Assignment 19: quicksort and quickmedian Due in class Monday, 2/4

January 31, 2019
CS: DS&A
Proof School

Coding Part. (Please submit via Dropbox, with the correct filename, as usual.)

1. Code up the quicksort routine, in a function called `quicksort`. Your function should take three arguments: a list, a start index, and an end index. (But a user should be able to call the function on just a list, and have the function supply the correct initial start and end indices.) The function should not return anything, but should modify the list in-place, and produce a sorted list according to the standard quicksort algorithm. Use the class separation algorithm we talked about for separating into left and right sets. You shouldn't need any additional space.

2. Code up the quickmedian routine, in a function called `quickmedian`. Your function should take four arguments: a list, a start index, an end index, and a value of k, and should return the kth smallest element of the list. (A user should be able to call the function on just a list, and have the function return the median of the list, i.e. the $\lceil n/2 \rceil$th smallest element.) The function can (and should) alter the list in-place during the call.

Non-Coding Part. (Please submit in class.)

Suppose that you are given access to a function `magic_median` that takes a list and returns its median in $\Theta(n)$ time. (Of course, the algorithm we described in the last ten minutes of class on Thursday *is* such an algorithm, although we haven't proved it yet!) Suppose, however, that `magic_median(input)` only works on lists with non-duplicates; it gives an error if given a list with a duplicated element.

Treat the function `magic_median` as a black-box–you have no idea how it works, but you can use it. Describe how to write a function that gives the median of any list, including lists with duplicates, in $\Theta(n)$ time. Your function, of course, should call `magic_median` at a crucial moment.