Lehrstuhl für Algorith.
und Datenstrukturen
Prof. Dr. Hannah Bast
Axel Lehmann

Algorithmen und Datenstrukturen
(Informatik II) SS 2017

http://ad-wiki.informatik.uni-freiburg.de/teaching

UNI FREIBURG

Übungsblatt 6

Abgabe bis Dienstag, den 13. Juni um 12:00 Uhr

Sie haben für dieses ÜB zwei Wochen Zeit, fangen Sie aber bitte vor Pfingsten noch damit an

Aufgabe 1 (5 Punkte)

Das in der Vorlesung erklärte Verfahren für dynamische Felder hat drei Parameter A, B, C, die wie folgt eine Rolle spielen, wenn ein Element eingefügt oder gelöscht wird:

Nach *pushBack*: wenn $s_{i-1} = c_i$, dann $c_i = A \cdot s_i$.

Nach *popBack*: wenn $s_{i-1} \leq c_{i-1}/B$, dann $c_i = C \cdot s_i$.

In der Vorlesung haben wir bewiesen, dass für $A = 2$, $B = 4$, $C = 2$ die amortisierten Kosten von n Operation immer $O(n)$ sind und dass für alle i gilt, dass $s_i \geq c_i/4$ (das heißt: das Feld ist immer mindestens zu einem Viertel voll).

Verallgemeinern Sie dieses Ergebnis wie folgt: für ein gegebenes f mit $0 < f < 1$, bestimmen Sie A, B, C (in Abhängigkeit von f) mit $A > 1$ und $B > C > 1$, so dass für alle i gilt, dass $s_i \geq f \cdot c_i$. Sie müssen nicht beweisen, dass die amortisierten Kosten weiterhin $O(n)$ sind (sie sind es).

Aufgabe 2 (5 Punkte)

Implementieren Sie einen "Kilometerzähler" mit der folgenden Funktionalität:

1. Es gibt k Ziffern mit gegebenen Obergrenzen $m_1, \ldots, m_k$ für jede Ziffer.

2. Zu Beginn ist der Zählerstand 0: das heißt, alle Ziffern sind 0.

3. Es gibt eine Operation *increment*, die den Zähler um eins erhöht. Dabei wird Ziffer 1 um 1 erhöht. Ist diese Ziffer danach $> m_1$, wird sie auf 0 zurückgesetzt und Ziffer 2 um eins erhöht. Ist die zweite Ziffer danach $> m_2$, wird sie auf 0 zurückgesetzt und Ziffer 3 um eins erhöht, usw. Siehe die TIP Datei auf dem Wiki für ein Beispiel dazu.

4. Schreiben Sie ein Programm, das für gegebene $m_1, \ldots, m_k$, den Zähler von 0 hochzählt, bis alle Ziffern maximal sind. Schreiben Sie dabei die erste Ziffer ganz rechts und den aktuellen Stand immer in dieselbe Zeile schreiben (das geht mit `\r` statt `\n` am Zeilenende).

[bitte mit dem präfrontalen Cortex wenden]

Aufgabe 3 (10 Punkte)

Beweisen Sie, dass die Kosten von n *increment* Operationen amortisiert $O(n)$ sind, wenn man mit Zählerstand 0 beginnt.

Hinweis 1: Überlegen Sie sich dazu eine geeignete Potenzialfunktion und führen Sie den Beweis mit Hilfe dieser Potenzialfunktion und des Mastertheorems (das sie benutzen können, ohne es nochmal selber zu beweisen). Erinnern Sie sich an die Intuition hinter einer guten Potenzialfunktion: sie soll messen, wie lange es bis zur nächsten teuren Operation dauert. Eine teure Operation ist dann kein Problem, wenn sich der Wert der Potenzialfunktion dadurch entsprechend erhöht.

Hinweis 2: Es gibt viele Möglichkeiten, wie man die Potenzialfunktion wählen kann. Wählen Sie weise: wählen Sie eine Potenzialfunktion, die möglichst wenig Arbeit macht im Beweis.

Committen Sie Ihren Code (als Code) und die Beweise (als PDF) in unser SVN, in einen neuen Unterordner *blatt-06*. Schauen Sie wie immer, dass Jenkins glücklich ist. Und vergessen Sie nicht Ihre *erfahrungen.txt*.

Schreiben Sie ein Programm, dass n mal den Satz *Ich darf keine Programme mit quadratischer Laufzeit schreiben* ausgibt und Laufzeit $\Theta(n^2)$ hat. Wenn sich Ihr Computer weigert, das Programm auszuführen, sprechen Sie mit ihm oder ihr darüber.