Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Einführung in die Informatik
Sommersemester 2018

Prof. Dr. Wolfram Burgard
Andreas Kuhner
Daniel Büscher

Übungsblatt 10

Abgabe bis Sonntag, 01.07.2018, 23:59 Uhr

Hinweis:

Aufgaben immer per E-Mail (eine E-Mail pro Blatt und Gruppe) an den zuständigen Tutor schicken (Bei Programmieraufgaben Java Quellcode und evtl. benötigte Datendateien).

Aufgabe 10.1

a) Wieso können Activation Records keine beliebige Tiefe erreichen?

b) Worin liegt der Vorteil bei einer rekursiven Lösung und worin bei einer iterativen Lösung?

c) Wozu wird Vererbung benutzt?

d) Wozu dient das Schlüsselwort `super()`?

Aufgabe 10.2

Zwei Matrizen $A, B \in \mathbb{R}^{n \times n}$ werden wie folgt multipliziert[1]:

$$C = AB$$
$$C_{ij} = \sum_{k=0}^{n-1} A_{ik} B_{kj}$$

Hinweis: Auf der Vorlesungswebseite finden Sie eine vorhandene Klasse `Matrix`. Nutzen Sie diese für die Lösung der folgenden Aufgaben.

1. Implementieren Sie eine Methode `public Matrix mult(Matrix b)` zur Multiplikation zweier Matrizen, die das Ergebnis als Objekt der Klasse `Matrix` zurückgibt.

2. Nutzen Sie das Projekt `MatrixProject`[2] von der Vorlesungswebseite, um Ihre implementierte Methode aus der ersten Teilaufgabe mit JUnit zu testen.

[1] Matrix-Multiplikation: `https://en.wikipedia.org/wiki/Matrix_multiplication#Matrix_product_.28two_matrices.29`

[2] MatrixProject: `http://ais.informatik.uni-freiburg.de/teaching/ss18/info/exercices/`

3. Führen Sie eine Aufwandsabschätzung der implementierten Methode aus Teilaufgabe 1 in Abhängigkeit von der Größe n der Matrix durch.

Aufgabe 10.3

Betrachten Sie die folgende Funktion $f : \mathbb{N}_0 \to \mathbb{N}_0$ ($\mathbb{N}_0$ sind die natürlichen Zahlen inklusive 0):

$$f(n) = \begin{cases} 0 & , n = 0 \\ 1 & , n = 1 \\ f(n-1) + f(n-2) & , n > 1 \end{cases}$$

1. Schreiben Sie eine rekursive Java-Methode, die die Funktion f implementiert.
2. Schreiben Sie eine *nicht*-rekursive Java-Methode, die die Funktion f implementiert.

Aufgabe 10.4

Betrachten Sie den folgenden Algorithmus:

```
static int f(int x, int y) {

  if (y > x) {
    return f(y, x);   // Zeile 4
  }
  int z = x % y;

  if (z == 0) {
    return y;         // Zeile 9
  } else {
    return f(y, z);   // Zeile 11
  }
}
```

Zeichnen Sie die Activation Records für den Aufruf $f(12, 21)$ bis zu dem Zeitpunkt, an dem die maximale Rekursionstiefe erreicht ist.

Aufgabe 10.5

Für einen Fuhrpark bestehend aus PKWs, LKWs, Bussen und Fahrrädern soll eine Klassenhierarchie entworfen werden. Verwenden Sie die folgenden Klassen:

```
Fahrzeug
Kraftfahrzeug
Bus
Fahrrad
PKW
LKW
```

Die unterschiedlichen Fahrzeuge besitzen sowohl gemeinsame als auch unterschiedliche Attribute:

- Jedes Fahrzeug besitze eine Seriennummer.
- Jedes Kraftfahrzeug besitze einen TÜV-Termin.

- Zu jedem Bus gehören die Angaben: Baujahr, amtl. Kennzeichen, Anzahl Sitzplätze, Anzahl Stehplätze sowie die Leistung (ganzzahlig).
- Zu jedem Fahrrad gehören die Angaben: Baujahr und Rahmengröße.
- Zu jedem PKW gehören die Angaben: Baujahr, amtl. Kennzeichen, Anzahl Sitzplätze sowie die Leistung (ganzzahlig).
- Zu jedem LKW gehören die Angaben: Baujahr, amtl. Kennzeichen, Anzahl Sitzplätze, Leistung (ganzzahlig) sowie Zuladung (ganzzahlig).

Darüber hinaus soll die `toString` Methode von jedem Objekt den Typ und zusätzlich die spezifischen Daten des Objektes ausgeben.

1. Implementieren Sie eine Klassenhierarchie. Machen Sie dabei Gebrauch von Vererbung, abstrakten Klassen und Methoden. Vermeiden Sie dabei Wiederholungen.
2. Testen Sie Ihre Implementierung anhand der Klasse `TestHierarchy`, die Sie auf der Homepage zur Übung finden.
3. Visualisieren Sie Ihre Klassenhierarchie als Graphen. Zeichnen Sie ein Rechteck für jede Klasse und einen Pfeil für jede Vererbung (jeweils von der Subklasse zur Superklasse).