i206: Lecture 15: Review for Test

Tapan Parikh

Spring 2013

Some slides courtesy Marti Hearst, John Chuang and others

Preparing for Exam 2

- Make a table or some other representation showing:
  - Each data structure we've covered
  - Its running time for insertion, deletion, searching for an item, traversing or visiting all items.
  - What are some of the trade-offs between different data structures.
  - Pseudocode for key algorithms.

Topics for Exam 2

- Data structures:
  - (See previous slide for what to know)
    - For inserting/deleting/traversing you only need to know the ones we did in detail in class
    - You do need to know their main properties and what trade-offs between them
  - When/how to apply them to different problems
- Python and pseudo code
  - Be able to understand and modify code similar to what we've seen.
    - Lists, queues, sets, stacks, hash tables, trees.
  - Be able to create your own code or pseudo code for specific algorithms.

Choosing Data Structures

- Questions to think about:
  - Is the data static, or does it change a lot?
  - Does the data need to be sorted?
    - Do you need the full range of values?
    - Or just the largest (smallest)?
  - Accessing data via specific keys or ranges of keys?
  - What kind of data? How large is each item? How many items?
  - How much memory is available?
  - Using distributed / parallel processing?

Algorithm / Data Structure Practice

- Assume you have a large set of data, say 1,000,000 elements. Each data element has a unique string associated with it.

- What data structure is the best for looking up any element when the only thing you know about the element is the string associated with it?

- Show pseudo code to implement this.

Algorithm / Data Structure Practice

- Say you want to design a program to represent a textbook. The book is subdivided into parts, chapter, sections, and paragraphs.
- What are some different ways that you might want to access the content of this book?
- What data structure(s) and algorithm(s) would you use to support this?
- Show pseudo code to implement this.

Algorithm / Data Structure Practice

- Name a problem for which a binary search tree would be useful.

Algorithm / Data Structure Practice

- Say you are designing a map program. It has a huge number of points representing geographic location positions, where those points are entered once and don't change often.
- What data structure and algorithm(s) would you use to compute the number of points that are within distance D of a given point?
- Show pseudo code to implement this.

Algorithm / Data Structure Practice

- Often for a given major, a student has to take some required courses. Some of those courses also have required prerequisites.
- What data structure would you use to represent these courses to devise an algorithm to find a sequence of courses that allows the student to complete her requirements?
- A specific example:
  - Judy needs to take i15, i16, i22, i31, i32, i126, i127, i141, and i169. Prerequisites are:
    - i15: none
    - i16: i15
    - i22: none
    - i31: i15
    - i32: i16, i31
    - I126: i22, i32
    - I127: i16
    - I141: i22, i16
    - I169: i32

Algorithm / Coding Practice

- Write a recursive algorithm to compute the product of two numbers (m*n) using only addition.
- Write a pseudocode to perform Boolean operations on three sets of data. (union, intersection)

Algorithm / Coding Practice

- Use a list comprehension to multiply the elements of two lists x and y together (multiply the first by the first, second by second, etc), yielding a list.
- Use a list comprehension to make a list of possible pairs from the elements of two lists x and y.