

# Resitklausur – Objektorientierte Programmierung

## 1. Theoretischer Teil: 45min

### Aufgabe A – Grundbegriffe

12 P.

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

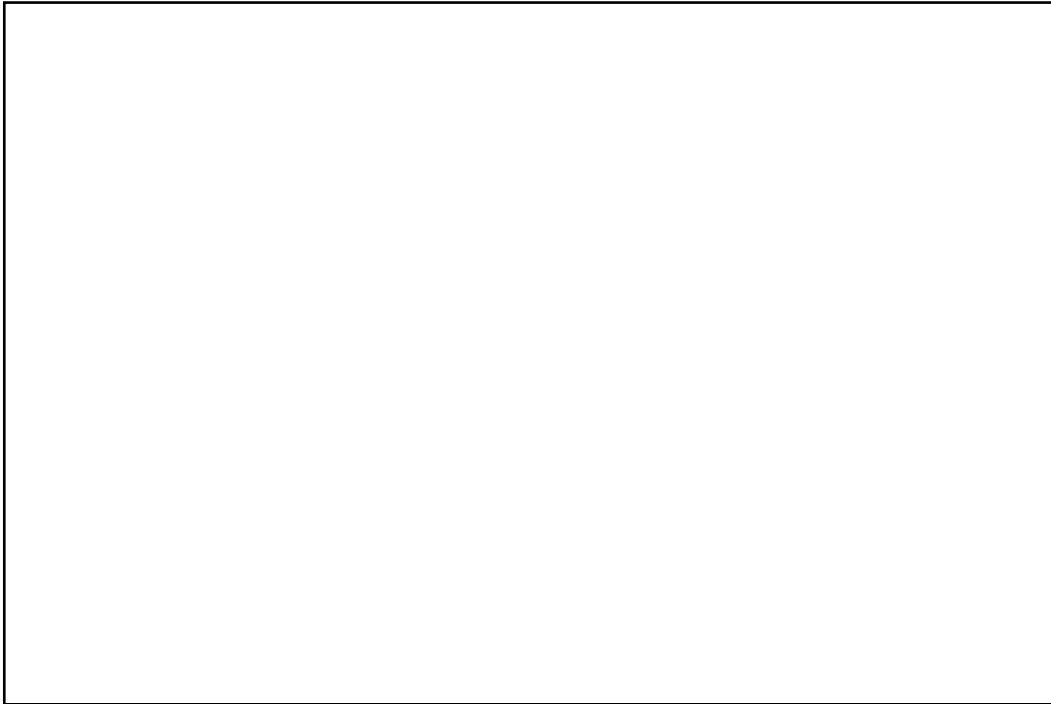
- *Klassenmethode*
- *Objekt*
- *Klasse*
- *Methode*
- *Attribut*
- *Vererbung*

### Aufgabe B – Konzepte der Objektorientierten Programmierung

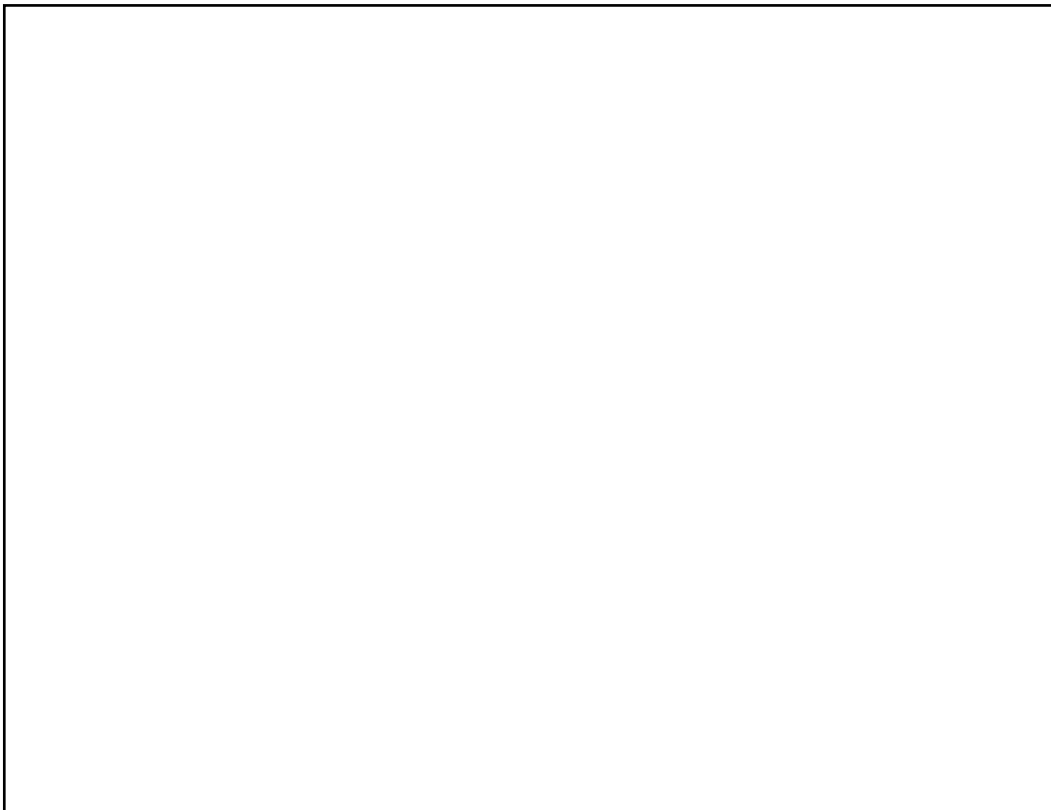
*Beschreiben Sie kurz, was die verschiedenen Begriffe und Konzepte in Java bzw. der objektorientierten Programmierung bedeuten.*

1. Definieren Sie den Begriff „Überladen“ (Overloading). Geben Sie ein Beispiel an, das zeigt, wie eine Überladung (Overloading) in Java implementiert wird. Was ist dafür wichtig? 5 P.

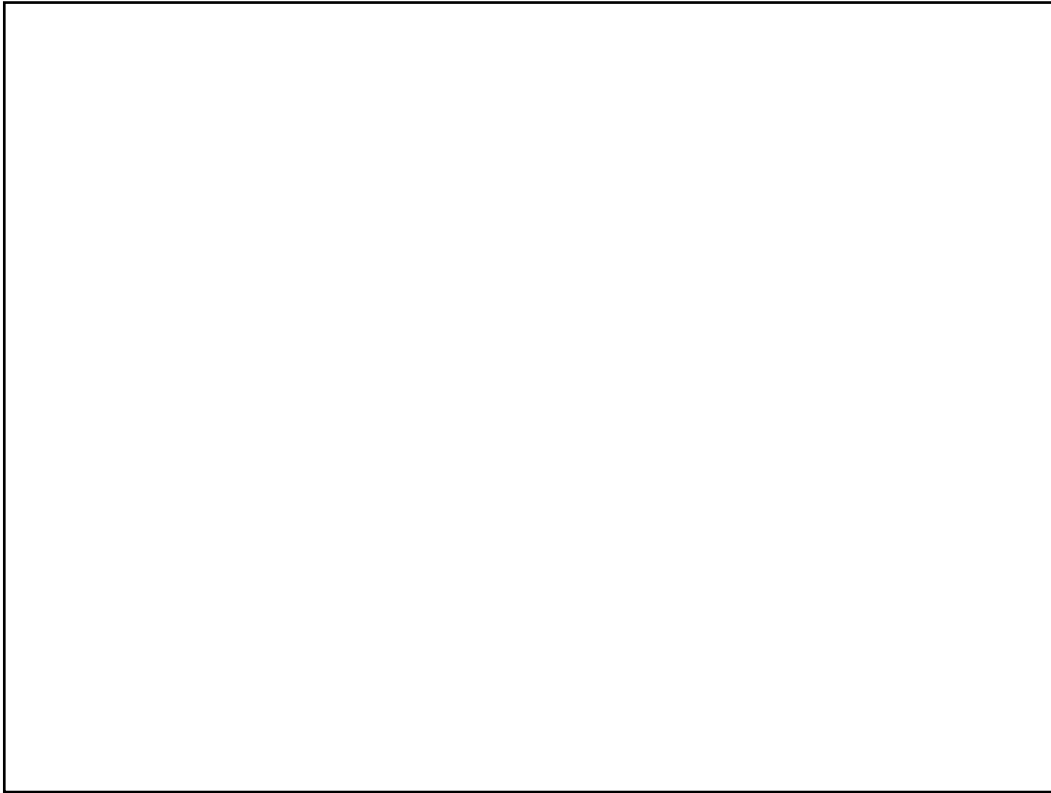
2. Erklären Sie die Unterschiede zwischen einer abstrakten Klasse und einem Interface in Java. Nennen Sie jeweils ein Beispiel für den Einsatz. 5 P.



3. Was ist ein Thread? Wofür wird dieser verwendet? Geben Sie ein Beispiel für dessen Verwendung an. 5 P.



4. 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.		
In Java gibt es keine Klassen, sondern nur <i>structs</i> .		
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 nur maximal ein Interface implementieren.		
Mit dem Schlüsselwort <code>final</code> können Sie definieren, dass beispielsweise eine Variable ihren Wert nicht mehr ändern kann.		
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 System erzeugt, in dem Sie Rezepte verarbeiten können. 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 Recipe {
2      private String name;
3      private String category;
4      private String recipeId;
5      private int prepTimeMinutes;
6      private boolean isFavorite;
7
8      public Recipe(String name, String category, String recipeId, int
prepTimeMinutes) {
9          this.name = name;
10         this.category = category;
11         this.recipeId = recipeId;
12         this.prepTimeMinutes = prepTimeMinutes;
13         this.isFavorite = false;
14     }
15
16     public String getName() {
17         return name;
18     }
19
20     public String getCategory() {
21         return category;
22     }
23
24     public String getRecipeId() {
25         return recipeId;
26     }
27
28     public int getPrepTimeMinutes() {
29         return prepTimeMinutes;
30     }
31
32     public boolean isFavorite() {
33         return isFavorite;
34     }
35
36     public void addToFavorites() {
37         if (!isFavorite) {
38             isFavorite = true;
39             System.out.println(name + " has been added to favorites.");
40         } else {
41             System.out.println(name + " is already in favorites.");
```

```
42     }
43 }
44
45 public void removeFromFavorites() {
46     if (isFavorite) {
47         isFavorite = false;
48         System.out.println(name + " has been removed from favorites.");
49     } else {
50         System.out.println(name + " is not in favorites.");
51     }
52 }
53
54 @Override
55 public String toString() {
56     return "Recipe: " + name +
57         "\nCategory: " + category +
58         "\nID: " + recipeId +
59         "\nPrep Time: " + prepTimeMinutes + " minutes" +
60         "\nStatus: " + (isFavorite ? "Favorite" : "Regular");
61 }
62 }
```



## 2. Die Klasse *RecipeManager*

```
1  import java.util.ArrayList;
2  import java.util.List;
3
4  public class RecipeManager {
5      private List<Recipe> recipeCollection;
6
7      public RecipeManager() {
8          recipeCollection = new ArrayList<>();
9      }
10
11     public void addRecipe(Recipe recipe) {
12         recipeCollection.add(recipe);
13         System.out.println(recipe.getName() + " has been added to your cookbook.
14     }
15
16     public void removeRecipe(String recipeId) {
17         Recipe recipeToRemove = findRecipeById(recipeId);
18         if (recipeToRemove != null) {
19             recipeCollection.remove(recipeToRemove);
20             System.out.println(recipeToRemove.getName() + " has been removed from
21             your cookbook.");
22         } else {
23             System.out.println("Recipe with ID " + recipeId + " not found.");
24         }
25
26     public Recipe findRecipeById(String recipeId) {
27         for (Recipe recipe : recipeCollection) {
28             if (recipe.getRecipeId().equals(recipeId)) {
29                 return recipe;
30             }
31         }
32         return null;
33     }
34
35     public List<Recipe> findRecipesByCategory(String category) {
36         List<Recipe> result = new ArrayList<>();
37         for (Recipe recipe : recipeCollection) {
38             if (recipe.getCategory().equalsIgnoreCase(category)) {
39                 result.add(recipe);
40             }
41         }
42     }
```

```
42         return result;
43     }
44
45     public void displayAllRecipes() {
46         if (recipeCollection.isEmpty()) {
47             System.out.println("Your cookbook is empty.");
48             return;
49         }
50
51         System.out.println("Cookbook Collection:");
52         for (Recipe recipe : recipeCollection) {
53             System.out.println("-----");
54             System.out.println(recipe);
55         }
56         System.out.println("-----");
57     }
58
59     public static void main(String[] args) {
60         // Create recipe manager
61         RecipeManager cookbook = new RecipeManager();
62
63         // Add some recipes
64         cookbook.addRecipe(new Recipe("Spaghetti Carbonara", "Italian", "R001",
65 30));
66         cookbook.addRecipe(new Recipe("Chicken Curry", "Indian", "R002", 45));
67         cookbook.addRecipe(new Recipe("Margherita Pizza", "Italian", "R003", 60));
68         cookbook.addRecipe(new Recipe("Butter Chicken", "Indian", "R004", 50));
69
70         // Display all recipes
71         cookbook.displayAllRecipes();
72
73         // Find recipes by category
74         System.out.println("\nItalian Recipes:");
75         List<Recipe> italianRecipes = cookbook.findRecipesByCategory("Italian");
76         for (Recipe recipe : italianRecipes) {
77             System.out.println(recipe.getName() + " (" + recipe.getPrepTimeMinutes()
78 + " minutes)");
79         }
80
81         // Add a recipe to favorites
82         System.out.println();
83         Recipe pizzaRecipe = cookbook.findRecipeById("R003");
84         if (pizzaRecipe != null) {
```

```
83         pizzaRecipe.addToFavorites();
84     }
85
86     // Try to add to favorites again
87     if (pizzaRecipe != null) {
88         pizzaRecipe.addToFavorites();
89     }
90
91     // Remove from favorites
92     System.out.println();
93     if (pizzaRecipe != null) {
94         pizzaRecipe.removeFromFavorites();
95     }
96
97     // Remove a recipe
98     System.out.println();
99     cookbook.removeRecipe("R002");
100
101     // Display updated collection
102     System.out.println();
103     cookbook.displayAllRecipes();
104 }
105 }
```

## 2. Praktischer Teil: 45min

### Aufgabe E – Array-Verdopplung

Gegeben ist ein Integer-Array *nums* mit der Länge *n*. Ihre Aufgabe ist es, ein neues Array *ans* mit der Länge  $2n$  zu erstellen, das folgende Eigenschaften erfüllt:

Für jeden Index *i*, wobei  $0 \leq i < n$  gilt (nullbasierte Indizierung):

- $ans[i] == nums[i]$  (Die ersten *n* Elemente von *ans* entsprechen exakt dem ursprünglichen Array *nums*)
- $ans[i + n] == nums[i]$  (Die letzten *n* Elemente von *ans* sind eine Wiederholung des Arrays *nums*)

Anders ausgedrückt: Das Array *ans* ist die Konkatination (Aneinanderreihung) von zwei identischen *nums* Arrays. Beispiel zur Verdeutlichung: Wenn  $nums = [1, 2, 3]$  (hier ist  $n = 3$ ), dann soll  $ans = [1, 2, 3, 1, 2, 3]$  sein. Die ersten drei Elemente ( $ans[0]$ ,  $ans[1]$ ,  $ans[2]$ ) entsprechen den Elementen von *nums*. Die letzten drei Elemente ( $ans[3]$ ,  $ans[4]$ ,  $ans[5]$ ) sind ebenfalls die Elemente von *nums*.

Implementiere eine Funktion, die das Array *nums* als Eingabe nimmt und das oben beschriebene Array *ans* zurückgibt. Hinweis: Denke daran, dass wir eine nullbasierte Indizierung verwenden, das heißt, der erste Index eines Arrays ist 0, nicht 1.

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. Legen Sie eine Variable an, die den Wert des neuen Arrays enthält. 5 P.
3. Schreiben Sie eine `for`-Schleife, die durch das gegebene Array iteriert. 5 P.
4. Bauen Sie eine Logik ein, die das Array zweimal in das neue Array verlegt. 10 P.
5. Lassen Sie sich von der Methode einen Wert zurückgeben, der dem geforderten Array entspricht. 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.

Insgesamt sind 95 + 0 P. erreichbar. Sie haben \_\_\_\_ P. von 95 P. erreicht.

Punkte	95–85	84–76	75–66	65–57	56–0
Wert	sehr gut	gut	befriedigend	ausreichend	n.b.

## Lösungsvorschläge – Resitklausur

Aufgabe	Erreichte Punkte
<b>Aufgabe A – Grundbegriffe</b>	<b>_____ / 12</b>
Methode, die zu einer Klasse gehört und das Keyword <code>static</code> hat.	_____ / 1
Objekt als Instanz einer Klasse	_____ / 2
Klasse als Bauplan für ein oder mehr Objekte	_____ / 2
Methode als Funktion oder Fähigkeit einer Klasse/Objekt.	_____ / 2
Attribut als Variable oder Eigenschaft einer Klasse/Objekt.	_____ / 2
Vererbung als Möglichkeit, Code zu organisieren. Weitergabe von Methoden und Attributen.	_____ / 2
<b>Aufgabe B – Konzepte der Objektorientierten Programmierung</b>	<b>_____ / 20</b>
Eine Methode, die in der selben Klasse mit unterschiedlichen Implementierungen existieren kann, während sie den gleichen Methodennamen, aber eine andere Signatur trägt.	_____ / 5
Eine abstrakte Klasse ist eine Klasse, die nicht instanziiert werden kann. Ein Interface ist eine Sammlung aus abstrakten Methoden.	_____ / 5
Ein weiterer Strang im Programm, der gleichzeitig zu einem anderen Strang Code ausführen kann.	_____ / 5
Klarheit und Struktur, Sicherheit, Wartbarkeit. Kapselung mittels <code>private</code> , sowie Getter und Setter.	_____ / 5
<b>Aufgabe C – Wahr oder Falsch</b>	<b>_____ / 16</b>
Falsch	_____ / 2
Falsch	_____ / 2
Falsch	_____ / 2
Wahr	_____ / 2

Falsch	_____ / 2
Falsch	_____ / 2
Wahr	_____ / 2
<b>Aufgabe D – Beschreibung eines bestehenden Programms</b>	<b>_____ / 10</b>
Override von <code>toString</code>	_____ / 2
Datenkapselung mittels <code>private</code> und Getter und Setter	_____ / 2
<code>ArrayList</code> statt normalem Array.	_____ / 2
Scanner wird als Eingabe über die Kommandozeile verwendet.	_____ / 2
Erweiterungen: Abstrakte-Klassen, Exceptions	_____ / 2
<b>Aufgabe E – Array-Verdopplung</b>	<b>_____ / 37</b>
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.	_____ / 10
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 <code>main</code> -Methode in einer der Klassen, die ausführbar ist und den Code gegen Testinput testet.	_____ / 5
	<u>_____ / 95 + 0 P.</u>