Aufgaben Tag 4

Aufgabe 1: Linked List

Schreiben Sie eine Implementierung einer einfach verketteten Liste. Gehen Sie dafür wie folgt vor:

a) Erstellen Sie die Klassen *Node* und *List*. Eine Node repräsentiert hierbei ein Element der Liste. Eine Node muss einen Wert speichern und seinen Nachfolger kennen. Die Listenklasse muss die erste Node der Liste kennen.
b) Erstellen Sie Konstruktoren für die beiden Klassen. Eine Node muss mit ihrem Wert initialisiert werden. Eine Liste wird nicht bereits mit dem ersten Element initialisiert. Erstellen sie außerdem eine *print()* Methode in der Klasse *Node*.
c) Schreiben Sie die beiden Methoden *isEmpty()* und *insertFront(Node n)* in der Klasse *List*. Die Methode *isEmpty()* soll *true* zurückgeben, falls die Liste keine Elemente enthält. Die Methode *insertFront(Node n)* fügt ein Element am Anfang der Liste ein.
d) Schreiben Sie die Methode *count()*, welche die Anzahl der Elemente in der Liste zurückgibt.
e) Implementieren Sie die Methoden *insert(Node n)* und *insertAt(Node n, int pos)*, welche ein Element am Ende bzw. an Position *pos* in der Liste einfügt. Achten Sie auch darauf, ggf. zu aktualisieren, welches das erste Element der Liste ist.
f) Implementieren Sie eine Methode *get(int k)*, welche das *k*-te Element der Liste ausgibt. Implementieren Sie weiterhin eine Methode *printAll()*, welche alle Elemente der Liste ausgibt.

Aufgabe 2: Binary Trees

a) Zeichnen Sie einen Binärbaum, in den in dieser Reihenfolge folgende Elemente eingefügt werden: {10, 5, 15, 3, 7, 99, 19, 12, 1, 4, 14, 111}
b) In diesem Binärbaum, suchen Sie nun nach dem Element 4. Notieren Sie alle Zwischenschritte.

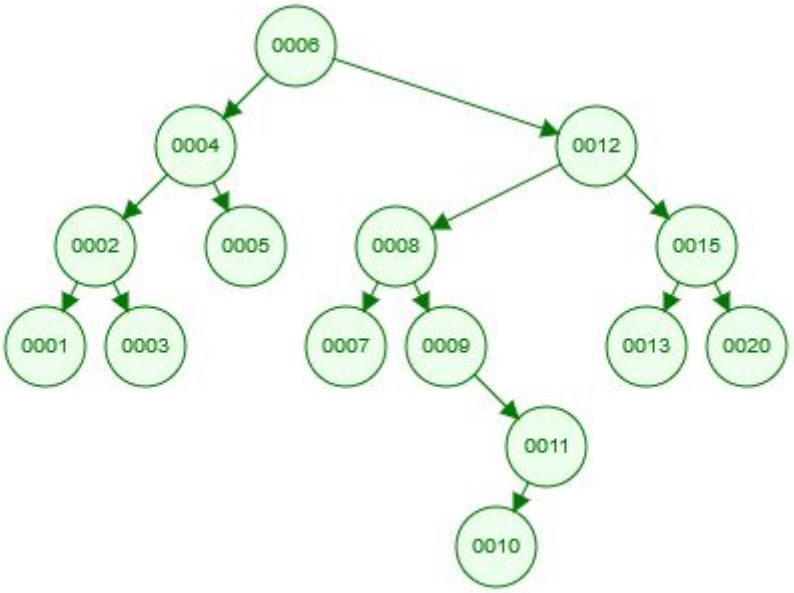
c) Gegeben sei die folgende Implementierung eines Binärbaums:

```java
class Node {
    public Node left;
    public Node right;
    public int value;

    public Node(int value) {
        this.value = value;
    }
}

public class BinaryTree {
    private Node root;

    public void insert(int value) {
        // Assume this is implemented
    }

    public Node min() {
        // Implement this
        // Finds the Node with the smallest value in the tree
    }

    public Node max() {
        // Implement this
        // Finds the Node with the biggest value in the tree
    }
}
```

Implementieren Sie die beiden Methoden *min()* und *max()*.

d) Implementieren Sie nun die Methode *insert(int value)*. Ob Sie hierbei iterativ oder rekursiv vorgehen, ist ihnen überlassen. Wir empfehlen jedoch, als Übung beide Versionen zu schreiben.

e) Geben Sie alle Elemente dieses Binärbaumes in Infix, Prefix und Postfix Notation an.



Aufgabe 3: Symbol Tables (Hash-Tabellen)

Geben Sie die Belegung einer Hash-Tabelle der Größe 11 an, wenn die Werte 38, 93, 16, 27 und 115 nacheinander in die anfangs leere Tabelle eingefügt werden. Verwenden Sie als Hash-Funktion *h(k) = k mod 11*. Also würde z.B. der Wert 12 and der Stelle 12%11 = **1** eingefügt werden. Im Falle einer Kollision, verwenden Sie als Ausweichfunktion *h'(k) = h(k) * h(k) mod 11*.