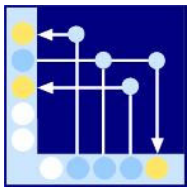
Hochschule Aalen
Fakultät Elektronik und Informatik
Studiengang Informatik



Programmieren 2

Vorlesung im Wintersemester 2014/2015
Prof. Dr. habil. Christian Heinlein

2. Übungsblatt (21. Oktober 2014)

Aufgabe 2: Arrays in Java

Definieren Sie in einer Datei mit dem Namen `Arrays.java` eine Java-Klasse `Arrays` mit den nachfolgend beschriebenen öffentlichen statischen Methoden! Übersetzen Sie die Klasse (unter Linux) mit `javac Arrays.java` und führen Sie ihre Hauptmethode mit `java Arrays` aus!

a) Die Methode `print` erhält als Parameter ein Array von `double`-Werten mit beliebiger Länge und gibt die Werte nacheinander mit `System.out.print` aus; zwischen den Werten soll jeweils ein Leerzeichen und nach dem letzten Wert mit `System.out.println` ein Zeilentrenner ausgegeben werden.

b) Die Methode `mirror` erhält als Parameter ebenfalls ein Array von `double`-Werten mit beliebiger Länge und dreht die Reihenfolge seiner Elemente „an Ort und Stelle“ um, d. h. ohne hierfür ein neues Array zu erzeugen.

c) Die Methode `rotate` erhält als Parameter ebenfalls ein Array von `double`-Werten mit beliebiger Länge sowie eine nicht-negative ganze Zahl `k` und liefert als Ergebnis ein neues Array, das die Elemente des ursprünglichen Arrays um `k` Positionen nach rechts „rotiert“ enthält. (Das ursprüngliche Array soll nicht verändert werden.)
Beispiel: Für ein Array mit den Elementen `2, 8, 6, 4, 10` und `k = 2` (oder auch `k = 7`) soll `rotate` ein Array mit den Elementen `4, 10, 2, 8, 6` liefern.

d) Die Methode `zip` erhält als Parameter zwei Arrays von `double`-Werten mit beliebiger Länge und liefert als Ergebnis ein neues Array, dessen Elemente wie bei einem „Reißverschluss“ abwechselnd aus den Elementen der beiden ursprünglichen Arrays bestehen. Wenn ein Array länger ist als das andere, werden seine überschüssigen Elemente einfach am Ende des Ergebnis-Arrays angefügt.
Beispiel: Für Arrays mit den Elementen `3, 7, 9` und `2, 8, 6, 4, 10` soll `zip` ein Array mit den Elementen `3, 2, 7, 8, 9, 6, 4, 10` liefern.

e) Die Hauptmethode `main` enthält für jede der obigen Methoden einige unterschiedliche Testaufrufe und gibt deren Ergebnisse jeweils mit `print` aus.
Die aktuellen Aufrufparameter der Methoden sollen jeweils durch Array-Initialisierer konstruiert werden.

Sie können davon ausgehen, dass keines der auftretenden Arrays `null` ist; Arrays der Länge 0 sollen aber korrekt behandelt werden.

Außer `System.out.print` und `System.out.println` sollen keinerlei Java-Bibliotheksmethoden verwendet werden!

Gerüst der Klasse `Arrays`:

```
class Arrays {
    // Array a elementweise in einer Zeile ausgeben.
    public static void print (double [] a) {
        ......
    }

    // Reihenfolge der Elemente des Arrays a umdrehen.
    public static void mirror (double [] a) {
        ......
    }

    // Rotation des Arrays a um k Positionen nach rechts
    // als neues Array liefern.
    public static double [] rotate (double [] a, int k) {
        ......
    }

    // Elemente der Arrays a und b per "Reißverschlussverfahren"
    // zu einem neuen Array zusammenfügen.
    public static double [] zip (double [] a, double [] b) {
        ......
    }

    // Test.
    public static void main (String [] args) {
        // Arrays a und b durch Array-Initialisierer konstruieren.
        // Zum Beispiel:
        double [] a = { 3, 7, 9 }, b = { 2, 8, 6, 4, 10 };

        System.out.println("mirror(a)");
        mirror(a); print(a);
        ......

        System.out.println("rotate(b, 2)");
        print(rotate(b, 2));
        ......

        System.out.println("zip(a, b)");
        print(zip(a, b));
        ......
    }
}
```

Lösung

```
class Arrays {
    // Array a elementweise in einer Zeile ausgeben.
    public static void print (double [] a) {
        for (int i = 0; i < a.length; i++) {
            System.out.print(a[i]);
            System.out.print(" ");
        }
        System.out.println();
    }

    // Reihenfolge der Elemente des Arrays a umdrehen.
    public static void mirror (double [] a) {
        int n = a.length;
        for (int i = 0; i < n/2; i++) {
            double tmp = a[i];
            a[i] = a[n-i-1];
            a[n-i-1] = tmp;
        }
    }

    // Rotation des Arrays a um k Positionen nach rechts
    // als neues Array liefern.
    public static double [] rotate (double [] a, int k) {
        int n = a.length;
        double [] b = new double [n];
        for (int i = 0; i < n; i++) {
            b[(i+k) % n] = a[i];
        }
        return b;
    }

    // Elemente der Arrays a und b per "Reißverschlussverfahren"
    // zu einem neuen Array zusammenfügen.
    public static double [] zip1 (double [] a, double [] b) {
        int m = a.length, n = b.length, sum = m + n;
        double [] c = new double [sum];
        for (int ai = 0, bi = 0, ci = 0; ci < sum; ) {
            if (ai < m) c[ci++] = a[ai++];
            if (bi < n) c[ci++] = b[bi++];
        }
        return c;
    }
```

```
    // Andere Implementierung von zip.
    public static double [] zip2 (double [] a, double [] b) {
        int m = a.length, n = b.length, min = m < n ? m : n;
        double [] c = new double [m+n];
        int ci = 0;
        for (int i = 0; i < min; i++) {
            c[ci++] = a[i];
            c[ci++] = b[i];
        }
        for (int i = min; i < m; i++) {
            c[ci++] = a[i];
        }
        for (int i = min; i < n; i++) {
            c[ci++] = b[i];
        }
        return c;
    }

    // Test.
    public static void main (String [] args) {
        // Arrays a und b durch Array-Initialisierer konstruieren.
        double [] a = { 3, 7, 9 }, b = { 2, 8, 6, 4, 10 };

        // Zwei Testaufrufe von mirror.
        System.out.println("mirror(a)");
        mirror(a); print(a);    // 9.0 7.0 3.0
        System.out.println("mirror(b)");
        mirror(b); print(b);    // 10.0 4.0 6.0 8.0 2.0

        // Zwei Testaufrufe von rotate.
        System.out.println("rotate(b, 2)");
        print(rotate(b, 2));    // 8.0 2.0 10.0 4.0 6.0
        System.out.println("rotate(b, 7)");
        print(rotate(b, 7));    // 8.0 2.0 10.0 4.0 6.0

        // Zwei Testaufrufe von zip.
        System.out.println("zip(a, b)");
        print(zip1(a, b));      // 9.0 10.0 7.0 4.0 3.0 6.0 8.0 2.0
        System.out.println("zip(b, a)");
        print(zip2(b, a));      // 10.0 9.0 4.0 7.0 6.0 3.0 8.0 2.0
    }
}
```