Informatik II: Algorithmen und Datenstrukturen SS 2015

Vorlesung 7a, Dienstag, 9. Juni 2015
(Verkettete Listen)

Prof. Dr. Hannah Bast
Lehrstuhl für Algorithmen und Datenstrukturen
Institut für Informatik
Universität Freiburg

UNI FREIBURG

Blick über die Vorlesung heute

UNI FREIBURG

- Organisatorisches
  - Ihre Erfahrungen mit dem Ü6 (Dynamische Felder)
- Inhalt
  - Verkettete Listen … Prinzip + Implementierung

    Laufzeitgleich wurde auf Vorlesung 7b verschoben
  - Übungsblatt 7, Aufgabe 2: Erweitern Sie die Klasse LinkedList aus der Vorlesung um lookup, remove, und size

Erfahrungen mit dem Ü6 1/2

UNI FREIBURG

- Zusammenfassung / Auszüge Stand 9. Juni 12:00
  - Im Prinzip für die meisten gut machbar
  - Gute Vorlage + noch schön was zum Nachdenken
  - Das "Runden" haben die meisten weggelassen
  - Für einige zeitintensiv, weil Sie bisher alles in Python gemacht haben, und noch nie was in Java oder C++

    Man konnte allerdings eins-zu-eins alles aus public/code kopieren und musste nur vier Zeilen ändern
  - C++ Vorlage wäre nett gewesen

    Jetzt für das Ü7, Vorlage in Python, Java und C++ !

Erfahrungen mit dem Ü6 2/2

- Musterlösung, Grundidee
  - Bei pushBack, wenn Feld ganz voll, dann $c_i = \lfloor X \cdot s_i \rfloor$
  - Bei popBack, wenn $s_i \leq f \cdot c_{i-1}$, dann auch $c_i = \lfloor X \cdot s_i \rfloor$
  - Einzige Bedingung: $X > 1$ und $f < 1/X$, damit keine Vergrößerung kurz nach Verkleinerung oder andersrum

    In der Analyse von Vorlesung 6b war $X = 2$ und $f = 1/4$

    Mit $f < 1/X$ ist das Feld am wenigsten gefüllt kurz vor dem Verkleinern und damit immer size $\geq f \cdot$ capacity
  - Für beliebiges f mit $0 < f < 1$ kann man einfach $X = 1 / f'$ mit f' sodass $f < f' < 1$ wählen, zum Beispiel $f' = (1 + f) / 2$

    Also für $f = 1/2$ zum Beispiel $f' = 3/4$ und damit $X = 4/3$

Verkettete Listen 1/7

- Grundprinzip
  - Wir betrachten hier **doppelt** verkettete Listen
  - Jedes Element kennt seinen Vorgänger (previousItem) und seinen Nachfolger (nextItem)
  - Außerdem kennt man den Anfang (firstItem) und das Ende (lastItem) der Liste
  - Das ermöglicht uns Einfügen und Löschen **an beliebiger Stelle** in O(1) Zeit

    Das können Felder nicht, siehe spätere Folie 12
  - Beispiele von solchen Listen auf den nächsten Folien …

Verkettete Listen 2/7

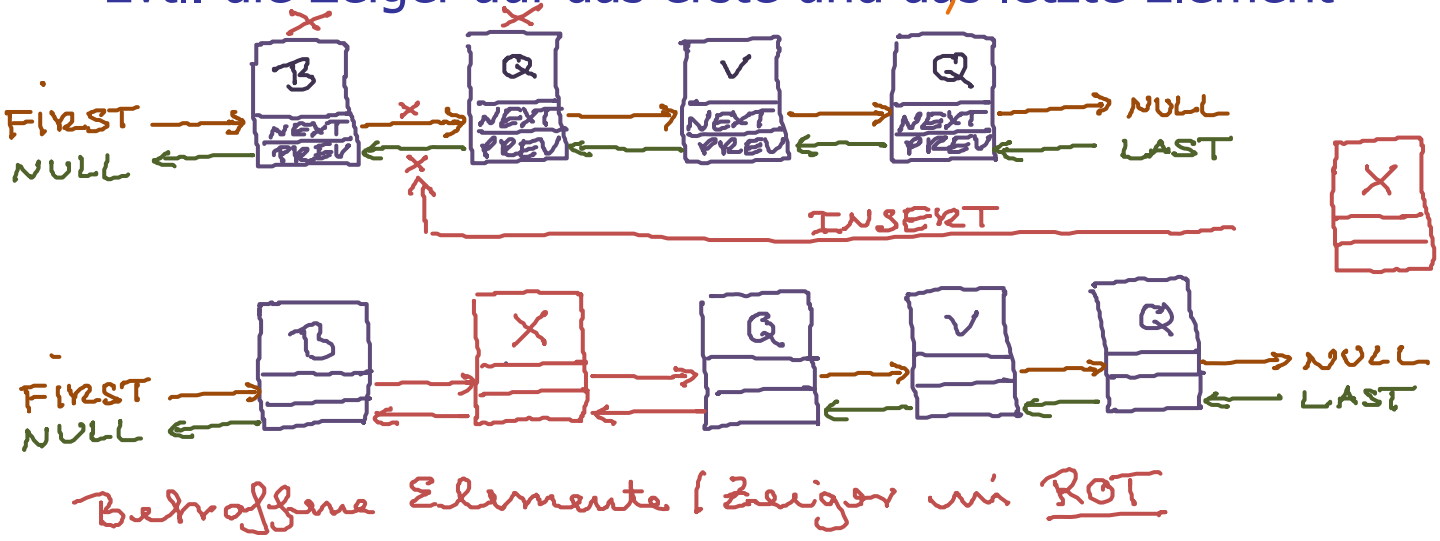
- Einfügen (insert) eines neuen Elementes
  - Im Prinzip nicht schwer: man muss nur die betroffenen Zeiger / Referenzen (siehe Folie 11) richtig "umbiegen"

    Den Nachfolgerzeiger vom Vorgänger

    Den Vorgängerzeiger vom Nachfolger

    Die beiden Zeiger des eingefügten Elementes

    Evtl. die Zeiger auf das erste und das letzte Element



Verkettete Listen 3/7

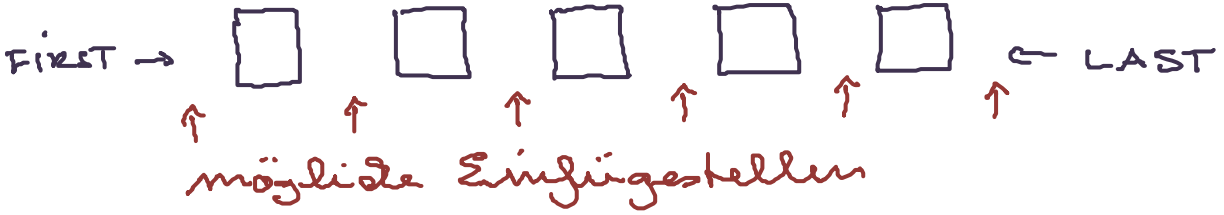
- Angabe der Einfügestelle
  - **Variante 1:** Referenz auf den neuen Nachfolger nextItem

    Dann nennt man die Operation oft insertBefore

    Einfügen am Ende, indem man nextItem = null übergibt
  - **Variante 2:** Referenz auf den neuen Vorgänger previousItem

    Dann nennt man die Operation oft insertAfter

    Einfügen am Anfang, indem man previousItem = null übergibt
  - **Variante 3:** Über einen zusätzlichen Typ ListIterator, der auf die n + 1 möglichen Stellen in einer Liste mit n Elementen zeigt … so wird es in Java und C++ gemacht



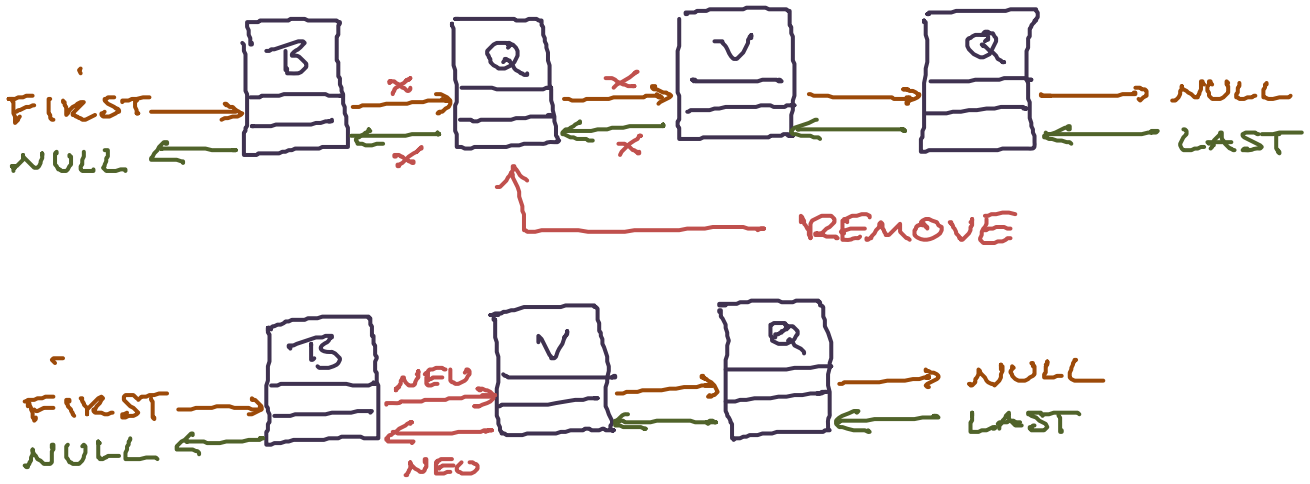
Verkettete Listen 4/7

- Entfernen (remove) eines geg. Elementes item
  - Auch hier müssen einfach nur die richtigen "Zeiger" "umgebogen" werden

    Nachfolgezeiger des Vorgängers von item

    Vorgängerzeiger des Nachfolgers von item

    Evtl. die Zeiger auf das erste und das letzte Element



Verkettete Listen 5/7

UNI FREIBURG

- Anzahl der Elemente
  - Ohne Weiteres muss man einmal von vorne nach hinten durch die Liste gehen, um die Anzahl Elemente zu ermitteln

    Laufzeit dafür $\Theta(n)$, wenn n = Anzahl Elemente
  - Es geht aber auch leicht in $O(1)$ Zeit

    Das ist Teil von Aufgabe 1 vom Ü7 und wird nicht verraten

Verkettete Listen 6/7

- Zugriff auf das i-te Element
  - In einer verketteten Liste können die Elemente **beliebig** im Speicher verteilt stehen
  - Will man das i-te Element haben, bleibt einem nichts anderes übrig, als sich "durchzuhangeln"

    Von vorne oder von hinten, was immer schneller ist
  - Die Laufzeit dafür ist also $\Theta(\min\{i, n - i\})$
  - Zum Vergleich: in einem Feld stehen die Elemente immer garantiert hintereinander im Speicher
  - Von daher Zugriff auf das i-te Elemente in $O(1)$ Zeit

    Steht an Stelle Anfangsadr. + i * Größe eines Elementes

Verkettete Listen 7/7

UNI FREIBURG

- In Java, C++ und Python
  - In Java: java.util.LinkedList<T> … in C++: std::list<T>
  - In Python gibt es keine native verlinkte Liste

    Einfügen an beliebiger Stelle kein häufiger Anwendungsfall

    Hier in der VL vor allem als Grundlage für Vorlesung 8a+b
  - Für die Vorlesung heute (und das Übungsblatt) wollen wir verkettete Listen **von Grund auf selbst** implementieren
  - Das Konzept des "Zeigers" gibt es in allen drei Sprachen:

    In C++ benutzt man wörtlich Zeiger … LinkedListItem*

    In Java und Python sind Namen von Objekten grundsätzlich Referenzen, also auch Zeiger … LinkedListItem

Literatur / Links

- Doppelt verkettete Liste
  - Wikipedia

    https://en.wikipedia.org/wiki/Linked_list
- Allozieren
  - http://www.duden.de/suchen/dudenonline/allozieren