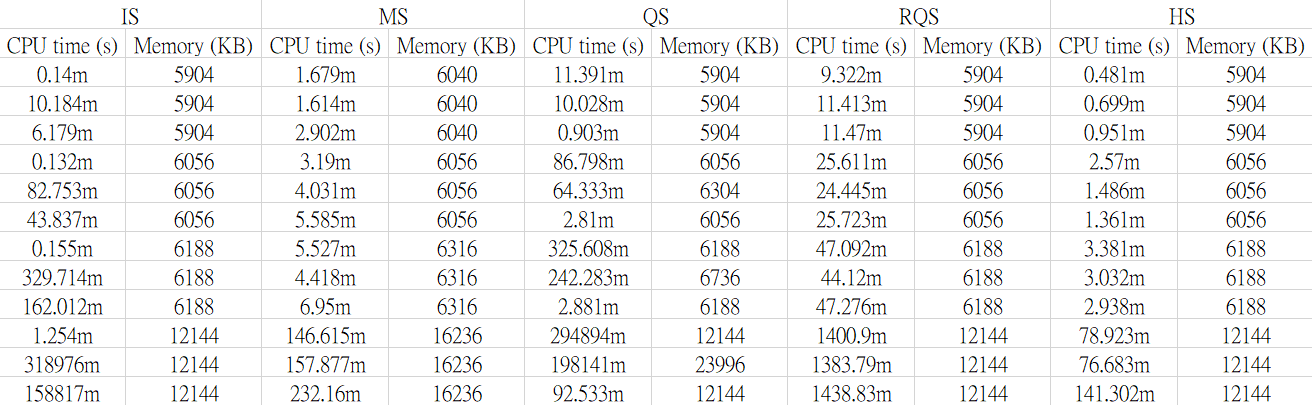
**Algorithms:**

**Programming Assignment 1 Report**

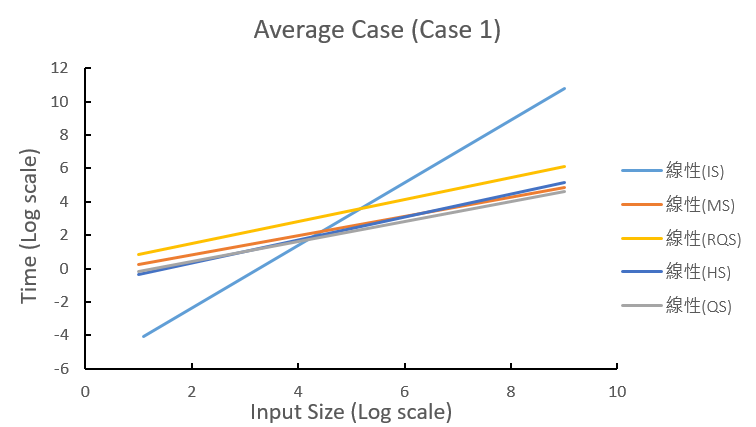
B10901006 電機三 鄒昀樺

1. Performance of Algorithms with Different Input Sizes

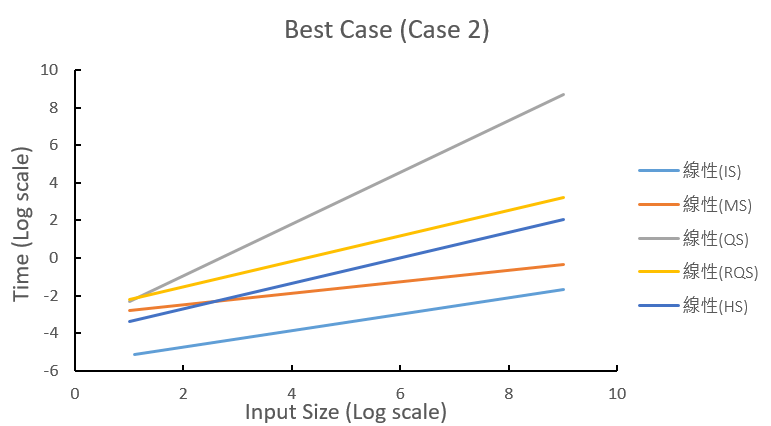
* Data is run on the EDA union lab machines



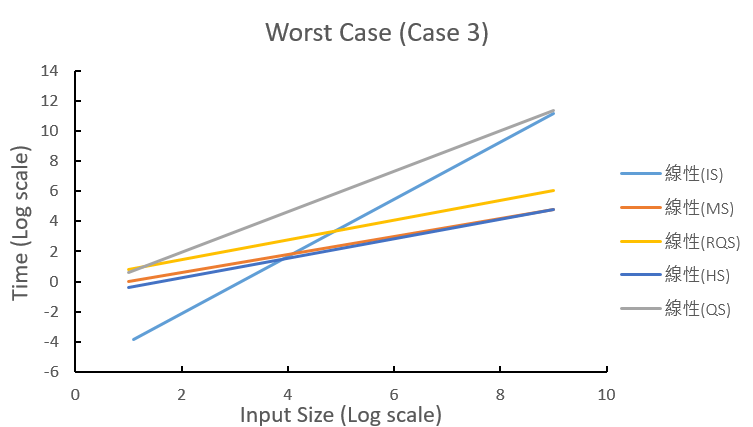
2. Growth of Run Time



In the average case, insertion sort has the least performance while the other four algorithms have similar performance. The slope of the insertion sort is about 1.88 ≈ 2. This corresponds to the run time conclusion of insertion sort made in class, which is O(n2). Other lines have slope about 0.6 to 0.7, corresponding to O(n∙lgn) after applying log.



As for the best case, Quicksort has shown the worst performance as the input size grows. Its slope is about 1.4, while others’ are between 0.3-0.6. The possible reason behind this could be that Quicksort always uses the first element as its pivot. For the best case which the elements have already been arranged, the array would always be cut into 1 vs. (n-1). Therefore, the run time would be O(n2), which is 2 (1.4 in this case) of the slope in the graph. Randomized Quick Sort has solved the problem by choosing the pivot randomly, resulting run time of O(n∙lgn). Insertion sort has the best performance because the array has already been sorted, so no insertion is needed in the process. As for slopes of Mergesort and Heapsort (which is under 1), they are simply the outcome of O(n∙lgn).



Finally in the worst case, both insertion sort and Quicksort have the worst performance. The reason for Quicksort is same as the above case 2, while the reason for insertion sort is same as the above case 1. Both are from the run time O(n2). Others have the same performance as the outcome of O(n∙lgn).