Übungsblatt 5 (Abgabe bis Dienstag, den 19.11.2024, um 12:00 Uhr)

Prof. Dr. J. Giesl

D. Cloerkes, J.-C. Kassing, N. Lommen, E. Meyer

Allgemeine Hinweise:

- Die **Deadline** zur **Abgabe** der Hausaufgaben ist am **Dienstag, den 19.11.2024, um 12:00 Uhr**.
- Der **Workflow** sieht wie folgt aus. Die Abgabe der Hausaufgaben erfolgt **im Moodle-Lernraum** und kann nur in **Zweiergruppen** stattfinden. Dabei müssen die Abgabepartner*innen **dasselbe Tutorium** besuchen. Nutzen Sie ggf. das entsprechende **Forum** im Moodle-Lernraum, um eine*n Abgabepartner*in zu finden. Es darf **nur ein*e** Abgabepartner*in die Abgabe hochladen. Diese*r muss sowohl die **Lösung** als auch den **Quellcode** der Programmieraufgaben hochladen. Die Bepunktung wird dann von uns für **beide** Abgabepartner*innen **separat** im Lernraum eingetragen. Die Feedbackdatei ist jedoch nur dort sichtbar, wo die Abgabe hochgeladen wurde und muss innerhalb des Abgabepaars **weitergeleitet** werden.
- Die **Lösung** muss als PDF-Datei hochgeladen werden. Damit die Punkte beiden Abgabepartner*innen zugeordnet werden können, müssen **oben** auf der **ersten Seite** Ihrer Lösung die **Namen**, die **Matrikelnummern** sowie die **Nummer des Tutoriums** von **beiden** Abgabepartner*innen angegeben sein.
- Der **Quellcode** der Programmieraufgaben muss als `.zip`-Datei hochgeladen werden und **zusätzlich** in der PDF-Datei mit Ihrer Lösung enthalten sein, sodass unsere Hiwis ihn mit Feedback versehen können.

  Auf diesem Blatt muss Ihre Codeabgabe Ihren vollständigen Java-Code in Form von `.java`-Dateien enthalten. Aus dem Lernraum heruntergeladene Klassen, etwa die Datei `SimpleIO.java`, dürfen nicht mit abgegeben werden.

  Stellen Sie sicher, dass Ihr Programm von `javac` **akzeptiert** wird, wenn die entsprechenden Klassen aus dem Lernraum hinzugefügt werden. Ansonsten werden keine Punkte vergeben.
- Einige Hausaufgaben müssen im Spiel Codescape gelöst werden. Klicken Sie dazu im Lernraum rechts im Block "Codescape" auf den angegebenen Link. Diese Aufgaben werden getrennt von den anderen Hausaufgaben gewertet.

Aufgabe 4 (Programmanalyse): (2+1+3+3+3+3+1+3+1 = 20 Punkte)

Lösen Sie die folgende Aufgabe ohne Einsatz eines Computers. Bedenken Sie, dass Sie in einer Prüfungssituation ebenfalls keinen Computer zur Verfügung haben.
Betrachten Sie das folgende kurze Programm:

```
public class A {
    private int i1;
    private Integer i2;
    private short s;

    public A(Short s, int i) {
        this.i1 = s;
        this.i2 = i;
        this.s = s;
    }

    public void f(Integer i) {
        System.out.println("f1");
    }
    public void f(String s) {
        System.out.println("f2: " + s);
    }
    public void f(double d) {
        System.out.println("f3");
    }

    public void g(int i, float d) {
        System.out.println("g1");
    }
    public void g(Integer i, Long j) {
        System.out.println("g2");
    }
    public void g(int... is) {
        System.out.println("g3");
    }
    public void g(double... is) {
        System.out.println("g4");
    }

    public static void main(String[] args) {
        A a = new A((short)1,2);
        a.f(a.i1);                    //a)
        a.f(a.i2);                    //b)
        a.f(a.s);                     //c)
        a.f(.0f);                     //d)

        a.g(1,2);                     //e)
        a.g(1,2L);                    //f)
        a.g(a.i2,Long.valueOf(2));    //g)
        a.g(1,2.0);                   //h)
        a.g(1,2.0f);                  //i)
    }
}
```

Geben Sie die Ausgabe dieses Programms an, wenn die `main`-Methode ausgeführt wird. **Begründen Sie Ihre Antwort!** Ordnen Sie jeder Teilaufgabe die aufgetretenen Effekte zu und erklären Sie, warum gerade diese zu beobachten sind.

Aufgabe 7 (Rekursion): (10 + 10 + 10 = 30 Punkte)

Diese Aufgabe ist die Fortsetzung von Aufgabe 5. Wir verwenden wieder die Klasse `Tagesgeld` mit allen dort eingeführten Attributen und Methoden. Sie finden entsprechende Implementierungen im Lernraum, auf die Sie zurückgreifen können und sollen.
Überlegen Sie in jeder Teilaufgabe für jede von Ihnen geschriebene Methode, welchen Zugriffsmodifikator Sie setzen wollen und ob die Methode statisch sein soll. Begründen Sie Ihre Entscheidungen jeweils in einem Kommentar.
Sie dürfen in der gesamten Aufgabe *keine* Schleifen verwenden. Die Verwendung von Rekursion ist hingegen erlaubt. Außerdem dürfen Sie Methoden aus Aufgabe 5 und aus früheren Aufgabenteilen aufrufen, auch wenn Sie diese nicht selbst implementiert haben. Sie dürfen jedoch keine vordefinierten Methoden benutzen, es sei denn, es ist in einer Teilaufgabe explizit erlaubt.

Hinweise:

- Sie können die vordefinierten Funktionen `Math.min(double a, double b)` und `Math.max(double a, double b)` verwenden, die den kleineren bzw. den größeren der beiden übergebenen Werte zurückgeben. Diese können in mehreren Teilaufgaben hilfreich sein.

a) Bei der Implementierung von `verzinse` sind wir davon ausgegangen, dass zuerst für `angebotsmonate` Monate mit `angebotszinsen` % verzinst wird und ab dann mit `normalzinsen` %. In dieser Teilaufgabe betrachten wir nun, ob es einen Unterschied macht, in welcher Reihenfolge die verschiedenen Zinssätze zur Anwendung kommen.

Erweitern Sie die Klasse `Tagesgeld` um eine rekursive Methode `double optimaleVerzinsung(double init, int nm)`. Die Methode soll den maximalen Endbetrag berechnen, der aus einem Anfangsbetrag `init` erzielt werden kann, nachdem eine Anzahl von Normalmonaten `nm` und die durch das `Tagesgeld`-Objekt gegebene Anzahl von Angebotsmonaten vergangen ist. Die Reihenfolge der Monate darf dabei beliebig gewählt werden. Für diese Teilaufgabe lassen wir die Höchstgrenzen für zu verzinsende Einlagen außer Acht.

Warum spielt die Reihenfolge, in der die verschiedenen Zinssätze angewendet werden, überhaupt eine Rolle? Notieren Sie Ihre Antwort in einem Kommentar.

Beispiel:

Unmittelbar nach der Zuweisung `Tagesgeld t = new Tagesgeld(50000,1,5.5,1.5);` soll der Aufruf `t.optimaleVerzinsung(10000,1)` den Wert `10057.187381927695` zurückgeben, der sich ergibt, wenn zuerst mit den Angebots- und dann mit den Normalzinsen verzinst wird. Bei der umgekehrten Reihenfolge ergibt sich ein marginal kleinerer Wert.

b) Ergänzen Sie die Klasse `Tagesgeld` um eine Methode `int verkuerzeUmKuerzesteLaufzeit(boolean verkuerze, Tagesgeld... ts)`. Es soll die kleinste *echt positive* Anzahl von Angebotsmonaten zurückgegeben werden, für die ein `Tagesgeld` in `ts` die Einlagen mit `angebotszinsen` % verzinst. Außerdem soll die Anzahl von Angebotsmonaten in jedem `Tagesgeld`-Objekt in `ts` um diesen Wert verringert werden, genau dann wenn `verkuerze` den Wert `true` hat. Wenn kein Objekt übergeben wird, das einen oder mehr solcher Angebotsmonate aufweist, soll `Integer.MAX_VALUE` zurückgegeben werden. Sie dürfen dazu nur eine einzige rekursive Methode nutzen.

Beispiel:

Wenn die drei `Tagesgeld`-Objekte `t1`, `t2` und `t3` die `angebotsmonate`-Werte `3`, `0` und `1` haben, soll der Aufruf `verkuerzeUmKuerzesteLaufzeit(true,t1,t2,t3)` den Wert `1` zurückgeben. Unmittelbar danach sollen die `angebotsmonate`-Attribute die Werte `2`, `0` und `0` haben.

Hinweise:

- Überlegen Sie, wie Sie die Ermittlung des Minimums und die (davon abhängige) Änderung des Attributs `angebotsmonate` in einer einzigen rekursiven Methode umsetzen können.
- Diese (einzige) rekursive Methode darf auch eine Hilfsmethode sein.

c) Bisher haben wir nur die Verzinsung durch einzelne Tagesgelder betrachtet. Bei entsprechend hohen Summen kann es aber wegen der Maximalbeträge und der gesetzlichen Einlagensicherung nötig werden, mehrere Tagesgeldkonten zu nutzen. Diesen Fall betrachten wir nun:

Ergänzen Sie die Klasse `Tagesgeld` dazu um eine Methode `double verzinseParallel(double init, int monate, Tagesgeld... ts)`. Die Methode soll den Betrag zurückgeben, der sich ergibt, wenn der anfängliche Betrag `init` für `monate` Monate gleichzeitig nach den in Aufgabe 5 erläuterten Angebotsbedingungen der `Tagesgeld`-Objekte in `ts` verzinst wird. Dabei sollen diejenigen Objekte mit einem *echt positiven* `angebotsmonate`-Wert in der gegebenen Reihenfolge jeweils für den gesamten Zeitraum maximal genutzt werden, bis die gesamte Summe verteilt ist. Das heißt, dass auf jedem solchen Konto der jeweilige maximale Betrag (bzw. 100.000€, wenn der maximale Betrag höher ist) liegen muss. Das letzte genutzte Konto muss nicht maximal genutzt werden. Wenn der anfängliche Betrag unter diesen Bedingungen nicht vollständig verteilt werden kann, wird der überschüssige Teilbetrag nicht verzinst.

Beispiel:

Sie finden im Lernraum ein ausführliches Beispiel in der `main`-Methode der Klasse `Tagesgeld`. (Diese enthält auch die Implementierungen aus Aufgabe 5.) Wenn Ihre Werte von den erwarteten Werten marginal abweichen, ist das wahrscheinlich auf Rundungsfehler durch eine leicht andere Art der Berechnung zurückzuführen und daher unproblematisch.

Hinweise:

- Sie können eigene Hilfsmethoden verwenden.

Aufgabe 8 (Deck 5): (Codescape)

Lösen Sie die Missionen von Deck 5 des Codescape Spiels. Ihre Lösung für die Codescape Missionen wird nur dann für die Zulassung gezählt, wenn Sie Ihre Lösung vor der einheitlichen Codescape Deadline am Freitag, den 24.01.2025, um 23:59 Uhr abschicken.