University of Zurich[UZH]

s.e.a.l. software evolution & architecture lab

Programmierung für Mathematik HS11

Übung 5

1 Aufgabe: Eclipse IDE

1.1 Lernziele

1. Die Entwicklungsumgebung Eclipse kennen lernen.

1.2 Aufgabenstellung

1. Installieren Sie die IDE[1] Eclipse (wählen Sie Eclipse Classic).
2. Arbeiten Sie das zur Verfügung gestellte Eclipse-Tutorial durch.
3. Legen Sie für die nachfolgenden Aufgaben ein neues Eclipse Projekt an und geben Sie ihm einen sinnvollen Namen. Nutzen Sie Java Packages, um die einzelnen Teilaufgaben zu organisieren.

[1] *IDE* steht für *Integrated Development Environment*

2 Aufgabe: Rechnen mit Matrizen

2.1 Lernziele

1. Mit zweidimensionalen Arrays rechnen können.

2.2 Aufgabenstellung

Jede Matrix A ist durch folgendes Schema gegeben:

$$A = \begin{pmatrix} \alpha_{11} & \alpha_{12} & \cdots & \alpha_{1n} \\ \alpha_{21} & \alpha_{22} & \cdots & \alpha_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{m1} & \alpha_{m2} & \cdots & \alpha_{mn} \end{pmatrix}$$

(Im Fall, dass A m Zeilen und n Spalten hat, sagt man auch, dass A eine $m \times n$ Matrix sei). Dabei ist $\alpha_{ij} \in \mathbb{R} \ \forall i \in \{1, ..., m\}, j \in \{1, ..., n\}$.

In Java können wir eine Matrix auf sehr einfache Weise durch ein zweidimensionales Array darstellen.

a) Darstellung einer Matrix

Schreiben Sie eine Klassen `Matrix` mit einer Methode `printMatrix`, welche als Übergabeparameter eine Matrix in Form eines zweidimensionalen Arrays erwartet und diese auf der Kommandozeile ausgibt.

b) Addition von Matrizen

Schreiben Sie eine Methode `add` welche die Summe von zwei Matrizen, welche als Parameter übergeben werden, berechnen und als neues Array zurückgeben soll. Falls diese nicht definiert ist, so soll eine Fehlermeldung ausgegeben werden.

Bemerkung: die Summe von zwei $m \times n$ Matrizen A, B ist komponentenweise definiert, das heisst:

$$A + B = \begin{pmatrix} \alpha_{11} & \cdots & \alpha_{1n} \\ \vdots & \ddots & \vdots \\ \alpha_{m1} & \cdots & \alpha_{mn} \end{pmatrix} + \begin{pmatrix} \beta_{11} & \cdots & \beta_{1n} \\ \vdots & \ddots & \vdots \\ \beta_{m1} & \cdots & \beta_{mn} \end{pmatrix} = \begin{pmatrix} \alpha_{11} + \beta_{11} & \cdots & \alpha_{1n} + \beta_{1n} \\ \vdots & \ddots & \vdots \\ \alpha_{m1} + \beta_{m1} & \cdots & \alpha_{mn} + \beta_{mn} \end{pmatrix}$$

(Falls die Dimensionen von A und B nicht übereinstimmen, so ist die Summe nicht definiert).

c) Multiplikation von Matrizen

Schreiben Sie eine Methode `multiplicate`, welche das Produkt von zwei Matrizen berechnen soll, falls es definiert ist und sonst eine entsprechende Fehlermeldung ausgibt.

Bemerkung: Die Matrixmultiplikation $A \cdot B$ ist für eine $n \times k$ Matrix A und eine $k \times m$ Matrix B wie folgt definiert:

$$A \cdot B = \begin{pmatrix} \alpha_{11} & \cdots & \alpha_{1k} \\ \vdots & \ddots & \vdots \\ \alpha_{n1} & \cdots & \alpha_{nk} \end{pmatrix} \cdot \begin{pmatrix} \beta_{11} & \cdots & \beta_{1m} \\ \vdots & \ddots & \vdots \\ \beta_{k1} & \cdots & \beta_{km} \end{pmatrix} = \begin{pmatrix} \gamma_{11} & \cdots & \gamma_{1m} \\ \vdots & \ddots & \vdots \\ \gamma_{n1} & \cdots & \gamma_{nm} \end{pmatrix}$$

wobei

$$\gamma_{ij} = \sum_{l=1}^{k} (\alpha_{il} \cdot \beta_{lj}) \quad \forall i \in \{1, ..., n\}, j \in \{1, ..., m\}$$

(Falls die Anzahl **Spalten** von A nicht mit der Anzahl **Zeilen** von B übereinstimmt, ist $A \cdot B$ nicht definiert).

3 Aufgabe: OOP und Information Hiding

3.1 Lernziele

1. Ein Problem von natürlicher Sprache in Java Code übersetzen können.
2. Eine bestehende Klasse intern abändern und dabei die Schnittstellen beibehalten können.

3.2 Aufgabenstellung

a) Die Klasse `Student`

Ein Student hat folgende Eigenschaften:

- *Name*
- *Alter* (um an der Universität zu studieren muss man mindestens 18 sein)
- *Matrikelnummer*

Schreiben Sie eine Klasse `Student`, welche diese Ausgangslage in Java abbildet. Es soll möglich sein den Namen und das Alter direkt bei der Initialisierung oder auch noch nachträglich zu setzen (Tipp: dazu muss ein Konstruktor geschrieben werden). Allerdings soll der Name nur einmal gesetzt werden können und bevor das Alter gesetzt wird, soll überprüft werden, ob das Alter gültig ist. Wird ein ungültiges Alter übergeben, soll eine Meldung ausgegeben und das Alter auf -1 gesetzt werden. Die Matrikelnummer soll automatisch bei jeder Initialisierung eines `Student` Objektes vergeben werden und zwar so, dass zwei Studenten nie dieselbe Nummer haben. Der erste Student bekommt die Nummer 1, der zweite die Nummer 2 etc. (Tipp: verwenden Sie eine statische Variable).

Implementieren Sie die Methode `toString()`, welche einen sinnvollen `String` mit Name, Alter und Matrikelnummer zurückgeben soll. (Tipp: Lesen Sie in der Java-API nach, wie die Methode `toString` von der `Integer` Klasse verwendet werden kann.)

b) Die Klasse `University`

Schreiben Sie eine Klasse `University`, welche die Studenten der Universität speichern soll. Benutzen Sie dazu als Instanzvariable ein Array vom Typ `Student[]`. Die Sekretärin soll die Möglichkeit haben einen Studenten zu dieser Liste hinzuzufügen und zwar so, dass die Matrikelnummern der Studenten in dem Array automatisch aufsteigend geordnet sind. Ausserdem soll sie auch die Möglichkeit haben das Alter aller Studenten um ein Jahr zu erhöhen (dies tut sie einfachheitshalber nur einmal pro Jahr, für alle Studenten gleichzeitig).

c) TestDriver

Schreiben Sie einen `TestDriver`, in welchem Sie die beiden Klassen testen. (Um die Klassen richtig testen zu können, ist es sinnvoll, die `University` Klasse um die Methode `printStudents` zu ergänzen, welche alle Studenten auf der Kommandozeile ausgibt.)