

AB 3.2-1: While-Schleifen

Übung 3.2 #2

Schreibt eine Methode, die alle Potenzen von 2 ausgibt, die kleiner als 1000 sind.

Aufgabe 3.2 #3

Was macht der folgende Quelltext? Wie funktioniert er? Welche Aufgabe hat `var2` ?

```
public void keineAhnung(int zahl)
{
    int var1 = 0;
    int var2 = zahl;

    while (zahl > 0)
    {
        int irgendwas = zahl % 10;
        var1 += irgendwas;
        zahl = zahl / 10;
    }

    System.out.printf("Ergebnis von %6d ist %4d%n", var2, var1);
}
```

Übung 3.2 #4

Schreibt eine Methode

```
public int fakultaet(int zahl)
```

die die **Fakultät** einer ganzen Zahl (1, 2, 3, ..., 10) berechnet. Bei Zahlen, die nicht in dem Intervall [1, 10] liegen, soll der Wert -1 zurückgegeben werden.

Übung 3.2 #5a

Schreibt eine Methode

```
public int kleinsterTeiler(int zahl)
```

die den kleinsten Teiler größer 1 dieser Zahl bestimmt. Beispiel zahl = 15, kleinster Teiler = 3. Zahl = 35, kleinster Teiler = 5. Zahl 13, kleinster Teiler = 13.

Tipp: Verwendet dazu den Modulo-Operator %

Übung 3.2 #5b

Schreibt eine Testmethode für `kleinsterTeiler()` aus Aufgabe 5a, und lasst mithilfe einer `while`-Schleife die kleinsten Teiler der Zahlen von 1 bis 150 ausgeben.

Übung 3.2 #5c

Modifiziert die Testmethode so, dass nur noch die Primzahlen im Bereich 1 bis 200 ausgegeben werden. Die Konsolenausgabe dieser Testmethode könnte beispielsweise so aussehen:

```
Die Zahl      1 ist die   1.te Primzahl
Die Zahl      2 ist die   2.te Primzahl
Die Zahl      3 ist die   3.te Primzahl
Die Zahl      5 ist die   4.te Primzahl
Die Zahl      7 ist die   5.te Primzahl
Die Zahl     11 ist die   6.te Primzahl
Die Zahl     13 ist die   7.te Primzahl
Die Zahl     17 ist die   8.te Primzahl
```

Übung 3.2 #6

Schreibt eine Methode

```
public void funktionstabelle(double xStart, double xEnde, double sw)
```

welche die `x`- und `y`-Werte einer Funktion $y = f(x)$ tabellarisch ausgibt. Dabei bestimmen `xStart` und `xEnde` den Anfang und das Ende des Definitionsbereichs der Funktion, und `sw` gibt die Schrittweite an, mit der der `x`-Wert inkrementiert werden soll.

Die Funktion $f(x)$ selbst wird als eigene Methode implementiert, zum Beispiel

```
private double funktion(double x)
{
    return 2*x*x - 3*x +3;
}
```

In der Methode `funktionstabelle()` könnt ihr dann auf diese private-Methode zugreifen:

```
y = funktion(x)
```

wobei `y` eine `double`-Variable und `x` die Laufvariable der `while`-Schleife ist.

Übung 3.2 #7

Modifiziert die Methode aus 3.2 #6 so, dass sie am Ende der Funktionstabelle ausgibt, wo das Minimum der Funktion liegt (also `x`-Wert des Minimums ausgeben).

Ergänzung für Experten

Wenn das Minimum lokalisiert wurde, wird der Definitionsbereich verengt (`xmin-sw` bis `xmin+sw`) und die Schrittweite verringert (`sw = sw/10`) und das Minimum erneut gesucht. Modifiziert eure Methode so, dass sie das automatisch macht.