Name: Matrikelnummer:

# Bachelorprüfung: Objektorientierte Softwareentwicklung

**WS16/17**

**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**: | | | | | | | | |

## Aufgabe 1 (Modellieren und Schreiben von Klassen)

Sie sollen in dieser Aufgabe Klassen entwickeln, die es erlauben, Rauminformationen zu speichern.

**a)** Entwickeln Sie eine Klasse `Raum`. Ein Objekt dieser Klasse repräsentiert einen Raum in einem Gebäude. Es soll zwei Eigenschaften haben: die Raumnummer, die durch eine Zeichenkette dargestellt ist und die Anzahl der Sitzplätze in dem Raum.

Schreiben Sie einen Konstruktor und überschreiben Sie die Methode `equals` auf adequate Weise. Überschreiben Sie auch die Methode `toString` auf sinnvolle Weise.

**b)** Schreiben Sie eine Unterklasse `Rechnerraum` der Klasse `Raum`. Rechneräume haben an jedem Arbeitsplatz einen PC. Die Klasse `Rechnerraum` soll als zusätzliche Eigenschaft einen Aufzählungswert haben, der angibt, welches Betriebssystem auf den Rechnern installiert ist. Gültige Aufzählungswerte sollen sein: Unix, Linux, Windows, MacOS.

Die Methode `toString` soll überschrieben werden, um die zusätzliche Information mit zu berücksichtigen.

**c)** Schreiben Sie eine Klasse `Gebäude`. Diese Klasse soll als Eigenschaft eine Liste von Räumen enthalten. Benutzen Sie hierzu Klassen aus dem Paket `java.util`.

Schreiben Sie eine Methode `int wievielPlatz(String raum)`, die angibt, wieviel Sitzplätze der Raum mit der übergebenen Raumnummer hat.

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

Listing 1: Aufgabe2a.java

**b)** Betrachten Sie folgende Klasse:

```
1
2   class Aufgabe2b{
3   static int p(int a, int b) {
4      if (!(a>b)) return b-343;
5      return (a+b)%2==1 ? (a+p(a-1,b+2)) : (a+p(a-2,b+1));
6    }
7
8   public static void main(String [] args){
9     System.out.println(p(49,25));
10  }
11 }
```

Listing 2: Aufgabe2b.java

Berechnen Sie schrittweise das Ergebnis des Ausdrucks `p(49,25)`.

## Aufgabe 3 (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("Drücken");
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 `FactorialGUI`, 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 Fakultät dieser Zahl auf dem Feld `output` angezeigt wird.

**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.

**c)** Lösen Sie jetzt Aufgabe a) mit Hilfe der Klasse `GeneralGUI` und einem Lambda-Ausdruck im super-Aufruf.

**d)** Schreiben Sie eine Applikation mit einer `main`-Methode, in der ein Fenster mit einem `SquareGUI`-Objekt geöffnet wird.

## Aufgabe 4 (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 get(int i) {
19     return array[i];
20   }
21   public int size(){return size;}
22   public boolean isEmpty(){return size == 0;}
23 }
```

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:

```
boolean contains(A o);
```

Es soll genau dann wahr sein, wenn ein Element der Liste, gleich dem übergebenen Parameters `o` ist.

**b)** `public boolean containsAll(Li<A> o)`

Es soll genau dann `true` zurück gegeben werden, wenn alle Elemente der übergebenen Liste in der Liste enthalten sind.

**c)** Gegeben sei zusätzlich folgende Schnittstelle:

```
1 public interface Comp<A>{
2   boolean isSmaller(A a1, A a2);
3 }
```

Listing 7: Comp.java

`public boolean isSorted(Comp<A> comp)`

Sie soll genau dann wahr sein, wenn die Elemente der Liste aufsteigend sortiert sind. Als Kleiner-Test wird dabei die Methode `isSmaller` des übergebenen `Comp`-Objekts verwendet.

## Aufgabe 5 Zusammenhänge

Erklären Sie in kurzen Worten.

**a)** Was passiert, wenn eine Rekursion oft aufgerufen wird. Woran liegt das? Was sollte man in solchen Fällen stattdessen machen?

**b)** Was beschreibt ein *Encoding*. Wo wird es verwendet?

**c)** Können Schnittstellen Konstruktoren haben und können Schnittstellen konkrete Methoden haben?

**d)** Was wird mit dem Schlüsselwörtern `break` und `continue` ausgedrückt?

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