- Java kennt folgende Datentypen für ganze Zahlen:
  - `byte`: besteht aus 8 Bits und hat den Wertebereich -128 bis +127
  - `short`: besteht aus 16 Bits und hat den Wertebereich -32768 bis 32767
  - `int`: besteht aus 32 Bits und hat den Wertebereich -2147483648 bis 2147483647
  - `long`: besteht aus 64 Bits und hat den Wertebereich -9223372036854775808 bis 9223372036854775807
  - in den meisten Fällen ist der `int` die geeignete Wahl
- Das Zweierkomplement ist ein Darstellungs bzw. Interpretationssystem für vorzeichenbehaftete ganze Zahlen. Mit dem Zweierkomplement können wir negative und positive ganze Zahlen kodieren.
  - alle positiven Zahlen haben den Wert 0 in ihrem höchstwertigen Bit
  - alle negativen Zahlen haben den Wert 1 in ihrem höchstwertigen Bit
  - um die Binärdarstellung einer negativen Zahl zu ermitteln, geht man wie folgt vor:
    - man ermittelt die Binärdarstellung der positiven Zahl
    - anschließend invertiert man die Binärzahl (0 -> 1 und aus 1 -> 0)
    - anschließend addiert man den Wert 1
    - das Ergebnis ist die Binärdarstellung der negativen Zahl
    - Beispiel: +1 = 0001 -> Invertieren -> 1110 -> 1 addieren -> 1111 = -1
    - Vorteil: Bei Verwendung des Zweierkomplements kann die bestehende Schaltlogik für positive Zahlen in der CPU verwendet werden.
- Eine Gleitkommazahl / Fließkommazahl ist eine Zahl, die sich wie folgt darstellen lässt: (-1)^VZ * 2^E * Mantisse
  - nicht alle Dezimalzahlen lassen sich exakt in Binärform darstellen
  - bei der Verrechnung von Gleitkommazahlen treten Rundungsfehler auf
  - Gleitkommazahlen können sehr groß, aber auch sehr klein sein, dazwischen liegen jedoch sehr viele "Lücken"
  - Der Vergleich zweier Gleitkommazahlen führt oft nicht zum erwarteten Ergebnis, z.B. kann (0.1 + 0.1 + 0.1) != 0.3 sein. Tipp: Berechne die Differenz zweier zu vergleichender Zahlen und schaue nach, ob diese einen bestimmten Schwellwert unterschreitet. Ist das der Fall, kann man beide Zahlen als gleich betrachten.
- Java kennt zwei Datentypen für Gleitkommazahlen
  - `float`: er besteht aus 32 Bits. 1 Bit wird für das Vorzeichen verwendet, 8 Bits für den Exponenten zur Basis 2, sowie 23 weitere Bits für die eigentliche Zahl (Mantisse). Faustregel: Der float kann ca. 6-8 Ziffern exakt darstellen. Der Wertebereich ist: +/- 1.4E-45 bis +/- 3.4028235E38.
  - `double`: er besteht aus 64 Bits, 1 Bit wird für das Vorzeichen verwendet, 11 Bits für den Exponenten und 52 Bits für die Mantisse. Faustregel: ca. 15 Ziffern können exakt abgebildet werden. Wertebereich ist: +/- 4.9E-324 bis +/- 1.7976931348623157E308.
- Sehr große und sehr kleine Zahlen werden in Java meistens in der wissenschaftlichen Notation dargestellt, z.B. bedeutet 3.21E-12 nichts anderes als 3.21 * 10^(-12) bzw. 3.21 / 10^12.
- Tipp: Wenn du mit Geldbeträgen arbeiten möchtest, verzichte am besten auf Gleitkommazahlen. Verwende stattdessen Integer. Zum Beispiel kannst du einen Eurobetrag in Cent umrechnen. Alternativ kannst du in Java die Klasse BigDecimal nutzen.
- Java kennt den Datentyp `boolean` für Wahrheitswerte: `true` und `false`. Hinweis: Booleans lassen sich in Java nicht automatisch in Integer umwandeln.
- Der Datentyp `char` kann genau ein Zeichen abspeichern. Intern wird die UTF-16 Kodierung verwendet (eine Kodierung für den Unicode Zeichensatz). Achtung: Werte für char-Variablen werden in einfache Hochkommas gesetzt, also `'x'`.

- Wenn zwei Integer miteinander dividiert werden, führt Java eine Integer-Division durch, wodurch die Nachkommastellen abgeschnitten werden. Beispiel: `7 / 2` ergibt `3` statt 3.5.