Algorithmen und Datenstrukturen (ESE)
Entwurf, Analyse und Umsetzung von Algorithmen (IEMS)
WS 2012 / 2013

Vorlesung 4, Dienstag 13. November 2012
(Assoziative Arrays aka Maps)

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 Ü3 (O-Notation)
  - Feedback von Ihrem Tutor / Ihrer Tutorin
- Assoziative Arrays aka Maps aka = also known as
  - Nutzen anhand eines typischen Beispieles
  - Alternative dazu: Sortieren
  - **Übungsaufgabe:** die weltweit häufigsten Ortsnamen finden **...** einmal mit "Maps", und einmal mit Sortieren

Erfahrungen mit dem Ü3 (O-Notation)

UNI FREIBURG

- Zusammenfassung / Auszüge Stand 13. November 15:30
  - War für die meisten machbar, auch zeitlich
  - Beweise genau zu kriegen war nichts so einfach / klar
  - Wann ist ein Beweis genau genug?
  - Triviale Sachen zu beweisen ist doof
  - Wie zeigt man sowas wie $\log n < n$? z.B. Kurvendiskussion
  - Rechnen mit Logarithmus ungewohnt / schwierig ... eben
  - Bei Aufgabe 3, $b < 1$ Energieerhaltungssatz verletzt .. genau
  - Hoffentlich jetzt wieder mehr Programmieren ... aber ja doch
  - Rudi, Bambi, Igitt, Kaba, Desi, Nie, Kartoffelpü, Erbsenpü, ...
  - Ist das Klausurr-elefant?

Feedback von Ihrem/r Tutor/in

UNI FREIBURG

- Machen Sie dazu svn update
  - ... spätestens bevor Sie ein neues Übungsblatt bearbeiten
  - Sie bekommen dann (hoffentlich) eine Datei

    uebungsblatt_XY/feedback-tutor.txt
  - Dort sehen Sie unter anderem
    - warum wo wie viel Punkte abgezogen
    - Verbesserungsvorschläge für Code / Beweise
    - Antworten auf Fragen aus Ihrer erfahrungen.txt
    - Und was Ihrem/r Tutor/in noch alles so einfällt ...

Normale vs. Assoziative Arrays

UNI FREIBURG

- Normales Array
  - Zugriff auf eine (große) Anzahl von gleichartigen Elementen über einen fortlaufenden **Index**

    A_0, A_1, A_2, A_3, A_4, A_5, ...
  - Beispielanfrage: das Element mit Index 3
- Assoziatives Array
  - Zugriff auf eine (große) Anzahl von gleichartigen Elementen über einen beliebigen sog. **Schlüssel** (Key)
  - A_{Europa}, $A_{Amerika}$, A_{Asien}, A_{Afrika} , ...
  - Beispielanfrage: das Element mit Schlüssel Amerika
  - Die Schlüssel können auch (nicht fortlaufende) Zahlen sein

Und wofür braucht man das?

UNI FREIBURG

- In der Praxis ...
  - ... gibt es kaum ein größeres Programm wo man nicht irgendwo ein assoziatives Array braucht
- Unser Beispiel für diese Vorlesung
  - Eine Liste von Infos zu Ländern (von www.geonames.org)
  - Uns interessieren die Spalten 5 (Land) und 9 (Kontinent)
  - Welche Kontinente haben wie viele Länder?
  - Wir schauen uns zwei typische Lösungen dafür an
    - Mit Sortieren
    - Mit einem assoziativen Array aka Map

Lösung 1: Sortieren

UNI FREIBURG

- Idee
  - Wir sortieren die Zeilen (Spalte 5 und 9) nach Kontinent
  - Dann stehen alle Länder im selben Kontinent in einem Block hintereinander
  - Die Größe von jedem Block können wir dann zählen
  - Und dann den größten Block finden
- Nachteil
  - Sortieren braucht Zeit $\Theta(n \log n)$ und geht nicht in $O(n)$
  - Wir müssen **zweimal** über die ganzen Daten laufen
- Vorteil
  - Algorithmisch einfach
  - Kommandozeile: geht mit einfachen Unix/Linux-Befehlen

Lösung 2: Assoziatives Array / Map

UNI FREIBURG

- Idee
  - Ein assoziatives Array mit dem Kontinent als Index
  - Der Wert ist entweder einfach ein Zähler, oder die Liste aller Länder in dem Kontinent
- Vorteil
  - Laufzeit O(n)
  - Vorausgesetzt wir können in Zeit O(1) auf ein Element des Arrays zugreifen, wie bei einem normalen Array
  - Dass das geht, zeigen wir in der nächsten Vorlesung
  - Aber erstmal sollen Sie verstehen, wozu und wie man assoziative Arrays benutzt !

Map in Java und C++ 1/3

UNI FREIBURG

- In Java und C++ heißen die assoziativen Arrays **map**
  - Der Index heißt dort **key**, das Element heißt **value**
  - Eine map unterstützt u.a. die folgenden Operationen
    - **Einfügen** von Element value mit Schlüssel key
      - in Java put(key, value) ... in C++ insert(key, value)
    - **Zugriff** auf das Element mit Schlüssel key
      - in Java get(key) ... in C++ operator[](key)
    - **Löschen** des Elementes mit Schlüssel key
      - in Java remove(key) ... in C++ erase(key)
    - **Fragen** ob Element mit Schlüssel key da ist
      - in Java containsKey(key) ... in C++ count(key)

Map in Java und C++ 2/3

UNI FREIBURG

- Effizienz
  - Hängt von der Implementierung ab, es gibt insbesondere
    - In Java: java.util.HashMap und java.util.TreeMap
    - In C++: __gnu_cxx::hash_map und std::map
      - in C++11 ist die hash map std::unordered_map
  - Was das genau ist, sehen wir in der nächsten Vorlesung
    - da bauen wir uns unsere eigene **map**
    - und analysieren sie dann

Map in Java und C++ 3/3

UNI FREIBURG

- Vorsicht bei folgendem Feature der map in C++
  - Nehmen wir an, wir haben eine map mit Schlüsseln vom Typ std::string und Elementen vom Typ int

    ```
    std::map<std::string, int> M;
    ```

  - Dann kann man auf Elemente einfach mit dem [] Operator zugreifen wie bei einem normalen Array

    ```
    M["europe"] = 54;
    M["asia"] = 52;
    ```

  - Aber Vorsicht, wenn das Element vorher nicht in der map war, wird es durch [] angelegt, mit dem Defaultwert für den Typ von value, für int ist das 0.

    ```
    if (M["venus"] > 0) ...          // Inserts "venus" with value 0.
    if (M.count("venus") > 0) ...    // Only asks if "venus" is there.
    ```

Literatur / Links

UNI FREIBURG

- Assoziative Arrays
  - In Mehlhorn/Sanders:

    4 Hash Tables and Associative Arrays (das führt schon weiter)
  - In Cormen/Leiserson/Rivest

    12 Hash Tables (das ebenso)
  - In Wikipedia

    http://de.wikipedia.org/wiki/Assoziatives_Array

    http://en.wikipedia.org/wiki/Associative_array
- Map in Java und in C++

  http://download.oracle.com/javase/1.4.2/docs/api/java/util/Map.html
  - und ...HashMap bzw. ...TreeMap

  http://www.cplusplus.com/reference/stl/map/