Aufgabenblatt 1 **Abgabe: 11.11.2007, 24.00 Uhr**

1. Eingabe (leicht) (5 Punkte)

Wie Sie Werte auf der Konsole ausgeben können, wissen Sie bereits.

- Schreiben Sie ein Programm, das zwei der folgende Variablentypen (nach eigener Wahl) von der Konsole einliest und wieder ausgibt: (2 Punkte)

$$byte, short, int, long, double, char$$

- Erweitern Sie Ihr Programm so, dass es mehrere (gleiche) Werte aus einer Zeile einlesen kann. (3 Punkte)

Hinweis: Verwenden Sie dafür die Klasse *java.util.StringTokenizer*.

2. Variable Parameterlisten (leicht) (8 Punkte)

In Java gibt es sogenannte variable Parameterlisten. Dabei kann eine Methode nicht nur eine fest vorgegebene Anzahl von Parametern besitzen, sondern beliebig viele des gleichen Typs. Ein Beispiel:

```
public int addiere(int... summanden) { /* Schlüsselwort "..." */
    int summe = 0;
    for (int i = 0; i < summanden.length; i++) {
        summe += summanden[i];
    }
    return summe;
}
```

Implementieren Sie in Java einen binären Suchbaum für ganze Zahlen mit den Methoden *einfuegen* und *istEnthalten*. Dabei sollen alle drei Methoden mit beliebig vielen Parametern aufgerufen werden können. Es sollen also mehrere Elemente auf einmal gelöscht und eingefügt werden können. Welche Semantik *istEnthalten* hat, können Sie selbst entscheiden.

Hinweis: Wie Sie wissen gibt es in Java keine Zeiger, daher müssen Sie solche Verknüpfungen mittels Variablen für Objekten und dem Befehl *new* erzeugt werden.

```
private Baum links,rechts;
.
.
links = new Baum(..);
.
.
```

3. Abstrakte Klassen (Mittel) (8 Punkte)

Eine abstrakte Klasse ist eine Klasse mit der Einschränkung, dass keine Instanzen von ihr erzeugt werden können (es können aber Variablen dieses Typs existieren). Gegeben sei folgende abstrakte Klasse:

```
abstract class Zahl {
    private float floatWert;

    Zahl(float initFloatWert) {
        floatWert = initFloatWert;
    }

    float floatWert() {
        return floatWert;
    }
    abstract String zahlTyp();
    abstract void infoAusgabe();
}
```

Von dieser Klasse können nun weitere Klassen abgeleitet werden. Dabei ist zu beachten, dass die abstrakten Methoden $zahlTyp()$ und $infoAusgabe()$ der abstrakten Klasse Zahl in der abgeleiteten Klasse implementiert werden müssen! Wieder ein Beispiel:

```
class GanzeZahl extends Zahl {
    private int intWert;

    GanzeZahl(int initIntWert) {
        super((float)initIntWert); /* ruft Konstr. der Oberklasse auf*/
        intWert = initIntWert;
    }

    String zahlTyp() {
        return "natürliche Zahl";
    }

    void infoAusgabe() {
        System.out.println(" Wert: " + intWert);
    }
}
```

Erweitern Sie obiges Konzept mit einer abgeleiteten Klasse für rationale und reelle Zahlen. Dabei soll die rationale Zahl auch wirklich als eine ganze und eine natürliche Zahl gespeichert und ausgegeben werden. Der $floatWert$ der Oberklasse muss aber trotzdem immer gesetzt werden, um einen Vergleich zwischen den Zahlen zu ermöglichen.