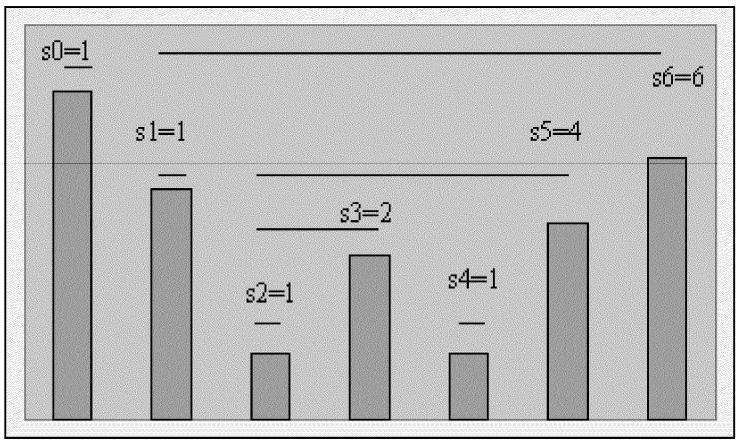
Algorithmen und Programmierung III

WS 2002/2003 — Übung 2 — H. Schweppe

Ausgabe: 24.10.2002
Abgabe: 5.11.2002

Aufgabe 2.1 (10 P)
Wir betrachten die Kurse a[0],..a[n-1] einer Aktie an n aufeinanderfolgenden Tagen. Die (positive) Spanne s_i am Tag i ist die Anzahl der aufeinander folgenden Tage bis zum Tag i einschließlich, an denen der Kurs kleiner oder gleich a[i] war.
Formal: s_i = max {k | k <= i+1,a[j] <= a[i] für j = i-k+1,...i}



Gesucht ist ein Programm, das in linearer Zeit s[i], 0 <= i < n berechnet.
Zu spezifizieren sind Voraussetzung und Effekte ("pre and post") sowie ggf. Schleifeninvarianten. Hinweis: verwenden Sie einen Stapel (stack).
Implementieren Sie das Programm als Java-Klasse, geben Sie eine knappe Begründung für die lineare Laufzeit und messen Sie die Laufzeit für verschiedene n. Die Ergebnisse sollen graphisch mit dem Benchmark-Werkzeug (VL, siehe auch Web-Seite) ausgegeben werden.

Aufgabe 2.2 (4 P)

Folgendes Codefragment ist gegeben:

```
public class Sort {
   public void quickSort(Sortable a[]) {
   ...}
   ...}

public interface Sortable {
    public boolean lessThan (Sortable s);
 }

class Country implements Sortable {....}

class Student implements Sortable {....}

class SortTest {
// sorts an array of country names (strings)
// and then outputs them

...
}
```

a) geben Sie ein Klassendiagramm mit der in der Vorlesung eingeführten graphischen Notation an.

b) Ergänzen Sie den Code der Klasse `SortTest`. Das Wort `Sortable` soll nicht im Code vorkommen. (Trockenübung).

Aufgabe 2.3 (6 P)
Während eines großen Software-Crash bei der Firma MacroSoft wurden alle Datenstrukturen mit Ausnahme der `class FastStack implements StackADT {...}` zerstört. `FastStack` implementiert einen Stapel (stack). Auf Anraten eines Beratungsunternehmens entschloss man sich, alle sonstigen linearen Datenstrukturen mit Hilfe von FastStack zu reimplementieren.

a) Tun Sie das für Schlangen (Queue) mit den Operationen `isEmpty (), enqueue () und dequeue ()`. Hinweis: 2 Stapel verwenden. (Trockenübung).
b) Geben Sie die Laufzeit im schlechtesten Fall an (O-Notation), wie immer mit Begründung.

Aufgabe 2.4 (6 P)
Gesucht ist eine algebraische Spezifikation des Datentyps Schlange (queue) mit den Operationen `newQueue` (leere Schlange), `enqueue, dequeue, head` (liefert nächstes Element ohne die Schlange zu verändern), `isEmpty.`

Aufgabe 2.5 (2P)
Geben Sie die Invariante für die innere Schleife des Programms `maxDiffSlow ()`

an (siehe VL - Folien).

Aufgabe 2.6 (2P)

a) Geben Sie mindestens 4 verschiedene Implementierungen für den Abstrakten Datentyp Schlange an.
b) Beschreiben Sie in maximal 3 Zeilen den Begriff "Geheimnisprinzip"
c) Richtig oder falsch (Begründung): wenn f(n) = O(n) ist, dann ist sie auch
   1/2 +O(n)
   O(n log n)
   O(log n)
d) Gegeben die naive Implementierung der Fibonacci-Folge:
   fib(0) = 0; fib(1)=1;fib(n) = fib(n-1)+fib(n-2) .
   Ist die Aussage: Der Algorithmus hat das gleiche asymptotische Laufzeitverhalten im besten, schlechtesten und mittleren Fall sinnvoll? ja/nein (kurze Begründung)
   Viel interessanter als diese Frage ist die Web-Seite:
   http://www.mcs.surrey.ac.uk/Personal/R.Knott/Fibonacci/fib.html

Trockenübung heißt: kein lauffähiger Code mit Test etc. erforderlich. Alle für die Bearbeitung der Aufgabe wichtigen Code-Stücke müssen aber auf dem Papier stehen (Java oder Haskell).