6. Übung Algorithmen und Datenstrukturen WS 2016/17

Klaus Kriegel **Abgabe: 05.12.2016, 12:00 Uhr**

Aufgabe 1: Heap-Sort (2 + 2 + 2 Punkte)

a) Fügen Sie die Zahlen $9, 4, 7, 3, 2, 6, 1$ schrittweise in eine Halde ein, die am Anfang leer ist. Skizzieren Sie dazu den Zustand der Halde nach jedem Einfügeschritt. Sie können zur Vereinfachung der Darstellung auf das Zeichnen der unmarkierten Blätter verzichten.

b) Löschen Sie aus der obigen Halde zweimal das kleinste Element und stellen Sie wieder den Zustand nach jeder Löschoperation dar.

c) Führen Sie für die Folge $2, 5, 7, 3, 4, 6, 8$ die Bottom-Up-Konstruktion einer Halde aus. Stellen Sie dazu den Zustand nach jeder Stufe der Konstruktion dar. Führen Sie danach eine Löschoperation aus.

Aufgabe 2: Implementierung von Halden (6 + 2 Punkte)

a) Entwerfen Sie eine Klasse MyHeap, mit der eine Halde auf einem `int`-Array `A` implementiert wird. Dem Konstruktor wird ein `int`-Wert `n` übergeben, um ein Array dieser Größe anzulegen. Benennen Sie das Attribut, welches auf die letzte gefüllte Stelle von `A` zeigt, mit `lastElem`.
Implementieren Sie Methoden `void insert(int k)` und `int deleteMin()`, welche bei Überfüllung der Halde oder Löschen auf einer leeren Halde entsprechende Exceptions werfen. Nennen Sie die dabei verwendeten Hlfsmethoden void bubbleDown(int k) bzw. void bubbleUp(int k).
Implementieren Sie Testmethode zum Sortieren einer Eingabefolge.

b) Die Halde soll auch als dynamische Prioritäswarteschlange genutzt werden können, d.h. es ist zusätzlich ein Mehode `void updateKey(int i, int k)` zu implementieren, die das Element an der Stelle `i` in `A` mit dem Wert `k` überschreibt und danach die Ordnungseigenschaft wieder herstellt.

Aufgabe 3: Mobile als Binärbaum (6 Punkte + 4 Zusatzpunkte)

Ein Mobile wird aus Kugeln und Stäben zusammengesetzt. Die einfachste Form ist eine einzelne Kugel, die durch ein Blatt repräsentiert wird. Bei jedem Stab hängen am linken und rechten Ende zwei Submobiles (die durch Fäden befestigt sind). Stäbe werden deshalb durch innere Knoten repräsentiert.
Das soll durch eine Unterklasse `Mobile<E>` der Klasse `BTNode<E>` aus Resources/Java realisiert werden (der Typparameter `E` spielt im Weiteren keine Rolle, aber man könnte ihn auch für das Gewicht verwenden). Die neue Klasse muss einige zusätzliche Attribute und Methoden enthalten, deren Bedeutung im nachfolgenden Text erläutert wird:
Mit `boolean ball, baton` wird zwischen Kugel und Stab unterschieden.
`float weight` repräsentiert das Gesamtgewicht. Kugeln bekommen ihr (positives!) Gewicht mit dem Konstruktor zugewiesen. Bei Stäben und den Fäden wird (zur Vereinfachung der Rechnungen) das Eigengewicht 0 angenommen, so dass sich das Gesamtgewicht als Gewichtssumme der Submobiles ergibt.

`float length, leftLenth, rightLength` bezieht sich auf die Länge eines Stabs und deren Aufteilung von Aufhängungspunkt bis zum linken bzw. rechten Ende.

a) In ersten Aufgabenteil soll jedes Submobile fixiert auf einer Seite liegen (es bewegt sich also nichts). Zur Vereinfachung nehmen wir an, dass alle Kugeln den gleichen Durchmesser haben (jeweils eine Längeneinheit) und sich auf dem gleichen Höhenniveau befinden. Jede Kugel soll frei hängen, aber benachbarte Kugeln können sich berühren (siehe Abbildung links). Zu berechnen sind die notwendigen Stablängen und die Positionen der Aufhängung der Stäbe, so dass sie sich im Gleichgewicht befinden (Hebelgesetz). Die Aufgabe besteht darin, korrekte `set`-Funktionen für das Gewicht und die Längen zu konstruieren. Für die Ausgaben Ihrer Testfunktion sollten Sie das folgende Format verwenden:
`rep(m):=[m.weight]` wenn das Mobile `m` nur eine Kugel ist.
`rep(m):=[(m.weight, m.length, m.leftLenth, m.rightLenth) rep(m1) rep(m2)]` wenn `m` ein Stab und `m1, m2` linkes/rechtes Submobile von `m` ist.
Verwenden Sie als Testobjekt zuerst ein Mobile mit 4 Kugeln und den Gewichten ((1,2),(4,2)).

b) **Freiwillige Zusatzaufgabe:** Wie man schon an einfachen Beispielen sieht, können die in a) konstruierten Mobiles nicht störungsfrei rotieren. Wie muss man die Stablängen verändern, damit eine störungsfreie Rotation möglich ist (Berührung erlaubt) und gleichzeitig die Stablängen kleinstmöglich gehalten werden (siehe Abbildung rechts).
Hinweis: Ofensichtlich braucht man dafür ein Attribut float maxWidth , welches die maximale horizontale Weite vom Aufhängungspunkt bis zum Rand der äußeren Kugel unter allen möglichen Rotationen beschreibt.