

$$A \rightarrow bS \mid b$$

$$C \rightarrow AS \mid b$$

Redukované gramatiky

Algoritmus, který zkonstruuje množinu

$$\mathcal{D} = \{X \in \Pi' \mid \exists \alpha, \beta \in (\Pi' \cup \Sigma)^* : S \Rightarrow^* \alpha X \beta\}$$

```
1   $\mathcal{D} \leftarrow \{S\}$ 
2  change  $\leftarrow$  TRUE
3  while change
4      do change  $\leftarrow$  FALSE
5          for each  $(X \rightarrow \alpha) \in P'$  where  $X \in \mathcal{D}$ 
6              do for  $i \leftarrow 1$  to  $|\alpha|$ 
7                  do if  $\alpha[i] \in \Pi'$  and  $\alpha[i] \notin \mathcal{D}$ 
8                      then  $\mathcal{D} \leftarrow \mathcal{D} \cup \{\alpha[i]\}$ 
9                          change  $\leftarrow$  TRUE
10 return  $\mathcal{D}$ 
```

Z gramatiky G' pak vytvoříme gramatiku G'' tak, že z G' odstraníme všechny neterminály nepatřící do \mathcal{D} a pravidla, kde se vyskytují.

Příklad: Gramatika $G' = (\{S, A, C\}, \{a, b\}, S, P')$

$$S \rightarrow A$$

$$A \rightarrow bS \mid b$$

$$C \rightarrow AS \mid b$$

Příklad: Gramatika $G' = (\{S, A, C\}, \{a, b\}, S, P')$

$$S \rightarrow A$$

$$A \rightarrow bS \mid b$$

$$C \rightarrow AS \mid b$$

$$\mathcal{D} = \{$$

Příklad: Gramatika $G' = (\{S, A, C\}, \{a, b\}, S, P')$

$$S \rightarrow A$$

$$A \rightarrow bS \mid b$$

$$C \rightarrow AS \mid b$$

$$\mathcal{D} = \{S,$$

Příklad: Gramatika $G' = (\{S, A, C\}, \{a, b\}, S, P')$

$$S \rightarrow A$$

$$A \rightarrow bS \mid b$$

$$C \rightarrow AS \mid b$$

$$\mathcal{D} = \{S, A$$

Příklad: Gramatika $G' = (\{S, A, C\}, \{a, b\}, S, P')$

$$S \rightarrow A$$

$$A \rightarrow bS \mid b$$

$$C \rightarrow AS \mid b$$

$$\mathcal{D} = \{S, A\}$$

Příklad: Gramatika $G' = (\{S, A, C\}, \{a, b\}, S, P')$

$$S \rightarrow A$$

$$A \rightarrow bS \mid b$$

$$C \rightarrow AS \mid b$$

$$\mathcal{D} = \{S, A\}$$

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

$$S \rightarrow A$$

$$A \rightarrow bS \mid b$$

Redukované gramatiky

Pořadí obou kroků je třeba dodržet. Pokud bychom je provedli v opačném pořadí, můžeme dostat gramatiku, která není redukovaná.

Příklad:

$$\begin{aligned} S &\rightarrow a \mid A \\ A &\rightarrow AB \\ B &\rightarrow b \end{aligned}$$

Pokud provedeme oba kroky algoritmu ve správném pořadí, dostaneme

$$S \rightarrow a$$

Pokud provedeme kroky algoritmu v opačném pořadí, dostaneme

$$\begin{aligned} S &\rightarrow a \\ B &\rightarrow b \end{aligned}$$

Předchozí algoritmus lze použít ke zjištění, zda $L(G) \neq \emptyset$.

Stačí ověřit, zda $S \in \mathcal{T}$:

- Pokud ano, je $L(G) \neq \emptyset$.
- Pokud ne, je $L(G) = \emptyset$.

Definice

Gramatika G je **nevypouštějící**, jestliže neobsahuje žádná ε -pravidla, tj. pravidla tvaru $X \rightarrow \varepsilon$, kde $X \in \Pi$.

Ke každé bezkontextové gramatice je možné sestrojít nevypouštějící gramatiku G' takovou, že $L(G') = L(G) - \{\varepsilon\}$.

Algoritmus, který zkonstruuje množinu

$$\mathcal{E} = \{X \in \Pi \mid X \Rightarrow^* \varepsilon\}$$

```
1   $\mathcal{E} \leftarrow \emptyset$ 
2  change  $\leftarrow$  TRUE
3  while change
4      do change  $\leftarrow$  FALSE
5          for each  $(X \rightarrow \alpha) \in P$ 
6              do if  $X \notin \mathcal{E}$  and  $\alpha \in \mathcal{E}^*$ 
7                  then  $\mathcal{E} \leftarrow \mathcal{E} \cup \{X\}$ 
8                      change  $\leftarrow$  TRUE
9  return  $\mathcal{E}$ 
```

Nevypouštějící gramatiky

Nevypouštějící gramatiku G' pak z G vytvoříme tak, že pro každé pravidlo $X \rightarrow \alpha$ z G přidáme do G' všechna možná pravidla

$$X \rightarrow \alpha'$$

kde α' vznikne z α vypuštěním libovolného počtu symbolů z \mathcal{E} , přičemž ale $\alpha' \neq \varepsilon$.

Příklad: Pokud například $\mathcal{E} = \{A, C, D\}$, pak místo pravidla

$$A \rightarrow aASCbA$$

přidáme pravidla

$$A \rightarrow aASCbA \mid aSCbA \mid aASbA \mid aASCb \mid aSbA \mid aSCb \mid aASb \mid aSb$$

Nevypouštějící gramatiky

Příklad: Gramatika $G = (\{S, A, B, C\}, \{a, b\}, S, P)$

$$S \rightarrow AB \mid \varepsilon$$

$$A \rightarrow aAAb \mid BS \mid CA$$

$$B \rightarrow BbA \mid CaC \mid \varepsilon$$

$$C \rightarrow aBB \mid bS$$

Nevypouštějící gramatiky

Příklad: Gramatika $G = (\{S, A, B, C\}, \{a, b\}, S, P)$

$$S \rightarrow AB \mid \varepsilon$$

$$A \rightarrow aAAb \mid BS \mid CA$$

$$B \rightarrow BbA \mid CaC \mid \varepsilon$$

$$C \rightarrow aBB \mid bS$$

$$\mathcal{E} = \{$$

Nevypouštějící gramatiky

Příklad: Gramatika $G = (\{S, A, B, C\}, \{a, b\}, S, P)$

$$S \rightarrow AB \mid \varepsilon$$

$$A \rightarrow aAAb \mid BS \mid CA$$

$$B \rightarrow BbA \mid CaC \mid \varepsilon$$

$$C \rightarrow aBB \mid bS$$

$$\mathcal{E} = \{S,$$

Nevypouštějící gramatiky

Příklad: Gramatika $G = (\{S, A, B, C\}, \{a, b\}, S, P)$

$$S \rightarrow AB \mid \varepsilon$$

$$A \rightarrow aAAb \mid BS \mid CA$$

$$B \rightarrow BbA \mid CaC \mid \varepsilon$$

$$C \rightarrow aBB \mid bS$$

$$\mathcal{E} = \{S, B,$$

Nevypouštějící gramatiky

Příklad: Gramatika $G = (\{S, A, B, C\}, \{a, b\}, S, P)$

$$S \rightarrow AB \mid \varepsilon$$

$$A \rightarrow aAAb \mid BS \mid CA$$

$$B \rightarrow BbA \mid CaC \mid \varepsilon$$

$$C \rightarrow aBB \mid bS$$

$$\mathcal{E} = \{S, B, A$$

Nevypouštějící gramatiky

Příklad: Gramatika $G = (\{S, A, B, C\}, \{a, b\}, S, P)$

$$S \rightarrow AB \mid \varepsilon$$

$$A \rightarrow aAAb \mid BS \mid CA$$

$$B \rightarrow BbA \mid CaC \mid \varepsilon$$

$$C \rightarrow aBB \mid bS$$

$$\mathcal{E} = \{S, B, A\}$$

Nevypouštějící gramatiky

Příklad: Gramatika $G = (\{S, A, B, C\}, \{a, b\}, S, P)$

$$\begin{aligned}S &\rightarrow AB \mid \varepsilon \\A &\rightarrow aAAb \mid BS \mid CA \\B &\rightarrow BbA \mid CaC \mid \varepsilon \\C &\rightarrow aBB \mid bS\end{aligned}$$

$$\mathcal{E} = \{S, B, A\}$$

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

$$\begin{aligned}S &\rightarrow AB \mid A \mid B \\A &\rightarrow aAAb \mid aAb \mid ab \mid BS \mid B \mid S \mid CA \mid C \\B &\rightarrow BbA \mid bA \mid Bb \mid b \mid CaC \\C &\rightarrow aBB \mid aB \mid a \mid bS \mid b\end{aligned}$$

Definice

Gramatika G je v **Chomského normální formě**, jestliže všechna její pravidla jsou pouze následujícího tvaru:

- $X \rightarrow YZ$, kde $X, Y, Z \in \Pi$, nebo
- $X \rightarrow a$, kde $X \in \Pi$ a $a \in \Sigma$.

Ke každé bezkontextové gramatice je možné sestrojít gramatiku G' v Chomského normální formě takovou, že $L(G') = L(G) - \{\varepsilon\}$.

Definice

Gramatika G je v **Greibachové normální formě**, jestliže všechna její pravidla jsou pouze následujícího tvaru:

- $X \rightarrow a\alpha$, kde $a \in \Sigma$, $X \in \Pi$ a $\alpha \in \Pi^*$.

Ke každé bezkontextové gramatice je možné sestrojít gramatiku G' v Greibachové normální formě takovou, že $L(G') = L(G) - \{\varepsilon\}$.

Definice

Gramatika G je **regulární**, jestliže všechna její pravidla jsou tvaru:

- $X \rightarrow wY$, kde $X, Y \in \Pi$, $w \in \Sigma^*$, nebo
- $X \rightarrow w$, kde $X \in \Pi$, $w \in \Sigma^*$.

Regulární gramatiky generují právě třídu regulárních jazyků, tj.:

- Jazyk generovaný regulární gramatikou je vždy regulární.
- Každý regulární jazyk je generován nějakou regulární gramatikou.