Übungen zu Vorlesung
Praktische Informatik I
Wintersemester 2002/03

Serie 4

Aufgabe 1: Binäre Suche (rekursiv) (50%)

a) Implementieren Sie eine Java–Methode zur binären Suche, welche als Parameter ein *sortiertes* Integer–Array beliebiger Länge sowie einen Integerwert bekommt, und feststellen soll, ob und falls ja, an welchem Index dieser Wert in dem Array steht. Geben Sie dazu die Precondition an, die beim Methodenaufruf gelten muss, und die Postcondition die *vor* dem Rücksprung aus der Methode gelten muss; d.h. entspechend dem Schema:

```
static int binsearch(int[] a, int ug, int og, int value){
// PRE: Precondition ueber a, ug, og, value

... Eure Loesung ...

// POST: Postcondition ueber a, ug, og, value, und ggf.
         weitere lokale  Variablen

return    // Rueckgabewert
        }
```

b) Lösen Sie obiges Problem mit Hilfe der *rekursiven* Binären Suche. Dabei sollen für die verschiedenen Rekursionsschritte keine Kopien des Arrays angelegt werden!
Die Suchmethode soll den entsprechenden Indexwert als Rückgabewert an das aufrufende Hauptprogramm zurückliefern, aber den Wert *nicht* selber auf dem Bildschirm ausgeben.
c) Schreiben Sie eine Klasse, in der die Verwaltung von Arrays mit den Methoden

- hinzufügen
- sortieren
- suchen
- drucken

möglich ist. Hier geben wir den Rahmen vor, einschliesslich Implementierung der Methoden für das Sortieren und Drucken. Zum Sortieren kann die Lösung zu 1 a) oder 1 b) geeignet verwendet werden. Die Methode zum Einfügen von Werten, `pushElement`, soll der in der Vorlesung diskutierten entsprechen.

Aufgabe 2: Die Türme von Hanoi (50%)

Das Problem der Türme von Hanoi stellt eine Aufgabe dar, die sich sehr gut mit den Mitteln der Rekursion lösen läßt: Gegeben seien `n` Scheiben

mit unterschiedlichem Durchmesser, die auf Platz A der Größe nach geordnet zu einem Turm aufgeschichtet sind, wobei die unterste Scheibe die größte ist. Die Scheiben sollen unter Verwendung des Hilfsplatzes C auf den Platz B transportiert werden. Dabei darf zu jeder Zeit nur *eine* Scheibe bewegt werden und diese darf *nur oben* von einem Turm abgenommen werden. Außerdem darf *niemals eine größere Scheibe auf einer kleineren liegen.*
Eine einfache rekursive Vorgehensweise zu diesem Problem besteht darin, es in kleinere Teilprobleme zu zerlegen:

1. Bewege alle bis auf die unterste Scheibe auf den Hilfsturm
2. Bewege die größte Scheibe auf den Zielturm
3. Bewege die restlichen Scheiben vom Hilfsturm auf den Zielturm

Implementieren Sie den hier angedeuteten Algorithmus in Java. Überlegen Sie sich hierzu eine geeignete Methodensignatur, um bei den rekursiven Aufrufen die notwendigen Parameter übergeben zu können. Alle tatsächlich notwendigen Bewegungen von Scheiben sollen auf dem Bildschirm in der folgenden Form ausgegeben werden: `<Scheibengröße>: <Von-Turm> -> <Nach-Turm>`, wobei die Scheibengrösse einfach mit jeder größeren Scheibe um Eins steigt und die anfangs oberste Scheibe per Definition immer die Größe 1 besitzt.

Abgabe: 16. – 20.12.2002 nach den jeweiligen Praktika. Die Abgabe soll sowohl elektronisch (Programm-Quellcode) als auch in gedruckter Form (mit LaTeX gesetzter kommentierter und erläuterter Quellcode) erfolgen. Dabei ist jeweils auch auf geeignete Testfälle und deren Dokumentation zu achten!