

Klausur — Objektorientierte Programmierung

1. Theoretischer Teil: 45min

Aufgabe A — Grundbegriffe

10 P.

Erklären Sie die folgenden fünf Grundbegriffe aus der objektorientierten Programmierung und geben Sie jeweils ein Beispiel an:

- *Attribut*
- *Methode*
- *Klasse*
- *Objekt*
- *Vererbung*

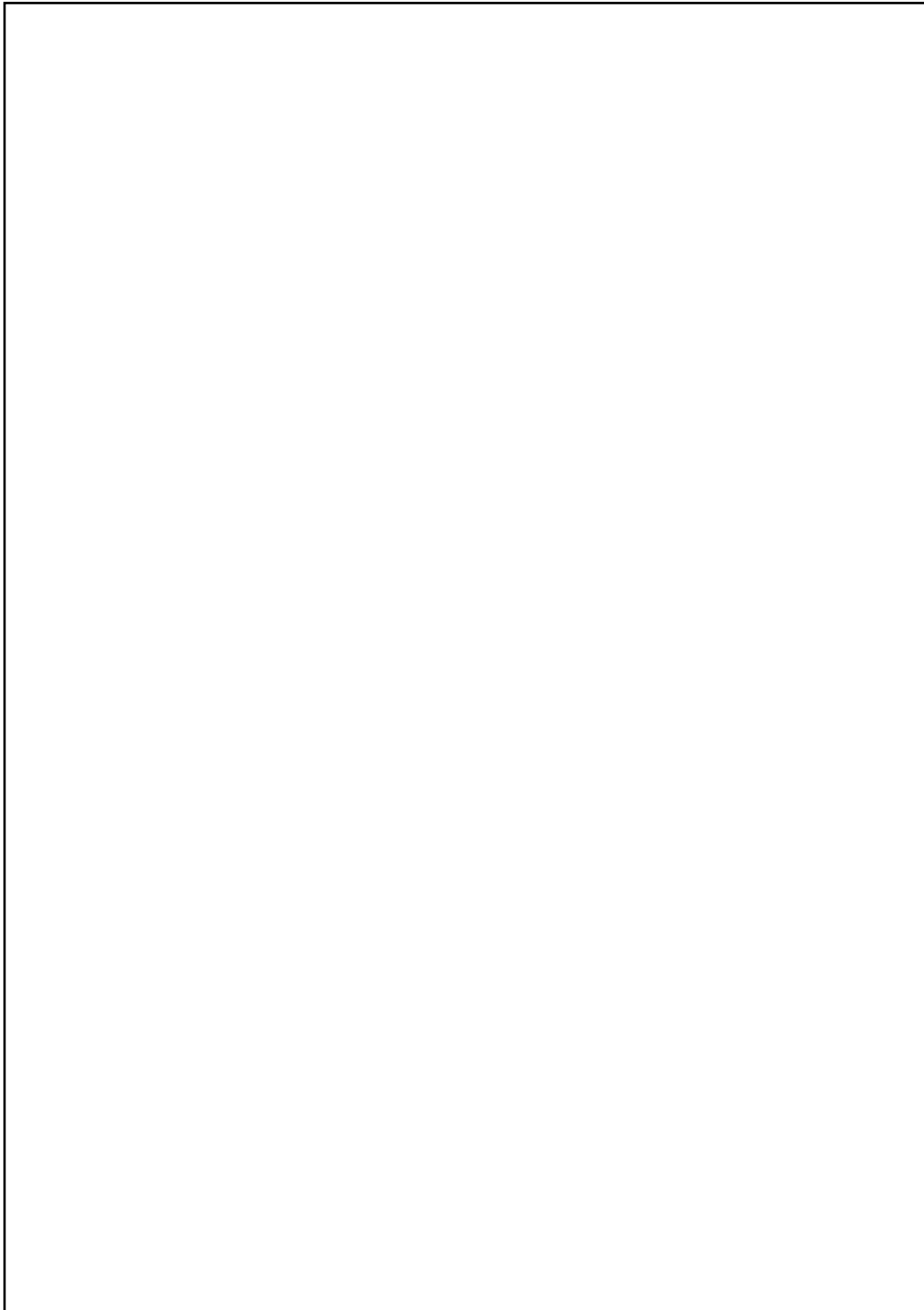
Aufgabe B – Konzepte der Objektorientierten Programmierung

Beschreiben Sie kurz, was die verschiedenen Begriffe und Konzepte in Java bzw. der objektorientierten Programmierung bedeuten.

1.

5 P.

Definieren Sie den Begriff „Polymorphismus“. Geben Sie ein Beispiel an, das zeigt, wie ein Polymorphismus in Java implementiert wird.



2.

5 P.

Erklären Sie die Unterschiede zwischen einer abstrakten Klasse und einem Interface in Java. Nennen Sie jeweils ein Beispiel für den Einsatz.

3. Warum ist es gut, wenn Daten gekapselt werden? Und wie wird diese Datenkapselung erzeugt? Erklären Sie dies und geben Sie ein Beispiel. 5 P.

Aufgabe C – Wahr oder Falsch

16 P.

Entscheiden Sie bei den folgenden Aussagen, ob Sie richtig oder falsch sind:

Frage	Wahr	Falsch
Eine Klasse kann von mehreren anderen Klassen erben.		
Vererbung und Komposition sind zwei Worte für die selbe Sache.		
Eine Referenz auf ein Objekt muss immer von genau der selben Klasse sein wie das Objekt selbst.		
Mit <code>public Auto()</code> definieren Sie einen Konstruktor für die Klasse <code>Auto</code> .		
Eine Klasse kann mehrere Interfaces implementieren.		
Mit dem Schlüsselwort <code>impl</code> können Sie definieren, dass eine Klasse ein Interface implementiert.		
Mit dem Ausdruck <code>let number : i32 = 0;</code> definieren Sie eine Variable mit dem Namen <code>number</code> und dem Wert 0.		
Der Hauptvorteil der Datenkapselung ist es, dass Sie Daten vor unerlaubten Zugriff schützen.		

Aufgabe D – Beschreibung eines bestehenden Programms

10 P.

In dem folgenden Programmcode wird ein einfaches Bibliothekssystem erzeugt. Beschreiben Sie, welche Funktionen mit dem Code abgedeckt werden. Welche Klassen und Methoden sind definiert? Fällt Ihnen ein, wie Sie das Programm erweitern könnten?

1. Die Klasse `Book`

```
1  public class Book {
2      private String title;
3      private String author;
4      private boolean isBorrowed;
5
6      public Book(String title, String author) {
7          this.title = title;
8          this.author = author;
9          this.isBorrowed = false;
10     }
11
12     public String getTitle() {
13         return title;
14     }
15
16     public String getAuthor() {
17         return author;
18     }
19
20     public boolean isBorrowed() {
21         return isBorrowed;
22     }
23
24     public void borrow() {
25         if (!isBorrowed) {
26             isBorrowed = true;
27             System.out.println("Das Buch '" + title + "' wurde ausgeliehen.");
28         } else {
29             System.out.println("Das Buch '" + title + "' ist bereits ausgeliehen.");
30         }
31     }
32
33     public void returnBook() {
34         if (isBorrowed) {
35             isBorrowed = false;
36             System.out.println("Das Buch '" + title + "' wurde zurückgegeben.");
37         } else {
38             System.out.println("Das Buch '" + title + "' war nicht ausgeliehen.");
39         }
40     }
41
42     public String toString() {
```

```
43         return "Buch: " + title + " von " + author + (isBorrowed ? " (ausgeliehen)"  
44             " (verfügbar)");  
45     }
```


2. Die Klasse `Library`

```
1  import java.util.ArrayList;
2
3  public class Library {
4      private ArrayList<Book> books;
5
6      public Library() {
7          books = new ArrayList<>();
8      }
9
10     public void addBook(Book book) {
11         books.add(book);
12         System.out.println("Das Buch '" + book.getTitle() + "' wurde der Bibliothek
hinzugefügt.");
13     }
14
15     public void listBooks() {
16         if (books.isEmpty()) {
17             System.out.println("Die Bibliothek enthält keine Bücher.");
18         } else {
19             System.out.println("Liste der Bücher in der Bibliothek:");
20             for (Book book : books) {
21                 System.out.println(book);
22             }
23         }
24     }
25
26     public void borrowBook(String title) {
27         for (Book book : books) {
28             if (book.getTitle().equalsIgnoreCase(title)) {
29                 book.borrow();
30                 return;
31             }
32         }
33         System.out.println("Das Buch '" + title + "' wurde nicht gefunden.");
34     }
35
36     public void returnBook(String title) {
37         for (Book book : books) {
38             if (book.getTitle().equalsIgnoreCase(title)) {
39                 book.returnBook();
40                 return;
41             }

```

```
42      }  
43      System.out.println("Das Buch '" + title + "' wurde nicht gefunden.");  
44      }  
45  }
```

3. Die Klasse `Main`

```
1  import java.util.Scanner;
2
3  public class Main {
4      public static void main(String[] args) {
5          Library library = new Library();
6          Scanner scanner = new Scanner(System.in);
7          boolean running = true;
8
9          // Beispielbücher hinzufügen
10         library.addBook(new Book("Der Herr der Ringe", "J.R.R. Tolkien"));
11         library.addBook(new Book("Harry Potter und der Stein der Weisen", "J.K.
Rowling"));
12         library.addBook(new Book("1984", "George Orwell"));
13
14         while (running) {
15             System.out.println("\nBibliothekssystem:");
16             System.out.println("1. Bücher anzeigen");
17             System.out.println("2. Buch ausleihen");
18             System.out.println("3. Buch zurückgeben");
19             System.out.println("4. Beenden");
20             System.out.print("Wählen Sie eine Option: ");
21             int choice = scanner.nextInt();
22             scanner.nextLine(); // Eingabezeile leeren
23
24             switch (choice) {
25                 case 1:
26                     library.listBooks();
27                     break;
28                 case 2:
29                     System.out.print("Titel des auszuleihenden Buches: ");
30                     String borrowTitle = scanner.nextLine();
31                     library.borrowBook(borrowTitle);
32                     break;
33                 case 3:
34                     System.out.print("Titel des zurückzugebenden Buches: ");
35                     String returnTitle = scanner.nextLine();
36                     library.returnBook(returnTitle);
37                     break;
38                 case 4:
39                     running = false;
40                     System.out.println("Programm beendet.");
41                     break;
```

```
42         default:
43             System.out.println("Ungültige Option. Bitte erneut versuchen.");
44             break;
45     }
46 }
47 scanner.close();
48 }
49 }
```

2. Praktischer Teil: 45min

Aufgabe E – Römisch zu Ganzzahl

Die römischen Ziffern bestehen aus sieben verschiedenen Symbolen: I, V, X, L, C, D und M.

Symbol	Wert
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

Zum Beispiel wird die Zahl 2 als II geschrieben, da es zwei Einsen sind. Die Zahl 12 wird als XII dargestellt, was X (10) plus II (2) entspricht. Die Zahl 27 wird als XXVII geschrieben, was XX (20) plus V (5) plus II (2) ist.

Römische Ziffern werden normalerweise von links nach rechts in der Reihenfolge von der größten zur kleinsten Zahl geschrieben. Allerdings wird die Zahl 4 nicht als IIII geschrieben, sondern als IV. Da die Eins vor der Fünf steht, subtrahieren wir sie, und so erhalten wir vier. Das gleiche Prinzip gilt für die Zahl 9, die als IX geschrieben wird. Es gibt sechs Fälle, in denen Subtraktion angewendet wird:

- I kann vor V (5) und X (10) stehen, um 4 und 9 zu bilden.
- X kann vor L (50) und C (100) stehen, um 40 und 90 zu bilden.
- C kann vor D (500) und M (1000) stehen, um 400 und 900 zu bilden.

Schreiben Sie ein Programm, welches Ihnen eine römische Zahl in eine ganze (numerische/integer) Zahl umrechnet!

Tip

Sie werden durch den String iterieren müssen. Nutzen Sie dafür die Methode `String.getChar(int i)`, wobei hier `i` der Index ist, an dem Sie den Wert auslesen möchten. Beispiel:

```
1 String s = "Hallo";
2 System.out.println("Buchstabe an der ersten Stelle von " + s + " ist " +
   s.charAt(0) + "."); //Buchstabe an der ersten Stelle von Hallo ist H.
```

1. Legen Sie ein neues Projekt an. Geben Sie dabei dem Projekt einen Namen, der Ihre Matrikelnummer und Ihren Namen enthält. Erstellen Sie eine Klasse mit einer Methode, die einen entsprechenden Namen und Parameter sowie Rückgabewert enthält. 3 P.
2. Überprüfen Sie den String auf eine ungültige Eingabe von Zeichen. Legen Sie eine Variable an, die den Wert der römischen Zahl im numerischen Format enthält. 5 P.
3. Schreiben Sie eine for-Schleife, die durch den gegebenen String iteriert. 5 P.
4. Bauen Sie eine Logik ein, die den String auswertet und entscheidet, ob einer der sechs Sonderfälle eingetreten ist. 10 P.
5. Lassen Sie sich von der Methode einen Wert zurückgeben, der dem Wert der römischen Zahl im numerischen Format angibt. 5 P.
6. Schreiben Sie eine main-Methode, welche die Methode gegen Testinput laufen lässt. 5 P.
7. Achten Sie bei der Programmierung Ihrer Lösung auf die gängigen Coding Styles, die in der Vorlesung festgelegt worden sind. 4 P.
8. Legen Sie Ihr Projekt auf der externen Festplatte ab, die Sie von der Klausuraufsicht zur Verfügung gestellt bekommen. Fragen Sie einfach, falls Sie Ihr Projekt ablegen möchten. Nachdem das Projekt abgelegt worden ist, gibt es keine weiteren Möglichkeiten, Änderungen zu machen. 3 P.

Insgesamt sind 91 + 0 P. erreichbar. Sie haben _____ P. von 91 P. erreicht.

Punkte	91–81	80–72	71–63	62–54	53–0
Wert	sehr gut	gut	befriedigend	ausreichend	n.b.

Lösungsvorschläge – Klausur

Aufgabe	Erreichte Punkte
Aufgabe A – Grundbegriffe	____ / 10
Attribut als Variable oder Eigenschaft einer Klasse/Objekt.	____ / 2
Methode als Funktion oder Fähigkeit einer Klasse/Objekt.	____ / 2
Klasse als Bauplan für ein oder mehr Objekte	____ / 2
Objekt als Instanz einer Klasse	____ / 2
Vererbung als Möglichkeit, Code zu organisieren. Weitergabe von Methoden und Attributen.	____ / 2
Aufgabe B – Konzepte der Objektorientierten Programmierung	____ / 15
Eine Methode in unterschiedlichen Klassen mit unterschiedlichen Implementierungen existieren kann, während sie den gleichen Methodennamen und die gleiche Signatur trägt. Überladen und Überschreiben.	____ / 5
Eine abstrakte Klasse ist eine Klasse, die nicht instanziiert werden kann. Ein Interface ist eine Sammlung aus abstrakten Methoden.	____ / 5
Klarheit und Struktur, Sicherheit, Wartbarkeit. Kapselung mittels private, sowie Getter und Setter.	____ / 5
Aufgabe C – Wahr oder Falsch	____ / 16
Falsch	____ / 2
Falsch	____ / 2
Falsch	____ / 2
Wahr	____ / 2
Wahr	____ / 2
Falsch	____ / 2
Wahr	____ / 2
Aufgabe D – Beschreibung eines bestehenden Programms	____ / 10

Override von toString	____ / 2
Datenkapselung mittels private und Getter und Setter	____ / 2
ArrayList statt normalem Array.	____ / 2
Scanner wird als Eingabe über die Kommandozeile verwendet.	____ / 2
Erweiterungen: Abstrakte-Klassen, Exceptions, Benutzerverwaltung	____ / 2
Aufgabe E – Römisch zu Ganzzahl	____ / 40
Der Code hat den richtigen Coding Style und sieht ordentlich aus.	____ / 4
Der Code funktioniert wie beschrieben und gibt bei richtigem Input eine richtige Antwort zurück.	____ / 8
Das Programm ist gegen Fehler durch falsche Eingaben gesichert. Ein nicht definiertes Zeichen führt zu einem Abbruch der Operation.	____ / 4
Das Programm ist gegen Fehler durch falsche Eingaben gesichert. Es wurde ein entsprechendes Exception Handling implementiert.	____ / 4
Es ist ein Projekt mit entsprechendem Namen, sowie eine Klasse und eine Methode angelegt.	____ / 5
Die Methode ist entsprechend benannt, hat Parameter und Rückgabewert, die mit der Aufgabenstellung zusammenpassen.	____ / 5
Es gibt eine main-Methode in einer der Klassen, die ausführbar ist und den Code gegen Testinput testet.	____ / 5
Das Projekt ist auf der Festplatte, ist richtig benannt und kann geöffnet werden.	____ / 5
	____ / 91 + 0 P.