Arrays

Definition

Unter einem Array versteht man ein Feld oder Container, das mehrere Objekte vom gleichen Typ aufzunehmen und zu verwalten kann. Die folgende Deklaration definiert ein Array, das 5 Integer-Werte aufnehmen kann:

```
int[] zahlen = new int[5];
```

mit der folgenden Deklaration würde das Array auch direkt initialisiert werden:

```
int[] zahlen = {11,32,42,2,4};
```

Ein Array kann man als Instanzen einer spezielle Klasse verstehen. Arrays, Felder werden also als Objekte behandelt und müssen durch den Operator new instanziiert werden. Die Array-Klasse bringt spezielle Methoden und Operationen mit, auf die man beim Umgang mit Arrays zurückgreifen kann, z.B liefert die Methode `length`, die Länge des Arrays zurück:

```
zahlen.length // in unserem Fall: 5
```

Aufgabe 1

Schreibe ein Java-Programm, welches das oben definierte Array `zahlen` enthält. Lass dir die Länge des Arrays auf der Konsole ausgeben.

Zugriff auf Array-Elemente

Jedes Element eines Array hat einen Wert und einen Index. **Die Zählung des Index beginnt immer bei Null**. Für unser Beispiel-Array sieht das also folgendermaßen aus:

Über den Index eines Elements, kann man auf dessen Wert zugreifen:

```
zahlen[1] // hier: 32
zahlen[4] // hier: 4
```

Aufgabe 2

Erweitere das Programm aus Aufgabe 1, so für alle Elemente des Arrays eine Zeile wie die folgende ausgegeben wird:

```
Das Array-Element mit dem Index 0 hat den Wert 11
Das Array-Element mit dem Index 1 ...
Das Array-Element mit dem Index 2 ...
```

Verwende dazu eine Zählschleife (for-Schleife).

Aufgabe 3

Gegeben ist eine Klasse „Messreihe“ mit einigen Methoden. Bei der Erzeugung einer Instanz des Typs Messreihe wird ein Array mit zufällig generierten „Messwerten“ vom Typ `double` erzeugt.

array_uebung4.java

```java
/** Fachklasse: Messreihe  (=eine Reihe von nummerierten Messdaten)
 *  @author: thh
 *  @author: fs
 *  @version: 20200115
 */

public class Messreihe {
    // Objektvariablen deklarieren
    int anzahl = 45;
    double[] gewicht = new double[anzahl];

    /** Konstruktor fuer Objekte der Klasse Messreihe
     *  Jede Messreihe enthaelt eine Reihe von positiven Messdaten
(Gewichten);
     */
    public Messreihe() {
        for (int i=0; i<anzahl; i++) {    // Alle Gewichte
            gewicht[i] = erzeugeZZahl();  // der Reihe nach festlegen
        }
    }

    /** das Element der Reihung mit dem Index i zurueckgeben
     *  Der gewuenschte Index i muss eingegeben werden
     *  Bei Eingabe eines nicht vorhandenen Index wird
     *  -8.888 als Fehlersignal zurueckgemeldet   */
    public double gibGewicht(int i)   {
        if (i<0 || i>anzahl) {              //<-- 2.
            return -8.888;       // als Fehlersignal!
        }
        else {
            return gewicht[i];
        }
    }

    /** setzt fuer zwei Elemente der Messreihe neue Werte fest.
```

```
     *  Das Element mit dem Index 5 in Reihung gewicht[ ] wird auf
555.55 gesetzt
     *  Das Element mit Index 9  auf den Wert 99.99    */
    public void setzeAn5und9() {
        // deine Aufgabe            //<-- 3.a) b)
    }


    /*# <-- 4. Aufgabe */


    // ----------- Hilfsfunktionen
    /** dient zum Anzeigen der Reihung am Bildschirm;
     *  kann durch GUI oder INSPECT ersetzt werden */
    public void anzeigen() {
       System.out.println("\n   Aktuelle Messreihe:");
        for (int i=0; i< anzahl; i++) {
            schreibe(i, gewicht[i]);
       }
    }

    //------ interne Hilfsfunktionen
    /** interne Methode, um eine Zufallszahl im Bereich 200.0 - 799.999
     *  mit 3 Nachkommastellen zu erzeugen;
     *  Math.random() liefert eine Zahl von 0 (inkl.) bis 1 (exkl.)  */
    private double erzeugeZZahl()   {
        double zufZahl = 200 + 600*Math.random();
        return Math.round((zufZahl*1000))/1000.0;
    }

    /** interne Hilfsfunktion zur Anzeige;
     *  setzt ein- bis zweistelligen Zahlen stellenrichtig ein. */
    private void schreibe(int i, double wert) {
        String erg = "Index";
        if (i<10) {
            erg = "Index  " + i;   // Zwei Leerzeichen drin !!
        }
        else  {
            erg = "Index " + i;    // hier nur eines !!
        }
        System.out.println(erg+" :  "+wert);
    }

    /**
    main Methode um den Programmablauf zu steuern
    **/
    public static void main(String[] args)
    {
        Messreihe reihe1 = new Messreihe();
        reihe1.anzeigen();
        // Erzeuge eine zweite Messreihe reihe2 und gebe sie aus
```

```
        // Teste weitere Methoden/bearbeite die Aufgaben unten/im Wiki
        double g=reihe1.gibGewicht(20);
        System.out.println("Gewicht " + g);
    }

}

/** Aufgaben:
 *
 * 1. Erprobe die Methode gibGewicht(). Wie muss sie aufgerufen werden.
 *    Klappt der Aufruf immer?
 *    Was wird in der Abfrage Z.28 geprueft? Was versucht man hier
abzufangen?
 *    Erlaeutere diese Pruefabfrage im Detail.
 *
 * 2.a) Vervollstaendige diese Methode zum Setzen eines neuen Wertes
fuer
 *    die Elemente gewicht[5] und gewicht[9] dieser Reihung.
 *   b) Schreibe eine Methode zum Setzen eines neuen Wertes fuer ein
 *    Element dieser Reihung mit waehlbarem Index.
 *   c) Teste deine Methoden mit entsprechenden Anweisungen in main()
 *
 *
 * 3. Ermittle das  Durchschnittsgewicht der gesamten Messreihe.
 *    Notiere zuerst deine Idee und setze sie in Quelltext um.
 *    Warum sollte dies eine eigenständige Funktion(Methode) werden?
 *
 * 4. Schreibe eine Methode, die je eine Interger Zahl als Start- (s)
und Endindex  (e) erhält
 *    und damit den Durchschnitt aller Werte mit Indizes (i) zwischen s
und e ermittelt.
 *
 */
```

From:
https://wiki.qg-moessingen.de/ - **QG Wiki**

Permanent link:
https://wiki.qg-moessingen.de/faecher:informatik:oberstufe:java:algorithmen:arrays:start?rev=1579095062

Last update: **15.01.2020 14:31**