Heinrich-Heine-Universität Düsseldorf
Institut für Informatik
Dr. Markus Brenneis

28. Juli 2022

Drittklausur

Programmierung
WS 21/22

Nachname:		Vorname:	
Matrikelnummer:		Sitzplatznummer:	
Zusätzliche Blätter:		Unterschrift:	

Hinweise:

- Diese Klausur enthält 19 nummerierte Klausurseiten. Prüfen Sie bitte zuerst, ob die Klausur alle Seiten enthält. Sie dürfen die Heftung der Klausur nicht auftrennen.
- Sie erhalten außerdem von uns leere Blätter. Sollten Sie auf diesen Blättern für die Korrektur relevante Teile Ihrer Lösung notieren, markieren Sie dies deutlich an der entsprechenden Aufgabe und auf dem zusätzlichen Blatt. Schreiben Sie auf alle zusätzlichen Blätter Ihren Namen. Geben Sie außerdem oben auf dem Deckblatt an, wie viele zusätzliche Blätter Sie zur Korrektur abgeben. Wenn Sie weiteres Papier benötigen, melden Sie sich bitte.
- Alle Fachbegriffe in dieser Klausur werden wie in der Vorlesung definiert verwendet. Alle Fragen beziehen sich auf die in der Vorlesung vorgestellte Java-Version 11. Programmcode muss in Java geschrieben werden.
- Antworten dürfen auf Deutsch oder Englisch gegeben werden. Sofern keine ausformulierten Sätze verlangt sind, reichen nachvollziehbare Stichworte als Antwort.
- Zugelassene Hilfsmittel: eine beidseitig beschriebene oder bedruckte DIN-A4-Seite, Wörterbuch (Wörterbücher müssen vor Beginn der Klausur den Aufsichtspersonen zur Kontrolle vorgelegt werden.)
- Schalten Sie Ihr Mobiltelefon aus. Täuschungsversuche führen zum sofortigen Ausschluss von der Klausur. Die Klausur wird dann als nicht bestanden gewertet.
- Schreiben Sie **nicht** mit radierbaren Stiften und auch **nicht** mit rot!

Diesen Teil bitte nicht ausfüllen:

Aufgabe	1	2	3	4	5	6	7	8	9	Σ
Punktzahl	8	5	5	13	13	7	6	6	27	90
Erreicht										

Aufgabe 1

_____ / 8 Punkte

(a) [1 Punkt] Ihr Terminal sieht gerade wie folgt aus:

```
/mnt/projects/blatt02 % ls
Hello.java Test.java
/mnt/projects/blatt02 %
```

Sie wollen die Klasse `Hello` kompilieren (aber nicht ausführen). Welchen Befehl müssen Sie dazu eingeben?

(b) [4 Punkte] Gegeben sei die folgende Klasse:

Studi.java — Java

```java
public class Studi {
    private String name;
    private int nummer;

    public Studi(String name, int nummer) {
        this.name = name;
        this.nummer = nummer;
    }

    public int getNummer() {
        return numer;
    }

    @Override
    public String tostring() {
        return name + ", " + nummer;
    }
}
```

Beim Compilieren der Klasse gibt es folgende Fehlermeldungen:

```
Studi.java:11: error: cannot find symbol
        return numer;
               ^
  symbol:   variable numer
  location: class Studi
Studi.java:14: error: method does not override or implement a method from a
   supertype
    @Override
    ^
2 errors
```

Geben Sie an, in welchen Zeilen die **Ursachen** für die Fehler sind, beschreiben Sie die Fehler jeweils kurz (max. 1 Satz) und geben Sie die korrigierte Codezeile **vollständig** an, sodass der Code das tut, was bei der Programmierung wahrscheinlich vorgesehen war. Geben Sie keine Folgefehler an, die durch Korrektur eines vorherigen Fehlers behoben werden.

Zeilennummer: _____

Fehlerbeschreibung: _____

Korrektur: _____

Zeilennummer: _____

Fehlerbeschreibung: _____

Korrektur: _____

Zeilennummer: _____

Fehlerbeschreibung: _____

Korrektur: _____

(c) [3 Punkte] Gegeben seien folgende Dateien und Java-Programme:

- Textdatei `/projekt/werte.txt`: enthält ganzzahlige Messwerte; ungültige Messwerte sind als -1 gespeichert
- Java-Programm `Filter`: liest zeilenweise Integer von der Standardausgabe ein und gibt alle Integer, die nicht -1 sind, zeilenweise auf der Standardausgabe aus
- Java-Programm `Auswertung`: liest zeilenweise gültige Messwerte von der Standardeingabe ein und gibt eine Auswertung auf der Standardausgabe aus

```
/projekt % ls
Auswertung.class Auswertung.java Filter.class Filter.java werte.txt
/projekt %
```

Geben Sie einen Befehl an, mit dem Sie eine Auswertung aller **gültigen** Messwerte aus `werte.txt` auf der Standardausgabe erhalten:

_____

Aufgabe 2

_____ / 5 Punkte

Formen Sie die Kontrollstrukturen in den folgenden Codeausschnitten um, sodass sich die Semantik des Codes nicht ändert. Benutzen Sie in Ihrem Code jeweils nur die von der Aufgabe vorgegebene Art von Kontrollstruktur.

(a) [2 Punkte] Formen Sie die folgende for-Schleife in eine for-each-Schleife um:

Vorgabe Java

```java
double[] noten = {2.0, 1.3, 2.6};
for(int index = 0; index < noten.length; index++) {
    System.out.print(noten[index]);
}
```

Ihre Lösung Java

```java
double[] noten = {2.0, 1.3, 2.6};
```

(b) [3 Punkte] Formen Sie die folgende if-Verzweigung in eine switch-Verzweigung um:

Vorgabe Java

```java
int choice = 2;
if(choice == 1 || choice == 3) {
    System.out.print("0,8");
} else if(choice == 2) {
    System.out.print("1,2");
} else {
    System.out.print("invalid");
}
System.out.println("You entered " + choice);
```

Ihre Lösung Java

```java
int choice = 2;

System.out.println("You entered " + choice);
```

Aufgabe 3

_____ / 5 Punkte

(a) [1 Punkt] Gegeben sei der reguläre Ausdruck `[a-z][a-z0-9]+`. Kreuzen Sie alle Strings an, die vollständig von diesem Ausdruck gematcht werden:

- ☐ `123a`
- ☐ `a`
- ☐ `1ab`
- ☐ `A12`
- ☐ `a19z`

(b) [1 Punkt] Der oben angegebene Ausdruck ist gleichbedeutend mit:

- ☐ `[A-Z][A-Z0-9]+`
- ☐ `[a-z0-9]*`
- ☐ `[a-z][a-z0-9][a-z0-9]*`

(c) [2 Punkte] Sie speichern in einem Programm mehreren Millionen Kundendatensätze mit einer Kundennummer. Die Reihenfolge, in der die Datensätze gespeichert sind, ist Ihnen egal. Sie wollen aber häufig und schnell herausfinden können, zu welchem Datensatz eine gegebene Kundennummer gehört.

Welche Datenstruktur würden Sie für diesen Anwendungsfall benutzen, um die Kundendatensätze zu speichern? Geben Sie eine nachvollziehbare Begründung in 1–3 ausformulierten Sätzen an.

(d) [1 Punkt] Angenommen, es soll in einer Java-Anwendung ein Hashset benutzt werden. Warum ist es sinnvoller, die JDK-Klasse `java.util.HashSet<E>` zu verwenden, anstatt ein Hashset selbst zu implementieren? Geben Sie einen nachvollziehbaren Grund in 1–2 ausformulierten Sätzen an.

Aufgabe 4

_____ / 13 Punkte

Für eine Finanzanwendung soll ein Programm für Zinsrechnung erstellt werden (ohne Zinseszins). Vervollständigen Sie dafür die Klasse `Zinsen`:

(a) [3 Punkte] Schreiben Sie eine **private, statische** Methode `gesamtkapital(double k0, double p, int n)`, die berechnet, auf welches Gesamtkapital K_n das Startkapital K_0 nach n Jahren bei einem jährlichen Zinssatz p gewachsen ist, und K_n zurückgibt. Die Berechnungsvorschrift lautet:

$$K_n = K_0 \cdot (1 + p \cdot n)$$

(b) [10 Punkte] Legen Sie eine `main`-Methode an, die das Startkapital K_0 (Kommazahl größer 0) und den Zinssatz p (Kommazahl größer 0) in dieser Reihenfolge als Konsolenargumente entgegennimmt. Es soll dann jeweils das Gesamtkapital K_n nach $n = 1, 2, 3$ usw. Jahren ausgegeben werden, bis das Endkapital **größer** als das doppelte Startkapital ist.

Wenn **etwas anderes** als Zahlen, **zu kleine** Zahlen oder **nicht genau zwei** Zahlen angegeben werden, soll die Fehlermeldung „invalid input“ ausgegeben werden. *Zur Erinnerung: Die Methoden zum Parsen werfen im Fehlerfall eine* `NumberFormatException`.

Beispiele:

```
% java Zinsen 100 0.25
125.0
150.0
175.0
200.0
225.0
% java Zinsen 100
invalid input
% java Zinsen 100 0
invalid input
% java Zinsen 100 zwei
invalid input
```

Zinsen.java Java

```
public class Zinsen {
```

Zinsen.java (Fortsetzung) Java

```
}
```

Aufgabe 5

_____ / 13 Punkte

Gehen Sie in dieser Aufgabe davon aus, dass $n \in \mathbb{N}$ und $n \geq 1$ gilt.

Eine natürliche Zahl n ist eine Rechteckzahl, wenn sie das Produkt zweier aufeinanderfolgender natürlicher Zahlen ist. Beispielsweise ist $12 = 3 \cdot 4$ eine Rechteckzahl.

Unten ist ein Programm `Rechteckzahlen` mit einer Beispielausgabe vorgegeben. Erweitern Sie dieses Programm um folgende **private**, **statische** Methoden, damit das Programm wie im Beispiel funktioniert:

(a) [4 Punkte] Schreiben Sie eine Methode `boolean istRechteckzahl(int n)`, die genau dann `true` zurückgibt, wenn `n` eine Reckteckzahl ist.

(b) [5 Punkte] Schreiben Sie eine Methode `int[] reckteckzahlen(int anzahl)`, die ein Array zurückgibt, das genau die ersten `anzahl` Rechteckzahlen enthält.

(c) [4 Punkte] Schreiben Sie eine Methode `int rechteckSumme(int anzahl)`, die die Summe der ersten `anzahl` Rechteckzahlen zurückgibt. Falls `anzahl < 0` ist, soll diese Methode eine `IllegalArgumentException` werfen.

Beispiel:

```
% java Rechteckzahlen
false
true
true
70
2 6 12 20 30
```

Rechteckzahlen.java — Java

```java
public class Rechteckzahlen {

    public static void main(String[] args) {
        System.out.println(istRechteckzahl(1));
        System.out.println(istRechteckzahl(2));
        System.out.println(istRechteckzahl(12));

        System.out.println(rechteckSumme(5));

        for(int zahl: rechteckzahlen(5)) {
            System.out.print(zahl + " ");
        }
    }

    // Ergänzen Sie hier die Methoden istRechteckzahl, rechteckzahlen, rechteckSumme.
```

Rechteckzahlen.java (Fortsetzung) Java

```
}
```

Aufgabe 6

_____ / 7 Punkte

Aus der Vorlesung kennen Sie folgende Implementierung von Insertion Sort, die ein Array von Integern aufsteigend sortiert:

Java

```
public static void sort(int[] numbers) {
    for(int currentIndex = 0; currentIndex < numbers.length; currentIndex++) {
        int currentNumber = numbers[currentIndex];
        int insertionPosition = currentIndex;
        while(insertionPosition > 0 && numbers[insertionPosition - 1] > currentNumber) {
            numbers[insertionPosition] = numbers[insertionPosition - 1];
            insertionPosition--;
        }
        numbers[insertionPosition] = currentNumber;
    }
}
```

Gegeben sei die folgende Klasse:

File.java Java

```
import java.nio.file.Files;
import java.nio.file.Path;
import java.io.IOException;

public class File {
    private final String path;
    private final long size;

    public File(String path) {
        this.path = path;
        long size;
        try {
            size = Files.size(Path.of(path));
        } catch(IOException e) {
            size = -1;
        }
        this.size = size;
    }

    // gibt die Dateigröße zurück
    // zur Erinnerung: Der Datentyp long kann wie int verwendet werden.
    public long size() {
        return size;
    }
```

Ergänzen Sie die Klasse um eine **öffentliche, statische** Methode `sorted`, die ein Array von `File`-Objekten übergeben bekommt, **absteigend** nach ihrer Dateigröße sortiert und diese als neues `File`-Array zurückgibt; die Reihenfolge der Objekte im übergebenen Array soll dabei von der Methode **nicht** verändert werden. Gehen Sie davon aus, dass das übergebene Array nicht `null` ist und kein Wert im Array `null` ist.

File.java (Fortsetzung) Java

File.java (Fortsetzung) Java

```
}
```

Aufgabe 7

_____ / 6 Punkte

Gegeben sei die Klasse `SearchTree` für einen binären Suchbaum, in dem Doubles gespeichert werden können:

SearchTree.java — Java

```java
public class SearchTree {

    private class BinaryNode {
        private double element;
        private BinaryNode left, right;

        private BinaryNode(double element) {
            this.element = element;
        }
    }

    private BinaryNode root;

    // fügt newNumber in den Baum ein
    public void insert(double newNumber) {
        // ... (Implementierung nicht abgedruckt)

    }
```

Vervollständigen Sie die Methode `int countNegative()`, die die **Anzahl** aller negativen Einträge im Baum zurückgibt. Nutzen Sie in Ihrem Code die Eigenschaften eines binären Suchbaumes aus, um die Anzahl der betrachteten Knoten minimal zu halten. Sie dürfen zusätzliche Hilfsmethoden mit minimaler Sichtbarkeit schreiben.

SearchTree.java (Fortsetzung) — Java

```java
public int countNegative() {
```

SearchTree.java (Fortsetzung) Java

```
    }
}
```

Aufgabe 8

_____ / 6 Punkte

Gegeben sei die folgende Klasse `LinkedList`, die eine einfach verkettete Liste implementiert, in der Integer gespeichert werden können:

LinkedList.java — Java

```java
public class LinkedList {
    private class Node {
        private int data;
        private Node next;

        private Node(int data, Node next) {
            this.data = data;
            this.next = next;
        }
    }

    private Node head;

    // fügt value vorne in die LinkedListe ein
    public void prepend(int value) {
        head = new Node(value, head);
    }
```

Vervollständigen Sie die Implementierung der Methode `boolean isSorted()`, die genau dann `true` zurückgeben soll, wenn die Zahlen in der Liste aufsteigend sortiert sind; in einer sortieren Liste dürfen Zahlen doppelt vorkommen, eine leere Liste ist sortiert.

LinkedList.java (Fortsetzung) — Java

```java
  public boolean isSorted() {

    if(head == null || _____) {

      _____;

    }

    _____;

    while(_____) {

      if(_____ > _____) {

        _____;

      }

      _____;

    }

    _____;
  }
}
```

Aufgabe 9

_____ / 27 Punkte

In dieser Aufgabe sollen Sie Interfaces, Klassen und Methoden für eine Marketinganwendung eines Restaurants implementieren.

Hinweise:

- Sie dürfen alle Variablennamen frei wählen.
- Wählen Sie sinnvolle Datentypen für Ihre Variablen.
- Alle Instanzvariablen müssen **privat** sein.
- Wenn kein Konstruktorverhalten vorgeschrieben ist, reicht der Default-Konstruktor.
- Das genaue Format von Textausgaben ist Ihnen überlassen.
- Innerhalb dieser Aufgabe müssen Sie keine Exceptions abfangen. Sie müssen keine Parameter validieren.
- Lesen Sie sich die Aufgabenstellung vor Beginn der Implementierung **vollständig** durch, um einen besseren Überblick über das Gesamtbild zu erhalten.
- Gehen Sie davon aus, dass alle in dieser Aufgabe genannten Klassen und Interfaces im selben Package liegen.

Gegeben sind folgende Interfaces:

Weather.java

```java
public interface Weather {
    // gibt die aktuelle Temperatur in Grad Celsius zurück
    double getTemperature();
}
```

Mailer.java

```java
public interface Mailer {
    // sendet an die E-Mail-Adresse den angegebenen Inhalt
    void sendMail(String mailadress, String content);
}
```

(a) [5 Punkte]
Schreiben Sie eine **nicht abstrakte, öffentliche** Klasse `FakeWeather`, die das `Weather`-Interface implementiert. Als aktuelle Temperatur soll irgendeine beliebige Zahl zurückgeben werden.

FakeWeather.java Java

Schreiben Sie eine **nicht abstrakte, öffentliche** Klasse `FakeMailer`, die das `Mailer`-Interface implementiert und Mails wie folgt „verschickt“: Die Methode gibt die übergebenen Strings auf der Standardausgabe aus.

FakeMailer.java Java

(b) [6 Punkte]
Erstellen Sie eine nicht abstrakte, **öffentliche** Klasse `Customer`. Jedes `Customer`-Objekt speichert eine E-Mail-Adresse, einen Namen und ein Geburtsjahr, die alle dem **öffentlichen** Konstruktor als Parameter übergeben werden. Die drei Eigenschaften sollen außerdem von **öffentliche** Methoden `getMail()`, `getName()` bzw. `getYearOfBirth()` zurückgegeben werden.

Customer.java Java

```
public class Customer {









}
```

(c) [10 Punkte] Schreiben Sie eine nicht abstrakte, **öffentliche** Klasse `Marketing`. Der **öffentliche** Konstruktor der Klasse bekommt eine `Weather`-Instanz, ein Array von `Customer`-Objekten und eine `Mailer`-Instanz übergeben und speichert diese ab.

Die Klasse soll eine **öffentliche** Methode `doMarketing` ohne Rückgabewert haben, die sich wie folgt verhält: Ist die aktuelle Temperatur (wie von dem `Weather`-Objekt angegeben) ...

- unter 20 °C, tut sie nichts.
- größer oder gleich 20 °C und kleiner als 30 °C, wird an alle Customer mit einem Geburtsjahr kleiner 1960 mithilfe des gespeicherten Mailers eine Mail geschickt.
- größer oder gleich 30 °C, wird an alle Customers eine Mail geschickt.

Der Mailinhalt soll jeweils nur der Name sein.

Marketing.java Java

```
public class Marketing {









}
```

(d) [6 Punkte]

Ergänzen Sie die `main`-Methode der Klasse `Bendispasta`, sodass folgendes passiert:

1. Eine Instanz von `FakeWeather` wird erstellt.
2. Eine Instanz von `FakeMailer` wird erstellt.
3. Zwei `Customer`-Objekte mit folgenden Namen, Mailadressen und Geburtsjahren werden erstellt:
   - Jan, j@hhu.de, 1993
   - Ute, u@hhu.de, 1951
4. Eine Instanz von `Marketing`, der alle oben genannten Objekte in geeigneter Weise übergeben werden, wird erstellt.
5. `doMarketing` wird aufgerufen.

Bendispasta.java Java

```
public class Bendispasta {
    public static void main(String[] args) {









    }
}
```