Technische Universität München
Fakultät für Informatik
Prof. Dr. Seidl

WS 2004/2005
Übungsblatt 6
2. Dezember 2004

Übungen zu Einführung in die Informatik I

Aufgabe 18 Vergleiche und Referenzen in Java

Die Sprache Java bietet zum Vergleichen den Vergleichsoperator == und die für alle Objekte verfügbare Methode `public boolean equals(Object obj)` (Aufruf: `x.equals(y)`).

a) Gegeben seien folgende Variablendeklarationen:

```
int a = 1;
int b = 1;
Integer c = new Integer(1);
Integer d = new Integer(1);
```

Die Klasse `Integer` speichert ganzzahlige Werte als Objekte.

- Welche Vergleiche sind möglich? Welche Ergebnisse liefern diese Vergleiche? Schreiben Sie ein kleines Testprogramm, um dies herauszufinden.
- Wie unterscheiden sich Variablen vom Typ `int` und Variablen, die durch `new` instanziiert wurden?
- Was unterscheidet den Operator == von der für alle Objekte verfügbaren Methode `equals`?

b) Implementieren Sie eine Klasse `Person` mit den beiden Attributen `vorname` und `nachname`. Diese soll einen Konstruktor zum Initialisieren der Attribute, sowie eine Methode `public boolean equals(Person p)` zum Testen auf Gleichheit zweier Personen enthalten.

Aufgabe 19 Doppelt verkettete Listen

In der Vorlesung wurde das Konzept der einfach verketteten Liste vorgestellt. Schreiben Sie eine Klasse, die *doppelt verkettete* `int`-Listen realisiert. Eine Liste nennt man doppelt verkettet, wenn jedes Element einen Verweis auf den Nachfolger *und* den Vorgänger besitzt. Die Klasse soll folgende Methoden enthalten:

- `insert` fügt vor dem aktuellen Element ein
- `append` fügt hinter dem aktuellen Element ein
- `delete` löscht das aktuelle Element
- `length` ermittelt die Anzahl der Listenelemente
- `getHead` liefert das erste Element (den Kopf) der Liste
- `reverse` dreht die Reihenfolge der Listenelemente um

- `toString` gibt jedes Listenelement genau einmal aus

Bei `insert`, `append` und `delete` ist das aktuelle Element immer das Objekt, dessen Methode aufgerufen wird. Die anderen Methoden können von jedem Objekt aufgerufen werden, beziehen sich aber auf die Liste als Ganzes.

Aufgabe 20 Dynamische Felder bei Schlangen

Eine Schlange ist eine Datenstruktur zur Speicherung von Elementen nach dem FIFO-Prinzip (First-In-First-Out). Die zweite der in der Vorlesung vorgestellten Implementierungen von Schlangen (Klasse: `Queue`) verwendet zur Speicherung der Elemente ein Feld. Zwei Indizes, `first` und `last`, geben dabei die Position des ersten bzw. letzten Wertes an. Bei Bedarf wird das Feld beim Einfügen von Elementen durch `enqueue()` durch ein größeres Feld ersetzt. Eine Verkleinerung erfolgt nicht.

Erweitern sie die in der Vorlesung vorgestellte Methode `dequeue()` (zum Entnehmen des nächsten Elements), so dass das Array bei Bedarf auch verkleinert wird. Überlegen sie sich ein geeignetes Verkleinerungskriterium.

Auszug aus der Klasse `Queue`:

```
public class Queue {
  private int first, last; // Index des ersten bzw. letzten Elements
  private int[] a;

  public Queue() {
      first = last = -1;
      a = new int[4];
  }

  public boolean isEmpty() {
      return first == -1;
  }

  public String toString() { ... }

  public void enqueue(int x) { ... }

  public int dequeue() {
      int result = a[first];
      if (last == first) {
          first = last = -1;
      } else {
          first = (first + 1) % a.length;
      }
      return result;
  }
}
```

Die gesamte Klasse `Queue` finden sie auf der Internetseite zur Übung.

Hinweis:
Betrachte die in der Vorlesung vorgestellten Stack-Implementiertung. Diese verwendet ebenfalls ein Array, das bei Bedarf vergrößert/verkleinert wird.