technische universität
dortmund

Fakultät für Informatik
Lehrstuhl 14
Lars Hildebrand, Marcel Preuß,
Iman Kamehkhosh, Marc Bury, Diana Howey

Praktikum zu

Einführung in die Informatik für LogWilngs und WiMas

Wintersemester 2015/16

Übungsblatt 13

Besprechung:
08.02.16 -
12.02.16

Vorbereitende Aufgaben

Aufgabe 13.1: Dynamische Datenstrukturen

In dieser Aufgabe sollen Sie verschiedene Datenstrukturen miteinander vergleichen.

- Welche Vorteile haben verkettete Listen gegenüber Arrays?

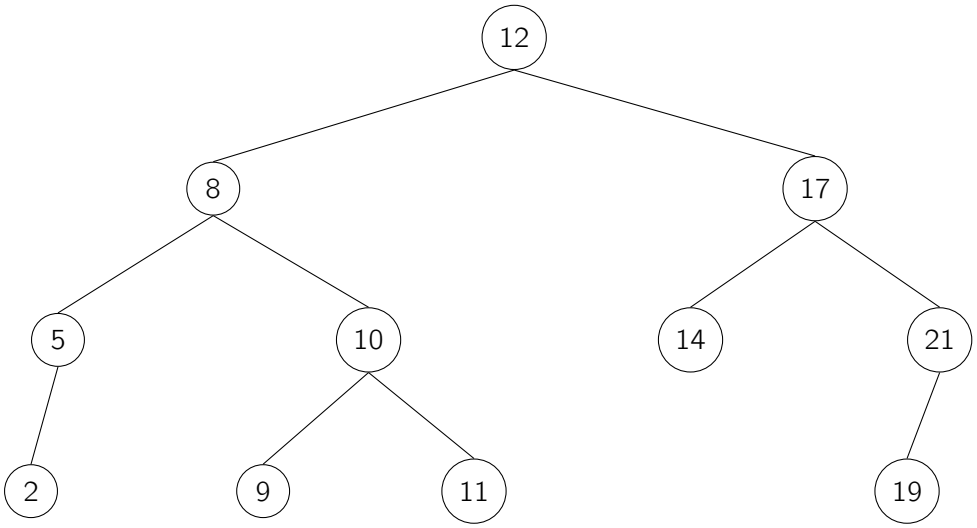
- Und welche Nachteile?

- Welche Vorteile haben binäre Suchbäume gegenüber Listen?

Aufgabe 13.2: Binäre Bäume: Verständnis 1

In dieser Aufgabe sollen Sie sich mit dem Aufbau von binären Bäumen beschäftigen.

Gegeben sei ein binärer Baum:



- Handelt es sich bei diesem Baum um einen binären Suchbaum?

- In welcher Reihenfolge werden die Knoten bei einem Pre-Order Durchlauf ausgegeben?

- Welchem Durchlauf entspricht die Folge $2, 5, 8, 9, 10, 11, 12, 14, 17, 19, 21$?

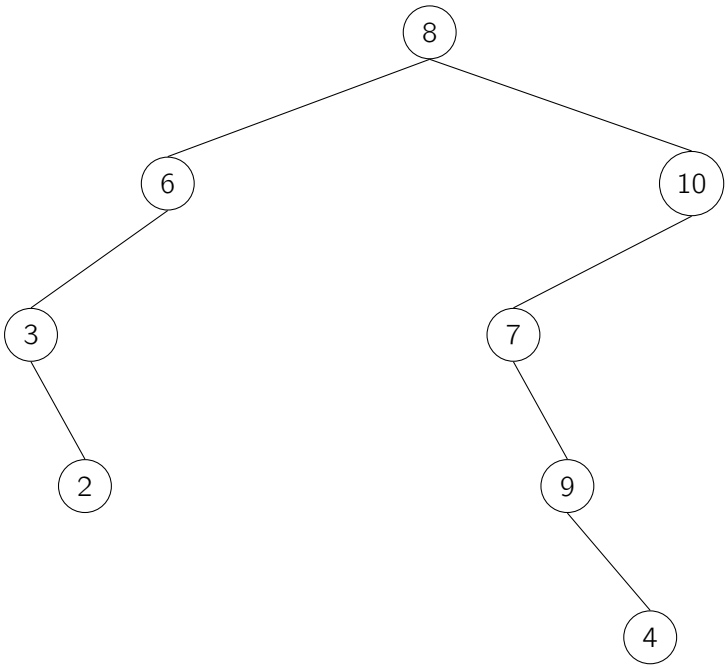
- Als welches Kind von welchem Knoten muss die Zahl 18 eingeordnet werden?

- Was ist die Höhe des Baumes?

Präsenzaufgaben

Aufgabe 13.3: Binäre Bäume: Verständnis 2

Gegeben sei ein binärer Baum:



- In welcher Reihenfolge werden die Knoten bei einem In-Order Durchlauf ausgegeben?

- Warum ist der In-Order Durchlauf dieses Baumes nicht aufsteigend sortiert?

- In welcher Reihenfolge werden die Knoten bei einem Post-Order Durchlauf ausgegeben?

- Welcher Datenstruktur ähneln die Unterbäume der 8?

Aufgabe 13.4: Bäume und Knoten

In dieser Aufgabe sollen Sie Knoten eines Baumes modellieren, der Integerwerte verwaltet.

- Erstellen Sie im Paket **blatt13** eine neue Klasse mit dem Namen **TreeNode**.
- Objekte dieser Klasse sollen zwei Referenzen auf weitere **TreeNode**-Objekte (die Kinder **leftChild** und **rightChild**) sowie den zu speichernden Wert (**value**) als Attribut haben.
- Erstellen Sie zwei Konstruktoren:
  - Einen Konstruktor, der nur den Wert entgegen nimmt und die Referenzen der Kinder auf **null** setzt.
  - Einen Konstruktor, der zusätzlich zu dem Wert die Referenzen des linken und rechten Kindes entgegen nimmt und setzt.
- Fügen Sie der Klasse Methoden hinzu, die den Wert und die Kind-Referenzen zurückgeben.
- Fügen Sie der Klasse ebenso Methoden hinzu, die den Wert und die Kind-Refenzen setzen können.

Aufgabe 13.5: Binäre Suchbäume

In dieser Aufgabe sollen Sie letztendlich einen Binärbaum implementieren.

- Die folgende Klasse **BinarySearchTree** ist teilweise vorgegeben. Die Vorgabe sowie eine Testklasse sind auf der EINI-Webseite zu finden.
- Diese Klasse soll das Einfügen, Suchen und die Ausgabe der einzelnen Baumdurchläufe realisieren. Die Methode **add** ist bereits vorgegeben, implementieren Sie die folgenden Methoden:
  - `boolean contains(int value)`
  - `void printInOrder()`, `void printPreOrder()` und `void printPostOrder()`
- Die Methode **contains** soll im Baum nach dem gesuchten Wert suchen und **true** oder **false** zurückgeben, je nachdem ob er im Baum existiert oder nicht.
- Die Methode **printInOrder** soll den In-Order Baumdurchlauf durch den Baum ausgeben.

  Schreiben Sie dafür die private Hilfsmethode `printInOrder(TreeNode node)`. Diese soll **rekursiv** mit den Kindern des übergebenen Knoten aufgerufen werden und zwischen der Ausgabe der Kinder den eigenen Wert ausgeben.
- Implementieren Sie die Methoden **printPreOrder** und **printPostOrder** in ähnlicher Weise.

Ergänzende Aufgaben

Aufgabe 13.6: Höhe eines Baumes

Schreiben Sie eine Methode **height**, die die Höhe eines **BinarySearchTree**-Objektes zurückgibt.