

Bachelorprüfung: Objektorientierte Softwareentwicklung

SS21

Studiengang (AI, AI Dual): _____

Erlaubte Hilfsmittel: keine

Jeder Griff zu einem elektronischen Gerät (z.B. Smartphone) wird als Täuschungsversuch gewertet.

Lösung ist auf den Klausurbögen anzufertigen. (eventuell Rückseiten nehmen)

Bitte legen Sie den Studentenausweis auf den Tisch.

Bearbeitungszeit: 60 Minuten

Unterschrift

Benotung

Aufgabe:	1	2	3	4		Gesamt	Note
Punkte:	25	20	30	25		100	
erreicht:							

Aufgabe 1 (Modellieren und Schreiben von Klassen)

Sie sollen in dieser Aufgabe Klassen entwickeln, die Wetterdaten repräsentieren.

- a) Entwickeln Sie eine Klasse **Wetter**. Ein Objekt dieser Klasse repräsentiert Informationen über eine Wettermessung. Sie soll drei Eigenschaften haben: einen Namen des Ortes der Messstation, der durch eine Zeichenkette dargestellt ist, die gemessene Temperatur in Grad Celcius als ganze Zahl und den gemessenen Luftdruck in Grad Celsius als ganze Zahl.

Schreiben Sie einen Konstruktor, der alle Felder initialisiert.

Überschreiben Sie die Methode `equals` auf adequate Weise.

Überschreiben Sie die Methode `toString`.

Name:

Matrikelnummer:

- b) Schreiben Sie eine Unterklasse **Alpenwetter** der Klasse **Wetter**. Diese Klasse soll ein zusätzliches Feld haben, das eine Lawinenwarnstufe aus einen der vier folgenden Aufzählungswerte speichert: keine, gering, erhoeht, hoch.

Name:

Matrikelnummer:

c) Gegeben Sei die Klasse **Zeitreihe**, die eine Liste von Wetterdaten ist:

```
1 class Zeitreihe extends java.util.ArrayList<Wetter>{  
2 }
```

Listing 1: Zeitreihe.java

Schreiben Sie für diese Klasse eine Methode:

```
double temperaturDurchschnitt(),
```

die die Durchschnittstemperatur über alle Temperaturwerte der Liste darstellt.

Name:

Matrikelnummer:

d) Schreiben Sie für die Klasse `Zeitreihe` eine Methode

```
int anzahlWarnungenHoch(),
```

die die Anzahl der in der Liste enthaltenen Alpenwettermessungen mit einer Lawinenwarnstufe ›hoch‹ berechnet.

Aufgabe 2 (Programmfluss)

- a) Führen Sie die folgende Klasse von Hand aus. Schreiben Sie dabei auf, in welcher Reihenfolge die Zeilen durchlaufen werden und mit welchen Werten die einzelnen Variablen während des Programmdurchlaufs belegt sind. Schreiben Sie auf, was auf dem Bildschirm ausgegeben wird.

```
1 class Aufgabe2a{
2     public static void main(String [] args){
3         int y = 13;
4         int x = 14;
5
6         for (int i=y;i>0;i=i-4){
7             System.out.println("A:" +i+ " "+x+ " "+y);
8             while (x<y){
9                 x++;
10                if (x%3==0){
11                    System.out.println("weiter");
12                    continue;
13                }
14                System.out.println("I:" +i+ " "+x+ " "+y);
15            }
16            y += 3;
17            i--;
18        }
19    }
20 }
```

Listing 2: Aufgabe2a.java

Name:

Matrikelnummer:

b) Betrachten Sie folgende Klasse:

```
1 class Aufgabe2b{  
2     static int e(int a, int b, int c) {  
3         if (a <= 0) return c;  
4         return e(a-1,b,c*b);  
5     }  
6  
7     public static void main(String [] args){  
8         System.out.println(e(4,3,1));  
9     }  
10 }
```

Listing 3: Aufgabe2b.java

Berechnen Sie schrittweise das Ergebnis des Ausdrucks `e(4,3,1)`.

Was berechnet die Funktion?

Aufgabe 3 (Arraylisten)

Gegeben sei folgende Klasse, die eine einfache Array-basierte Liste realisiert, wie aus der Vorlesung bekannt.

```
1 public class AL<A> {  
2     private int size = 0;  
3     private A[] array = (A[]) new Object[10];  
4  
5     public void add(A el){  
6         if (size >= array.length){  
7             enlargeArray();  
8         }  
9         array[size++] = el;  
10    }  
11    private void enlargeArray(){  
12        A[] newarray = (A[]) new Object[array.length + 10];  
13        for (int i=0; i<array.length; i++) newarray[i] = array[i];  
14        array = newarray;  
15    }  
16    public A get(int i) {  
17        return array[i];  
18    }  
19    public int size(){return size;}  
20    public boolean isEmpty(){return size == 0;}  
21 }
```

Listing 4: AL.java

Implementieren Sie folgende Methoden für diese Listenklasse:

a) Schreiben Sie in der Klasse **AL** eine Methode **startsWith**:

Es soll getestet werden, ob die übergebene Liste ein Präfix der this-Liste ist.

```
22 boolean startsWith(AL<A> prefix){  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38 }
```

Listing 5: AL.java

b) Schreiben Sie in der Klasse **AL** eine Methode **reverse**:

Die this-Liste soll so verändert werden, dass sie anschließend die Elemente in umgekehrter Reihenfolge enthält.

```
39 void reverse(){  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56 }
```

Listing 6: AL.java

c) Gegeben sei zusätzlich folgende Klasse:

```
1 public class Pair<A,B>{  
2     A fst;  
3     B snd;  
4     Pair(A a,B b){fst=a;snd=b;}  
5 }
```

Listing 7: Pair.java

Sie soll verwendet werden, um Paare von Objekten zu bilden.

Schreiben Sie für die Klasse `AL` eine Methode `makeIndex()`.

Es soll eine neue Liste erstellt werden, die aus der ursprünglichen Liste entsteht, indem für jedes Element mit seinem Index ein Paar gebildet wird.

```
57 static <A> AL<Pair<Integer,A>> makeIndex(){  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75 }
```

Listing 8: AL.java

- d) Schreiben Sie eine Methode `getValue`, die aus einer Liste von Paaren, das Objekt im Feld `snd` aus dem Paar zurück gibt, in dem das Feld `fst` ein Objekt gleich zum Parameter `a` erhält.

Ist kein solches Element enthalten, soll `null` als Ergebnis zurück gegeben werden.

```
76 static <A,B> B lookUp(AL<Pair<A,B>> ps,A a){  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96 }
```

Listing 9: AL.java

- e) Wie lässt sich ein Ausdruck der Form: `lookUp(xs.makeIndex(),x)` klüger umsetzen?

Aufgabe 4 Zusammenhänge

Erklären Sie in kurzen Worten.

a) Können Schnittstellen andere Schnittstellen erweitern und wenn ja wie viele?
Wie geht das syntaktisch?

b) Was kann bei der Implementierung eines Konstruktors die erste Anweisung sein.
Was passiert, wenn keine dieser Anweisungen die erste Anweisung des Konstruktors ist?

c) Was versteht man unter der *diamond notation*? Wofür ist diese nützlich?

Name:

Matrikelnummer:

d) Was ist der Unterschied zwischen Überladen und Überschreiben von Methoden.

e) Welche Eigenschaft haben funktionale Schnittstellen? Wozu kann man diese Eigenschaft nutzen?