

**252-0027**

# **Einführung in die Programmierung**

## **1.0 EBNF**

*Thomas R. Gross*

**Department Informatik  
ETH Zürich**

- **EBNF Regel besteht aus:**

LHS  $\Leftarrow$  RHS

- Linke-Seite (Left-Hand Side, LHS)
- Rechte-Seite (Right-Hand Side, RHS)
- $\Leftarrow$  (trennt LHS von RHS, ausgesprochen «ist definiert als»)

- **LHS**

- Ein Wort (kursiv, kleingeschrieben) – der Name der EBNF Regel

- **RHS**

- Die genaue Beschreibung für den Namen (d.h., der LHS) durch
  - Zeichen (stellen das Zeichen da, d.h. wir erwarten dieses Zeichen und kein anderes) – nicht kursiv
  - Namen (von EBNF Regeln) – kursiv und kleingeschrieben
  - Kombinationen der vier Kontrollelemente («control forms») (auf folgenden Seiten)

# Control forms (zum Kombinieren)

- Aufreihung
- Auswahl, Option (Entscheidung)
- Wiederholung (kann auch Option «0 Wiederholungen» sein)
- **Rekursion**

# 1.5 Rekursion

# Positive (ganze) Zahlen

- EBNF Regel *pos\_integer* soll ganze Zahlen ohne Vorzeichen beschreiben
  - Wir haben eine Regel für Ziffern: *digit*  $\Leftarrow$  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
- Erster Versuch

*pos\_integer*  $\Leftarrow$  *digit* .....

Mehr davon

# Positive (ganze) Zahlen

- EBNF Regel *pos\_integer* soll ganze Zahlen ohne Vorzeichen beschreiben
  - Wir haben eine Regel für Ziffern: *digit*  $\Leftarrow$  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

- Erster Versuch

*pos\_integer*  $\Leftarrow$  *digit* .....

*pos\_integer*  $\Leftarrow$  { *digit* }  
ε ist legale positive Zahl

- Einfache Wiederholung nicht was wir wollen

# Positive (ganze) Zahlen

- EBNF Regel *pos\_integer* soll positive Zahlen ohne Vorzeichen beschreiben

- Wir haben eine Regel für Ziffern: *digit*  $\Leftarrow$  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

- Erster Versuch

*pos\_integer*  $\Leftarrow$  *digit* .....

*pos\_integer*  $\Leftarrow$  { *digit* }  
ε ist legale positive Zahl

- Einfache Wiederholung nicht was wir wollen
  - Aber vielleicht ein Anfang

# Option – kann, muss aber nicht dabei sein

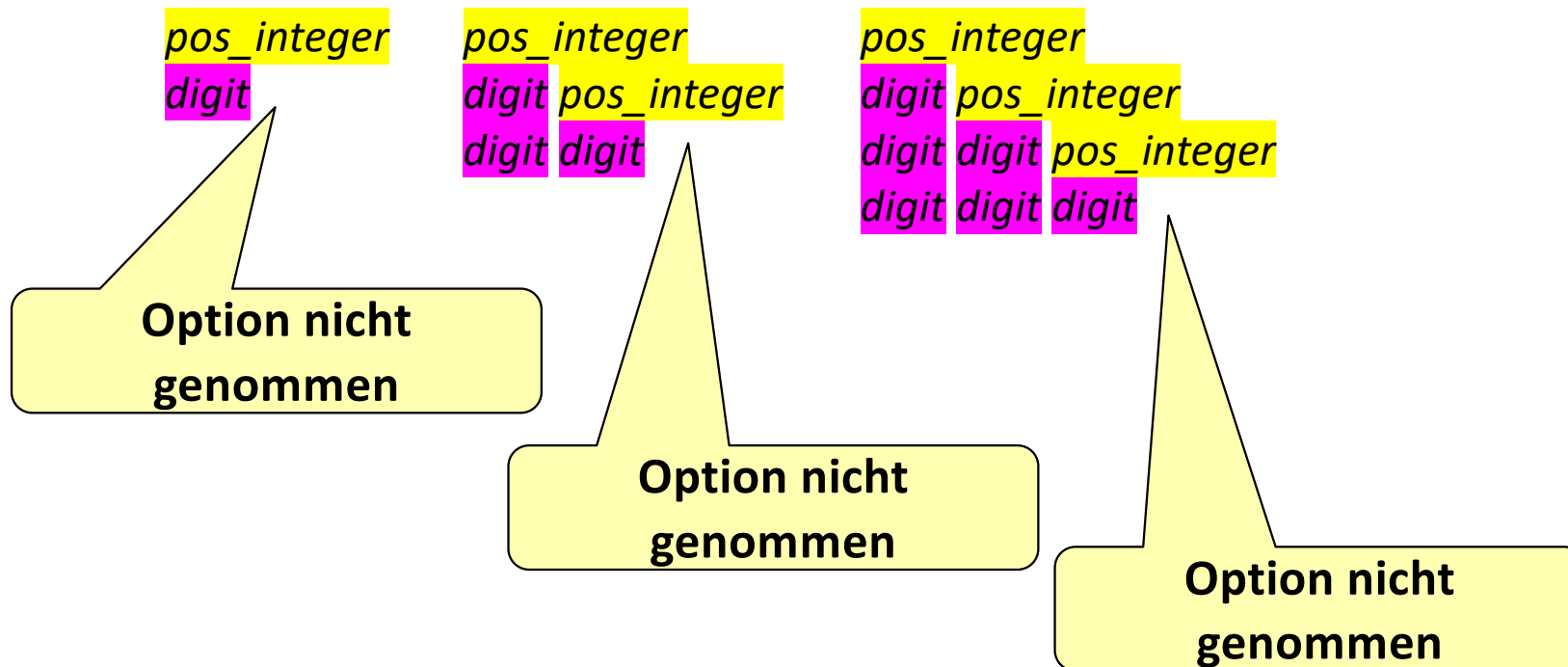
- Was wenn wir den Namen einer Regel auf der rechten Seite *dieser* Regel verwenden?
  - Mit Option – können alle ganzen Zahlen ohne Vorzeichen beschreiben

***pos\_integer***  $\Leftarrow$  ***digit*** [ ***pos\_integer*** ]



*pos\_integer*  $\Leftarrow$  *digit* [ *pos\_integer* ]

- Beispiele für legale Symbole



# Name der LHS auch auf der RHS

- Es muss (mindestens) einen Weg geben, *Namen der LHS* durch eine RHS *ohne diesen Namen* zu ersetzen

$pos\_integer \Leftarrow digit [ pos\_integer ]$

- Nicht-Wahl der Option: Nur  $digit$  auf der RHS

$pos\_integer$   
 $digit pos\_integer$   
 $digit \epsilon$   
 $digit$

- Andere Möglichkeit

$pos\_integer \Leftarrow ( digit pos\_integer ) | \epsilon$

Ohne Klammern: Aufreihung  
bindet stärker als Auswahl

$pos\_integer$   
 $digit pos\_integer$   
 $digit digit pos\_integer$   
 $digit digit \epsilon$   
 $digit digit$

# Name der LHS auch auf der RHS

- Es muss (mindestens) einen Weg geben, *Namen der LHS* durch eine RHS *ohne diesen Namen* zu ersetzen

***pos\_integer***  $\Leftarrow$  ***digit*** [ ***pos\_integer*** ]

- Nicht-Wahl der Option: Nur ***digit*** auf der RHS

***pos\_integer***  
***digit*** ***pos\_integer***  
***digit***  $\epsilon$   
***digit***

- Andere Möglichkeit

***pos\_integer***  $\Leftarrow$  ***digit*** ***pos\_integer*** |  $\epsilon$

Lässt  $\epsilon$  als  
«Zahl» zu ...

***pos\_integer***  
***digit*** ***pos\_integer***  
***digit*** ***digit*** ***pos\_integer***  
***digit*** ***digit***  $\epsilon$   
***digit*** ***digit***

# Rekursive Regel

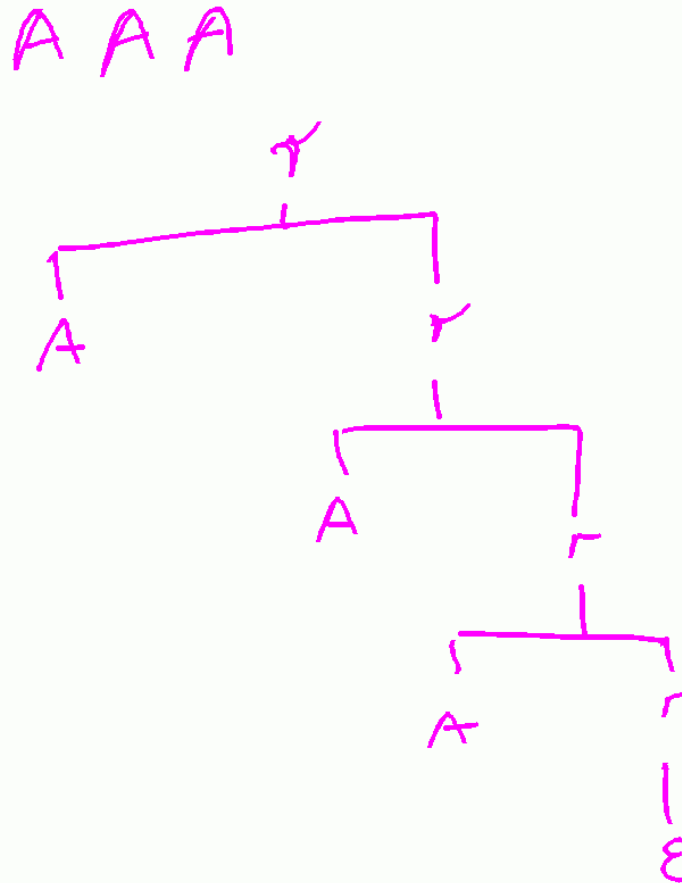
- Regel ist rekursiv: ihr Name wird in der Definition verwendet  
$$pos\_integer \Leftarrow digit \ [ \ pos\_integer \ ]$$
- Beschreibung ist rekursiv: mindestens eine rekursive Regel

# Rekursion

- **Rekursive Beschreibung enthält rekursive Regeln**
  - Eine Regel ist *direkt rekursiv* wenn ihr Name in der Definition verwendet wird
  - Also die LHS erscheint auch auf der RHS
  - $r \Leftarrow \mid A r$
  - $r \Leftarrow \mid ( A r )$  falls Sie Unklarheit vermeiden wollen
  - $r \Leftarrow \varepsilon \mid ( A r )$  falls Sie alle Unklarheiten vermeiden wollen
- **EBNF Description  $r$  beschreibt Folge von null oder mehr  $A$  Zeichen**

# Diskussion

Aus dem Archiv ...



# Rekursive Regel

- Regel ist rekursiv: ihr Name wird in der Definition verwendet

*$pos\_integer \Leftarrow digit [ pos\_integer ]$*

- Hätten wir auch anders machen können

*$pos\_integer \Leftarrow digit \{ digit \}$*

# Diskussion

- **Warum der Aufwand?**

- Wären nicht (in den Beispielen)

$$r \Leftarrow \{ A \}$$

$$pos\_number \Leftarrow digit \{ digit \} \quad \text{gut genug?}$$

- **Kann jede Rekursion durch Wiederholung(en) ausgedrückt werden?**
- **Kann jede Wiederholung durch Rekursion ausgedrückt werden?**



# Diskussion

$A \dots A B \dots B$

$\underbrace{\hspace{1cm}}$   
Anzahl A = Anzahl B

$A^n B^n$

$n = 0, 1, 2, \dots$

$d \Leftarrow \{A\} \{B\}$   $\times$

$AB B$

$\Leftarrow \{A B\}$   $\times$

$ABAB$

# Diskussion

- Kann jede Rekursion durch Wiederholung(en) ausgedrückt werden?
  - Nein
  - Finden Sie Beschreibung für  $A^n B^n$  ( $n$  Zahl  $\geq 0$ : also gleiche Anzahl A, B)

EBNF Description *balance*

*balance*  $\Leftarrow \epsilon \mid A \text{ balance } B$

*balance*  $\Leftarrow \epsilon \mid ( A \text{ balance } B )$

# Diskussion

- **Kann jede Wiederholung durch Rekursion ausgedrückt werden?**
  - Ja
  - Werden später noch mehr über Rekursion vs. Wiederholung(en) erfahren

- **Direkte Rekursion**

- $r \Leftarrow A \mid A r$

- **Indirekte Rekursion**

- Folge von Regeln  $N_1 \dots N_k$  so dass  $N_2$  auf der RHS von  $N_1$ ,  $N_3$  auf der RHS von  $N_2$ , ... und  $N_1$  auf der RHS von  $N_k$  erscheint

$$name2 \Leftarrow ( name1 \ B ) \mid ( X \ B )$$

$$name1 \Leftarrow A \ name2$$

- **Direkte Rekursion**

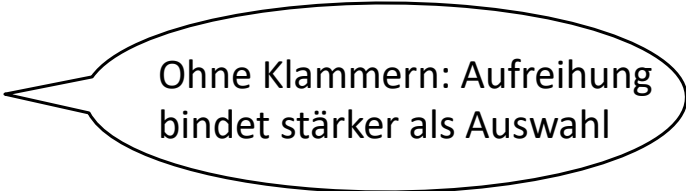
- $r \Leftarrow A \mid A r$

- **Indirekte Rekursion**

- Folge von Regeln  $N_1 \dots N_k$  so dass  $N_2$  auf der RHS von  $N_1$ ,  $N_3$  auf der RHS von  $N_2$ , ... und  $N_1$  auf der RHS von  $N_k$  erscheint

$$name2 \Leftarrow name1 B \mid X B$$

$$name1 \Leftarrow A name2$$



Ohne Klammern: Aufreihung  
bindet stärker als Auswahl

- **Beschreibung von  $name1$ :  $AXB, AAXBB, \dots$**

# Zusammenfassung: Ableitungsbaum, Tabellen, Graphen

- **Ein Ableitungsbaum oder eine Tabelle demonstrieren, dass ein Symbol legal gemäss einer EBNF Beschreibung ist.**
  - In beiden Fällen kürzen wir die Schritte manchmal ab wenn keine Verwechslungsgefahr besteht.
- **Ein (EBNF) Graph ist eine andere Darstellung einer EBNF Beschreibung**
  - Ein Pfad durch den Graphen entspricht einem Symbol das legal ist
  - Umgekehrt: um zu zeigen, dass ein Symbol legal ist, finden wir einen Pfad
  - Graph für rekursive Beschreibung: nicht elegant (muss endlich sein!)

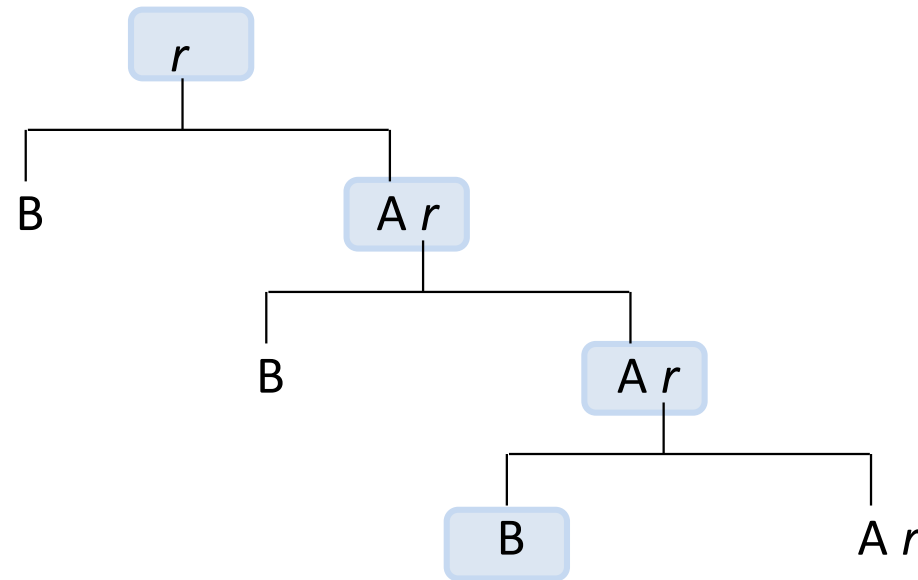
$r \Leftarrow B \mid A r$

■ Ist AAB legal? -- Tabelle

	Regel
$r$	Anfang jeder Tabelle
$B \mid A r$	Ersetzen von $r$ durch RHS (1)
$A r$	2. Auswahlmöglichkeit gewählt (2)
$A ( B \mid A r )$	Ersetzen von $r$ durch RHS (1), $()$ zur Vermeidung von Missverständnissen
$A A r$	2. Auswahlmöglichkeit gewählt (2)
$A A ( B \mid A r )$	Ersetzen von $r$ durch RHS (1)
$A A B$	1. Auswahlmöglichkeit gewählt (2)

$r \Leftarrow B \mid A r$

- Ist AAB legal? -- Ableitungsbaum (Version 1)

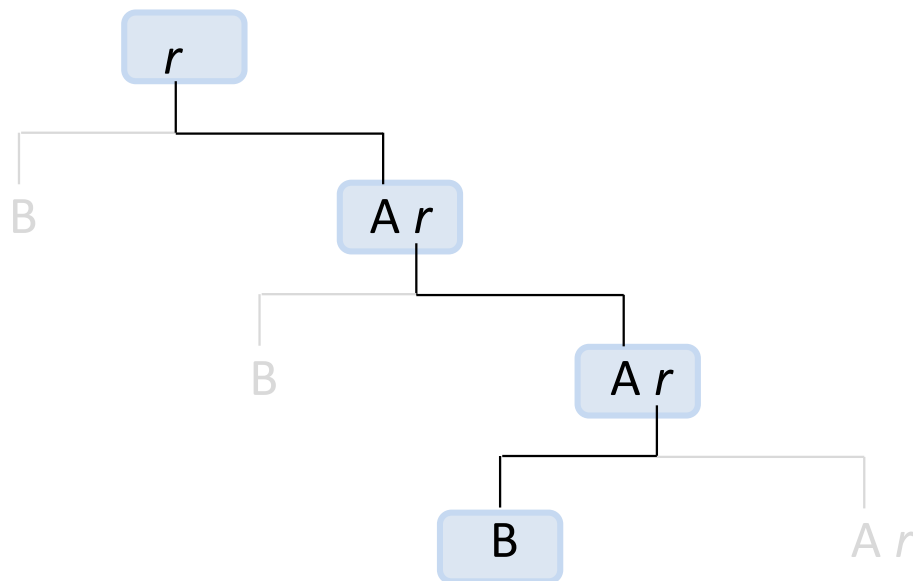


Auswahl getroffen

- In jeder Zeile wird eine EBNF Beschreibung durch eine rechte Seite ersetzt (hier immer  $A r$ , bis auf den letzten Schritt)

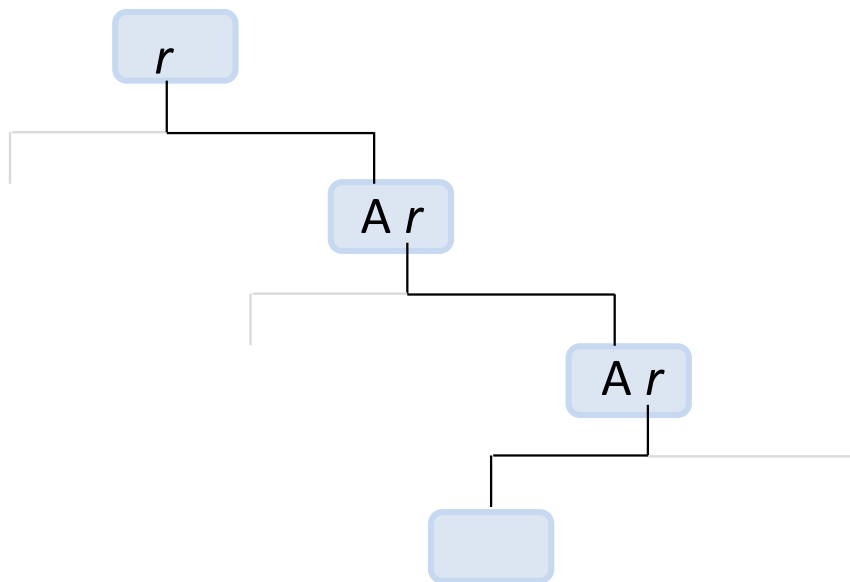


- Ist AAB legal? -- Ableitungsbaum (Version 2)
- Wir fassen Schritt 1 (Ersetzen der RHS) mit Schritt 2 (Auswahl treffen) zusammen
  - Unwichtiges lassen wir weg



$r \Leftarrow \varepsilon \mid A r \quad \text{oder} \quad r \Leftarrow \mid A r$

- Ist AA legal? -- Ableitungsbaum (Version 2)
- Wir fassen Schritt 1 (Ersetzen der RHS) mit Schritt 2 (Auswahl treffen) zusammen
  - Unwichtiges lassen wir weg



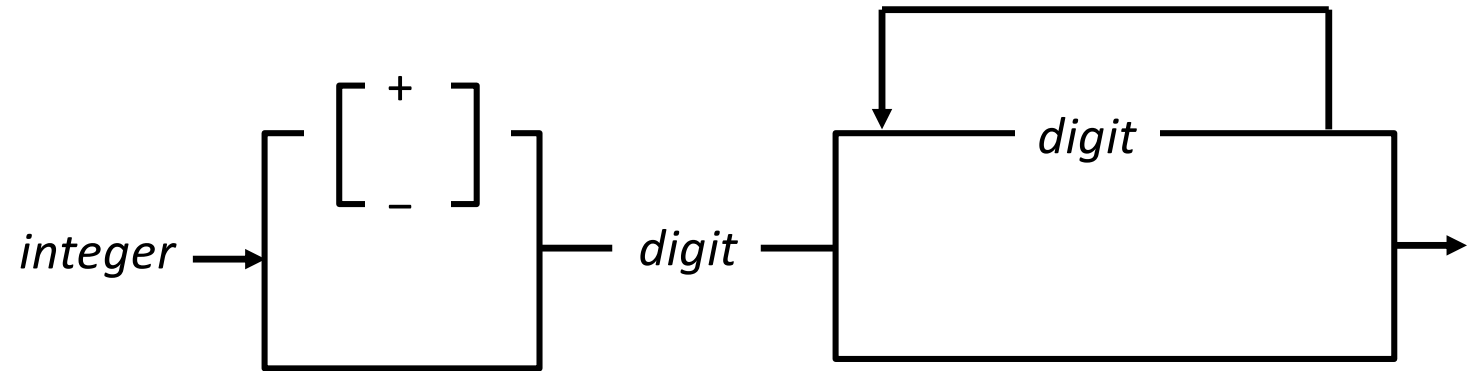
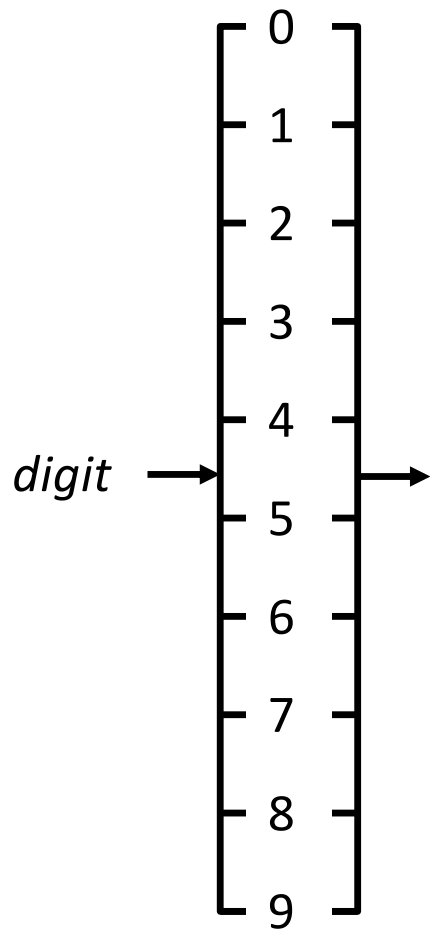
# EBNF Beispiel (i1) nochmal

**EBNF Description:** *integer*

*digit*  $\Leftarrow$  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

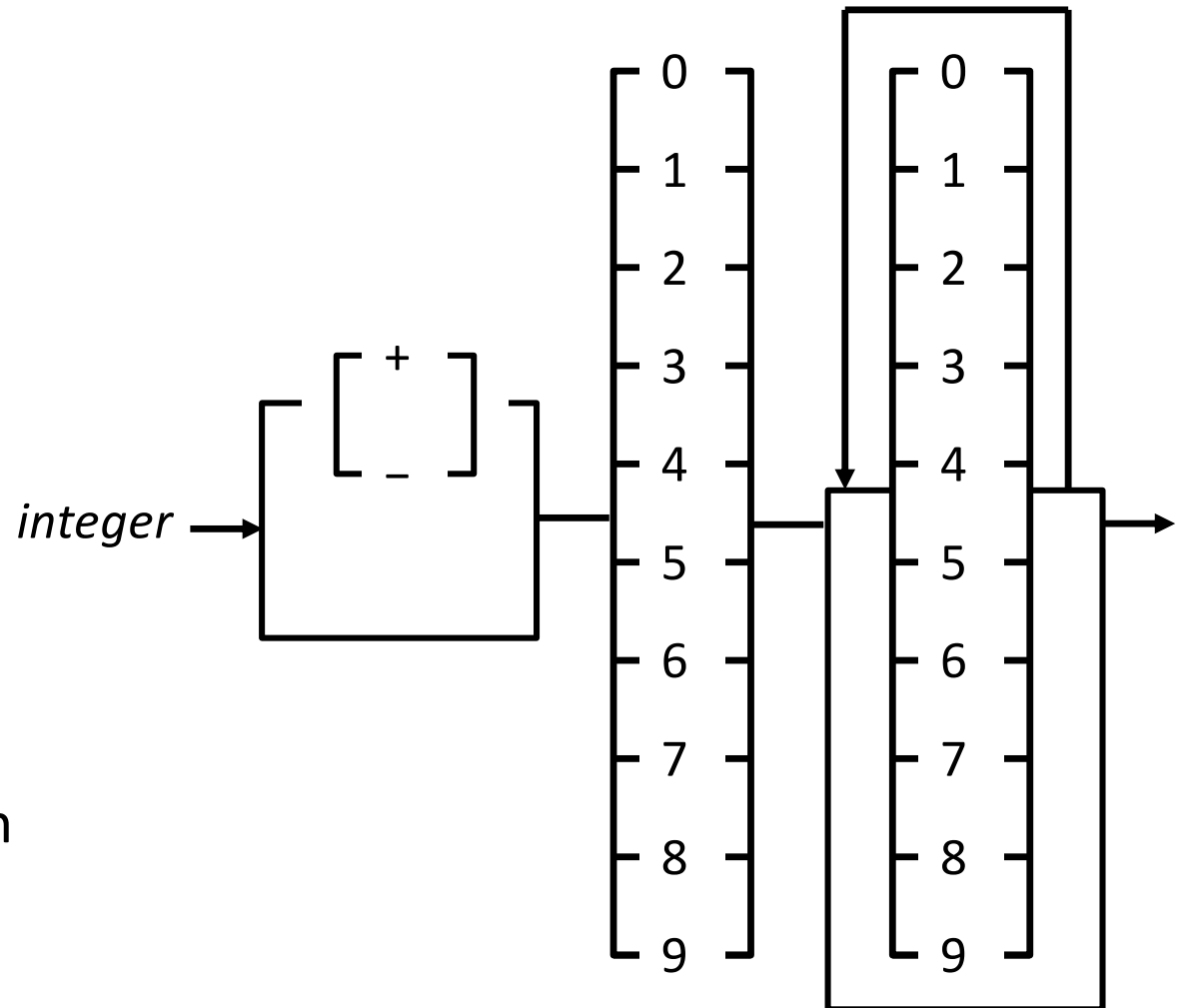
*integer*  $\Leftarrow$  [+|-]*digit*{*digit*}

# Beispiel



# Substitution

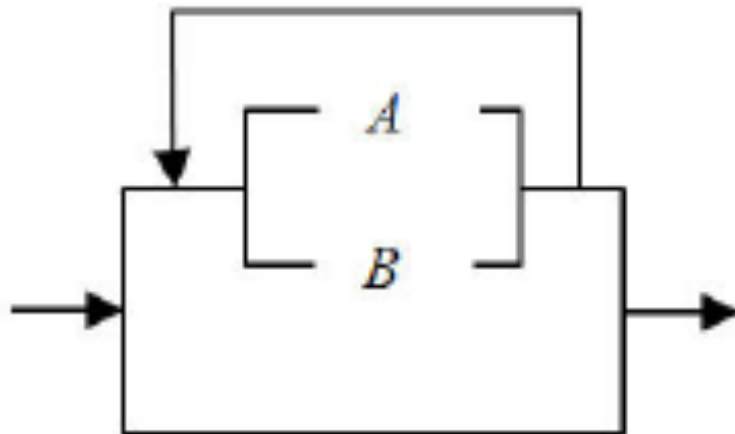
- Können einen Syntax Graphen in einen anderen einsetzen
  - «interne» Namen verschwinden
- EBNF Beschreibung
  - Alle Graphen für Regeln der Beschreibung zusammen



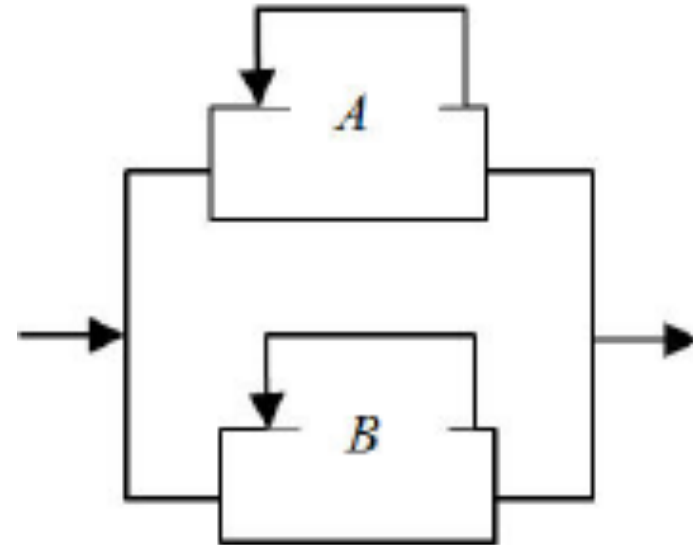
# Was für Symbole sind legal?

Eine EBNF Regel zur Beschreibung der Menge der legalen Symbole.  $g \Leftarrow ???$

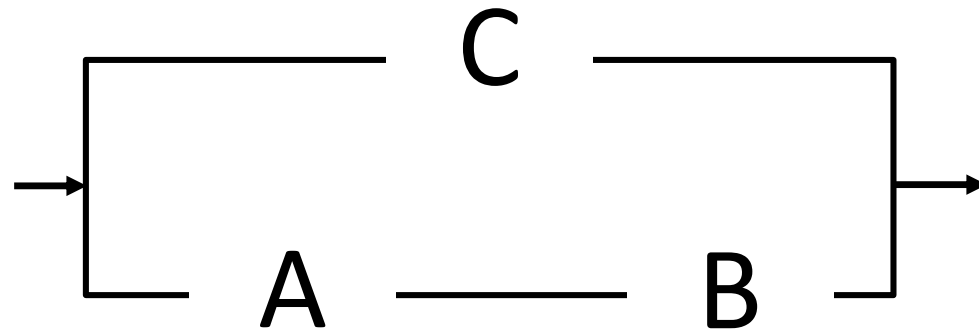
1.



3.



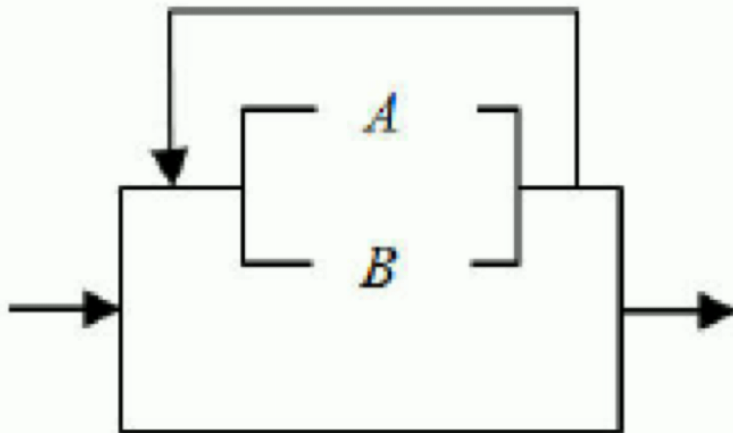
2.



# Was für Symbole sind legal?

Eine EBNF Regel zur Beschreibung der Menge der legalen Symbole.  $g \Leftarrow ???$

1.

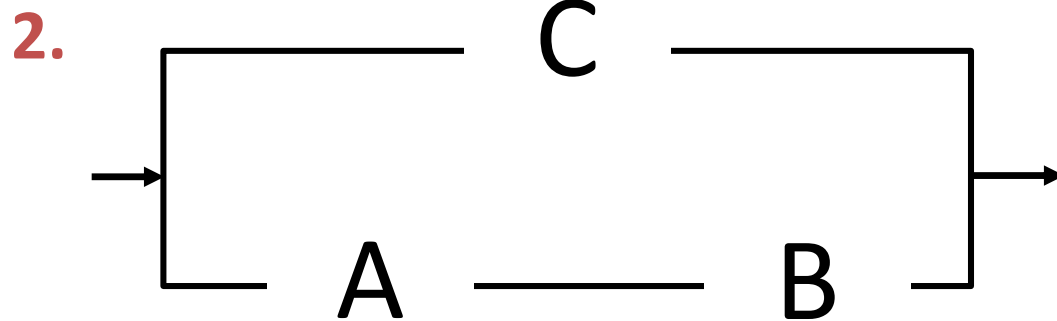


$g \Leftarrow \{A | B\}$

$\epsilon$   
A  
B  
AA  
AB  
BA  
BB ..  
..

# Was für Symbole sind legal?

Eine EBNF Regel zur Beschreibung der Menge der legalen Symbole.  $g \Leftarrow ???$



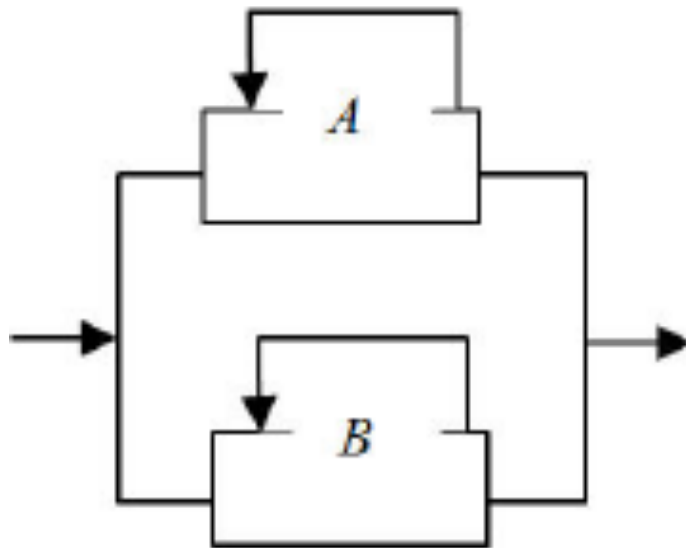


# Was für Symbole sind legal?

Poll

Eine EBNF Regel zur Beschreibung der Menge der legalen Symbole.  $g \Leftarrow ???$

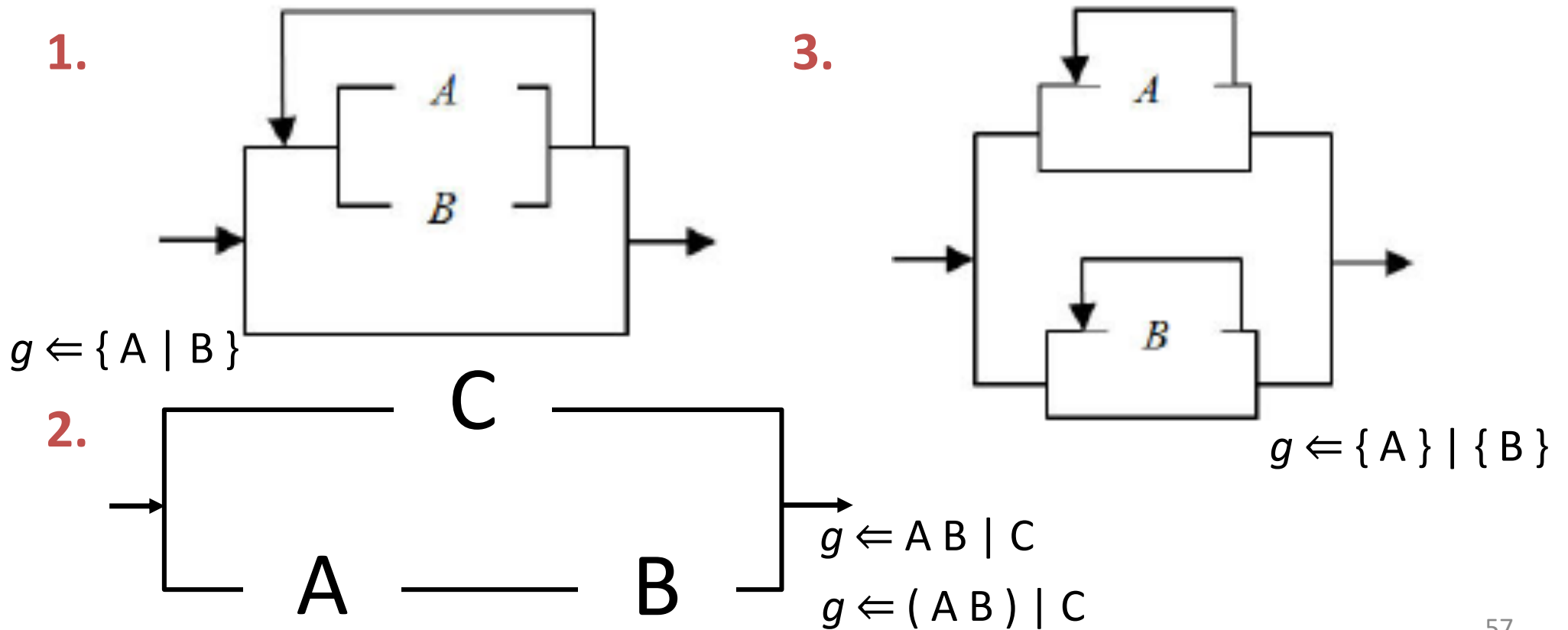
3.



# Was für Symbole sind legal?

Poll

Eine EBNF Regel zur Beschreibung der Menge der legalen Symbole.  $g \Leftarrow ???$



# EBNF Geschichte

- **BNF enthielt erst nur Rekursion und Auswahl**
  - Diese sind essential
- **Option und Wiederholung von Niklaus Wirth hinzugefügt**
  - Daher «E» – extended
  - Machen Beschreibung einfacher zu lesen
    - Motivation: Beschreibung von Pascal

# Nochmal *integer*

**EBNF Description:** *integer* (using BNF not EBNF)

*sign*  $\Leftarrow$  | + | -

*digit*  $\Leftarrow$  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

*digits*  $\Leftarrow$  | *digit digits*

*integer*  $\Leftarrow$  *sign digit digits*

# EBNF

- Das war's.