Backus-Naur-Form

Programmieren und Software-Engineering
Homomorphismen, Formale Sprachen und Syntax-Analyse

22. Februar 2023

Backus-Naur-Form

Anläßlich der Definition der Programmiersprache ALGOL-58/ALGOL-60 wurde von John Backus und Peter Naur eine neue Darstellungsform **(Backus-Naur-Form, BNF)** für die formale Definition der Syntax von Programmiersprachen entwickelt, die folgende Metasprachlichen Symbole verwendet:

Definition (Backus-Naur-Form: Notation)

- **::=** Definitions-, bzw. Ersetzungszeichen
- | Entweder-Oder-Zeichen
- **<, bzw. >** Spitze Klammern zur Kennzeichnung von Variablen

Backus-Naur-Form

Beschreibung des Sprachelementes ⟨ identifier ⟩ der Programmiersprache ALGOL-60 mittels BNF:

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

$\langle \text{letter} \rangle ::= a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|$
$r|s|t|u|v|w|x|y|z|A|B|C|D|E|F|G|$
$H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|$
$W|X|Y|Z$

$\langle \text{identifier} \rangle ::= \langle \text{letter} \rangle | \langle \text{identifier} \rangle \langle \text{letter} \rangle | \langle \text{identifier} \rangle \langle \text{digit} \rangle$

Backus-Naur-Form

Beschreibung des Sprachelementes ⟨ identifier ⟩ der Programmiersprache ALGOL-60 mittels BNF:

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

$$\begin{aligned}\langle \text{letter} \rangle ::= {} & a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q| \\ & r|s|t|u|v|w|x|y|z|A|B|C|D|E|F|G| \\ & H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V| \\ & W|X|Y|Z\end{aligned}$$

$$\langle \text{identifier} \rangle ::= \langle \text{letter} \rangle | \langle \text{identifier} \rangle \langle \text{letter} \rangle | \langle \text{identifier} \rangle \langle \text{digit} \rangle$$

Anmerkung: Das syntaktische Objekt ⟨identifier⟩ wird aus einer beliebigen Kombination von Buchstaben und Ziffern gebildet, wobei an der ersten Stelle ein Buchstabe stehen muss. Dies wird durch den rekursiven zeichenweisen Aufbau von rechts nach links und durch das Verlassen der Rekursion mittels eines Buchstaben sichergestellt.

Backus-Naur-Form

Beispiel

- Syntaktisch richtige ⟨identifier⟩:
  - buchstabe
  - V17a
  - Feld27
- Syntaktisch falsche ⟨identifier⟩:
  - 1bz
  - $abc
  - f(x)

Ableitung (1)

Rechtsableitung von V17a

$$\langle \text{identifier} \rangle \quad \Rightarrow \quad \langle \text{identifier} \rangle \langle \text{letter} \rangle$$

Ableitung (1)

Rechtsableitung von V17a

$$\begin{aligned} \langle \text{identifier} \rangle &\Rightarrow \langle \text{identifier} \rangle \langle \text{letter} \rangle \\ &\Rightarrow \langle \text{identifier} \rangle \text{a} \end{aligned}$$

Ableitung (1)

Rechtsableitung von V17a

$$
\begin{aligned}
\langle \text{identifier} \rangle &\Rightarrow \langle \text{identifier} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \text{a} \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \text{a}
\end{aligned}
$$

Ableitung (1)

Rechtsableitung von V17a

$$
\begin{aligned}
\langle \text{identifier} \rangle &\Rightarrow \langle \text{identifier} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \text{a} \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \text{a} \\
&\Rightarrow \langle \text{identifier} \rangle \text{7a}
\end{aligned}
$$

Ableitung (1)

Rechtsableitung von V17a

$$
\begin{aligned}
\langle \text{identifier} \rangle &\Rightarrow \langle \text{identifier} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \text{a} \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \text{a} \\
&\Rightarrow \langle \text{identifier} \rangle \text{7a} \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \text{7a}
\end{aligned}
$$

Ableitung (1)

Rechtsableitung von V17a

$$
\begin{aligned}
\langle \text{identifier} \rangle &\Rightarrow \langle \text{identifier} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \text{a} \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \text{a} \\
&\Rightarrow \langle \text{identifier} \rangle \text{7a} \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \text{7a} \\
&\Rightarrow \langle \text{identifier} \rangle \text{17a}
\end{aligned}
$$

Ableitung (1)

Rechtsableitung von V17a

$$
\begin{aligned}
\langle \text{identifier} \rangle &\Rightarrow \langle \text{identifier} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \text{a} \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \text{a} \\
&\Rightarrow \langle \text{identifier} \rangle \text{7a} \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \text{7a} \\
&\Rightarrow \langle \text{identifier} \rangle \text{17a} \\
&\Rightarrow \langle \text{letter} \rangle \text{17a}
\end{aligned}
$$

Ableitung (1)

Rechtsableitung von V17a

$$
\begin{aligned}
\langle \text{identifier} \rangle &\Rightarrow \langle \text{identifier} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \text{a} \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \text{a} \\
&\Rightarrow \langle \text{identifier} \rangle \text{7a} \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \text{7a} \\
&\Rightarrow \langle \text{identifier} \rangle \text{17a} \\
&\Rightarrow \langle \text{letter} \rangle \text{17a} \\
&\Rightarrow \text{V17a}
\end{aligned}
$$

Ableitung (2)

Linksableitung von V17a

$\langle \text{identifier} \rangle \quad \Rightarrow \quad \langle \text{identifier} \rangle \langle \text{letter} \rangle$

Ableitung (2)

Linksableitung von V17a

$$\begin{aligned} \langle \text{identifier} \rangle &\Rightarrow \langle \text{identifier} \rangle \langle \text{letter} \rangle \\ &\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \end{aligned}$$

Ableitung (2)

Linksableitung von V17a

$$
\begin{aligned}
\langle \text{identifier} \rangle &\Rightarrow \langle \text{identifier} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle
\end{aligned}
$$

Ableitung (2)

Linksableitung von V17a

$$
\begin{aligned}
\langle \text{identifier} \rangle &\Rightarrow \langle \text{identifier} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{letter} \rangle \langle \text{digit} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle
\end{aligned}
$$

Ableitung (2)

Linksableitung von V17a

$$
\begin{aligned}
\langle \text{identifier} \rangle &\Rightarrow \langle \text{identifier} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{letter} \rangle \langle \text{digit} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \text{V} \langle \text{digit} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle
\end{aligned}
$$

Ableitung (2)

Linksableitung von V17a

$$
\begin{aligned}
\langle \text{identifier} \rangle &\Rightarrow \langle \text{identifier} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{letter} \rangle \langle \text{digit} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \text{V} \langle \text{digit} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \text{V1} \langle \text{digit} \rangle \langle \text{letter} \rangle
\end{aligned}
$$

Ableitung (2)

Linksableitung von V17a

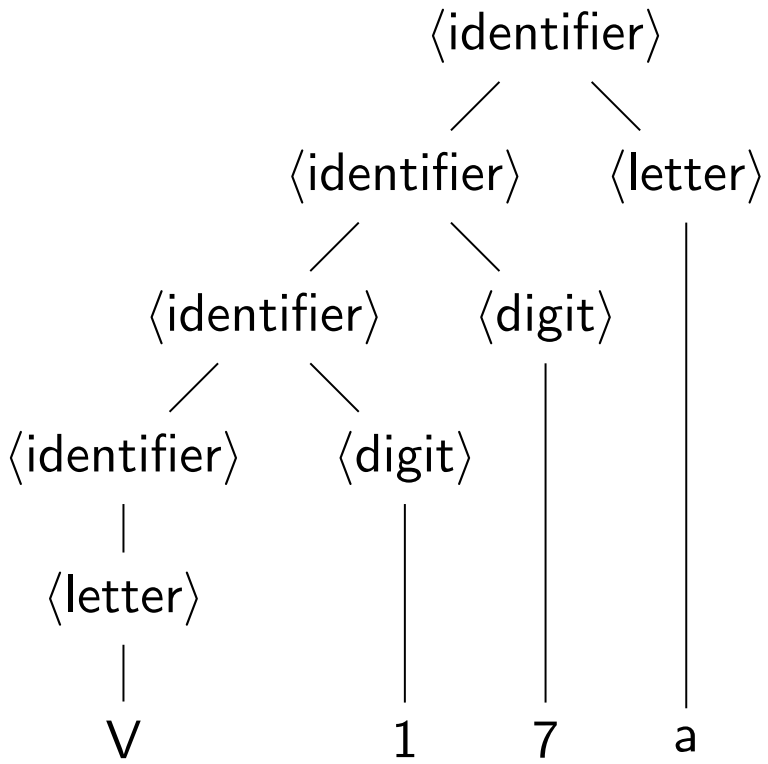
$$
\begin{aligned}
\langle \text{identifier} \rangle &\Rightarrow \langle \text{identifier} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{letter} \rangle \langle \text{digit} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \text{V} \langle \text{digit} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \text{V1} \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \text{V17} \langle \text{letter} \rangle
\end{aligned}
$$

Ableitung (2)

Linksableitung von V17a

$$
\begin{aligned}
\langle \text{identifier} \rangle &\Rightarrow \langle \text{identifier} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{identifier} \rangle \langle \text{digit} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \langle \text{letter} \rangle \langle \text{digit} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \text{V} \langle \text{digit} \rangle \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \text{V1} \langle \text{digit} \rangle \langle \text{letter} \rangle \\
&\Rightarrow \text{V17} \langle \text{letter} \rangle \\
&\Rightarrow \text{V17a}
\end{aligned}
$$

Syntaxbaum / Ableitungsbaum



- Jedem solchen Ableitungs-Prozess kann man einen **Ableitungsbaum**, bzw. **Syntaxbaum**, bzw. **Produktionsbaum** zuordnen.
- Wir zeigen anhand des Ableitungsbaumes, dass V17a ein gültiger ⟨identifier⟩ ist.
- Reihenfolge ist im Ableitungsbaum nicht nachvollziehbar ⇒ somit klar, dass Reihenfolge unerheblich!

Erweiterte Backus-Naur-Form (EBNF)

- Ergänzung um Zeichen { und } in der **Erweiterten Backus-Naur-Form (EBNF)**
- Darin stehende Ausdrücke können *beliebig oft* verwendet werden.
- Verwendung für die Definition von PASCAL

Beschreibung des Sprachelementes ⟨identifier⟩ der Programmiersprache PASCAL mittels EBNF:

⟨digit⟩ ::= 0|1|2|3|4|5|6|7|8|9

⟨letter⟩ ::= $a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|$
$r|s|t|u|v|w|x|y|z|A|B|C|D|E|F|G|$
$H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|$
$W|X|Y|Z$

⟨letter or digit⟩ ::= ⟨letter⟩|⟨digit⟩

⟨identifier⟩ ::= ⟨letter⟩{⟨letter or digit⟩}

Anmerkung: Unterstrich war in der ursprünglichen Definition nicht vorgesehen.

Allgemeine Backus-Naur-Form (ABNF)

- Neben { und } können zusätzlich auch die Symbole **[** und **]** verwendet werden
- Die darin enthaltenen Ausdrücke können *höchstens einmal* verwendet werden.
- Anstatt der spitzen Klammern wird für Nonterminale (Variablen) **Fettdruck** verwendet.
- Die Programmiersprachen ADA und MODULA-2 sind in ABNF definiert.

Allgemeine Backus-Naur-Form (ABNF)

Beschreibung des Sprachelementes ⟨identifier⟩ der Programmiersprache ADA mittels ABNF:

digit ::= 0|1|2|3|4|5|6|7|8|9

letter ::= **lower_case_letter** | **upper_case_letter**

lower_case_letter ::= a | b | c | d | e | f | g | h | i | j | k |
l | m | n | o | p | q | r| s | t| u |
v | w | x | y | z

upper_case_letter ::= A | B | C | D | E | F | G | H | I |J |
K | L | M | N | O | P | Q | R | S | T|
U | V | W | X | Y | Z

letter_or_digit ::= **letter** | **digit**

underscore ::= _

identifier ::= **letter** | { **[underscore] letter_or_digit**}

Mehrdeutigkeit einer Grammatik

- Eine Grammatik G heißt **eindeutig**, wenn für jedes Wort w, das gemäß G aus S ableitbar ist, gilt: die mit unterschiedlichen Ableitungen von w verbundenen Ableitungsbäume sind identisch.
- ebenso, wenn es nur eine einzige Ableitung von w gibt (Spezialfall)
- Eine Grammatik heißt **mehrdeutig**, wenn sie nicht eindeutig ist.
- Ist G mehrdeutig, dann gibt es immer ein w und zwei Ableitungen von w, die unterschiedliche Ableitungsbäume erzeugen.
- In der Praxis von Programmiersprachen werden mehrdeutige Grammatiken nicht benutzt, da sie zu Problemen sowohl bei der Definition der Semantik als auch bei der Programmanalyse führen.
- Eine formale Sprache L kann im i. A. durch mehr als eine Grammatik beschrieben werden.
- Eine formale Sprache L heißt *inhärent mehrdeutig* genau dann, wenn alle Grammatiken für L mehrdeutig sind

Aufgabe 3.1

Die syntaktische Bildung von "sequence" gehorche folgender Menge P von Produktionsregeln:

$$\langle \text{letter} \rangle ::= \text{a} \mid \text{b} \mid \ldots \mid \text{z}$$
$$\langle \text{digit} \rangle ::= 0 \mid 1 \mid \ldots \mid 9$$
$$\langle \text{word} \rangle ::= \langle \textit{letter} \rangle \langle \text{word} \rangle \mid \langle \text{letter} \rangle \langle \text{digit} \rangle$$
$$\langle \text{assignment} \rangle ::= \langle \text{word} \rangle = \langle \text{digit} \rangle$$
$$\langle \text{sequence} \rangle ::= \langle \text{assignment} \rangle; \mid \langle \text{assignment} \rangle, \langle \text{sequence} \rangle$$

a) Wie sehen die restlichen Bestimmungsstücke der Grammatik $G = (N, T, P, S)$ aus?

b) Welche der folgenden Symbolketten sind keine richtige "sequence"? Lokalisieren Sie *alle* Fehler!

- x1=3,y1=7;
- aa1=bb1=0,cc1=1;
- i3=1,k2=i
- s5=25;
- klmno1=0
- a12b=3,
- a=1 bb1=7;
- r25=6
- A7=9;

SPL: Simple Programming Language

- Simple Programming Language (SPL) ist die Definition einer extrem einfachen Programmiersprache zur Veranschaulichung von Grammatiken formaler Sprachen.
- Die Sprache verfügt weder über ein *Typsystem*, noch *Methoden* oder gar *Klassen*.
- Dennoch ist die Sprache ausreichend um alle denkbaren Berechnungen durchzufühen, jedoch vergleichsweise umständlich und unkomfortabel.

SPL: Simple Programming Language

Produktionsregeln zur SPL in BNF:

⟨program⟩ ::= ⟨statement list⟩
⟨statement list⟩ ::= ⟨statement⟩|⟨statement list⟩;⟨statement⟩
⟨statement⟩ ::= ⟨io statement⟩|⟨assignment statement⟩|
⟨conditional statement⟩|⟨definite loop⟩|⟨indefinite loop⟩
⟨io statement⟩ ::= read ⟨variable list⟩
⟨io statement⟩ ::= write ⟨variable list⟩
⟨variable list⟩ ::= ⟨variable⟩ | ⟨variable list⟩,⟨variable⟩
⟨assignment statement⟩ ::= ⟨variable⟩ := ⟨expression⟩
⟨conditional statement⟩ ::= if ⟨comparison⟩ then ⟨statement list⟩ fi |
if ⟨comparison⟩ then ⟨statement list⟩ else ⟨statement list⟩ fi
⟨definite loop⟩ ::= to ⟨expression⟩ do ⟨statement list⟩ end
⟨indefinite loop⟩ ::= while ⟨comparison⟩ do ⟨statement list⟩ end

SPL: Simple Programming Language

⟨comparison⟩ ::= ⟨expression⟩⟨relation⟩⟨expression⟩
⟨expression⟩ ::= ⟨term⟩|⟨expression⟩⟨weak operator⟩⟨term⟩
⟨term⟩ ::= ⟨element⟩|⟨term⟩⟨strong operator⟩⟨element⟩
⟨element⟩ ::= ⟨constant⟩|⟨variable⟩|(⟨expression⟩)
⟨constant⟩ ::= ⟨digit⟩|⟨constant⟩⟨digit⟩
⟨variable⟩ ::= ⟨letter⟩|⟨variable⟩⟨letter⟩|⟨variable⟩⟨digit⟩
⟨relation⟩ ::= = | <= | < | > | >= | <>
⟨weak operator⟩ ::= + | -
⟨strong operator⟩ ::= * | /
⟨digit⟩ ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
⟨letter⟩ ::= a | b | c | d | e | f | g | h | i | j | k | l | m |
n | o | p | q | r| s | t| u | v | w | x | y | z

SPL: Simple Programming Language

Aufgabe 3.2

- Gibt es Syntaxfehler in folgendem SPL-Programm?
- Was würde es tun wenn es korrekt wäre?

```
1   read n;
2   to n do
3     read x;
4     if x>0 then
5       y := 1;
6       z := 1;
7       while z<>x do
8         z := z+1;
9         y := y*z;
10      end;
11      write y
12    fi
13  end
```

SPL: Simple Programming Language

Aufgabe 3.3

Entwickeln Sie einen SPL-Dialekt, indem der Strichpunkt Bestandteil *jeder* Anweisung ist (und nicht nur Anweisungen voneinander trennt).

Definition (Perfekte Zahl)

Unter einer perfekten Zahl versteht man eine natürliche Zahl, deren Wert gleich der Summe *aller* ihrer echten Teiler ist.

Beispiel: $6 = 1 + 2 + 3, \quad 28 = 1 + 2 + 4 + 7 + 14, \quad 496 = \ldots$

Aufgabe 3.4

Entwickeln Sie ein kleines SPL-Programm, das eine positive ganze Zahl n einliest, und die kleinste perfekte Zahl größer n ausgibt.

SPL: Simple Programming Language

Aufgabe 3.5

Zeigen Sie, dass es sich bei

```
while z <> x do
    z := z + 1;
    y := y * z
end
```

um ein gültiges SPL-Statement handelt.

Lösung (3.5): Wir beginnen die Ableitung mit ⟨*statement*⟩ (laut Angabe!). Bei der Ableitung werden einige Schritte übersprungen. Geben Sie bei Tests oder Abgaben aber stets alle Zwischenschritte an!

⟨statement⟩
⇒ ⟨indefinite loop⟩
⇒ while ⟨comparison⟩ do ⟨statement list⟩ fi
⇒ while ⟨expr⟩⟨rel⟩⟨expr⟩ do
⟨statement list⟩;
⟨statement⟩
end
⇒ . . . ⇒
while z <> x do
⟨statement⟩;
⟨statement⟩
end

Fortsetzung (Lösung 3.5)

$\Rightarrow \ldots \Rightarrow$

```
while z <> x do
        ⟨assignment statement⟩;
        ⟨assignment statement⟩
end
```

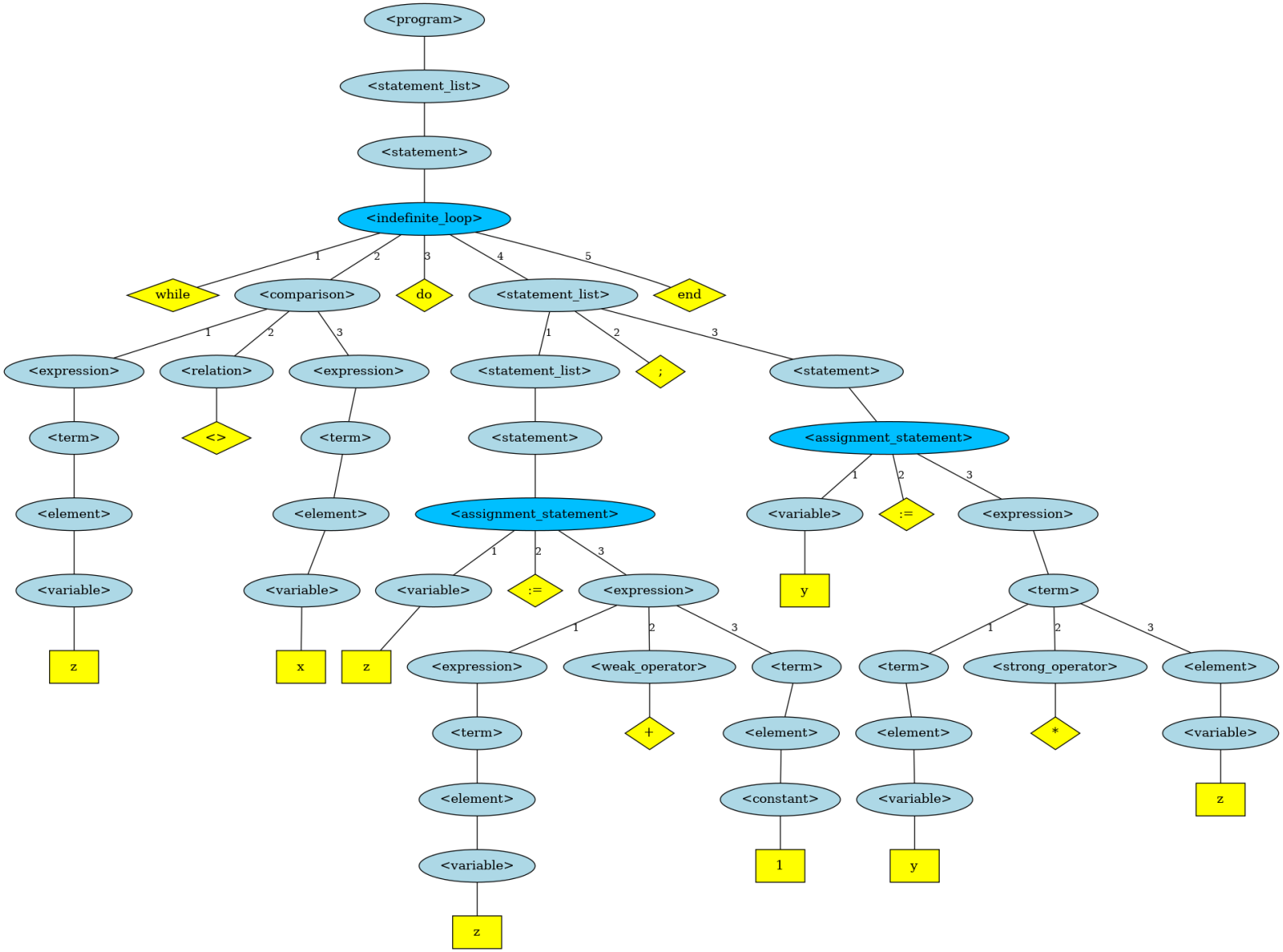
$\Rightarrow \ldots \Rightarrow$

```
while z <> x do
        ⟨variable⟩ := ⟨expression⟩;
        ⟨variable⟩ := ⟨expression⟩
end
```

$\Rightarrow \ldots \Rightarrow$

```
while z <> x do
        z := z + 1;
        y := y * z
end
```

Alternative Lösung zu 3.5 mittels Syntaxbaum:



Aufgabe 3.6

- Gibt es Syntaxfehler in folgendem SPL-Programm?
- Was würde es tun wenn es korrekt wäre?

```
1   read n
2   if n>0 then
3       a = 1;
4       b = 1;
5       write 1;
6       if n>1 then
7           write 1;
8           to n-2 do
9               c = a + b;
10              write c;
11              a = b;
12              b = c;
13          end;
14      end
15  fi
```

Literaturübersicht I

[1] **Berger, Krieger, Mahr: "Grundlagen der elektronischen Datenverarbeitung", Skriptum**

[2] Dirk W. Hoffmann: "Theoretische Informatik", Hanser, 3. Auflage

[3] Gernot Salzer: "Einführung in die Theorie der Informatik", Skriptum, TU Wien, 2001

[4] Wikipedia (Englisch): `https://en.wikipedia.org/`

[5] Wikipedia (Deutsch): `https://de.wikipedia.org/`

[6] `https://cs.au.dk/~amoeller/RegAut/JavaBNF.html`

[7] The Java Language Specification
`https://docs.oracle.com/javase/specs/jls/se8/jls8.pdf`

[8] SPL-Syntaxbaum-Generator:
`https://syntaxbaum.theoretische-informatik.at`