Assignment 2 Part 1

List 3 reasons why asymptotic analysis may be misleading.

1) The asymptotic analysis of an algorithm is just a generalization in terms of how it will perform. It could be misleading when you're running the algorithm through different programming languages where slight differences in the implementation could alter the runtime for better or worse.

2) Different machines could also play into a misleading analysis, if you run an algorithm on an old machine it will take a lot longer than if you ran it through a High Performance machine.

3) Along with machine differences there is bound to be differences between implementations. Just becau an algorithm is $O(\log n)$ doesnt mean that every time you use different implementations of the algorithm you can be altering the $O(\log n)$ analysis.

How long would it take to find an element in a search tree with 10,000 elements?

- We know that the time complexity for searching in a BST is $\Theta(\log(n))$ so if we have a tree of 1000 elements and it takes 5 seconds to find the element we need to solve for the intermediate value. $5 = x \cdot \log_2(1000) \approx .5$ now that we have x we can replace 5 with t and solve for time $t = .5 \cdot \log(10{,}000)$ which gives us $t \approx 6.6$ seconds

It actually takes 100 seconds, list 3 reasons why the two numbers differ.

- Balance throughout the tree with 10,000 elements could be different than the tree with 1,000 elements
- Referring back to the first question, the asymptotic analysis could have been misleading, or, ran on two different machines
- Implementation would be a big reason the times are so vastly different with only two trials of 1,000 and 10,000 we can not tell if the current implementation is the most effective available.