CS121 - Computer Science II
Lab Assignment #9
Fall 2011

The purpose of this exercise is to observe the behavior of several of the search algorithms we discussed in class, using a technique called program instrumentation. This involves adding code that counts occurrences of certain code sequences that characterize the algorithm implemented in the function. In the case of searching, the usual measure that is instrumented is the number of key comparisons performed during the search.

For this assignment, you are to write functions for the linear, binary, and hash searches, and instrument them to count key comparisons. Use arrays (ie, contiguous storage) for these functions. The keys can be `ints`. You can use the functions directly from the book if you wish, modified as necessary to include the instrumentation. Since the behavior changes depending on whether the key value is found or not, your instrumentation should keep separate track of the number of comparisons required if the key value is found, and if the value is not found.

For the linear search, the average number of comparisons should be around $\frac{n}{2}$ for a value that is found in the list, and n for a value that is not found. Verify that your results match this expectation.

For the binary search, the average comparisons should be $O(\log n)$. The book presents two variants of the binary search - one that always goes through the entire search, even when the value is found early, and another that checks to see if the value has been found (which requires a additional key comparison during each pass) and terminates early if found. Try both algorithms and see which one performs better.

The hash search is of order a constant for a sparse list, but approaches $O(n)$ as the search list becomes more full. Verify that this is the case, by using hash arrays varying in size from 50% of the search list up to 95%, in increments of 5% (that is, try a hash array of 50%, then 55%, etc.).

For this assignment you will need both a list to search and a list of keys to find within the list. You can generate these lists using a random number generator, and either a separate program to generate the lists, or one or more "utility" functions called from your main program to generate the lists.

Use lists that are large enough so that your results are meaningful.

For submission, you should supply the code necessary to actually execute the search functions - this can be either a single main program that calls all of your search functions, or separate main program for each function. In addition to your code, you should include a short report on your findings that discusses the results you found, including any discrepancies between what you expected and what you observed. Submit all of this in a single zip or tar file using `cscheckin`.