J. Rau 08/2024

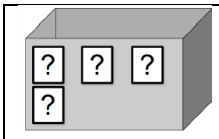
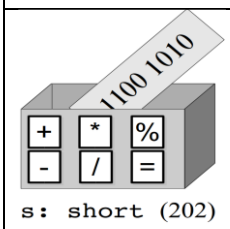
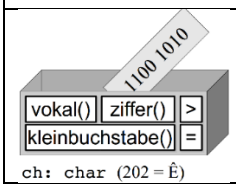
Datentypen und Variable

Wenn wir in Java möchten Werte speichern und später wieder verwenden, interessiert es uns überhaupt nicht, wo im Speicher der Wert abgelegt wird.
Wir vergeben einen Namen ein, der als Platzhalter für die Speicheradresse dient.

Es ist die Aufgabe des Compilers und der virtuellen Java Maschine dem Namen eine Speicheradresse zuzuordnen.

Es ist aber wichtig zu wissen, wieviel Platz zur Speicherung eines Wertes benötigt wird. Dies wird durch den Datentyp festgelegt, deshalb gehört zu einer Variablen immer ein Datentyp.
Wird der Speicherplatz der Variablen während des gesamten Programmlaufs reserviert, ist es eine globale Variable.
Wird der Speicherplatz nur kurz genutzt, ist es eine lokale Variable.

Der tatsächliche Wert wird der Variablen durch passende Parameter zugewiesen.

	*Eine Variable kann man sich als Kiste vorstellen. In diese Kiste werden die Werte der Variable gelegt. Die Variable hat einen Datentyp. Wir können uns eine Variable als Fernsteuerung für die Inhalte vorstellen.*
	*Das Bitmuster «1100 1010» wird in der Variablen s (vom Datentyp short) gespeichert. Die Variable können wir nun addieren, subtrahieren, vergleichen etc. Als ganze Zahl (short) hat diese Variable den Wert 202. Der Datentyp short besteht aus 16 Bit besteht. Beachten Sie, dass unser Bitmuster zwar nur 8 Bit benötigt, aber die vorangehenden 8 Bit werden automatisch mit Nullen «0000 0000» aufgefüllt.*
	*Bei der Variablen ch (vom Datentyp char), hat dasselbe Bitmuster eine ganz andere Bedeutung. Wir können die Variable prüfen (istZiffer( ) , istVokal()), aber auch verändern (z. B. kleinBuchstabe()).* *Als Unicode-Zeichen hat das Bitmuster den Wert Ê.*

Die Datentypen byte, short, int, double, char und boolean werden ohne zusätzliche Festlegungen angegeben:
z.B. `byte b=5; short s=5000; double b=1.2; char c='x'; boolean bo=true;`

Die Datentypen float und long müssen explizit als solche ausgewiesen werden:
`z.B. long l=1234567889L; float fl=4.7F`
Überschreitet der Wert den Zahlenbereich meldet der Compiler einen Fehler.

Gebrochene Zahlen

In Java ist der Datentyp einer gebrochenen Zahl mit double (oder float) festgelegt.

In Java handelt es sich bei Dezimalzahlen immer um einen Näherungswert**.**
Es ist nicht möglich die Zahl Pi oder 1/7 exakt darzustellen. Nicht einmal die Zahl 0.1 ist genau darstellbar, da die Brüche als Summe von Potenzen 2^{-1}, 2^{-2} …2^{-n} abgebildet werden.
Dabei ist eben 0.1 ein nicht abbrechender „Binärbruch“.

J. Rau 08/2024

Wissenschaftliche Notation

Die Exponentialschreibweise sagt einfach aus, um wie viele Stellen der Dezimalpunkt verschoben werden soll:

z.B. `5.7 e+6 = 5.700.000; 4.35353535354 e-2 = 0.0435353535354`

`double d = 3.00 e+8; float fl = 3.00E+8F;`

Typumwandlung (Casting)

In Java muss oft zwischen gebrochenen und ganzen Zahlen gewechselt werden. Aus einer ganzen Zahl eine gebrochene zu erhalten, ist trivial.

Weil durch die Umwandlung in der Regel Nachkommastellen verloren gehen, bedarf die Umwandlung in die andere Richtung der Zustimmung durch den Programmierer (Casting = explizite Typumwandlung),

Beispiel (1): `int i; double d;`

```
// Diese Umwandlung ist trivial:      i = 4; d = i;                  // 4.000
// Dies ist nur mit Casting erlaubt:  d = 3.1104; i = (int) d;       // i hat Wert 3
```

Beispiel (2): `int i = 22; float fl;`

```
// Diese Umwandlung ist trivial:      fl = i;                        // Wert  22.0f
```

Der Java-Compiler verhindert folgende Zuweisung (Lost of precision):

```
                                      fl = 3.14f; i = fl;
// Casting erlaubt:                   i = (int) fl; // i ==> 3fl ==> 3.14f
```

Aufgaben:

Erkläre kurz, was beim Deklarieren der Variablen passiert und ob die Deklaration gültig ist:

```
byte b1=-128;
```

```
byte b2=128;
```

```
short s=6000;
```

```
int i=23456;
```

```
long l1=23456782345678;
```

```
double d=4,0;
```

```
float f=3.14;
```

```
double d2=4.0d;
```

```
int a,b,c=5;
```
