CSC207.01 2014S, Class 40: Implementing Dictionaries with Binary Search Trees

Overview

- Preliminaries.
  - Upcoming work.
  - Admin.
  - Questions.
- About binary search trees.
- Basic analysis.
- Lab.

Preliminaries

Admin

- Printed makeups due NOW!
- No writeup today.
- Project, phase 1, due next Wednesday.
- Fix the following problem for the rest of the semester and earn $100 from SamR.

```
$ ssh home w
09:57:48 up 1004 days, 23:53,  3 users,  load average: 15.15, 10.17, 5.94
```

Extra Credit

- http://www.strikingly.com/pioneerweekend
- Iowater projects this weekend, next weekend, or the following weekend. Mail iowater@grinnell.edu for details.
  - Saturday, April 12, 9-11am, clean up little bear creek
  - Saturday, April 12, 1-4pm, clean up Arbor lake
- Get and wear one of the "1 in 4" shirts next week.
- CS table today: Lambda in Java 8.
- CS extra next Monday: Walker and Liberto on bluetooth.
- CS extra next Thursday: Charlie Eddy on Kinect and more.

About the Project

- Parse JSON (used for RESTful services)
  - Numbers look like numbers, with optional e. 3e10.
  - Strings look like strings: "Sam says \"Strings look like Strings.\""
  - Arrays/lists: [val,val,val]a
  - Objects: {"class":"csc151","mentor":{"lname":"Wheelie","fname":"Earnie"}}
- Do it better!
- Put it out there for people to use.
- Pitch it.
  - Grade prizes.
  - Maybe other prizes.
- You can use Java's Hashtable (or HashMap or AssociationList) objects, even if you don't yet know how they are implemented.

About binary search trees

- Implementing dictionaries
- We have three implementations so far
  - Association list: O(n) put, O(n) get, O(n) remove
  - Sorted array: O(n) put, O(logn) get, O(n) remove
  - Skip lists: O(logn) put, O(logn) get, O(logn) remove
- So, how do we do better?
  - Often, divide and conquer
- Apply divide and conquer to data structure design, rather than algorithm design.
- Suppose you have a collection of key/value pairs that represent a dictionary
  - Divide into elements with "small" keys and elements with "large" keys.
  - And do that for each subset
  - Put the "pivot" between the two sets
- To look something up, recurse down the strcutrue.
- To put something, recurse down the structure and insert in place.
- To remove, punt
- How long does it take to find the appropriate place? O(number of levels)
- If we build the tree right, that's O(logn)
- A lot of work on balancing trees
  - A CSC 301 topic

Lab