Name: Matrikelnummer:

Bachelorprüfung: Objektorientierte Softwareentwicklung

WS15/16

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: 90 Minuten

Unterschrift

Benotung

Aufgabe:	1	2	3	4	5		Gesamt	Note
Punkte:	20	20	20	20	20		100	
erreicht:								

Name: Matrikelnummer:

Aufgabe 1 (20 Punkte)
(Modellieren und Schreiben von Klassen)
In dieser Aufgabe sollen Sie eine Minibibliothek zur Beschreibung von Tieren einer Zoohandlung entwerfen.

a) Schreiben Sie eine Klasse `Haustier` zur Beschreibung von Haustieren. Ein Haustier sei charakterisiert durch einen Gattungsnamen (z.B. Schäferhund, Hauskatze, Meerschweinchen), der als String gespeichert wird, die Größe in Zentimetern und das Gewicht in Gramm.

Schreiben Sie für die Klasse einen geeigneten Konstruktor und überschreiben sie die Methode `equals` der Klasse `Object` auf adequate Weise.

(Define a class for *pets*. It has a field of type String for the species (e.g. dog, cat.) and fields for size and weight. Define a constructor and override equals.)

b) Schreiben Sie eine Unterklasse von `Haustier`, die Fische beschreibt. Für Fische soll in einem zusätzlichen Feld vermerkt sein, ob es sich um einen Salzwasserfisch oder um einen Süßwasserfisch handelt.

Überschreiben Sie die Methode `equals`. Benutzen Sie dabei möglichst viele Aufrufe von Code der Oberklasse.

(Write a subclass for fishes. Additional field signifies if it is a freshwater fish. Override equals.)

c) Lassen Sie die Klasse `Haustier` jetzt zusätzlich die folgende Standardschnittstelle `Comparable` implementieren, so dass es möglich ist, Haustiere mit anderen Haustieren mit der Methode `compareTo` zu vergleichen. Zum Vergleich von Haustieren soll die Größe herangezogen werden und bei gleicher Größe das Gewicht ausschlaggebend sein.

```
1 package java.lang;
2 public interface Comparable<A>{
3   /** Compares this object with the specified object for order.
4       Returns a negative integer, zero, or a positive integer as
5       this object is less than, equal to, or greater than the
6       specified object. **/
7   int compareTo(T o)
8 }
```

(Implement the interface above for pets.)

Name: Matrikelnummer:

Aufgabe 2 (20 Punkte)
(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 = 5;
4      for (int x=13;x>=y;x++){
5        System.out.println(x+" "+y);
6        int i=y;
7        while (i>0){
8          System.out.println("I:"+i+" "+x+" "+y);
9          y += 3;
10         i = i-4;
11       }
12       if (x == 13){
13        x++;
14        continue;
15       }
16       System.out.println(x+" "+y);
17     }
18   }
19 }
```

Listing 1: Aufgabe2a.java

b) Betrachten Sie folgende Klasse:

```
1
2  class Aufgabe2b{
3    static int pas(int a, int b) {
4      if (a==b) return a;
5
6      if (a>b) return pas(a-b,b);
7      return pas(b-a,a);
8    }
9
10   public static void main(String [] args){
11     System.out.println(pas(21,51));
12   }
13 }
```

Listing 2: Aufgabe2b.java

Berechnen Sie schrittweise das Ergebnis des Ausdrucks `pas(21,51)`.

Was berechnet die Funktion pas.

Name: Matrikelnummer:

Aufgabe 3 (20 Punkte)
(SwingGUI)
Gegeben Sei folgende Swing-Komponente:

```
1  import javax.swing.*;
2  class SimpleGUI extends JPanel{
3    JTextField input = new JTextField();
4    JLabel output = new JLabel();
5    JButton button = new JButton();
6    SimpleGUI(){
7      this.add(input);
8      this.add(button);
9      this.add(output);
10   }
11 }
```

Listing 3: SimpleGUI.java

a) Schreiben Sie eine Unterklasse `SquareGUI`, die von der Klasse `SimpleGUI` erbt. Es soll eine Komponente entstehen, bei der auf Knopfdruck der Text aus dem Feld `input` als Zahl interpretiert und die Quadratzahl dieser Zahl auf dem Feld `output` angezeigt wird.

(Write a subclass `SquareGUI`. Pushing the button shall calculate the square of the input.)

Name: Matrikelnummer:

b) Gegeben sei zusätzlich folgende Schnittstelle:

```
1 interface StringLogik{
2   String apply(String s);
3 }
```

Listing 4: StringLogik.java

Schreiben Sie eine Unterklasse `GerneralGUI`, die von der Klasse `SimpleGUI` erbt. Sie soll ein Objekt, das die Schnittstelle `StringLogik` implementiert, im Konstruktor übergeben bekommen. Es soll eine Komponente entstehen, bei der auf Knopfdruck, der Text aus dem Feld `input` der Methode des `StringLogik`-Objekts übergeben und das Ergebnis auf dem Feld `output` angezeigt wird.

(Write a subclass `GerneralGUI`. It has some object of interface `StringLogik`. Pushing the button shall use the `StringLogik` object for calculating the result.)

Notizen

Name: Matrikelnummer:

Aufgabe 4 (20 Punkte)
(Arraylisten)
Gegeben sei folgende Klasse, die eine einfache Array-basierte Liste realisiert, wie aus der Vorlesung bekannt.

```
1  public class Li<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
12   private void enlargeArray(){
13       A[] newarray = (A[])new Object[array.length + 10];
14       for (int i=0; i<array.length; i++) newarray[i] = array[i];
15       array = newarray;
16   }
17
18   public A getElement(int i) throws Exception{
19     if (i>=size) throw new IndexOutOfBoundsException();
20     return array[i];
21   }
22   public int size(){return size;}
23   public boolean isEmpty(){return size == 0;}
24 }
```

Listing 5: Li.java

Implementieren Sie folgende Methoden für diese Listenklasse:

a) Gegeben sei zusätzlich folgende Schnittstelle:

```
1 public interface Property<A>{
2   boolean test(A a1);
3 }
```

Listing 6: Property.java

Sie soll verwendet werden, um zu testen, ob ein Objekt eine bestimmte Eigenschaft hat.

Schreiben Sie jetzt in der Klasse Li folgende Methode:
`int howMany(Property<A> p);`

Es soll sie Anzahl der Elemente zurück gegeben werden, die die Eigenschaft p haben.

(Count elements in list with the given property.)

Name: Matrikelnummer:

```
int howMany(Property<A> p) {
    int result = 0;
    for(int i = 0; i < size(); i++){
        if( p.test(array[i])) result++;
    }
    return result;
}
```

Notizen

Name: Matrikelnummer:

b) `public boolean contains(A o)` {

Es soll genau dann `true` zurück gegeben werden, wenn die Liste ein Objekt enthält, das gleich dem Argument ist. Implementieren Sie diese Methode unter Benutzung der Methode `howMany` und eines Lambda-Ausdrucks.

(Is the argument `o` contained in the list? Use the method `howMany` and a lambda expression for the solution.)

return howMany((x) -> x.equals(o)) > 0;

}

Name: Matrikelnummer:

c) `public void insert(int i,A o) throws Exception`

Es soll Argument `o` in die Liste eingefügt werden und zwar am Index `i`. Wenn `i` außerhalb des Indexbereichs der Liste liegt, soll eine Exception geworfen werden.

(Insert element into the list at a certain index. Illegal index (too large or too small) will result in an exception.) */

```
if (i<0 || i > size)
        throw new IndexOutOfBoundsException();

if (size >= array.length)
        enlargeArray();

for (int j = size; j>i; j--){
    array[j] = array[j-1];
}
array[i] = o;

sizeit;

}
```

Name: Matrikelnummer:

Aufgabe 5 **(20 Punkte)**
Zusammenhänge
Erklären Sie in kurzen Worten.

a) Welche Eigenschaft muss ein Interface haben, damit es durch einen Lambda-Ausdruck implementiert werden kann?

(Which interfaces can be implemented by means of a lambda expression?)

Es muss eine funktionale Schnittstelle sein, d.h. es darf nur eine abstrakte Methode haben.

b) Warum ist es in Java sinnvoller die meisten Berechnungen mit Schleifen statt mit einer Rekursion umszusetzen.

(Advantage of loops vs recursions?)

Notizen

Name: Matrikelnummer:

c) Was sind Aufzählungen in Java.
(Explain enumerations in Java.)

Klassen ohne public Konstruktor, von denen nur endlich viele Objekte existieren, die in statische Felder gespeichert sind.

d) Was ist der Unterschied zwischen einem if-else-Konstrukt und dem Fragezeichen-Doppelpunkt-Operator?
(Difference between if-else and question mark colon operator.)

if ist eine Anweisung zum Kontrollfluss, ?: ein Ausdruck. Ausdrücke lassen sich immer zu einem Wert auswerten.

Notizen

Name: Matrikelnummer:

e) In welchen zwei unterschiedlichen Bedeutungen kann das Schlüsselwort `this` verwendet werden?

(Two different meanings of keyword `this`.)

this gefolgt von einer öffnenden Klammer ist der Aufruf eines Konstruktors der Klasse. Dieses darf nur als erste Befehl eines überladenen Konstruktors gestaltet. (statt super(...)).

Ansonsten referenziert es das Objekt, für dass eine Methode aufgerufen wurde.