Java08 – while-Schleife

Die grundlegenden Kontrollstrukturen in nahezu allen Programmiersprachen sind:

- die **`for`**-Schleife (Wiederholung mit fester Anzahl) [vorige Stunde],
- die **`while`**-Schleife (bedingte Wiederholung) **[heute]** und
- die **`if`**-Anweisung (bedingte Anweisung) [demnächst].

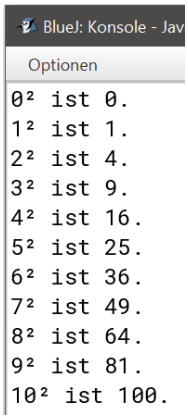
Schritt 1 – BlueJ-Projekt „Java_03_Konsolenausgabe“ öffnen

Öffne Dein BlueJ-Projekt „**Java_03_Konsolenausgabe**“ der letzten Stunden.

Schritt 2 – Quelltext der Klasse „Schleife“ öffnen

Öffne den Quelltext mit einem Doppelklick auf das Klassen-Symbol der Klasse „Schleife“.

Schritt 3 – Methoden mit while-Schleife zur Erzeugung einer Liste von Quadratzahlen



1. AUFGABE: Methode „`public void whileQuadratzahlen10()`“ **erstellen**

Wir wollen mit Hilfe einer while-Schleife eine *Liste der ersten **10** Quadratzahlen* ausgeben.

Kopiere dazu den folgenden Code in den Quelltext unterhalb der letzten „for-Methode“, aber noch vor der letzten schließenden geschweiften „Klassen-Klammer“:

```
public void whileQuadratzahlen10() {
    int i = 1;
    while (i <= 10) {
        System.out.println(i + "² ist " + (i*i) + ".");
        i = i + 1;      // alternativ: i++;
    }
}
```

WICHTIG: Übersetze, erzeuge ein Objekt der Klasse „Schleife“ und **teste**, ob die Methode „tut, was sie soll“!

Ablauf *dieser* while-Schleife:

1. Zuerst wird die Anweisung `int i = 1` ausgeführt:
   Die lokale Variable **`i`** wird als ganze Zahl („`int`“) deklariert und mit dem Wert 1 initialisiert.
2. In der „Bedingungsklammer“ nach „while“ wird geprüft, ob die Bedingung `i <= 10` wahr ist.
3. Wenn sie wahr ist, wird der Rumpf der **`while`**-Schleife ausgeführt. Andernfalls wird das Programm mit den nach der **`while`**-Schleife ggf. stehenden Anweisungen fortgesetzt.
4. Nun wird die Anweisung `i = i + 1` (alternativ: `i++`) ausgeführt .
   Der Wert der lokalen Variablen **`i`** wird dadurch um 1 erhöht (inkrementiert).
   Am Ende des Durchgangs wird die **`while`**-Schleife an der Stelle 2. wieder fortgeführt.

2. AUFGABE: Methode „`public void whileQuadratzahlenN(int n)`“ **erstellen**

Wir wollen auf der Konsole mit Hilfe einer while-Schleife eine *Liste der ersten **n** Quadratzahlen* ausgeben.

Beim Aufruf dieser Methode soll eine Abfrage erfolgen, wie viele Quadratzahlen erzeugt werden sollen.

Dies erreicht man, indem man im Methoden-Kopf in den **runden Parameter-Klammern** einen Parameter n vom Typ „ganze Zahl“ („`int`“) aufführt.

Implementiere (d. h.: Programmiere) diese Methode „`public void whileQuadratzahlenN(int n)`“.
Orientiere Dich ggf. an der Methode „`public void forQuadratzahlenN(int n)`“ der letzten Stunde.

WICHTIG: Übersetze, erzeuge ein Objekt der Klasse „Schleife“ und **teste**, ob die Methode „tut, was sie soll“!

Die while-Schleife im Vergleich zur for-Schleife

Bei der **`for`**-Schleife (Wiederholung mit fester Anzahl) muss vorher bekannt sein, wie oft die Schleife durchlaufen werden soll.

Die **`while`**-Schleife (bedingte Wiederholung) ist „mächtiger“. Bei dieser Kontrollstruktur wird die Schleife so oft durchlaufen, solange (while ≙ solange) eine bestimmte Bedingung erfüllt ist.

Schritt 4 − Methode zur Eingrenzung der Wurzel aus einer natürlichen Zahl

3. AUFGABE: Methode „`wurzelZiehen(int zahl)`“ erstellen

Eine Möglichkeit, um herauszufinden, zwischen welchen natürlichen Zahlen die Wurzel einer natürlichen Zahl (hier: die als Parameter der Methode übergebene Zahl **`zahl`**) liegt, ist die folgende:

Man quadriert von 0 beginnend die natürlichen Zahlen, solange das Quadrat kleiner als die Zahl ist.

Wenn das Quadrat nicht mehr kleiner als die Zahl ist, bricht man damit ab.

Kopiere den folgenden Code unterhalb von „`whileQuadratzahlenN(int n)`“, aber noch vor der letzten schließenden geschweiften „Klassen-Klammer“.
(Die hier erscheinenden Zeilenumbrüche bei ** musst Du löschen!)

```
    public void wurzelZiehen(int zahl) {
        int x0 = 0;
        while (x0 * x0 < zahl) {
            int q0 = x0 * x0;
            System.out.println("Das Quadrat von "+ x0 +" ist "+ q0 +" und
** somit kleiner als " + zahl + ".");
            x0 = x0 + 1;
        }
        int q0 = x0 * x0;
        System.out.println("Das Quadrat von "+ x0 +" ist "+ q0 +" und größer
** oder gleich "+ zahl +".");
        int unt0 = x0 - 1;
        int ob0 = x0;
        System.out.println("-----");
        System.out.println("Die Wurzel aus "+ zahl +" liegt also zwischen den
** Zahlen "+ unt0 +" und "+ ob0 +".");
        System.out.println("-----");
    }
```

WICHTIG: Übersetze, erzeuge ein Objekt der Klasse „Schleife“ und **teste**, ob die Methode „tut, was sie soll“!

Bei welchen Zahlen, die beim Methodenaufruf als Parameter **`zahl`** übergeben werden, gibt es ein Problem?

Dieses Problem bei „`wurzelZiehen(int zahl)`“ können wir erst lösen, wenn wir auch die dritte Kontrollstruktur, die **`if`**-Anweisung (bedingte Anweisung), kennengelernt haben.
(Das ist sozusagen der „Cliffhanger“.)