

Übungsblatt 7: Klassen & Vererbung

Deadline: 27.11.2025 12:00 Uhr

Besprechung: 01.12.–05.12.2025

Hinweise für die Bearbeitung der Aufgaben

- Bitte lesen Sie die Aufgabenstellungen genau durch.
- Testen Sie Ihre Lösungen vor der Abgabe selbst!
- Halten Sie sich an die Java Coding Konventionen (siehe vorhergehende Übungsblätter).
- Damit die automatischen Tests korrekte Ergebnisse liefern können, achten Sie bitte auf die folgenden Punkte:
 - Halten Sie sich bitte genau an die Vorgaben/Vorlagen aus der Aufgabenstellung.
 - Wenn Sie Umlaute verwenden, stellen Sie die Kodierung Ihrer Dateien unbedingt auf UTF-8.
 - Achten Sie darauf, dass Ihr Projekt mit Java Version 21 kompiliert und ausgeführt werden kann.
 - Ihre Java-Klassen müssen sich im default package befinden (es darf kein package angegeben werden).



Aufgabe 1 Abgabe über GATE mit Peer-Review **Modellierung Polymorphismus**

Modellieren Sie folgende Situation:

Es gibt verschiedene Personen mit genau einer Rolle an einer Universität. Jede Person hat einen Namen und kann sich vorstellen. Jede Vorstellung beginnt mit: „Hallo, mein Name ist NAME.“

Es gibt die Rollen Student, Professor, Wissenschaftliche Mitarbeitende (WiMi) und Nicht-wissenschaftliches Personal. Studierende haben zusätzlich eine Matrikelnummer und einen Studiengang. Wissenschaftliches Personal (Professoren, WiMi) und Nicht-wissenschaftliches Personal haben eine Personalnummer. Professoren haben zusätzlich ein Forschungsgebiet. Sie können das Forschungsgebiet wechseln. WiMi sind genau einem Professor zugeordnet mit dessen Forschungsgebiet. Nicht-wissenschaftliches Personal arbeitet in einem bestimmten Bereich (z. B. Bibliothek, IT, Sekretariat).

Bei der Vorstellung sagen sie zusätzlich zum Namen Folgendes:

Wenn sie Studierende sind: „Ich bin Student im STUDIENGANG und meine Matrikelnummer ist MATRIKELNR.“

Wenn sie Professor sind: „Ich bin Professor und mein Forschungsgebiet ist FORSCHUNGSGEBIET.“

Wenn sie WiMi sind: „Ich bin WiMi bei Professor PROFESSOR im Forschungsgebiet FORSCHUNGSGEBIET.“

Wenn sie nicht-wissenschaftliches Personal sind: „Ich bin nicht-wissenschaftliches Personal und arbeite im Bereich BEREICH.“

Für Uni-Personal soll am Ende zusätzlich ausgegeben werden: „Ich arbeite an der Uni und meine Personalnummer lautet: PERSONALNUMMER.“

- a) Modellieren Sie die Situation in einem UML-Diagramm.
- b) Implementieren Sie die Modellierung aus a) in Java.
- c) Erstellen Sie ein Programm `Modelling`. Modellieren Sie folgende Vorstellung von Personen in dessen `main`-Methode:
 - Professorin Maria Weber arbeitet im Forschungsgebiet Theoretische Informatik mit der Personalnummer 142.
 - WiMi Theodor Grün arbeitet für Professorin Weber und hat die Personalnummer 816.
 - Studierende Anna Schmidt ist im Studiengang „Medieninformatik“ eingeschrieben mit der der Matrikelnummer 10815.

- Mitarbeiterin Sabine Koch arbeitet im Bereich „IT-Support“ und hat die Personalnummer 112.

Alle Personen sollen sich dieser Reihenfolge nach vorstellen.

Aufgabe 2 Abgabe mit Tests in GATE

Abstrakte Klassen und Polymorphismus

In dieser Aufgabe soll ein einfaches Programm zur Ausgabe von geometrischen Figuren entwickelt werden. Der Anwender soll zunächst aufgefordert werden, eine Auswahl an geometrischen Figuren zu treffen, die er später ausgeben möchte (mithilfe der `scanner.nextLine()`-Methode).

Der Anwender kann dann eine der vier Zeichenketten („Dreieck“, „Quadrat“, „Linie“ und „Ende“) eingeben – andere Zeichenketten sollen ignoriert werden. Der Anwender soll so maximal 10 Figuren eingeben können. Möchte er weniger eingeben, so kann er die Eingabe mittels Eingabe von „Ende“ beenden.

Im Anschluss an diese Eingabephase sollen dann die ausgewählten geometrischen Figuren in der Anzahl, in der sie eingegeben wurden, untereinander ausgegeben werden. Die einzelnen Figuren sollen dabei wie unten abgebildet aussehen (ohne Leerzeichen am Ende):

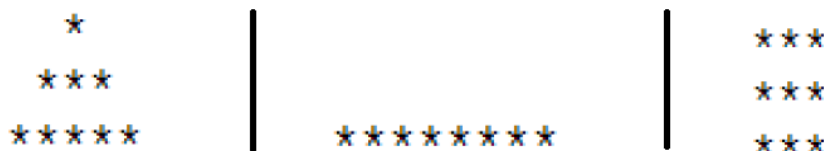


Abbildung 1: mögliche Ausgaben

Wichtige Hinweise zur Lösung dieser Aufgabe:

- Verwenden Sie zur Lösung der Aufgabe die folgenden Klassen: `Triangle`, `Square`, `Shape` und `Line`.

Überlegen Sie sich, welche dieser Klassen als abstrakt definiert werden sollte (mind. eine dieser Klassen sollen Sie als abstrakt deklarieren) und wie die Klassen voneinander erben sollten. In jeder dieser Klassen soll genau eine Methode definiert werden mit der Signatur **public void printShape()**. Diese soll in der abstrakten Klasse als abstrakt definiert werden und in den anderen Klassen implementiert werden und die oben gezeigten passenden Ausgaben auf der Konsole erzeugen. Ferner sollen die Klassen lediglich den Standardkonstruktor besitzen.

- Die Klasse, in der die `main`-Methode steht, soll `ShapeApplication` heißen und u.a. zuständig sein für die Anzeige des Eingabemenüs und für das Füllen des Arrays `shapes`. Ein Template der Klasse `ShapeApplication` kann über GATE heruntergeladen werden und soll bis auf den Schleifenrumpf in dieser Aufgabe auch nicht verändert werden.

- Auf GATE finden Sie die Formen in einer Textdatei. Achten Sie darauf, dass sich an den Zeilenenden keine Leerzeichen befinden und am Ende keine zwei Leerzeilen.

ShapeApplication.java

```
import java.util.Scanner;

public class ShapeApplication {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Shape[] shapes = new Shape[10];

        System.out.println("Bitte geben Sie eine geometrische Form
            ein. Zur Auswahl stehen:\nDreieck\nQuadrat\nLinie\n\nZum
            Beenden des Programmes geben Sie \"Ende\" ein.");
        for(int i=0;i<shapes.length;i++) {
            // Hier bitte Loesung eintragen.
            // Benutzereingabe bitte über scanner.nextLine() abfragen
            .
        }

        for(int i=0;i<shapes.length;i++) {
            if (shapes[i] != null) {
                shapes[i].printShape();
                System.out.println();
            }
        }
    }
}
```

Aufgabe 3 Abgabe in GATE
Quiz

Welche der folgenden Aussagen sind für Java korrekt?

- a) Eine Klasse kann mehrere Interfaces implementieren.
- b) Erbende Klassen können alle Methoden der übergeordneten Klasse aufrufen.
- c) Eine Klasse darf mehrere Konstruktoren haben.
- d) Eine Klasse, die abstrakte Methoden enthält, muss als abstrakt deklariert sein.
- e) Aus einer statischen Methode, kann auf die Instanzvariablen mit dem this-Schlüsselwort zugegriffen werden.

Aufgabe 4 Abgabe mit Tests in GATE

Fun with Strings

Implementieren Sie eine Klasse `FunWithStrings` mit den folgenden statischen Methoden:

- **int** `countWords(String text)`, die die Wörter (Zeichenfolgen abgetrennt durch Leerzeichen) in einem Text zählt.
- **String** `getQuotedText(String text)`, die den in einer übergebenen Zeichenkette enthaltenen Text, der in Anführungszeichen eingeschlossen ist, zurück gibt. Sie können davon ausgehen, dass maximal zwei doppelte Anführungszeichen in der Zeichenkette vorhanden sind. Beispiele:
 - `getQuotedText("keine Anführungszeichen") -> null`
 - `getQuotedText("nur ein \"Anführungszeichen\") -> null`
 - `getQuotedText("vorher \"Text\" nachher") -> "Text"`
 - `getQuotedText("vorher \"\" nachher") -> ""`
- **String** `commonPrefix(String text1, String text2)`, die den längsten gemeinsamen Prefix von zwei Strings berechnet.

Beispiele:

- `commonPrefix("First", "Second") -> ""`
- `commonPrefix("xyzTest", "xyzHello") -> "xyz"`
- `commonPrefix("Hello", "Hello") -> "Hello"`

Tipp: Sie können die [String-Methoden der Java Standard Library](#) benutzen.

Abgabe

Die Abgabe erfolgt bis zum 27.11.2025 12:00 Uhr auf GATE, <https://gate.ifi.lmu.de>.

Zuletzt aktualisiert: 17. November 2025