Operatoren in Kontrollstrukturen

Operatoren

Arithmetische Operatoren

Grundrechenarten

- + Addition
- Subtraktion
- * Multiplikation
- / Division

Modulo-Operator

% bestimmt den Rest

Involviert sind immer zwei Operanden.

Es gelten die selben Regeln wie in der Mathematik:

- Punktrechnung vor Strichrechnung
- Durch Klammernsetzung () wird der Ausdruck in den Klammern zuerst ausgewertet

Arithmetische Operatoren

Beispiele

Arithmetische Operatoren: Verkürzt geschrieben

Da oft der Wert einer Variable mit sich selbst berechnet wird, gibt es dafür eine verkürtzte Schreibweise mit folgenden Operatoren:

- += Addition mit sich selbst
- -= Subtraktion mit sich selbst
- /= Division mit sich selbst
- *= Multiplikation mit sich selbst
- %= Rest von Division mit sich selbst

In gut Deutsch

Ein Wert wird mit sich selber berechnet

In Java

```
int i = 15; // Startwert von i ist 15

int k = 3; // Startwert von k ist 3

i += 5; // i = i + 5; \rightarrow i = 20

i -= 7; // i = i - 7; \rightarrow i = 16

i /= k; // i = i / k; \rightarrow i = 5

i *= 7; // i = i * 7; \rightarrow i = 35

i %= 4; // i = i % 4; \rightarrow i = 3
```

Unäre Operatoren: ++ und --

- besitzen kein Gleichheitszeichen
- beziehen sich auf sich selbst
- also nur den "linken" Teil einer Operation
- werden ausschliesslich zum Hochund Runterzählen verwendet
 - ++ Inkrementierung
 - o -- Dekrementierung

In gut Deutsch

In Einerschritten sich selbst Hoch- oder Runterzählen

In Java

```
int i = 3; // Startwert für i ist 3 double d = 1.5; // Startwert für d ist 1.5 i++; // i = i + 1 \rightarrow i = 4 d--; // d = d - 1.0 \rightarrow d = 0.5
```

Vergleichsoperatoren

- Mit Vergleichsoperatoren lassen sich Ausdrücke formulieren, welche einen Rückgabewert vom Typ boolean liefern.
- Eine Variable vom Typ boolean kann die Werte true oder false annehmen.
- Durch Vergleichsoperatoren können Wahrheitswerte ermittelt werden
- Diese können für Entscheidungen im Programmverlauf verwendet werden.
 - o if, while
- Alle Vergleichsoperatoren:
 - o == , != , < , > , <= , >=

Vergleichsoperatoren: Tabelle

Gegeben: int a = 2; int b = 3;

Operator	Beschreibung	Beispiel	Resultat
==	überprüft auf <u>Gleichheit</u>	a == b	false
!=	überprüft auf <u>Ungleichheit</u>	a != b	true
>	ist linker Operand <u>grösser</u>	a > b	false
<=	ist linker Operand <u>grösser oder gleich</u>	a <= b	true
<	ist linker Operand <u>kleiner</u>	a < b	true
>=	ist linker Operand <u>kleiner oder gleich</u>	a >= b	false

^{*} nur bei primitiven Datentypen. Nicht bei String!

Logische Operatoren: Kombination von bool'shen Ausdrücken

Gegeben: boolean a = true; boolean b = false;

Operator	Beschreibung	Beispiel	Resultat
&&	UND: beide Ausdrücke sind true	a && b	false
	ODER: mindistens ein Ausdruck ist true	a b	true
٨	XOR: genau einer der Ausdrücke ist true	a ^ b	true
!	NOT: wandelt ein boolean ins Gegenteil um	! b	true

Kontrollstrukturen

Vergleichs- und logische Operatoren kommen häufig dann zum Einsatz, wenn man etwas nur unter einer bestimmten Bedingung ausführen soll.

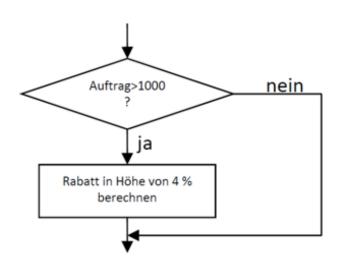
Kontrollstruktur: if / else if / else

- Ein if-Statement wird dafür verwendet, Bedingungen zu überprüfen
- Kontrolle über einen Codeabschnitt, der nur unter besonderen Bedingungen durchlaufen wird.
- Als Bedingung dient ein **Bool'scher Wert** (true, false), welche über ein **Vergleichsoperator** erzeugt wird.

Kontrollstruktur: if / else if / else

Beispiel

Wenn, if, ein Kunde einen Auftrag über 1000.-- erteilt, bekommt er 4 % Rabatt.



In Java

```
double price = StdInput.readDouble();

if (price > 1000) {
    price *= 0.96;
}

System.out.println("Your price " + price);
```

- Bedingung: price > 1000
 - Operator: > grösser als
- Anweisung: price *= 0.96
 - ∘ Oder: price = price * 0.96

Kontrollstruktur: if / else if / else

Mit else if kann priorisiert auf weitere Bedingungen reagiert werden

Schema

if-Statements können beliebig verschachtelt werden!

Code-Beispiel

```
int age; // beliebiges alter
double betrag; // beliebiger Betrag
if (betrag > 10000 && age < 18) {
    // mehr als 10000 ausgegeben
    // UND unter 18 Jahre alt
    betrag *= 0.9;
} else if(betrag > 1000) {
    betrag *= 0.96;
} else { // Für alle andern
    betrag *= 0.98;
}
```