

Aufgabe 01 (12 Punkte)

Programmieren Sie die Methode `public static void printMinMax` aus. Die Methode soll den Wert des kleinsten und des größten Elementes im Array `a` mit Hilfe von `System.out.print` ausgeben. Denken Sie auch an den Fall eines leeren Feldes.

```
public class Aufgabe1 {  
  
    public static void printMinMax(int[] a) { // jetzt sind Sie dran!  
  
        if (a.length==0) System.out.println(„Array ist leer“);  
        else {  
            int max=a[0], min=a[0];  
            for (int i=1;i<a.length;i++) {  
                if (a[i]<min) min = a[i];  
                if (a[i]>max) max = a[i];  
            }  
            System.out.println(“Min = ”+min+”, Max = ”+max);  
        }  
    }  
}
```

Aufgabe 02 (12 Punkte)

Die rekursive Funktion $f(n) = 2 \cdot n - 1 + f(n-1)$ mit der Initialisierung $f(1) = 1$ berechnet für positive ganze Zahlen n : $f(n) = n \cdot n$.

Schreiben Sie eine iterative (nicht-rekursive!) Funktion `itSquare(int n)`, die durch fortgesetztes Aufaddieren von $2 \cdot i - 1$ ($i=1..n$) $n \cdot n$ berechnet!

Den Rahmen geben wir wieder vor:

```
/**
 * Iterative Berechnung von n*n
 */
public static int itSquare(int n) {
// hier können Sie die Lösung codieren

    int sum=0;
    for (int i=1; i<=n; i++) sum += i+i-1;
    return sum;
}
```

Aufgabe 03 (12 Punkte)

Implementieren Sie eine Klasse **Zeitspanne**, die eine Zeitspanne in Tagen, Stunden und Minuten festlegt. Die interne Speicherung sei normiert, d.h. die Werte von Minuten liegen zwischen 0 und 59, die Stunden zwischen 0 und 23. Tag kann beliebig groß (im Wertebereich von long) werden. Wir betrachten nur positive Zeitspannen. Wir machen folgende Vorgaben:

```
public class Zeitspanne {
    protected int m; // Minuten 0<=m<60
    protected int h; // Stunden 0<=h<24
    protected long d; // Tage d>=0
}
```

- a) Implementieren Sie einen Konstruktor, der die Zeitspanne der Dauer d Tage, h Stunden, m Minuten erzeugt: Denken Sie an den Aufruf der Methode `normiere()` (siehe b)), damit auch nicht normierte Übergabewerte verarbeitet werden können.

```
public Zeitspanne (long d, int h, int m) { // hier ergänzen
    this.m=m;
    this.h=h;
    this.d=d;
    normiere();
}
}
```

- b) Vervollständigen Sie die Methode `normiere`, die die Zeitwerte wie anfangs beschrieben normiert.

```
void normiere() {
    // normiere Minuten
    while (m>=60) {
        m -=60;
        h++;
    }

    // hier ergänzen: normiere Stunden

    while (h>=24) {
        h -=24;
        d++;
    }
}
```

- c) Implementieren Sie eine Methode `add`, die eine Zeitspanne aufaddiert. Denken Sie an die notwendige Normierung! Beispiel `Zeitspanne(0,0,20).add(0,0,100)` liefert dieselbe Zeitspanne wie `Zeitspanne(0,2,0)`.

```
public void add(long d1, int h1, int m1) {
    m +=m1;
    h +=h1;
    d +=d1;
    normiere();
}
```

Aufgabe 04 (12 Punkte)

Diese Aufgabe umfasst 3 Multiple-Choice Cluster mit je 4 Ankreuzfragen. Für jedes Cluster gilt: wenn alle 4 Kreuze an der richtigen Stelle stehen, gibt es 4 Punkte für das Cluster. Ein falsches Kreuz gibt einen Punkt Abzug. Wer 2 richtige und 2 falsche Kreuzchen in einem Cluster macht, erhält $1+1-1-1=0$ Punkte. Zum Trost: es gibt keine negativen Gesamtpunktzahlen, jedes Cluster bringt 0 bis 4 Punkte.

| | | Ja | Nein | Frage |
|----|----|-------------------------------------|-------------------------------------|---|
| a) | | | | Grundlagen |
| | 1. | <input type="checkbox"/> | <input checked="" type="checkbox"/> | Einige wenige Algorithmen benötigen unbedingt eine Sprache mit Goto-Anweisungen. |
| | 2. | <input type="checkbox"/> | <input checked="" type="checkbox"/> | Klassendiagramme beschreiben das Verhalten von Objekten. |
| | 3. | <input checked="" type="checkbox"/> | <input type="checkbox"/> | UML kennt Mehrfachvererbung, Java nicht. |
| | 4. | <input type="checkbox"/> | <input checked="" type="checkbox"/> | Jede Assoziation ist auch eine Aggregation. |
| b) | | | | Java |
| | 1. | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <code>long int real;</code> ist korrekter Java-Code. |
| | 2. | <input type="checkbox"/> | <input checked="" type="checkbox"/> | Jede nicht abstrakte Java-Klasse muss eine main-Methode besitzen. |
| | 3. | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Ein Java-Objekt kann aus dem Speicher entfernt werden, wenn keine Referenz mehr auf das Objekt zeigt. |
| | 4. | <input type="checkbox"/> | <input checked="" type="checkbox"/> | Wenn eine Klasse keinen Konstruktor hat, meldet der Compiler einen Fehler. |
| c) | | | | Programmieren |
| | 1. | <input type="checkbox"/> | <input checked="" type="checkbox"/> | Nicht jede rekursive Java-Funktion kann in eine nicht-rekursive Funktion umgeschrieben werden. |
| | 2. | <input checked="" type="checkbox"/> | <input type="checkbox"/> | Der Ausdruck $9/2/2.0$ hat den Wert 2.0. |
| | 3. | <input checked="" type="checkbox"/> | <input type="checkbox"/> | "a" ist ein String und 'b' ist ein Character-Literal. |
| | 4. | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <code>for (int i; i<10; i--) System.out.print(i);</code> ist korrekter Java-Code. |