

AB 2.2-8: Weitere zwölf Aufgaben zu if-else

Übung 2.2.8 #1

Der folgende Quelltext soll die Werte von x und y vertauschen, wenn x größer ist als y. Überprüfe, ob der Quelltext korrekt funktioniert. Falls das nicht der Fall sein sollte, schreibe den Quelltext so um, dass er korrekt funktioniert.

```
if (x > y)
    int swap = x;
    x = y;
    y = x;
```

Übung 2.2.8 #2

Eine Methode soll Flüssigkeitsmengen in eine leicht lesbare Form umwandeln und als String zurückgeben. Die Flüssigkeitsmenge **in Litern** wird als double-Parameter `volumen` übergeben.

```
public String getVolumenLesbar(double volumen)
```

Hier ein paar Beispiele für mögliche Ausgaben:

double volumen	String
4.0	"ca. 4 Liter"
4.3	"ca. 4 Liter"
3.8	"ca. 4 Liter"
0.2	"ca. 20 Zentiliter"
0.18	"ca. 18 Zentiliter"
0.0032	"ca. 3 Milliliter"
0.0017	"ca. 2 Milliliter"

Zum Runden könnt ihr die Math-Methode `Math.round()` verwenden. Beispiel

```
int x = (int) Math.round(7.6); // ⇒ 8
int y = (int) Math.round(7.4); // ⇒ 7
```

Fehlerquelle: `Math.round()` liefert für double-Werte einen long-Wert zurück, keinen int-Wert.

Daher ist ein **Typecasting** notwendig.

Alternativ könnte man `volumen` als `float` deklarieren. Dann ergibt `round()` einen `int`-Wert.

Übung 2.2.8 #3

Für ein Reisevergleichsportal sollen Reiserücktritt-Versicherungen von drei Versicherungs-Unternehmen verglichen werden.

- A. Prämie = 50 Euro + 4% vom Reisepreis
- B. Prämie = 5% vom Reisepreis
- C. Prämie = 100 Euro + 3,5% vom Reisepreis

Schreibt eine Methode, die berechnet, welche Versicherung für den gegebenen Reisepreis die günstigste ist. Der Reisepreis wird als double-Parameter übergeben, die ausgewählte Versicherung kann als char-Wert zurückgegeben werden oder mit `System.out.println()` direkt ausgegeben werden.

Übung 2.2.8 #4a (einfache Version)

Schreibt eine Methode

```
public void zeigeJahreszeit(int monat)
```

die - abhängig von dem Parameter `monat` - die jeweilige Jahreszeit mit `System.out.println()` ausgibt.

Hinweis: Der Winter umfasst die Monate Dezember bis Februar, der Frühling die Monate März bis Mai, der Sommer den Juni bis August und der Herbst schließlich den September bis November.

Übung 2.2.8 #4b (anspruchsvollere Version)

Schreibt ähnlich wie in der einfachen Version eine Methode

```
public void zeigeJahreszeit(int monat, int tag)
```

In dieser anspruchsvolleren Version müsst ihr auch den Parameter `tag` berücksichtigen. Der Winter beginnt nämlich am 21. Dezember, der Frühling am 21. März, der Sommer am 21. Juni und der Herbst am 21. September.

Übung 2.2.8 #5

Erstellt eine Methode

```
public int getSchraubenTyp(int durchmesser, int laenge)
```

die den Typ der Schraube zurückliefert.

Dabei gilt:

- Durchmesser bis zu 3 mm, Länge bis zu 20 mm = Typ 1
- Durchmesser von 4 bis 6 mm, Länge von 21 bis 30 mm = Typ 2
- Durchmesser von 7 bis 20 mm, Länge von 31 bis 50 mm = Typ 3
- andere Werte = Typ 0 (unbekannter Schraubentyp)

Übung 2.2.8 #6

Die sogenannte **Diskriminante** D einer quadratischen Gleichung $ax^2 + bx + c$ wird folgendermaßen berechnet: $D = b*b - 4*a*c$.

Mit der Diskriminante kann man die Zahl der Lösungen dieser Gleichung bestimmen: $D > 0$: Zwei Lösungen; $D < 0$: keine Lösung; $D = 0$: eine Lösung.

Erstellt eine Methode mit den Parametern a , b und c , die dann die Anzahl der Lösungen der Gleichung ausgibt.

Übung 2.2.8 #7

Schreibt eine Methode

```
public void sortiere(int a, int b, int c)
```

die die drei Zahlen a, b und c in sortierter Reihenfolge ausgibt. Ihr dürft dabei nur if- oder if-else-Anweisungen benutzen.

Übung 2.2.8 #8

Schreibt eine Methode

```
public boolean sortiert(int a, int b, int c, int d)
```

die überprüft, ob die vier Zahlen aufsteigend sortiert sind.

Übung 2.2.8 #9

Schreibt eine Methode

```
public boolean istDreieck(int a, int b, int c, int d)
```

die überprüft, ob die drei Längen a, b und c ein Dreieck bilden können.

Dazu muss gelten:

$$a + b > c, \quad a + c > b, \quad b + c > a$$

Übung 2.2.8 #10

Schreibt eine Methode

```
public boolean istReihe(int a, int b, int c)
```

die überprüft, ob die drei Zahlen aufeinanderfolgen – also z. B. 3 4 5 oder 10 11 12.

Die Reihenfolge muss aufsteigend sein, und der Abstand zwischen den Zahlen soll jeweils genau 1 betragen.

Übung 2.2.8 #11

Schreibt eine Methode

```
public boolean istArithmetisch(int a, int b, int c)
```

die überprüft, ob die drei Zahlen eine arithmetische Reihe bilden. Der Abstand zwischen den drei Zahlen muss also gleich groß sein, beispielsweise 3, 5, 7 oder 2, 6, 10.

Übung 2.2.8 #12

Schreibt eine Methode

```
public String dreiecksTyp(int a, int b, int c)
```

die die drei Längen a, b, c eines Dreiecks überprüft und dann den Typ des Dreiecks als String zurückgibt:

- "kein Dreieck"
- "gleichseitiges Dreieck"
- "gleichschenkliges Dreieck"
- "ungleichseitiges Dreieck"

Auf "rechtwinkliges Dreieck" muss aber nicht getestet werden, das ist zu schwer.

Wer aber Ahnung von Mathe bzw. Geometrie hat, kann sich ja mal daran versuchen.

Als Hilfsmethode könnte ihr

```
public boolean istDreieck(int a, int b, int c, int d)
```

aus Übung #9 einsetzen.