Programmierung I – Praxis – Test 2

Nachname, Vorname Matrikel-Nr.

Aufgabe 1: Aufbau von Klassen

Was sind die Bestandteile einer Klassendeklaration? (1)

Umkreisen Sie die 3 Fehler in der folgenden Version des Konstruktors der Klasse *Ticketautomat*? (1)

```
public Ticketautomat(int ticketpreis)
    {
        int preis = ticketpreis;
        bisherGezahlt = 0
        gesamtsumme = 0;
    }
}
```

Vervollständigen Sie den Rumpf der folgenden Methode, die die Aufgabe hat, den Wert des Attributs *punktestand* um den Wert des Methodenparameters zu erhöhen: (1)

```
public void erhoehen(int punkte)
{



}
```

Aufgabe 2: Ausdrücke und Anweisungen

Welches sind die möglichen Werte des Ausdrucks (*n* % *m*) mit *n* und *m* als Variable vom Typ *int*? (2)

Schreiben Sie hinter die *boolschen* Ausdrücke, ob sie *true* oder *false* liefern. (2)

```
!false
!(4 < 5)
(2 > 2) || (4 == 4)
(2 > 2) || (4 == 4) && (1 < 0)
(34 != 33) && !false
```

Schreiben Sie eine Methode `void gibTarifinformation(int alter)` die - abhängig vom eingegebenen Alter - per Konsolenausschrift den Tarif für ein Ticket ausgibt. Die Tarife sind folgendermaßen festglegt: Jugendliche (9-16 Jahre) 150 Cent, Erwachsene (über 16) 240 Cent. Die Ausschrift kann ruhig sehr knapp ausfallen, z.B. `Jugend: 150` (3)

Programmierung I – Praxis – Test 3

Nachname, Vorname Matrikel-Nr.

Variablen und Datentypen

Sie wissen, dass es der Java-Compiler nicht zulässt, Werte aus großen Bechern in kleine Becher zu stecken, dies umgekehrt aber möglich ist. Versuchen Sie, auf der Basis Ihrer Kenntnisse über die elementaren Datentypen, die Zuweisungen zu umkreisen, die der Compiler **nicht** erlauben würde. (2)

1.) int x = 34.5;
2.) boolean boo = x;
3.) int g = 17;
4.) int y = g;
5.) y = y + 2.5;
6.) short s;
7.) s = y;
8.) float b = 3.4f;
9.) double v = b;
10.) short n = 12;
11.) v = n;

2

Deklarieren Sie zwei Referenzvariablen `buch1` und `buch2` vom Typ *Buch* und erzeugen Sie **eine** Instanz der Klasse *Buch* mit dem Konstruktor

`Buch(String autor, String titel)` und verknüpfen Sie beide Variablen mit dem Objekt. (2)

Erzeugen Sie eine Instanz der Klasse *Rechner* mit dem Konstruktur `Rechner()` und weisen Sie diese der Variablen `rechner` zu, die den passenden Typ haben sollte. (1)

Schreiben Sie für die Klasse Rechner den Rumpf der folgende Methode, die die Summe der als Argumente übergebenen Zahlen berechnet und das Ergebnis zurück liefert. (3)

```
public int gibSumme (int x, int y)
{




}
```

Rufen Sie für die oben deklarierte Variable `rechner` die Methode `gibSumme` mit den Werten 3 und 4 auf und weisen Sie das Ergebnis einer Variablen `resultat` vom passenden Typ zu. (2)

Programmierung I – Praxis – Test 4

Nachname, Vorname Matrikel-Nr.

Aufgabe 1: Objektinteraktion

Die Klasse *Stoppuhr* modelliert eine einfache Stoppuhr, die nur Sekunden und Minuten hoch zählen kann (bis max. 90 Minuten, danach fängt sie wieder bei 0 an). Die Stoppuhr soll für die Verwaltung der Minuten und Sekunden die Klasse *Nummernanzeige* nutzen, die eine zweistellige Nummernanzeige realisiert. Im Konstruktor der Nummernanzeige kann man den Grenzwert angeben, nach dem die Nummernanzeige wieder auf 0 springen soll. Deklarieren Sie die benötigten Attribute der Klasse *Stoppuhr* und schreiben Sie einen Konstruktor, der die Stoppuhr in einen definierten Anfangszustand versetzt. Den Rest des Klassenrumpfes können Sie auslassen. (5)

```
public class Stoppuhr {
```

Schreiben Sie eine Methode `reset()`, die die Stoppuhr wieder auf 00:00 setzt. Ein externer Methodenaufruf der Methode `setzeWert(int wert)` der Klasse *Nummernanzeige* kann genutzt werden, um bei den beiden Nummernanzeigen eine bestimmte Zeit einzustellen. Denken Sie daran, die Zeitanzeige zu aktualisieren. Dazu können Sie einen internen Methodenaufruf der Methode `anzeigeAktualisieren()` nutzen, die in der Klasse *Stoppuhr* für diesen Zweck angeboten wird. (5)

Programmierung I – Praxis – Test 5

Nachname, Vorname Matrikel-Nr.

Aufgabe 2: Switch-Anweisung

Schreiben Sie im Rumpf der Methode *antwortAusgeben(..)* eine Switch-Anweisung, die, je nach Wert des Parameters *antwort,* eine bestimmte Antwort zurück liefert. Hierbei soll bei Eingabe von *j* die Antwort „Ja.“, bei *n* die Antwort „Nein.“ und bei allen anderen Zeichen die Antwort „Keine Ahnung.“ geliefert werden. (3)

```
public String antwortAusgeben(char antwort) {
```

Schreiben Sie im Rumpf der Methode *zahlenAbsteigendAusgeben(..)* eine while-Schleife, die von der eingegebenen Zahl *zahl* an alle Zahlen bis 0 absteigend auf dem Bildschirm ausgibt. (3)

```
public void zahlenAbsteigendAusgeben(int zahl) {

}
```

Schreiben Sie im Rumpf der Methode *gebeAusGeradeZahlenBis(..)* eine For-Schleife, die alle geraden Zahlen von 0 bis zu der eingegebenen Zahl auf dem Bildschirm ausgibt. (4)

```
public void gebeAusGeradeZahlenBis(int zahl) {

}
```

Programmierung I – Praxis – Test 6

Nachname, Vorname	Gruppe	Matrikel-Nr.

Objektsammlungen I

Schreiben Sie eine Deklaration für eine Array-Variable *leute*, die auf ein Array von *Person*-Objekten verweisen kann. Schreiben Sie eine Deklaration für eine Array-Variable *verfügbar*, die auf ein Array von *boolean*-Werten verweisen kann. (2)

Korrigieren Sie die folgende Array-Erzeugung, notieren Sie die korrekte Version: (1)

```
double[] preise = new double(50);
```

Schreiben Sie eine For- Schleife, die allen Variablen in *preise* den Wert 1.99 zuweist. (3)

Schreiben Sie eine Deklaration plus Zuweisung, die einer Variablen *meinPreis* vom Typ *double* den Wert des 5. Eintrags aus dem obigen Array *preise* zuweist. (1)

Schreiben Sie die Deklaration eines privaten Attributs *mitglieder*, das ein *Array* halten kann. Die Elemente des *Arrays* sollen vom Typ *Mitglied* sein. (1)

Gegeben sei ein Array *mitglieder* in der diverse Objekte vom Typ *Mitglied* enthalten sind. Auf den Namen eines Mitglieds lässt sich über die Methode *gibName()* von *Mitglied* zugreifen. Schreiben Sie eine Methode *mitgliederAusgeben()*, die auf der Konsole die Namen aller Mitglieder in dem Array *mitglieder* ausgibt. Nutzen Sie hierfür die For-Each-Schleife. (2)

Programmierung I – Praxis – Test 7

Nachname, Vorname | Gruppe | Matrikel-Nr.

5

7

Objektsammlungen II

Schreiben Sie eine import-Anweisung, die die Klasse *MenueKeyEvent* aus dem Paket *javax.swing.event* importiert. (1)

Vervollständigen Sie die Klasse *Bibliothek*. Schreiben Sie die Deklaration des privaten Attributs *buecher*, das eine *ArrayList* halten kann. Die Elemente der *ArrayList* sollen vom Typ *Buch* sein. Initialisieren Sie das Attribut *buecher* im Konstruktor mit einem passenden Objekt vom Typ *ArrayList*. Fügen Sie die beiden Instanzen *buch1* und *buch2* in die *ArrayList* ein. (3)

```
public class Bibliothek {

   private____                                    _________buecher;

   public Bibliothek() {
     buecher = __                                          _______;
     Buch buch1 = new Buch("Robinson Crusoe ", "Daniel Defoe");
     Buch buch2 = new Buch("Faust I", "Johann Wolfgang Goethe");

     -                                          _______________;

     -                                          _______________;
     }
}
```

1

1

0,5

(2,5)

Gehen Sie davon aus, dass die *ArrayList buecher* fünf Einträge enthält. Schreiben Sie eine Anweisung, die den 3. Eintrag aus der *ArrayList buecher* entfernt gefolgt von einer Anweisung, die einer Variablen *meinBuch* vom Typ *Buch* den letzten Eintrag aus der *ArrayList buecher* zuweist. (2)

Ergänzen Sie die Klasse *Bibliothek* um eine Methode *buchTitelAusgeben(String autor)*, die auf der Konsole die Titel aller vorhandenen *Buch*-Objekte des Autors *autor* ausgibt (Sie müssen die Klasse hier nicht noch einmal hinschreiben, nur die Methode). Nutzen Sie die Klasse *Iterator* für diese Aufgabe.

Hinweis: Auf den Autor eines Buchs lässt sich über die Methode *gibAutor()* der Klasse *Buch* zugreifen, mit *gibTitel()* lässt sich der Titel abfragen. (4)

Programmierung I – Praxis – Test 8

Nachname, Vorname Gruppe Matrikel-Nr.

Aufgabe 1: Bibliotheksklassen nutzen

Nehmen Sie an, sie hätten eine Variable *generator* vom Typ *Random*. Schreiben Sie eine Anweisung, die der Variablen *zufallszahl* vom Typ *int* eine Zufallszahl zwischen 1 und 6 zuweist. Die Variable sollte vorher deklariert werden. (1)

Aufgabe 2: HashMap

Schreiben Sie die Deklaration und Initialisierung eines privaten Attributs *cdSammlung*, das eine *HashMap* aufnimmt. Die Schlüssel der *HashMap* sollen vom Typ *String* sein, die Werte sollen vom Typ *CD* sein. (1)

Als Schlüssel für die Verwaltung der CDs in der *HashMap cdSammlung* soll der Name des Interpreten des *CD*-Objekts verwendet werden. Die Klasse *CD* besitzt die Methode *gibInterpret()* um auf diese Zeichenkette zuzugreifen. Schreiben Sie eine Anweisung, die eine Instanz *bestOf* der Klasse *CD* in die *HashMap cdSammlung* einfügt. (2)

Nehmen Sie an, es gäbe in der *HashMap cdSammlung* einen Eintrag zu dem Namen „Heino". Schreiben sie eine Anweisung, die das *CD*-Objekt des Interpreten namens Heino aus der *HashMap* holt und einer lokalen Variablen *magIchNicht* vom Typ CD zuweist. Die lokale Variable sollte vorher deklariert werden. (2)

Programmierung I – Praxis – Test 9

Nachname, Vorname | Gruppe | Matrikel-Nr.

8,5

Aufgabe 1: Klassenattribute

Schreiben Sie eine Klasse *Hund*, die mit einem „Mechanismus" versehen ist, mit dem gezählt werden kann, wie viele *Hund*-Objekte bislang erstellt wurden. Die Klasse *Hund* soll außerdem zwei Attribute *name* und *groesse* besitzen (den Typ der Attribute bitte geeignet wählen), deren Werte über den Konstruktor der Klasse *Hund* gesetzt werden. (5)

Aufgabe 2: Konstanten und Klassenmethoden

Schreiben Sie eine Klasse *MehrwertsteuerRechner*, in der es eine Klassenmethode *berechneMwst* gibt, die für einen eingegebenen Geldbetrag vom Typ float die zu entrichtende Mehrwertsteuer als float-Wert zurück liefert. Der aktuelle Mehrwertsteuersatz (19%) soll hier als Konstante der Klasse festgelegt werden. (5)

Programmierung I – Praxis – Test 10

Nachname, Vorname | Gruppe | Matrikel-Nr.

15

10

Aufgabe 1: Vererbung

Bringen Sie die folgenden Konzepte in eine geeignete Vererbungshierarchie: Studenten und Dozenten haben einen Namen und eine Adresse (String). Ein Student ist in einem bestimmten Fachsemester (int) und hat eine Matrikelnummer (int). Ein Dozent lehrt ein bestimmtes Fachgebiet und besitzt eine email-Adresse (String). Schreiben Sie dazu eine geeignete Oberklasse und geeignete Subklassen (nur Klassen mit Attributdeklarationen, ohne Methoden). (5)

Aufgabe 2: Konstruktoren

Schreiben Sie die Konstruktoren (Köpfe und Methodenrümpfe) für die oben definierten Klassen. Den Konstruktoren sollen alle erforderlichen Attributwerte als Methodenparameter übergeben werden können. (5)