Karlsruher Institut für Technologie
Institut für Theoretische Informatik

Prof. Dr. Dennis Hofheinz
Lukas Barth, Lisa Kohl

3. Übungsblatt zu Algorithmen I im SoSe 2016

https://crypto.iti.kit.edu/index.php?id=algo-sose16
{lisa.kohl,lukas.barth}@kit.edu

Aufgabe 1 (*Amortisierte Analyse,* $2+2$ *Punkte*)

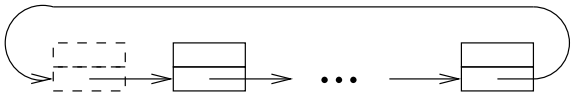
Gegeben ist ein binärer Zähler, bei dem pro Speicherstelle 1 Bit gespeichert ist. Der Zähler ist mit 0 initialisiert ist. Mit der Operation *inc* wird der Zähler um eins erhöht und mit der Operation *res* auf 0 zurückgesetzt. Nehmen Sie im Folgenden die Kosten für das Kippen eines Bits mit 1 an.

a) Zeigen Sie, dass die Kosten von n Operationen in $\mathcal{O}(n)$ liegen, indem Sie den Operationen je nach Zustand des Zählers geeignete amortisierte Kosten zuweisen.

b) Nun kommt eine weitere Operation *dec* hinzu, bei der der Zähler um eins erniedrigt wird. Was sind nun die Kosten von n Operationen im worst-case (im $\mathcal{O}$-Kalkül)?

Aufgabe 2 (*Einfach verkettete Liste,* $1+2+3$ *Punkte*)

Sie verwenden die einfach verkettete Listen-Datenstruktur mit Dummy-Header aus der Vorlesung in einem Programm. Die Datenstruktur mit n Listen-Elementen ist von Ihrem Kollegen programmiert worden. Der Kollege versichert Ihnen, dass jedes Listen-Element auf ein weiteres Listen-Element zeigt.



Sie möchten nun noch die Konsistenz der Liste überprüfen, indem sie ausschließen, dass eins der Listen-Elemente auf ein vorheriges Listen-Element zeigt.

a) Entwerfen Sie einen Algorithmus, der in $\mathcal{O}(n)$ Zeit prüft, ob eine Liste konsistent ist. Sie dürfen dabei annehmen, dass die Listenlänge bekannt ist und in jedem Listen-Element ein zusätzliches mit 0 initialisiertes Bit Speicher verfügbar ist, dass Sie verändern dürfen.

b) Entwerfen Sie wieder einen Algorithmus, der in $\mathcal{O}(n)$ Zeit und mit $\mathcal{O}(1)$ zusätzlichen Platz prüft, ob eine Liste konsistent ist. Diesmal darf Ihr Algorithmus dabei allerdings die Liste *nicht verändern* und die Listenlänge ist unbekannt.

c) Erweitern Sie Ihren Algorithmus aus a) oder b), indem Sie zusätzlich feststellen *an welchem* Element die Invariante verletzt wird. Wieder darf dieser die Liste nicht verändern und nur mit $\mathcal{O}(1)$ zusätzlichem Platz arbeiten. Die Listenlänge dürfen Sie hier aber als bekannt voraussetzen.
Ein Algorithmus in $\mathcal{O}(n^2)$ Laufzeit erhält 1 Punkt, einer in $\mathrm{o}(n^2)$ zusätzlich 2 Punkte.

Begründen Sie jeweils die Laufzeit und den Platzbedarf Ihrer Algorithmen.

Aufgabe 3 (*XOR Listen,* $1+2+1+1$ *Punkte*)

In der Vorlesung wurden doppelte verkettete Listen L mit zwei Zeigern pro Listenelement, $x.prev$ und $x.next$, implementiert. Nehmen Sie an, dass alle Zeigerwerte als k-bit Integer-Zahlen interpretiert werden können, auf denen Bit-Operationen durchgeführt werden können. Nehmen Sie weiter für die Aufgabenstellung an, dass jedes Listenelement ein Feld *key* hat, dass die eindeutige Identifizierung des Elements ermöglicht (dieses darf nicht verändert werden; es wird nur für die *search*-Funktion benötigt).

Hinweis: $\oplus$ ist die bitweise XOR-Operation, d.h. für $i,j \in \{0,1\}$ gilt $i \oplus j = \begin{cases} 0 & \text{falls } i = j, \\ 1 & \text{sonst} \end{cases}$.

a) Gegeben sind zwei k-Bit-Integer I und J. Zeigen Sie mit Hilfe der Definition $(I \oplus J) \oplus J = I$.

b) Erklären Sie, wie eine doppelt verkettete Liste mit nur einem Zeiger $x.np$ pro Listenelement implementiert werden kann. **Hinweis**: mit der Datenstruktur müssen nur die Funktionen aus Teilaufgabe c) und d) effizient implementiert werden können.

c) Implementieren Sie die Funktion *search*, die ein Listenelement mit einem gegebenen Key findet, in $\mathcal{O}(|L|)$ und die Funktion *insertFront* in $\mathcal{O}(1)$.

d) Beschreiben Sie, wie die Reihenfolge der Listenelemente in $\mathcal{O}(1)$ umgekehrt werden kann.

Aufgabe 4 (*Dr. Meta vs. Unbounded Arrays,* 3 *Punkte*)

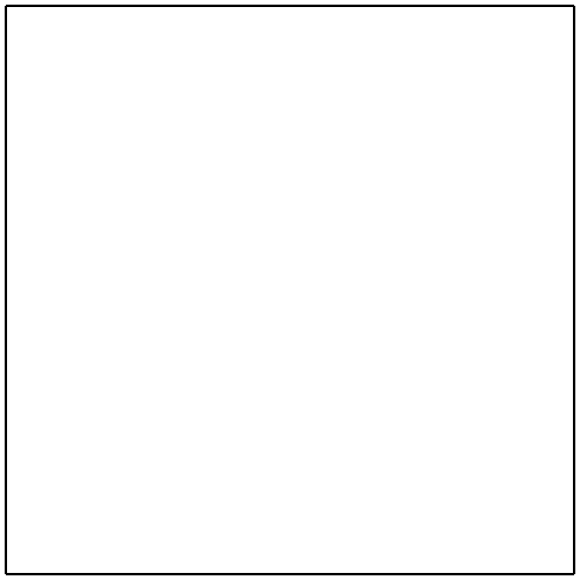
Der ebenso gerissene wie gefährliche Superschurke Dr. Meta hat wie jedes Jahr zur Weltherrschafts-Konferenz geladen. Einer seiner mittelmäßig intelligenten Handlanger ist mit der Organisation des Tagungshotels betraut. Eine seiner Aufgaben ist es, während der Konferenz sicherzustellen, dass immer ein für alle versammelten Bösewichte ausreichend großer Raum im Hotel zur Verfügung gestellt wird. Diese Aufgabe wird dadurch erschwert, dass Schurken sich nicht an Regeln halten, und nach Lust und Laune die Konferenz betreten oder verlassen. Das Hotel hat Räume, die beliebige Zweierpotenzen von Personen fassen können (also 2, 4, 8, 16, ... Personen).
Die Strategie des Handlangers sieht wie folgt aus: Wenn der aktuelle Raum zu klein wird, so bucht er den nächstgrößeren - also einen Raum, der doppelt so viele Personen fasst wie der aktuelle. Daraufhin muss das versammelte Verbrecherkollektiv den aktuellen Raum verlassen und in den neuen Raum umziehen. Wenn der aktuelle Raum nur noch zur Hälfte gefüllt ist, so bucht der Handlanger auf den nächst kleineren Raum um, um die bei Dr. Meta grundsätzlich knappen Geldmittel zu sparen. Wieder müssen alle Teilnehmer umziehen.
Gehen Sie nun davon aus, dass eine Teilgruppe der eingeladenen Gauner so durchtrieben ist, dass sie die Weltherrschafts-Konferenz selbst torpedieren wollen. Sie hecken daher eine Abfolge von „Konferenz betreten“ und „Konferenz verlassen“ aus, sodass ihre Kollegen möglichst oft umziehen müssen. Zeigen Sie: mit einer Abfolge von n „betreten“/„verlassen“-Aktionen ist es möglich, $\Theta(n^2)$ Personen-Umzüge zu erzwingen.
Irgendwann merkt selbst der Handlanger, dass er seine Strategie verändern muss. Was kann er besser machen?
Hinweis: Wenn k Personen den Raum wechseln müssen, so sind dies k Personen-Umzüge. Genauso gilt: Wenn eine Person l-fach den Raum wechseln muss, so sind dies l Personen-Umzüge.

Ausgabe: Mittwoch, 04.05.2016
Abgabe: Freitag, 13.05.2016, 12:45 im Briefkasten im Untergeschoss von Gebäude 50.34

Deckblatt Übungsblatt 3
Algorithmen I

A1	**A2**	**A3**	**A4**	$\sum$

Tutoriumsnummer:



Name	**Matrikelnummer**	**Unterschrift**

Mit unseren Unterschriften bestätigen wir, dass die Aufgaben von den Unterzeichnern eigenständig gelöst worden sind.

Hinweis: Das Übungsblatt darf in Gruppen von bis zu zwei Personen bearbeitet werden. Beide Personen müssen demselben Tutorium zugeteilt sein. Möchte jemand seine Abgaben-Gruppe innerhalb des Semesters wechseln, so ist dies im Voraus mit dem Tutor abzusprechen. **Bitte tragt in das obere Quadrat *groß* die Nummer eures Tutoriums ein.** Die Lösung des Übungsblattes ist in jedem Fall mit diesem Deckblatt abzugeben.