

## Übungen zu Algorithmen

Wintersemester 2003/2004

### Blatt 12

Wenn Sie in einem der folgenden Studiengänge studieren, **müssen** Sie sich bis zum 29.01.2004 zur Algorithmenklausur anmelden, da der Algorithmenschein für Sie eine Prüfungsleistung ist.

Studiengang	Anmeldung bei
Bachelor Physik mit Informatik	Frau Heinze (31/323)
Diplom Physik	Frau Heinze (31/323)
Bachelor Mathematik/Informatik	Frau Lohaus (31/448c)

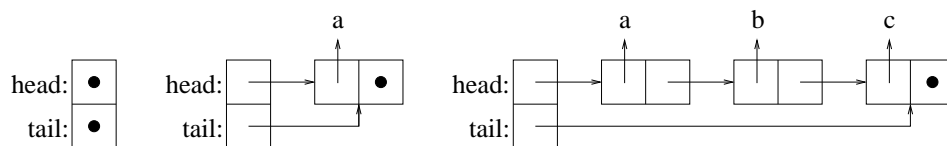
#### Aufgabe 12.1 (40 Punkte)

Implementieren Sie den ADT *Schlange* gemäß dem Interface `/home/ainf/Vorlesung/Schlange.java` aus dem Skript **mit Hilfe von Verweisen**.

Die Klasse `VerweisSchlange` hat folgende Definitionen:

```
private static class SchlangenEintrag{
    Object          inhalt; // Inhalt des SchlangenEintrags.
    SchlangenEintrag next; // Zeigt auf naechsten SchlangenEintrag.
}
private SchlangenEintrag head; // Zeigt auf den Anfang der Schlange.
private SchlangenEintrag tail; // Zeigt auf das Schlangenende.
```

Eine *VerweisSchlange* hat folgende Gestalt v.l.n.r.: neu erzeugt, mit einem Element 'a', mit drei Elementen 'a', 'b', 'c':



Testen Sie Ihre Implementation des ADT *Schlange* mit Hilfe einer Klasse `VerweisSchlangeTest` analog zu `/home/ainf/Vorlesung/ArraySchlangeTest`.

#### Aufgabe 12.2 (20 Punkte)

Implementieren Sie in einer Klasse `BaumIO` die **rekursive** Methode `void printBaum(Baum b, int t)`, die einen binären Baum liegend ausgibt (s. Beispiel). `printBaum` erhält als zweiten Parameter die Tiefe der aktuellen Ebene im Baum, die der Einrücktiefe bei der Ausgabe entspricht. Verwenden Sie nur die Methoden aus `/home/ainf/Vorlesung/Baum.java`.

#### Aufgabe 12.3 (15 Punkte)

Implementieren Sie in der Klasse BaumIO auch die **rekursive** Methode

VerweisBaum baueZufallsbaum(int n), die einen

/home/ainf/Vorlesung/VerweisBaum.java mit n Knoten liefert, die jeweils einen zufälligen Buchstaben enthalten. Jeder Knoten hat dabei zufällig 0,1 oder 2 Söhne (s. Beispiel). Verwenden Sie dazu folgende Methoden aus /home/ainf/Uebung/Blatt12/Zufall.java:

```
/** Liefert eine Zufallszahl zwischen 0 und max-1. max muss >= 1 sein. */
public static int zufallsZahl(int max)
```

```
/** Liefert einen zufaelligen Kleinbuchstaben zwischen 'a' und 'z' */
public static Character zufallsBuchstabe()
```

### Aufgabe 12.4 (10 Punkte)

Implementieren Sie in der Klasse BaumIO auch die **rekursive** Methode

int hoehe(Baum b),

die die Höhe des Baumes in Ebenen liefert. Ein leerer Baum hat die Höhe 0, ein Blatt die Höhe 1, eine Wurzel mit einem Blatt als Sohn hat die Höhe 2 usw..

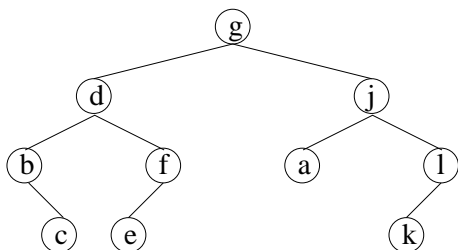
### Aufgabe 12.5 (15 Punkte)

Schreiben Sie eine Klasse BaumIOtest, die in ihrer main-Methode zunächst eine Zahl n einliest und anschließend durch den Aufruf Baum b=BaumIO.baueZufallsbaum(n); einen zufälligen binären Baum b aufbaut und zurückliefert (s. Beispiel). Traversieren Sie den Baum mit drei Traversierungen Ihrer Wahl aus dem Skript. Geben Sie dann den Baum liegend durch Aufruf von BaumIO.printBaum(b, 0); aus (s. Beispiel). Anschließend geben Sie die Höhe des Baumes auf dem Bildschirm aus, nachdem Sie sie durch Aufruf von BaumIO.hoehe(b) berechnet haben.

### Beispiel

Baum b =  
BaumIO.baueZufallsbaum(10);

liefert z.B. folgenden Baum b zurück:



Die Ausgabe dieses Baumes mit

```
BaumIO.printBaum(b, 0);
```

liefert den Baum folgendermaßen:

```

      l
      k
    j
    a
g
  f
  e
d
  c
  b
  
```