Klausur

Aufgabe	**Maximale Punktzahl**	**Erreichte Punktzahl**
Aufgabe 1	20	
Aufgabe 2	10	
Aufgabe 3	10	
Aufgabe 4	10	
Gesamt	**50**	

Diese Klausur ist eine Testklausur. Die Teilnahme ist freiwillig. Das Ergebnis geht nicht mit in die Note am Semesterende ein.

Hilfsmittel:

- Erlaubtes Material: 1 Blatt handschriftliche Notizen (mit Vor- und Rückseite)
- Nicht erlaubt: Elektronische Geräte in irgendeiner Form, also kein Taschenrechner, Notebook, Handy, usw.
- Dauer: 60 Minuten

1 Aufgabe 1

1.1 Main-Methode

Geben Sie die Signatur der Java-Main-Methode an.

Lösung
`public static void main(String [] args)`

1.2 Auswertung

Welchen Wert hat die Variable `ergebnis` nach folgendem Ausdruck:
`boolean ergebnis = 5 * '0' == 0;`

Lösung
`false`

1.3 Auswertung

Welchen Wert hat die Variable `ergebnis` nach folgendem Ausdruck:
`double ergebnis = 7/2;`

Lösung
`3.0`

1.4 Zusammengesetzte Logische Ausdrücke

Geben Sie für den folgenden umgangssprachlich beschriebenen Ausdruck einen entsprechenden Java-Ausdruck an. Der Java-Ausdruck soll `true` liefern, wenn die Bedingung zutrifft, und ansonsten `false` (`i` und `j` sind vom Typ `int`, `a` und `b` sind vom Typ `boolean`).
`i ist gleich j oder a ist gleich b.`

Lösung
`(i == j ) || (a == b)`

1.5 Schleifen

Geben Sie Quellcode mit einer for-Schleife an, der sich genauso verhält, wie diese do-while-Schleife:

```
int x = 23;
do {
  System.out.println(x);
  x += 1;
} while ( x < 29 );
```

Lösung

```
int x;
for ( x = 23; x < 29; x++){
  System.out.println(x);
}
```

1.6 Dreistelliger Bedingter Operator

Welchen Wert hat die Variable `ergebnis` nach dem folgenden Ausdruck?

```
int ergebnis = ((int)3.5 == 3)? 0 : 1;
```

Lösung

```
0
```

1.7 UML

Zeichnen Sie das UML-Klassendiagramm für folgende Klasse:

```
public class Affe {
  private int alter;
  public Affe(int alter){
    this.alter = alter;
  }
  public static void bruell(){
    System.out.println("Brüll!");
  }
}
```

Lösung

Affe
- alter : int
+ Affe(int) + bruell : void

1.8 Aufzählungstypen

Definieren Sie einen Aufzählungstyp für Sehenswürdigkeiten: Alster, Hafen, Rathaus.

Lösung

```
enum Sehenswuerdigkeiten {ALSTER, HAFEN, RATHAUS}
```

1.9 Methode

Schreiben Sie eine öffentlich sichtbare Methode, die einen ganzzahligen Parameter hat. Sie liefert wahr zurück, falls der Parameter-Wert gerade ist, ansonsten falsch.

Lösung

```
public boolean istGerade(int zahl){
  return zahl % 2 == 0;
}
```

1.10 Methode

Wie nennt man es, wenn es mehrere Methoden mit gleichem Namen in einer Klasse gibt (die aber unterschiedliche Argumentlisten haben)?

Lösung
Überladen von Methoden

2 Aufgabe 2

Gegeben ist folgende Klasse `Zahlenpaar`:

```
public class Zahlenpaar {
  private int zahl1;
  private int zahl2;
}
```

2.1 Selbstbeschreibung

Erweitern Sie die Klasse um eine Methode zur Ausgabe des aktuellen Zustands auf der Konsole.

Lösung

```
public String toString(){
  return zahl1 + ",  " + zahl2;
}
```

2.2 Konstruktor

Geben Sie zwei Konstruktoren an:

- einen Konstruktor, der die Initialwerte der beiden Zahlen als Argumente bekommt.
- einen Kopier-Konstruktor, der den ersten Konstruktor wiederverwendet

Lösung

```
public Zahlenpaar(int zahl1, int zahl2){
  this.zahl1 = zahl1;
  this.zahl2 = zahl2;
}
public Zahlenpaar(Zahlenpaar paar){
  this(paar.zahl1, paar.zahl2);
}
```

2.3 Unveränderlichkeit

Wie müssen Sie die ursprüngliche Klasse anpassen, sodass es eine unveränderliche Klasse wird?

Lösung
`final` vor die beiden Objektvariablen.

3 Aufgabe 3

In dieser Aufgabe entwickeln Sie ein Programm zu Berechnen der Nullstellen einer linearen Gleichung der Form

$$f(x) = ax + b.$$

Dazu ist folgende Klasse gegeben:

```
public class LineareGleichung {
  private double a;
  private double b;
  public LineareGleichung(double a, double b){
    this.a = a;
    this.b = b;
  }
  public double getWert(double x){
    return a * x + b;
  }
}
```

3.1 Nullstelle

Die Nullstelle der linearen Gleichung liegt natürlich bei

$$x = -\frac{b}{a}.$$

Schreiben Sie eine Methode `berechneNullstelle`, die die Nullstelle berechnet und zurückliefert.

Lösung

```
public double berechneNullstelle(){
  return - b / a;
}
```

3.2 Test

Schreiben Sie eine JUnit-Test-Methode für die Methode `berechneNullstelle` mit mindestens zwei Testfällen.

Lösung

```
@Test
public void testBerechneNullstelle() {
  LineareGleichung lineareGleichung1 = new LineareGleichung(1, 1);
  assertEquals("Nullstelle stimmt nicht.", -1,
      lineareGleichung1.berechneNullstelle(), 1e-5);
  LineareGleichung lineareGleichung2 = new LineareGleichung(0.5, 0);
  assertEquals("Nullstelle stimmt nicht.", 0,
      lineareGleichung2.berechneNullstelle(), 1e-5);
  LineareGleichung lineareGleichung3 = new LineareGleichung(0.5, -2);
  assertEquals("Nullstelle stimmt nicht.", 4,
      lineareGleichung3.berechneNullstelle(), 1e-5);
}
```

4 Aufgabe 4

In dieser Aufgabe entwickeln Sie eine Klasse `Wagen`. Ein `Wagen` hat eine gewissen Ladung in Tonnen. Außerdem kann (muss aber nicht) ein `Wagen` einen Anhänger haben (ebenfalls vom Typ `Wagen`). Für die Ladung und den Nachfolgewagen werden Getter und Setter benötigt.

4.1 UML-Klassendiagramm

Zeichnen Sie das Klassendiagramm für den Typ `Wagen`.

Lösung

Wagen
- ladung : int - nachfolger: Wagen
+ getLadung() : int + setLadung(int) : void + getNachfolger() : Wagen + setNachfolger(Wagen) : void

4.2 Implementierung

Schreiben Sie die Klasse `Wagen`.

Lösung

```
public class Wagen {
  private int ladung;
  private Wagen nachfolger;
  public int getLadung(){ return ladung; }
```

```
  public void setLadung(int l){ ladung = l;}
  public Wagen getNachfolger(){ return nachfolger; }
  public void setNachfolger(Wagen w){ nachfolger = w; }
}
```

4.3 Traversierung

Sie haben nur Zugriff auf ein Objekt `wagen` vom Typ `Wagen`. Schreiben Sie Code mit dem der Wagen und all seine Nachfolger durchlaufen werden und für jeden besuchten Wagen die Ladung auf der Konsole ausgegeben wird.

Lösung

```
Wagen aktuellerWagen = wagen;
while( aktuellerWagen != null ){
  System.out.println("Ladung: " + aktuellerWagen.getLadung());
  aktuellerWagen = aktuellerWagen.getNachfolger();
}
```