Assignment 8
Due: Wednesday, October 30
9 pts (3 pts of Bonus available)

1. (6 pts total) Skyline Problem

A. (3 pts) Write a function that takes in a list of buildings and forms a skyline by adding one building at a time. Note buildings have the following notation: (L,H,R), were L is the left x-coordinate of a building, H is a height, and R is the right x-coordinate of a building. Show that this implementation is in linear time; justify your estimated run-time (Word/PDF). (See top of page 117 for a hint.) (Code; LASTNAME_8.1A.py)

We loop through building list and then the connections. Then we loop through heights so that will give us O(n^2 + 2n) and we drop the 2n so we are left with O(n^2) (quadratic).

B. (3 pts) Write a function that creates a skyline from a list of buildings taking a divide and conquer approach. Show that this implementation has a loglinear running time; justify your estimated run-time (Word/PDF). (Exercise 6-1) (Code; LASTNAME_8.1B.py)

This would be loglinear time because by using divide and conquer you are continuously cutting n in half and n can only be cut in half log n times. And then when we iterate through the list that multiplies the log n portion resulting in O(n log n).

2. (3 pts) Quicksort. Rewrite quicksort such that the pivot is selected at random. (Exercise 6-13) Using the same set of input, list the number of times the data has to be partitioned and the number of recursive calls using the randomly selected pivot value and if pivot is selected as the first element in the data to be sorted. For instance, say I have 5 cards A, 2, 7, K, and J. With the randomly selected pivot approach, I randomly select J. With the first-element-is-my-pivot approach, I would select A.

You will turn in code for both the randomly selected pivot and the "first-element-is-my-pivot" functions. Write up your comparison in a Word/PDF. List your original data set and the number of calls as previously mentioned. (Code; LASTNAME_8.2.py)

So my data set was [0, 7, 3, 12, 1, 32, 56, 23, 4]. When I chose the first element as my pivot it partitioned 6 times. When I partitioned using a random element the results varied, I got anywhere from 5 partitions to 7, I have a feeling that if there were more elements in the list then the difference would be much more noticeable. By picking a random pivot, the pivot is more likely to be in the middle so it is generally a better idea to use a random pivot than the first index.

BONUS PROBLEM

3. Binary Search Trees

A. (2 pts) How would you delete a node from a binary search tree. (Exercise 6-5) (Code; LASTNAME_8.3A.py)

B. (1 pt) Let's say you insert n random values into an initially empty binary search tree. What would, on average, be the depth of the leftmost (that is, smallest) node? (Exercise 6-6)

The chance of each newly added node to the tree being the minimum is 1/the number of nodes that have been added. If it is the depth is increased by one. Therefore the depth is 1 + 1/2 + … 1/n. Also known as the harmonic number of n.