

2023-RO-02 Postfix-Notation

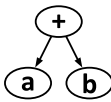
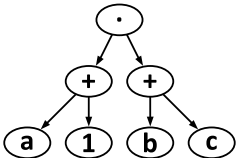
Body

Ein mathematischer Ausdruck besteht aus ...

- ... einem *Operator*: $+$, $-$, \cdot oder $:$
- ... und den *Operanden*: Zahlen wie 1, 2, ..., Buchstaben wie a, b, ... oder wieder Ausdrücke wie $(1 + 2)$.

Die Struktur eines mathematischen Ausdrucks kann man als *Strukturbaum* darstellen. Dieses Diagramm aus Operatoren und Operanden wird so gezeichnet: Ein Kringel mit dem Operator wird durch Pfeile mit den Strukturbäumen der Operanden verbunden. Das sind im einfachsten Fall Kringel mit einer Zahl oder einem Buchstaben.

Aus einem Strukturbaum wiederum kann man die *Postfix-Notation* eines mathematischen Ausdrucks ablesen. In dieser Notation werden für jeden Ausdruck zunächst die Operanden und dahinter der Operator geschrieben.

Mathematischer Ausdruck:	$a + b$	$(a + 1) \cdot (b + c)$
Strukturbaum:		
Postfix-Notation:	$a\ b\ +$	$a\ 1\ +\ b\ c\ +\ \cdot$

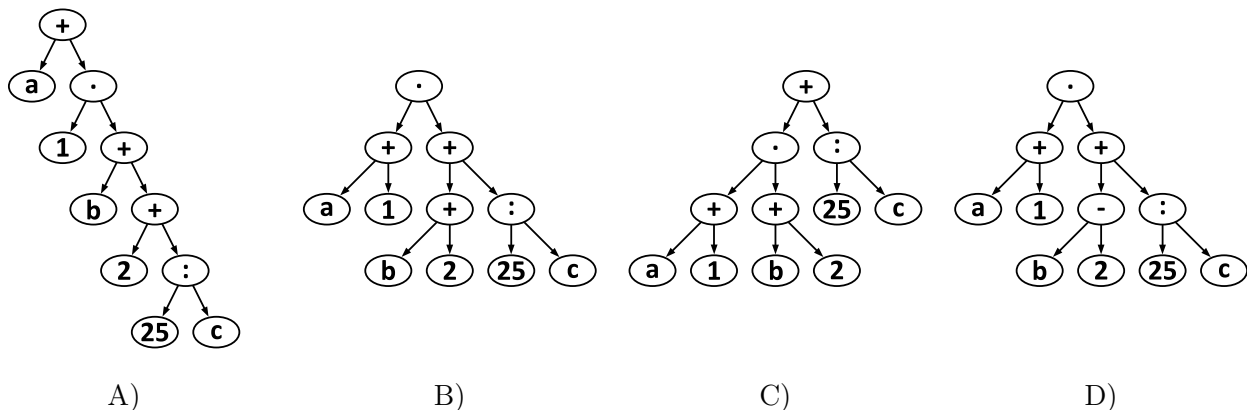
Hier ist die Postfix-Notation eines anderen Ausdrucks:

$a\ 1\ +\ b\ 2\ +\ \cdot\ 25\ c\ :\ +$

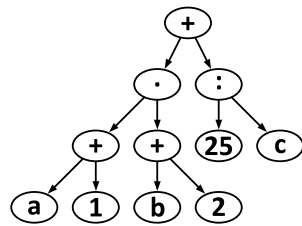
Question/Challenge - for the brochures

Welchen Strukturbaum hat dieser Ausdruck?

Answer Options/Interactivity Description



Answer Explanation



Antwort C ist richtig:

Wie in der Aufgabenstellung beschrieben ist, findet sich der zentrale Operator eines mathematischen Ausdrucks im Strukturbaum ganz oben (er ist dessen *Wurzel*) und in der Postfix-Notation ganz hinten. Möchte man den Strukturbaum eines Ausdrucks in Postfix-Notation finden oder erstellen, muss man das in der Postfix-Notation letzte Zeichen oben im Baum suchen, in diesem Fall das $+$. Nur bei den Bäumen der Antworten A und C findet sich ein $+$ in der Wurzel.

Der Operator $+$ hat zwei Operanden, einer links und einer rechts. In der Postfix-Notation sieht man direkt (am vorletzten Zeichen), dass der rechte Operand des Ausdrucks wiederum ein Ausdruck ist, der den Operator $:$ hat. Im Strukturbaum muss also rechts unter der Wurzel ein $:$ zu sehen sein. Das ist nur im Baum aus Antwort C der Fall. Also muss das die richtige Antwort sein.

Das kann man auch zeigen, indem man den Strukturbaum aus Antwort C vollständig in Postfix-Notation umwandelt:

- Die untersten drei «kleinsten» Teilbäume, die jeweils aus 3 Elementen bestehen, werden zu $a\ 1\ +$, $b\ 2\ +$ und $25\ c\ :$.
- Die beiden linken dieser drei «kleinsten» Teilbäume werden zum linken Operanden des oberen $+$, so dass die Umwandlung des linken Teilbaum nun $a\ 1\ +\ b\ 2\ +\ \cdot$ lautet. Der dritte der «kleinsten» Teilbäume ist bereits der rechte Operand.
- Damit lautet die Postfix-Notation des Strukturbaums aus Antwort C insgesamt so: $a\ 1\ +\ b\ 2\ +\ \cdot\ 25\ c\ : +$. Das ist genau der in der Aufgabenstellung vorgegebenen Ausdruck.

This is Informatics

Die *Postfix-Notation*, auch *umgekehrte polnische Notation* (engl. *reverse Polish notation RPN*) genannt, wird oftmals verwendet, um mathematische oder andere Ausdrücke (z. B. in Programmiersprachen) missverständnisfrei und kompakt zu formulieren. Würde man den durch den Strukturbaum aus Antwort C gegebenen Ausdruck in normaler Notation schreiben (also mit Operatoren zwischen den Operanden, deshalb auch *Infix-Notation* genannt), müsste man Klammern setzen $(a + 1) \cdot (b + 2) + 25 : c$, die man für die Postfix-Notation nicht braucht. Die Postfix-Notation wurde von Jan Łukasiewicz (1878–1956) zunächst als Präfix-Notation eingeführt, mit dem Operator vor den Operanden. So schreibt man u.a. die Anwendung von Funktionen auf: in der Mathematik $f(x, y)$, beim Programmieren `funktionsname(argument1, argument2, argument3)`. Im Computer wird sie unter anderem beim *Parsen* von Ausdrücken einer Programmiersprache verwendet.

In der jüngeren Vergangenheit haben viele Menschen die Postfix-Notation vor allem bei der Nutzung der

ersten wissenschaftlichen Taschenrechner kennengelernt: Dort konnte man damit schnell und zuverlässig und vor allem ohne Klammern komplexe mathematische Ausdrücke eingeben und berechnen lassen. Noch heute gibt es eine eingeschworene Gemeinschaft, die programmierbare Taschenrechner (wie z. B. den HP 35s) mit Postfix-Notation nutzen.

This is Computational Thinking

Um die Aufgabe zu lösen, muss ein komplexer mathematischer Ausdruck durch *Zerlegen* oder *Dekomposition* in kleinere mathematische Ausdrücke verwandelt werden, die für sich im Kontext verständlich sind. Zudem muss ein Vorgang dahingehend *analysiert* werden, dass seine Funktionsweise auch in rückwärtiger Reihenfolge verständlich wird. Im Grunde wird dabei dieselbe Information zwischen unterschiedlichen Darstellungsweisen konvertiert.

Informatics Keywords and Websites

- Umgekehrte polnische Notation UPN: https://de.wikipedia.org/wiki/Umgekehrte_polnische_Notation
- Syntaxbaum: <https://de.wikipedia.org/wiki/Syntaxbaum>
- Jan Łukasiewicz: https://de.wikipedia.org/wiki/Jan_Łukasiewicz
- HP 35s: https://en.wikipedia.org/wiki/HP_35s

Computational Thinking Keywords and Websites

- Zerlegen, Dekomposition: <https://de.wikipedia.org/wiki/Modell#Modellbildung>