- review search and big-Oh O()

Take Final exam online

- see Canvas for due date. Must submit before end of this day

Lab

- current lab was assigned in 'Finish trees; hashing; graphs'. See Canvas for due date

Review last week

- introduced sorting, and order of complexity analysis, big-Oh O()
- reviewed three classes of sort algorithms, with three examples of each
  - appreciate how the algorithms work
  - don't need to memorize implementations

Introduction to this week

- our last topic, review search
  - review the search algorithms we've seen already
  - consider big-Oh O() for search

Review search and big-Oh O()
Objective: quickly review the different searches we've seen, and their order of complexity big-Oh O()

Review the searches we've covered

- the simplest search is sequential, or linear search
  - start at the first item and look at each in turn, until found or end of list
- binary search is faster, but requires that data be in sorted order
  - guess the halfway point, reducing the search space by half with each comparison
  - or implement as binary search tree
- hashing is potentially the fastest search technique
  - applies a hash function to map directly from the key to a location
  - just one comparison in the best case
  - but more comparisons likely if collisions can occur

Big-Oh O() for search

- in order, fastest to slowest, as for sort algorithms:

search	O()	category
hashing	O(1)	constant time
binary	O(logN)	logarithmic
sequential	O(N)	linear

Summary

- we've already covered implementations of all of these searches

Take Final exam online

- see Canvas for due date. Must submit before end of this day

Lab

- current lab was assigned in ‘Finish trees; hashing; graphs’. See Canvas for due date