CS 61B – Summer 2005

Final exam review

Java

data hiding, encapsulation, interfaces, modularity, packages
subclassing – why? how?
exceptions and subclassing – how does it make programs better?
iterator pattern
polymorphism – code that supports future growth

algorithm analysis

why?

Prove or disprove:

$T(N) = 3N + 12$ is $O(1)$
$T(N) = 3N + 12$ is $O(N)$

Lists

linked data structures, running times (sorted, unsorted)
binary search. # comparisons it takes

Sorting

algorithms:

insert
shell
quick
merge
heap

using data structures (insert N times to sorted linked list, then call dequeue() N times)

Stacks and Queues

implementation, representation
S & Q and trees – BFS, DFS

Hash Tables

Overview of how they work. Avg bucket chain length—how to calculate. How does this affect running time? Relationship to lambda

Trees & Traversals

In-order, pre-order, post-order, level-order.

BST property. How to maintain despite insertions and removals?

Balanced trees – AVL trees

What is AVL property? Why is it good? How to maintain despite insertions? (Only handle case where one node is out of balance)

P Queues, Binary Heaps

Show that find operations in heaps take O(log N) time since it is a complete tree

The heap property. How to maintain despite insertions and deletions? Complete trees. How to keep a heap in an array.

Graphs

The datatype. G = (V, E) Edges and how they are represented.

Shortest path
- BFS – how to do it with a queue
- Dijkstra – how to do it with a P-queue
- Bellman-ford – how to do it with a queue

DAGs (Directed acyclic graphs)
- Topological sort – how to do it.
- Linear algorithm to find shortest paths in DAG
- Given appropriate graph, find earliest finishing times, latest finishing times, and slack time

Go to guest lecture tomorrow.