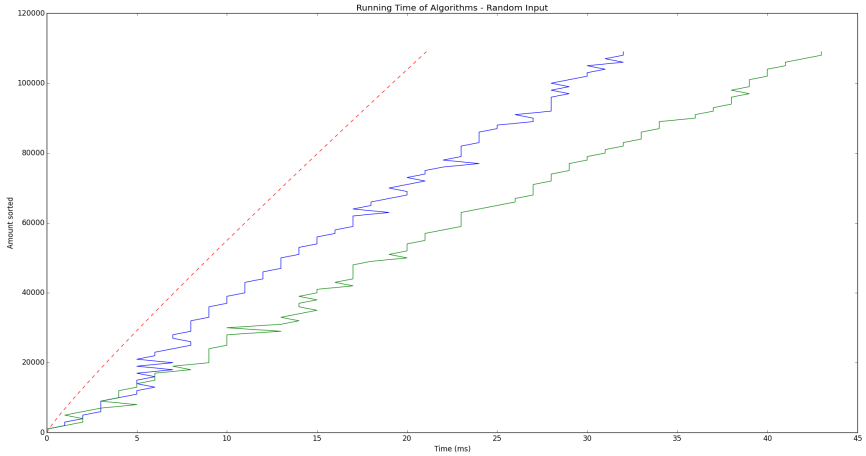
COMP20007 Design of Algorithms. Assignment 1 Report.

Luther Carroll. 391929. luthercarroll@runbox.com

This report will discuss the performance of my Quicksort and Mergesort implementations. In the following graphs: *nlog(n)* is red and dashed, *n^2* is red, Quicksort is blue, Mergesort is green. *nlog (n)* is calculated with *nlog(n)/6000,* with a natural logarithm and *n^2* is calculated with *(n^2)/6000*. This yields estimations of the shape of these functions, but is of course not an accurate conversion from instructions to milliseconds.

Figure 1: Running times of the algorithms with random input.



As expected, both Quicksort and Mergesort take *O(nlog(n))* time and Quicksort is marginally faster than Mergesort.

For testing with pathological input, sorted input and input with one unique, repeated number (namely one) were used.

Figure 2: Running times of the algorithms with sorted input.

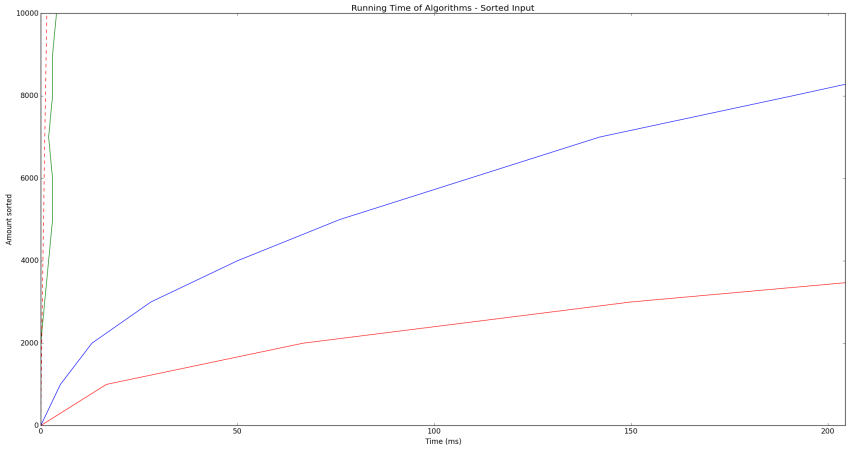
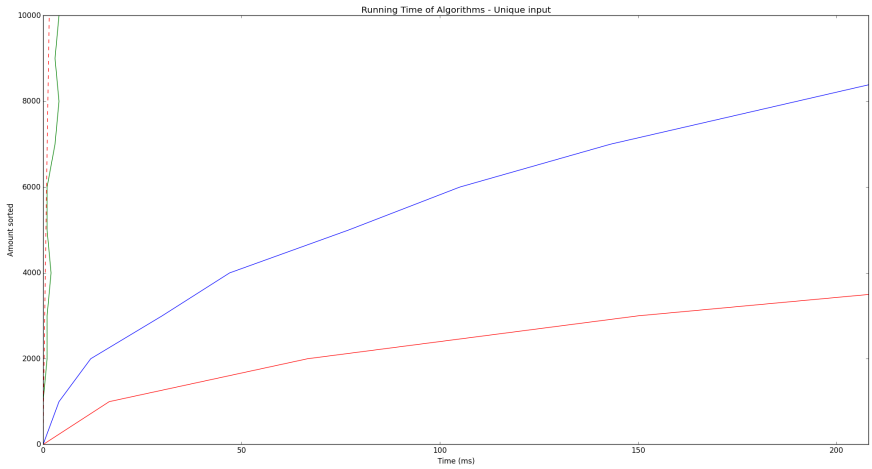


Figure 3: Running times of the algorithms with unique input.



Again, the run times of the algorithms matches the expected complexities. Quicksort takes *O(n^2)* time for sorted and unique input, while Mergesort only takes *O(nlog(n)).* My Quicksort does not use a pivoting strategy to escape the slow performance under sorted input - it chooses the first element of the list as the pivot.

The Python 3 script used to conduct these tests - *tests.py* - is in the *tests* directory. It requires the external library *matplotlib* to function.

Thank you.