

Einleitung

Zur Vorbereitung auf Prüfungen ist es immer ratsam die Aufgaben, die in den Klausuren der letzten Jahre gestellt wurden durchzuarbeiten. Diese kann man sich im BECI kostenlos ausdrucken lassen.

Als kleinen Einstieg haben wir uns die Klausur von Herrn Prof. Dr. Frühwirth aus dem Jahr 2004 geholt und wollen euch nun die Gelegenheit geben die in der Klausur vorkommenden Programmieraufgaben zusammen mit uns zu lösen.

Insgesamt waren damals **80 Punkte** zu vergeben und die hier gezeigten Programmieraufgaben schlugen mit **25 Punkten** zu Buche.

Aufgabe 1: Klassen und Objekte, 10 Punkte

Es soll eine Java-Klasse `Punkt` implementiert werden, die einen zweidimensionalen Punkt in einem Koordinatensystem repräsentiert. Dabei soll der Zustand des Objekts ausschließlich über Zugriffsmethoden verändert und ausgelesen werden können. Geben sie die Klasse `Punkt` mit folgenden Elementen an (**je 2 Punkte**):

- Alle nötigen Attribute zur Verwaltung eines Punktes und Methoden zum Lesen und Setzen dieser Attribute.
- Ein Konstruktor, der alle Attribute des Punktes zur Initialisierung übergeben bekommt.
- Eine Methode `double getDistanz()`, die den Abstand des Punktes zum Ursprung berechnet.
- Eine Methode `void verschiebe(double dx, double dy)`, die den Punkt in x- und y-Richtung um die übergebenen Werte verschiebt.
- Eine Methode `public static void main(String[] args)`, die zwei beliebige Punkte erzeugt und deren Distanz zum Ursprung auf dem Bildschirm ausgibt.

Tipp: Zur Berechnung der Quadratwurzel aus einer Zahl n können Sie die Methode `double Math.sqrt(n)` verwenden.

Aufgabe 2: Suchbaum, 5 Punkte

Gegeben ist folgende Klasse zur Repräsentation eines Knotens in einem Suchbaum:

```
public class BaumKnoten {
    int wert;
    BaumKnoten links;
    BaumKnoten rechts;
}
```

Geben Sie eine Methode `public void PrintOddOrEven(BaumKnoten b, boolean odd)` an. Wird für `odd` der Wahrheitswert `true` übergeben, so soll die Methode alle *ungeraden* Knotenwerte des Baumes aufsteigend sortiert auf dem Bildschirm ausgeben. Wird hingegen `false` übergeben, dann gibt die Methode alle *geraden* Knotenwerte sortiert aus.

Aufgabe 3: Listen, 10 Punkte

Gegeben sind folgende (aus der Vorlesung bekannte) Klassen zur Verwaltung von Listen. Implementieren Sie in der Klasse `List` eine Methode `Element getNextToLast()`, die den folgenden Rückgabewert haben soll:

- Hat die Liste mindestens zwei Elemente wird das vorletzte Element zurückgegeben.
- Für eine leere Liste oder eine Liste mit genau einem Element wird `null` zurückgegeben.

Die Methode `Element getNextToLast()` soll die Liste dabei nicht verändern.

```
class Element {
    Element next;
    int data;

    // Element-Methoden (aus dem Beispiel der Vorlesung),
    // die hier nicht verwendet werden, wurden entfernt.
}

class List {
    Element head;

    // List-Methoden (aus dem Beispiel der Vorlesung),
    // die hier nicht verwendet werden, wurden entfernt.

    // ZU IMPLEMENTIEREN:
    Element getNextToLast() { ... }
}
```