Babeș-Bolyai Universität
Cluj Napoca
Fakultät für Mathematik und Informatik
Dr. Cătălin Rusu

II - Klausur *Fortgeschrittene Programmierungsmethoden*

26 Jan 2024

Kandidat/in:

Name: _____

Allgemeine Hinweise

- *Neben Papier und Schreibutensilien sind keine weiteren Hilfsmittel erlaubt.*
- Verwenden Sie keine roten Stifte und keine Bleistifte.
- Sie dürfen alle Aufgaben in beliebiger Reihenfolge lösen, aber konzentrieren Sie sich jeweils auf eine Aufgabe, aber teilen Sie sich Ihre Zeit ein.
- Alle Mobiltelefone müssen vollständig ausgeschaltet sein.
- *Vergessen Sie nicht, Ihren Namen auf jedes Blatt zu schreiben.*
- Blätter ohne diese Angaben werden nicht bewertet.
- Bitte schreiben Sie in Ihrem eigenen Interesse deutlich. Unverständliche oder nicht begründete Antworten werden nicht bewertet.
- In den letzten 15 Minuten der Prüfung kann der Raum nicht mehr verlassen werden.
- *Im Fall von Täuschungsversuchen wird die Klausur sofort mit 0 Punkten bewertet. Eine Vorwarnung erfolgt nicht.*

Aufgabe	max. Punkte	erreicht
1-A	15 Punkte	
1-B	10 Punkte	
2-A	10 Punkte	
2-B	20 Punkte	
3-A	10 Punkte	
3-B	25 Punkte	
Summe	90 + 10 e.o. = 100 Punkte	

Aufgabe 1

A. Beschreiben Sie die zentrale Idee des **Dependency-Injection-Prinzip**. Klären Sie mit einem Code-Beispiel aus der realen Welt ab. **(15 Punkte)**

B. Was ist eine **Anonyme Klasse**? Geben Sie ein Code-Beispiel an. Dürfen Anonyme Klasse statische Elemente besitzen? **(10 Punkte)**

Aufgabe 2

A. Erläutern Sie anhand von konkreten Code-Beispielen den Unterschied zwischen **Komposition** und **Aggregation** in Java. **(10 Punkte)**

B. Eine Student-Klasse besitzt die folgenden Attributen:

- private String name;
- private String kurs;
- private int id;
- private int note;

1. Erstellen Sie die Student-Klasse und eine Liste von Studenten. **(5 Punkte)**

2. Implementieren Sie eine Java 8 Stream Funktionalität, die
   - die Studenten entfernt, deren Name mit ‘a’ startet;
   - die Note aller Studenten mit 1 erhöht;
   - die Anzahl der Studenten für jeden Kurs ausgibt. **(15 Punkte)**

   For/While-Schleife und Rekursion sind nicht erlaubt!

Aufgabe 3

A. Was gibt der folgende Code aus? **(10 Punkte)**

Code-Fragment

```
class Ex1{
  public static void main(String args[]){
    int x = 10;
    int y = new Ex1().change(x);
    System.out.print(x+y);
  }

  int change(int x){
    x = 12;
    return x;
  }
}
```

B. Ein Online-Shop möchte eine Liste der zum Verkauf verfügbaren Fahrzeuge anzeigen. Alle Fahrzeuge sind in einer Liste vom Typ "ArrayList<Fahrzeug>" gespeichert und haben als Attribute eine *Marke*, ein *Modell* und einen *Kilometerstand*. *Marke* und *Modell* dürfen nicht geändert werden. Der *Kilometerstand* kann nur um 1 inkrementiert werden.

Es gibt nur zwei aktuelle Fahrzeugtypen: **Verbrennungsfahrzeug** und **Elektrofahrzeug**. Ein **Verbrennungsfahrzeug** hat ein veränderbares "HorsePower"-Attribut und ein **Elektrofahrzeug** hat ein veränderbares Attribut "Kilowatt". Eine Methode "display()" soll bereitgestellt sein, die alle Fahrzeuge der Liste anzeigt. Jedes Fahrzeug "weiß", wie es auf dem Bildschirm angezeigt werden soll. **Verbrennungsfahrzeug** wird anzeigen: "Thermal: <Marke> <Modell> <Km> <HorsePower>", wobei z. B. "<Marke>" ein Platzhalter für die Marke des Fahrzeugs ist. **Elektrofahrzeug** wird anzeigen: "EV: <Marke> <Modell> <Kilowatt>".

1. Implementieren Sie alle benötigten Klassen. **(10 Punkte)**

2. Einige dieser Fahrzeuge sind beschädigt und nicht fahrbereit. Verwenden Sie das Proxy-Muster und stellen Sie sicher, dass jedes Mal, wenn das Fahrzeug angezeigt wird, das normale Textwort mit der Zeichenfolge "!WARNING DAMAGED!" startet. **(10 Punkte)**

Beispiel: "!WARNUNG BESCHÄDIGT! EV: Tesla Model 3 150kW".

3. Stellen Sie die Endlösung in Form eines UML-Diagramms grafisch dar. **(5 Punkte)**