

DAP2 Praktikum – Blatt 9

Ausgabe: 08. Juni — Abgabe: 14.–17. Juni

Studienleistung (Scheinkriterien)

- Zum Bestehen des Praktikums muss jeder Teilnehmer die folgenden Leistungen erbringen:
 - Es müssen mindestens 50 Prozent der Punkte in den Kurzaufgaben erreicht werden.
 - Es müssen mindestens 50 Prozent der Punkte in den Langaufgaben erreicht werden.
- An Feiertagen suchen Sie sich bitte einen Ersatztermin, ansonsten wird dies als “nicht erschienen” gewertet.
- Im Krankheitsfall kann ein Testat (bei Vorlage eines Attests) in der folgenden Woche nachgeholt werden.
- **Wichtig:** Notieren Sie sich Ihre Punkte nach jedem Testat! Dies dient der eigenen Kontrolle, da Ihnen Ihr vollständiger Punktestand aufgrund möglicher Gruppenwechsel während des laufenden Semesters nicht genannt werden kann.

Hinweis zu den Kurzaufgaben

Diese Aufgaben sollen im Rahmen der Praktika vor Ort erledigt werden. Anschließend präsentieren Sie Ihrem Tutor die Lösungen auf Ihrem Praktikumsrechner im Pool und bekommen dafür Punkte (nach Absprache mit dem Tutor ist es auch möglich, die Lösung auf ihrem eigenen Rechner zu präsentieren). **Sie dürfen und sollen die Kurzaufgaben in Gruppen von bis zu drei Studierenden bearbeiten und präsentieren!**

Wichtig: Der Quellcode ist natürlich mit sinnvollen Kommentaren zu versehen. Überlegen Sie außerdem in welchen Bereichen Invarianten gelten müssen und überprüfen Sie diese ggf. an sinnvollen Stellen mit *Assertions* (siehe Hinweis auf Blatt 2).

Kurzaufgabe 9.1

(2 Punkte)

Lernziel: Traversieren von Binärbäumen

Ein *Sortierbaum* ist ein binärer Baum mit der Eigenschaft, dass das linke Kind eines Knotens v kleiner, und das rechte Kind von v größer als v ist.

- (a) In einen Sortierbaum mit ganzen Zahlen werden der Reihe nach folgende Zahlen eingefügt:
 $13 \ 17 \ 5 \ 3 \ -10 \ 100 \ 40 \ -5 \ 4 \ 12 \ 11$
- (b) Erstellen Sie die Zahlenfolge, die wir erhalten, wenn wir den Baum in pre-order, in-order und post-order traversieren (auf Papier).

- (c) Beim einem alternativen Traversieren des obigen Baumes erhalten wir folgende Ausgabe:
 13 5 3 -10 -5 4 12 11 17 100 40
 Auf welche Art und Weise wurde der Baum traversiert, um diese Folge der Zahlen zu erhalten?
- (d) Implementieren Sie einen Sortierbaum in Java. Das Programm soll als Eingabe ein Array aus Zahlen erwarten und einen Sortierbaum aufbauen. Nachdem der Baum erstellt wurde, soll er in-order, pre-order und post-order traversiert werden. Das Ergebnis soll hierbei ausgegeben werden.

Kurzaufgabe 9.2

(2 Punkte)

Lernziel: Ternäre Bäume

Es gibt auch Bäume mit mehr als zwei Teilbäumen als Nachfolger. Beispielsweise könnte man in einem *ternären Sortierbaum* in einem Teilbaum alle kleineren Zahlen und in einem Teilbaum alle grösseren Zahlen als den Wurzelwert und in einem dritten Teilbaum alle gleich grossen Zahlen wie den Wurzelinhalt eingefügt werden.

- (a) Erstellen Sie zeichnerisch einen ternären Baum, in dem nacheinander die Zahlen 13, 4, 2, 15, 13, 100, 100, 1, 2, 50, 1, 50 eingefügt werden.
- (b) Implementieren Sie einen ternären Baum in Java. Das Programm soll als Eingabe ein Array aus Zahlen erwarten und einen ternären Baum aufbauen. Nachdem der Baum erstellt wurde, soll er in-order, pre-order und post-order traversiert werden. Das Ergebnis soll hierbei ausgegeben werden.

Kurzaufgabe 9.3

(4 Punkte)

Lernziel: Rucksackproblem (dynamische Programmierung)

Implementieren Sie den in der Vorlesung vorgestellten Algorithmus zum Rucksackproblem. Der Algorithmus verwendet eine Tabelle zur dynamischen Programmierung, um in $O(nz)$ Zeit zu laufen, wobei n die Anzahl der Elemente und z die Summe der Werte sind. Gehen Sie dabei wie folgt vor:

- Ihr Programm erwartet als Eingabeparameter die Zahl n und die Zahl z .
- Erstellen Sie die Klasse "Ware", die die Attribute "wert" und "gewicht" bereitstellt.
- Erstellen Sie n Waren, deren Werte summiert z ergeben. Dazu sollen die Werte zufällig gleichverteilt gewählt werden. Die Gewichte sind ebenfalls gleichverteilt zu wählen.
- Plotten Sie die Laufzeit des Algorithmus abhängig von n und z .
- Schreiben Sie einen Algorithmus, der die Waren mit den besten Verhältnis von Wert gegen Gewicht zurest wählt.
- Ist dieser Algorithmus ein Greedy-Algorithmus?
- Wie gut schneidet dieser Algorithmus in Gegensatz zu dem Algorithmus aus der Vorlesung ab?