Informatik II (D-ITET)
FS 2009
Prof. Dr. F. Mattern
Philipp Bolliger (bolligph@inf.ethz.ch)

ETH Zürich
Institut für Pervasive Computing
Verteilte Systeme
http://www.vs.inf.ethz.ch

Übungsserie 12

Ausgabe: 20. Mai 2009
Keine Abgabe

Die Aufgaben dieser Serie werden nicht mehr bewertet, sind jedoch relevant für die Prüfung.

1. Aufgabe: Heap

(1a) Wieviele Elemente sind in einem Heap der Höhe h minimal und maximal enthalten?

(1b) Ist ein aufsteigend sortiertes Array ein Heap (wenn es als Binärbaum interpretiert wird)?

(1c) Wenden Sie die Operationenfolge get_min; get_min; insert(12); get_min; insert(21) auf den in Klammerdarstellung notierten Heap 2(7(17(23,19),18(20)),9(15,25)) an und geben Sie den Array-Inhalt der niveauweisen Speicherung nach jeder Operation an.

(1d) Ist folgendes eine korrekte Alternative zur Methode get_min aus der Vorlesung? (Begründung!) Wenn ja, ist diese Variante besser oder schlechter?

- Wurzel entfernen.
- Lücke durch den kleineren der beiden Nachfolger ersetzen.
- Die dadurch entstandene neue Lücke entsprechend rekursiv behandeln.

(1e) Was geschieht bei der Methode get_min aus der Vorlesung (Folie 536), wenn der Heap leer ist? Ist das korrekt?

2. Aufgabe: Heapsort

Der Heapsort-Algorithmus erlaubt die Sortierung eines Arrays *in place*, d.h. ohne zusätzlichen Speicheraufwand: alle Schritte des Algorithmus operieren lediglich im Speicherbereich des ursprünglichen Arrays.

In dieser Aufgabe soll ein Array, das mit Zufallszahlen besetzt ist, sortiert werden. Dazu ist der Heapsort-Algorithmus anzuwenden. Dieser Algorithmus läuft in zwei Phasen ab. In der **ersten Phase** wird aus dem unsortierten Array ein Heap gebildet. In der **zweiten Phase** werden die Elemente dem Heap in der korrekten Reihenfolge entnommen. (Beachten Sie, dass ein Array als Binärbaum gelesen werden kann.)

Führen Sie dieses Verfahren an dem angegebenen Array durch. Dokumentieren Sie dabei die Zwischenzustände, die sich beim Aufbau des Heaps und bei der Bildung des sortierten Arrays ergeben. Benutzen Sie dafür das angegebene Schema. Das Verfahren kommt ohne zusätzlichen Speicherplatz aus, d.h. der Heap wird im gleichen Speicherbereich aufgebaut, in dem sich die Zahlenfolge befindet. Die angegebene Markierung zeigt an, wo die Grenze zwischen dem Heap (links) und der Zahlenfolge (rechts) verläuft.

Phase 1: Aufbau des Heaps

Unsortierte Folge:	7	4	3	0	2	5
Schritt 1:						
Schritt 2:						
Schritt 3:						
Schritt 4:						
Schritt 5:						
Heap:						

In der untersten Zeile steht als Zwischenergebnis ein Heap.

Phase 2: Aufbau des sortierten Arrays

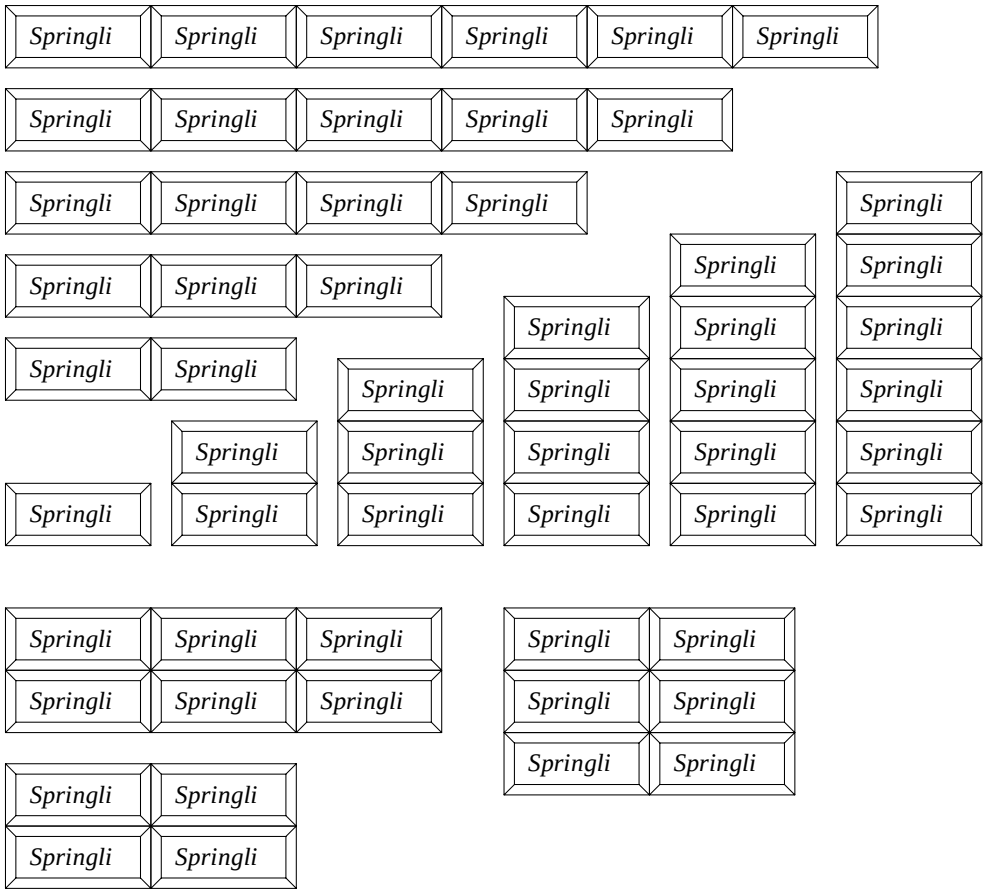
Heap des ersten Schemas:						
Schritt 1:						
Schritt 2:						
Schritt 3:						
Schritt 4:						
Schritt 5:						
Sortierte Folge:						

In der untersten Zeile steht ein (umgekehrt) sortiertes Array.

3. Aufgabe: Rekursives Problemlösen

(Nach einer Idee von Prof. Dr. W. Brauer, TU München)

Die Firma *Springli* beabsichtigt, eine neue Schokolade auf den Markt zu bringen. Da über Grösse und Format noch Uneinigkeit herrscht, soll die Akzeptanz aller rechteckigen Formate mit maximal n Stückchen auf dem Markt getestet werden. Nachfolgend sind als Beispiel alle 14 möglichen Formate dargestellt, die sich mit maximal 6 Stückchen konstruieren lassen.



Wieviele Formate muss die Firma *Springli* in Abhängigkeit von n auf dem Markt testen? Geben Sie eine rekursive Java-Methode an, die dies für beliebige n berechnet. Hinweis: Für $n = 1, 2, 3, 4, 5, 6$ sind $1, 3, 5, 8, 10, 14$ Formate zu testen.