KLAUSUR WS 07-08
Programmieren 2 - 90 mn

Name: [redacted]

Nr: [redacted] Punkte: 71/71 /75 Note: 1,0

Wenn zutreffend, bitte ankreuzen:
Zeitablauf ○ Dritter Versuch ○ WEITER SO im STUDIUM!

Aufgabe 1 (15 Punkte) Schreiben Sie den Rumpf der folgenden Methode, damit sie leistet, was im Kommentar geschrieben ist.

```
/** Wie viele Konstruktoren mit mindestens einem int-Parameter
  * hat die Laufzeitklasse des Parameters o?
  * Diese Funktion liefert die Antwort.
  **/

public static int anzahlKonstruktorenMitIntParameter(Object o){
    Class<?> c = o.getClass();
    int ret = 0;
    Constructor[] cons = c.getConstructors();
    for(Constructor con : cons){
      Class<?>[] params = con.getParameterTypes();

      for(Class<?> para : params){
        if(para == Integer.TYPE){
          ret++;
          break;
        }
      }
    }
    return ret;
}
```

15

V

Aufgabe 2 (15 Punkte).
Betrachten Sie die Klasse Punkt

```
class Punkt<E extends Number> {
  E x;
  E y;
  public Punkt(E x, E y){
     this.x = x;
     this.y = y;
  }
  public boolean equals(Object o){
     if (o == this) return true;
     if (! (o instanceof Punkt)) return false;
     Punkt<E> p = (Punkt<E>) o;
     return x.equals(p.x) && y.equals(p.y);
  }
} // class
```

Welche der folgenden Befehlen sind richtig und welche sind nicht richtig? Begründen Sie Ihre Antwort (keine Punkte ohne Begründung).

```
1. Punkt<Integer> p01 = new Punkt<Integer>(1, 3);
2. Punkt<Double> p02 = new Punkt<Double>(1.0, 3.0);
3. Punkt<Number> p03 = new Punkt<Integer>(1, 6);
4. Punkt<Number> p04 = new Punkt<Number>(1, 3.0);
5. Punkt<Object> p05 = new Punkt<Object>(1, 3.0);
6. System.out.println(p01.equals(p02));
7. p01 = p02;
8. p02 = new Punkt<Double>(1.2, 0.5);
9. p04 = p02;
```

	Richtig(J/N)	*Begründung*
1	J	Korrekter Typ, korrekter Konstruktoraufruf ✓
2	J	Auch hier alles in Ordnung (Parameter vom Typ Double) ✓
3	N	Punkt<Integer> ist keine Unterklasse von Punkt<Number> ✓
4	J	Die Parameter für den Konstruktor Aufruf passen in den Datentyp Number ✓
5	J	1, 3.0 können problemlos als Object gespeichert werden (alles erbt von Object) ✗
6	N	Cast von Punkt<Double> nach Punkt<Integer> nicht möglich ✗

equals hat als Parameter vom Typ Object.

	Richtig(J/N)	Begründung
7	N	Compile error : Incompatible types ✓
8	J	neues Object passt in poz (Punkt<Double>) ✓
9	N	Wie bei Aufgabe 7. Ein "cast" von ~~Punkt<Number>~~ ~~nach~~ Punkt<Number> nach Punkt<Double> ✓ ist nicht möglich

Aufgabe 3 (10 Punkte). Schreiben Sie den Rumpf der folgenden Methode, damit sie leistet, was im Kommentar geschrieben ist.

```
public class MeinHashSet<K> extends AbstractSet<K> implements
Set<K> {

  ArrayList<LinkedList<K>> all;

  public void prettyPrinting(){
    // Zeigt am Bildschirm der Inhalt von all
    // in zweidimensionaler Form an.  Zum Beispiel:
    // Falls all 6 verketteten Listen enthält und
    // die Liste an der Position 1 zwei Zeichenketten „aa"
    // und „bb" enthält, die Liste an der Position 4
    // eine Zeichenkette „ccc" enthält
    // und alle anderen  verkettete Listen leer sind,
    // wird ein Aufruf dieser Methode das Folgende anzeigen:
    // 0 []
    // 1 [aa, bb]
    // 2 []
    // 3 []
    // 4 [ccc]
    // 5 []
    bool first = true;
    for (int i=0; i<all.size(); i++) {
      System.out.print(i + " [");
      Iterator<LinkedList<K>> it = all.get(i);
      while (it.hasNext()) {
        if(first) {
          first = false;
          System.out.print(it.next().toString());
        }
        else
          System.out.print(", " + it.next().toString());
      } // while
      System.out.println("]");
    }
```

10

Es geht einfach: toString von LinkedList benutzen

Aufgabe 4 (10 Punkte) Schreiben Sie ein XML-Dokument, welches entsprechend der folgenden DTD *gültig* (engl. valid) ist.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- DTD fuer die Klausur -->
<!ELEMENT methodCall (methodName, params)>
<!ATTLIST methodCall
        modifier1 ( private | public | protected | absent) "public"
        modifier2 (static | absent) "absent">
<!ELEMENT params (param*)>
<!ELEMENT param (int | float | string )>
<!ELEMENT int #PCDATA>
<!ELEMENT float #PCDATA>
<!ELEMENT string #PCDATA>
```

10

<!ELEMENT methodName #PCDATA

<?xml version="1.0" encoding="UTF-8" ?>
<methodCall modifier1="public" modifier2="static">
<methodName> myFirstMethod </methodName>
<params>
<param>
<int> 12 </int>
</param>
</params>
</methodCall>

✓

Aufgabe 5 (10 Punkte) Betrachten Sie die Klasse `GOberflaeche02` und schreiben Sie dafür eine Klasse `MyListener`. Diese Klasse muss in einer eigenen Datei namens `MyListener.java` stehen. Wenn man ein `MyListener`-Objekt bei einem `JButton`-Objekt mittels der Methode `register` anmeldet, soll dies die folgende Wirkung haben: der Text, den eine Benutzerin in die Variable `textfeld` eingibt, soll in der Variable `button` erscheinen, wenn die graphische Komponente `button` geklickt wird.

```
import java.awt.BorderLayout;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.JTextField;

public class GOberflaeche02 extends JFrame {
  private JTextField textfeld;
  private JButton button ;

  public GOberflaeche02 () {
    this.setDefaultCloseOperation(EXIT_ON_CLOSE );
    textfeld = new JTextField();
    button = new JButton("OK");
    this.getContentPane().add(textfeld,BorderLayout.NORTH);
    this.getContentPane().add(button,BorderLayout.CENTER);
    this.setSize(300, 100);
    this.setVisible(true);
  }

  public void register(ActionListener al){
     button.addActionListener(al);
  }

  public String getTextTextfeld() {
     return textfeld.getText();
  }
  public String getTextButton() {
     return button.getText();
  }
  public void setTextTextfeld(String s) {
     textfeld.setText(s);
  }
  public void setTextButton(String s) {
     button.setText(s);
  }
}
```

```
public class MyListener implements ActionListener {
    Oberflaeche02 gui;
    public MyListener (Oberflaeche02 gui)
     this.gui = gui;
    }

    public void actionPerformed (ActionEvent aE) {
      gui.setTextButton(gui.getTextTextfeld());

    }
}
```

Aufgabe 6 (15 Punkte).

6.1 Die Methode `istVokal` liefert `true` zurück, falls ihr Parameter ein Vokal ist, das heißt a, e, i, o oder u ist, sonst liefert sie `false`:

```
static boolean istVokal(char c)
```

Benutzen Sie diese Methode, um die Methode `anzahlVokale` ihrem Kommentar entsprechend zu vervollständigen. Sie müssen `anzahlVokalen` rekursiv programmieren (die Methode ruft sich selber auf und enthält keine Schleife).

```
public static int anzahlVokale(String s){
// Liefert die Anzahl der Vokalen in S.
// Zum Beipiel liefert:
// anzahlVokalen("") die Zahl 0
// anzahlVokalen("a") die Zahl 1
// anzahlVokalen("carola") die Zahl 3
```

12

```
if(s.length() <= 0) return 0;
if( istVokal(s.charAt(0)))
   return (1+ anzahlVokale(s.substring(1, s.length()));
else
   return anzahlVokale(s.substring(1, s.length());
```

V

```
}
```

1*: ist auf der Rückseite!

6.2 Schreiben Sie eine Methode `printInt` ihrem Kommentar entsprechend. Sie können die Variable `digitTable` benutzen. Sie müssen `printInt` rekursiv programmieren (die Methode ruft sich selber auf und enthält keine Schleife).

```
private final static String digitTable="0123456789abcdef";

// Precondition   2<= b <= 16  und n >=0
public static void printInt(int n, int b){
// zeigt an der Konsole die Zahl n in der Basis b. Zum Beispiel:
// printInt(10, 2) zeigt 1010
// printInt(10, 3) zeigt 101
// printInt(10, 10) zeigt 10
// printInt(10, 16) zeigt a
// printInt(0, 16) zeigt 0
```

3

```
   if(n / b > 0) {
   printInt( n/b, b);
   }System.out.print( digitTable.charAt(n%b));
```

V

```
}
```

```
/*: if (s.length() == 1) {
    if (isVokal(s.charAt(0))) return 1;
    else return 0;
}
```