Albert-Ludwigs-Universität, Inst. für Informatik
Prof. Dr. Fabian Kuhn
Y. Maus, O. Saukh

06 Mai, 2016

Algorithmen und Datenstrukturen
Sommersemester 2016
Übungsblatt 3

Abgabe bis 12:00, Freitag, 13 Mai, 2016

Aufgabe 1: Wattebälle (10 Punkte)

Sie befinden sich in einem Traum in einem Gefängnis über den Wolken. Um sich zu befreien, müssen Sie N Bälle aus Watte nach ihrem Gewicht sortieren. Watte ist super leicht und daher muss man sie sehr genau wiegen. Leider gibt es in Ihrer Gefängniszelle nur die folgenden Werkzeuge:

- Eine magische Waage mit drei Waagschalen. Wenn man drei Wattebälle hineinlegt, zeigt die Waage an in welcher Waagschale der Ball mit dem Median-Gewicht liegt. Die Waage funktioniert nur, wenn in jeder Waagschale genau ein Watteball liegt.

  Genauer, sei `MEDIAN(x,y,z)=y`, so gilt entweder $x < y < z$ oder $z < y < x$.

- Eine Hochpräzisonswaage, die den leichteren von zwei Wattebällchen anzeigt. Da Watte jedoch so leicht ist, funktioniert die Waage nur, wenn man den leichtesten aller Bälle mit dem schwersten aller Bälle vergleicht. Wenn man andere Bälle miteinander vergleicht, so ist das Ergebnis nicht verlässlich.

  Genauer, sei `LIGHTEST(a,b)` das Resultat einer Wägung der Bälle a und b. Falls a der leichteste aller Bälle ist und b der schwerste aller Bälle, so gilt `LIGHTEST(a,b)`$= a$. Falls a der schwerste und b der leichteste Ball ist, so gilt `LIGHTEST(a,b)`$= b$. Ansonsten, ist der Rückgabewert von `LIGHTEST(a,b)` unverlässlich.

Um alles ein wenig einfacher zu machen, können Sie annehmen, dass alle N Bälle verschiedene Gewichte haben. Natürlich möchten Sie die Bälle mit möglichst wenigen Wiegevorgängen sortieren.
Um die Aufgabe ein wenig klarer zu sehen nehmen Sie zunächst ein Nickerchen. Dort fallen Sie innerhalb Ihres derzeitigen Traumes in einen zweiten Traum. Dort erscheint Ihnen eine Fee, die Ihnen den leichtesten und den schwersten Watteball zeigt, jedoch sagt sie Ihnen nicht welcher welcher ist.

a) Geben Sie ein kurzes Beispiel (mit $N \geq 3$) an um zu erklären, wieso die Funktion `MEDIAN` allein nicht ausreicht, um den leichtesten vom schwersten Ball zu unterscheiden.

b) Sei l der leichteste Ball. Benutzen Sie $\mathcal{O}(1)$ viele Aufrufe von `MEDIAN` um die Funktion `LIGHTER(a,b)` zu implementieren. Die Funktion `LIGHTER(a,b)` funktioniert für alle Bälle a und b und gibt `WAHR` zurück, wenn a leichter ist als b, ansonsten `FALSCH`.

Nachdem Sie aus Ihrem zweiten Traum aufgewacht sind und wieder im ersten Traum sind, merken Sie, dass es garkeine Fee gibt! Lösen sie die folgenden Aufgaben ohne die Information, die die Fee Ihnen gegeben hätte.

c) Geben Sie einen Algorithmus an, der $\mathcal{O}(N)$ Aufrufe der Funktion `MEDIAN` benutzt um den schwersten und leichtesten Ball zu finden. Dabei müssen Sie nicht herausfinden, welcher der beiden Bälle der jeweils leichteste bzw. schwerste ist.

d) Erklären Sie wie Sie die vorherigen Teile benutzen können um N Wattebälle mit $\mathcal{O}(N \log N)$ Aufrufen von `MEDIAN` und $\mathcal{O}(1)$ Aufrufen von `LIGHTEST` zu sortieren.

e) Begründen Sie, dass man mindestens $\Omega(N \log N)$ Aufrufe von `MEDIAN` benötigt um die N Wattebälle zu sortieren.

Aufgabe 2: Doubly-Linked-List (10 Punkte)

Implementieren Sie eine Datenstruktur *DoublyLinkedList*, in welcher jedes Element einen Schlüssel, sowie zwei Zeiger auf das Element davor und das danach enthält. Dafür werden Sie zwei Klassen schreiben müssen - *ListElement* und *DoublyLinkedList*.
Die erste Klasse enthält ein Feld *key* und zusätzlich zwei Zeiger *previous* und *next* auf die Nachbarelemente in der Liste.
Die zweite Klasse implementiert *Doubly Linked List* "= eine doppelt verkettete Liste aus Elementen vom Typ *ListElement*.
Im public-*Ordner finden Sie vorgefertigte Strukturdateien, die spezifizieren, welche Operationen Ihre doppelt verkettete Liste anbieten soll. Sie können die Strukturdateien herunterladen und anpassen, bzw. mit Programmcode füllen.*