Table of Contents

Laboreinheit: Algorithmen und Datenstrukturen – Aufgabenblatt 2

Tom Kamberg-Buhrtz <buhrtz@htw-berlin.de>

Zum Lösen des Aufgabenblattes verwenden Sie bitte die **Standarddatentypen und –definition von JAVA**. `LinkedList<T>`, `ArrayList` usw. **dürfen nicht verwendet werden.** Es darf nur für die Verwaltung der `ICommands` eine `LinkedList<ICommand>` oder `HashMap<Integer, ICommand>` verwendet werden (siehe für weitere Informationen erstes Aufgabenblatt), dies soll Ihnen die Erstellung des Command-line Menüs erleichtern.

Aufgaben

1. Erarbeitung einer einfach verketteten Liste

   (10 Pkt.)

   a. Erarbeiten Sie schriftlich die notwendige Datenstruktur einer einfach verketteten Liste.

   b. Zu speichern in unserer Liste sind Studenten mit Vor- und Nachnamen nebst Matrikelnummer und Studiengang. Entwerfen Sie die notwendigen Datentypen und begründen Sie gegebenenfalls Ihre Wahl schriftlich.

   c. Implementieren Sie die notwendigen Datenstrukturen bzw. Datentypen in JAVA unter Verwendung von **class**, **interface**, **enum** und den entsprechenden gewählten Datentypen.

2. Funktionalität zur Verarbeitung von Elementen einer einfach verketteten Liste

   (20 Pkt.)

   a. Implementieren Sie eine Funktion zum Einlesen der Eigenschaften eines Elements (z.B. Vorname, Nachname, Kurs und Matrikelnummer).

   b. Implementieren Sie eine Funktion zum Hinzufügen eines Elements vor dem ersten Element.

   c. Implementieren Sie eine Funktion zum Hinzufügen eines Elements nach dem letzten

Element.

d. Implementieren Sie eine Funktion zur Ausgabe eines Elements der Liste.

e. Implementieren Sie eine Funktion zur Ausgabe der gesamten Liste.

f. Implementieren Sie eine Funktion zur Ausgabe der Anzahl der Elemente.

g. Implementieren Sie eine Funktion zum Löschen eines Elements.

h. Implementieren Sie eine Funktion zum Löschen der gesamten Liste.

i. Implementieren Sie Funktionalität zum Suchen eines oder mehrerer Studenten nach Vorname, Nachname, Matrikelnummer oder Studiengang.

Es muss die Suche nach allen 4 Eigenschaften unterstützt werden!

j. Implementieren Sie Funktionalität zum Sortieren der Datensätze der Studenten, Matrikelnummer und Studiengang nach zwei selbstgewählten Sortierverfahren.

3. Einige Methoden obiger einfach verketteter Liste lassen sich (im Gegensatz zum Array oder einer doppelt verketteten Liste) effizient (in unterschiedlicher Hinsicht) implementieren, andere nicht unbedingt – welche sind das und warum?

   (6 Pkt.)

4. Implementieren Sie obige Datenstruktur und 4 der oben genannten Funktionalitäten (möglichst xlaufzeiteffizient) als doppelt verkettete Liste.

   (8 Pkt.)

5. Analysieren Sie die Komplexität der von ihnen implementierten Sortierverfahren **allgemein** und **im speziellen Fall** Ihrer Implementierung.

   (6 Pkt.)

Deadline und Punkte

Table 1. Maximale Punkte und Deadline

Deadline:	**28.12.2021 - 23:55**
Maximale KP:	**25 KP (Klausur Punkt)**
Maximale Punktanzahl:	**50 Punkte** == **100 %** == **25 KP**

Wird die volle Punktzahl beim Aufgabenblatt erreicht, dann haben Sie **100 %** erfüllt und somit **25 KP * 1.0** = **25 KP**. Haben Sie nur **50 %** erfüllt, dann berechnen sich die Klausurpunkte, wie folgt **25 KP * 0.5** = **12.5 KP** usw.

Feedback

Senden Sie mir gerne konstruktives Feedback zu den Aufgabenblättern und Dokumenten zu.

Verbesserungsvorschläge für dieses Aufgabenblatt können Sie gerne per Git-Patch oder schriftlich per E-Mail an mich senden. Git-Patches geben Ihnen die Möglichkeit, zusätzliche Punkte für ein Aufgabenblatt zu erhalten. Dafür müssen die Git-Patches atomar sein und im ZIP-Archive abgelegt, oder per E-Mail an mich gesendet werden.

Viel Spaß und Erfolg bei der Bearbeitung der Aufgabenblätter!

Erstellungsdatum und -zeit: 2021-10-12 - 22:52:45 +0200

Literaturverzeichnis

- [1] Thomas Ottmann and Peter Widmayer, Algorithmen und Datenstrukturen (German Edition), Spektrum Akademischer Verlag}, 2002, ISBN: 3827410290.
- [2] Sebastian Dworatschek, Grundlagen der Datenverarbeitung (de Gruyter Lehrbuch) (German Edition), de Gruyter, 1989, ISBN: 3110120259.
- [3] Robert Sedgewick, Algorithmen in C++ (German Edition), Pearson Studium, 2002, ISBN: 3827370264.
- [4] Ullenboom, Christian (Autor), Java ist auch eine Insel – Das Standardwerk für Programmierer. Über 1.000 Seiten Java-Wissen. Mit vielen Beispielen und Übungen, aktuell zu Java 14, 2020, ISBN: 9783836277372. https://www.buchhandel.de/buch/Java-ist-auch-eine-Insel-9783836277372
- [Java10] Openbook: Java ist auch eine Insel
- [Java11] Java ist auch eine Insel - Kapitel Vererbung
- [Java12] Java ist auch eine Insel - Kapitel Vererbung
- [Java13] JAVA Interfaces
- [Java14] JUnit-Testframework
- [Java15] Java ist auch eine Insel - Generics<T>
- [Gradle20] Gradle
- [Gradle21] Gradle Installation
- [Gradle22] Gradle Tutorials and Guides
- [Git-SCM] Git
- [Git-EN-v2-Book] Pro Git Book - Englisch (PDF etc.)
- [Git-DE-v2-Book] Pro Git Book - Deutsch
- [HTW70] HTW Berlin - Projekteserver
- [PlantUML-ClassDiagram] Plant-UML-Klassendiagramme
- [Asciidoctor] Asciidoctor
- [QuickAsciidoctor] Asciidoctor - Quick Reference
- [HTWMoodle] HTW Berlin - Moodle
- [DeepLTranslator] AI Assistance for Language
- [Eclipse Plugin for Gradle] Buildship Plugin for Gradle
- [GraphViz Pocket Reference] GraphViz Pocket Reference

Appendix A: Leitfaden für das Konsolenmenü

Die Reihenfolge und Zuordnung der Nummern (siehe unten) zu den verschiedenen `Optionen` bitte einhalten, um aufgrund der hohen Anzahl von Abgaben ein automatisiertes Skript zur Kontrolle einsetzen zu können.

Auswahl des Listentyps:

Bei der Auswahl der Option `1` oder `2` wird der entsprechende Listentyp für das folgende Menü ausgewählt:

```
Console-Application: Exercise-2              <Vorname> <Name> <Matrikelnummer>

Select list type before starting the main menu:

  1. SinglyLinkedList
  2. DoublyLinkedList

Please enter a number for an option: 1
```

- Konsolenhauptmenü*

Dieses Menü unterstützt die relevanten Hauptfunktionalitäten vom Interface `Listable<T>` für das Testen Ihrer Liste per Command Line Interface (CLI).

```
Console-Application: Exercise-2              <Vorname> <Name> <Matrikelnummer>

You selected SinglyLinkedList:

  1. Add Student to the end of this list.
  2. Inserts the Student at the specified position in this list.
  3. Inserts the Student at the beginning of this list.
  4. Appends the Student to the end of this list.
  5. Returns the Student at the specified position in this list.
  6. Print all students to console from list.
  7. Returns the number of Students in this list.
  8. Removes the Student at the specified position in this list.
  9. Removes all of the Students from this list.
 10. Search for student(s) by different characteristics.
 11. Sort list by different properties.
  0. Exit

Please enter a number for an option:
```

Eigenschaften des Studenten einlesen:

Bei der Auswahl der Option 1, 2, 3 oder 4 werden die folgenden Informationen im Untermenü abgefragt:

```
 Please enter prename: Sheldon
 Please enter surname: Cooper
 Please enter course number:  1
 Please enter matriculation number: 5
```

Suche eines oder mehrerer Studenten anhand einer Eigenschaft:

Bei der Auswahl der Option 10 werden die folgenden Informationen im Untermenü abgefragt:

```
Select a property to search for the student:

 1. Search by prename?
 2. Search by surname?
 3. Search by course number?
 4. Search by matriculation number?

Please enter a number for an option: 1
```

Zum Beispiel Suche alle Studenten mit den Vornamen Sheldon:

```
Please enter prename for the search: Sheldon<enter>
```

Sortieren von Studenten anhand von Eigenschaften:

Bei der Auswahl der Option 11 werden die folgenden Informationen im Untermenü abgefragt:

```
Select a sorting method for sorting:

 1. Bubblesort?
 2. Selectionsort?

Please enter a number for an option: 1
```

```
Please select a property for sorting with the 'Bubblesort' algorithm:

 1. Sort by course?
 2. Sort by matriculation number?

Please enter a number for an option: 1
```

Die Sortierverfahren Bubblesort und SelectionSort sind Beispiele von mir. Sie können gerne andere Sortierverfahren verwenden.

Vielen Dank!

Appendix B: Abgabebedingungen

Voraussetzung, damit eine Bewertung vorgenommen wird!

Bei jeder Abgabe muss der Gradle-Wrapper `./gradlew` im **Projektrootverzeichnis** vorhanden sein und **funktionieren**, sonst wird **keine Bewertung der Aufgabe vorgenommen**. Es dürfen nur relative Pfade verwendet werden. **Die folgenden Tasks müssen per Command-line im Projektrootverzeichnis ausführbar sein**:

```
./gradlew clean                    # Löscht automatisch generierte Dateien.
./gradlew build                    # Programm kompilieren.
./gradlew test                     # Alle Unittests ausführen.
./gradlew javadoc                  # API Dokumentation erstellen.
./gradlew run -q --console=plain   # Programm per Console ausführen.
./gradlew run                      # Programm per Console ausführen.
```

Bitte in der `build.gradle` den Parameter `System.in` setzen, damit die Eingabe von Parametern `./gradlew run -q --console=plain` auf der Konsole per Gradle funktioniert.

Halten Sie bitte die Reihenfolge der Optionen und die Zuordnung der Nummern ein, damit automatische Skripte für die Kontrolle der Abgaben eingesetzt werden können.

Verwalten Sie den Source Code über die Versionsverwaltung Git als Übung für zukünftige Projekte in der Angewandten Informatik (AI).

Appendix C: Bewertung und Verteilung der Klausurpunkte (KP)

Wie wird die Bewertung in diesem Semester vorgenommen?

Es **müssen folgende Aufgabenblätter in diesem Semester abgegeben werden**. Sie können insgesamt **50 Klausurpunkte (KP) bei mir in der Laboreinheit** erhalten. **Sie müssen mindestens 25 Klausurpunkte (KP) bei mir erreichen, damit Sie an der Prüfung bei Prof. Sieck teilnehmen können.**

Aufgabenblatt 1

15 Klausurpunkte - Themen: Einführung (Gradle, allgemeine Algorithmen, Anforderung an die Abgaben)

Aufgabenblatt 2

25 Klausurpunkte - Themen: Einfach und doppelt verkettete Liste, Sortierverfahren und Komplexitätsberechnung

Aufgabenblatt 3

15 Klausurpunkte - Themen: Adjazenzliste, Adjazenzmatrix, Graphen und ADTs

Aufgabenblatt 4

Entfällt

Aufgabenblatt 5

10 Klausurpunkte - Themen: Lineares Sondieren, Quadratisches Sondieren und Double Hashing

Aufgabenblatt 6

30 Klausurpunkte - Themen: Verlustfreie Kompressionsalgorithmen

Die Aufgabenblätter haben ein eigenes Punktesystem. Es sind keine Klausurpunkte (KP). Wenn Sie 100 % des Aufgabenblatts erfüllt haben, dann bekommen Sie die maximal möglichen Klausurpunkte pro Aufgabenblatt. Die maximal möglichen Klausurpunkte stehen auf dem jeweiligen Aufgabenblatt am Ende. Hier ein kurzes Rechenbeispiel der maximal möglichen Klausurpunkte pro Aufgabenblatt:

- Wenn Sie z.B. beim Aufgabenblatt 1 die volle Punktzahl (100%) erreichen, dann bekommen Sie die vollen 15 Klausurpunkte.
- Wenn Sie z.B. beim Aufgabenblatt 2 die Hälfte der angegebene Punktzahl (50%) erreichen, dann bekommen Sie 12,5 Klausurpunkte von 25 maximal möglichen Klausurpunkten.
- Wenn Sie z.B. beim Aufgabenblatt 3 die volle Punktzahl erreichen, dann bekommen Sie die 15 Klausurpunkte.

In Summe können Sie aus der Laboreinheit und dem seminaristischen Unterricht 100

Klausurpunkte erreichen. Bei mir können Sie genau die Hälfte der Klausurpunkte (also 50 KP) erreichen. Die restlichen 50 Klausurpunkte bekommen Sie von Prof. Sieck in einer mündlichen oder schriftlichen Prüfung. Wenn Sie alle Aufgabenblätter in der Laboreinheit mit 100 % absolvieren, dann haben Sie die maximal möglichen 50 Klausurpunkte (KP) bei mir aus der Laboreinheit erreicht.