CSCI 2270

Data Structures & Algorithms

Gabe Johnson

Lecture 9 Feb 4, 2013

Sorting Algorithms

Upcoming Homework Assignment

HW #3 **Due: Friday, Feb 8**

Sorting Algorithms

This time around we'll implement three particular algorithms (quicksort, mergesort, bubblesort) and one 'mystery sort' of your choosing. The definitions and tests are all up in GitHub, but they aren't active in RG yet, since I have some things to fix there first.

Lecture Goals

1. Announcements
2. Upcoming Test 1
3. Upcoming HW on B-Trees
4. Bubblesort
5. Quicksort
6. Mergesort

Announcements

1. **Editor Fu:** Wednesday 7:30pm ECCR 105 tutorial session on emacs and possibly vim. Be there, or not.

2. **RetroGrade**: Lots of confusion about RG errors. 0/0 means syntax error; n/N (N < 15) means segfault or other unexpected program halt. *Check the verbose output,* and **always** run the program driver on your end first and debug as appropriate.

Announcements

3. **Office Hours**: I am moving my *Tuesday* office hours to 9—12 to accommodate students who have class during my current slot. My *Thursday* office hours remain the same.

Upcoming Test 1

Test 1 will be open note, but nothing electronic will be allowed. This includes iPads and laptops and brain/cloud network interfaces you bought on eBay.

It will cover *concepts* and *code*.

Concepts: pointers, recursion, comp. complexity.

Code: linked lists, binary (search) trees, sorting algos, and coding issues (design, debugging) e.g. what's a segfault?

HW 4 (after test) on B-Trees

The upcoming homework after the test will be on **B-Trees**.

This will be the toughest assignment of the semester.

The point of this assignment is to test your ability to design an implementation plan. Without a flexible strategy to implement a *very tricky* data structure, you will not get very far. This is all about teaching you the value of developing a plan, as opposed to hacking around until it works.

Sorting Algorithms

All have a common goal: take an unsorted list of integers and put them in nondecreasing order.

input: 9, 5, 4, 1, 10, 5
output: 1, 4, 5, 5, 9, 10

We *sort* so we can find things more easily, and make informed decisions about the nature of our data.

Bubble Sort

Refer to BubbleSort.pdf in the sorting homework dir.

Quick Sort

Refer to QuickSort.pdf in the sorting homework dir.

Merge Sort

Refer to MergeSort.pdf in the sorting homework dir.