11. Übung **Informatik B** **SS 08**

Klaus Kriegel **Abgabe: 08.07.2008, 12:00 Uhr**

Alle Lösungen der Aufgaben 1 und 2 werden als Zusatzpunkte gewertet. Aufgabe 3 wird nicht bewertet.

Aufgabe 1: **Sortieren** (8 Punkte)

Auf den Punkten im Raum $\mathbb{R}^3$ definieren wir eine totale Ordnungsrelation $\leq$, die prmär die x–Koordinaten, sekundär die y–Koordinaten und letzlich die z–Koordinaten der Punkte vergleicht, d.h. für zwei Punkte $P = (x, y, z)$ und $P' = (x', y', z')$ gilt

$$P \leq P' \iff (x < x') \vee (x = x' \wedge y < y') \vee (x = x' \wedge y = y' \wedge z \leq z')$$

Sei $m \in \mathbb{N}$, $n = m^2$ und sei eine Folge A von n Punkten aus dem Bereich $[0, m)^3 \subseteq \mathbb{R}^3$ gegeben. Bereiche der Form $[x, x+1) \times [y, y+1) \times [z, z+1)$ werden halboffene Einheitswürfel und Bereiche der Form $[x, x+1) \times [y, y+1) \times [0, m)$ werden halboffene Einheitstäbe genannt. Der Vergleich von zwei reellen Zahlen sowie das Auf– und Arunden einer reellen Zahl auf den nächsten ganzzahligen Wert werden als Elementaroperationen betrachtet. Nutzen (und kombinieren) Sie die in der Vorlesung besprochenen Sortierverfahren, um möglichst schnelle Algorithmen (Laufzeiten in $\Theta(f(n))$ angeben) zum Sortieren dieser Folge zu entwerfen, wobei die folgenden Nebenbedingungen vorausgesetzt werden können:

a) In jedem halboffenen Einheitswürfel liegt höchstens ein Punkt aus A.

b) In jedem halboffenen Einheitsstab liegen höchstens 100 Punkte aus A.

c) In jedem halboffenen Einheitswürfel liegen höchstens $\log n$ Punkte aus A.

d) In jedem halboffenen Einheitsstab liegen höchstens $\sqrt{n}$ Punkte aus A.

Aufgabe 2: **Heap-Implementierung** ($4 + 2 + 2$ Punkte)

Wir wollen mit einer Klasse `Heap` eine Halde mit `int`–Werten auf einem Array implementieren. Zur Vereinfachung nehmen wir an, dass die maximale Heap–Größe dem Konstruktor als Parameter `n` übergeben wird. Als Attribute werden ein Feld `int[] A` und ein Index `int lastElem` verwendet.

a) Implementieren Sie zwei (rekursive) Methoden `void upHeapBubble(int i)` und `void downHeapBubble(int i)`, die das Verhalden und das Versickern des Eintrags in `A[i]` realisieren. Entscheiden Sie selbst, ob Sie dazu Hilfsmethoden `int left(int i)`, `int right(int i)` und `int parent(int i)` zur Berechnung der Indizes von Kinder– und Elternknoten implementieren wollen, oder ob Sie diese Berechnungen in den Bubble–Methoden ausführen.

b) Implementieren Sie Einfüge– und Streichmethoden `void insert(int a)` und int delete().

c) Testen Sie die Funktionalität Ihrer Implementiereung, indem Sie eine in der Kommandozeile übergebene Folge von `int`–Werten mit Heap–Sort sortieren.

Aufgabe 3: **Halden** (0 Punkte)

Bauen Sie die Halden für die Eingabefolge $5, 4, 6, 1, 2, 3, 7$ mit der inkrementellen Methode (schrittweises Einfügen) und mit der Bottom–Up–Methode auf.
Wie sehen diese Halden nach jeweils zwei Löschoperationen aus?