**DSA(2) Program Demo Homework 1**

Sorting Algorithms Practice and Discussion

從Insertion Sort可以得出資料愈大時，所耗的時間會趨近指數性的成長，因此當處理資料量大時，Heap or Quick Sort會是較好的選擇，但是當資料量小時，Insertion Sort的處理效率反而表現優異，有時甚至比其他更好，這是因為資料量小時，Insertion Sort僅需從尾部循序比對，但平均情況下並不用比完全程即可插入，所以會相比Heap先Create Heap或Quick移植找pivot來的快。

上圖中因為Insertion Sort與其他兩個的差距太大，於是我們將Heap Sort and Quick Sort拉出來比較，可以看到Heap Sort所耗費的時間明顯比Quick Sort大，趨勢線也是一樣，尤其資料愈大時，Quick Sort的處理效率會愈好

可以看到圖中Insertion Sort明顯較其他二者耗費較多時間，且從趨勢線可以看出當資料量愈大時，Insertion Sort 與其他兩個排序演算法的時間差距愈大，且趨近於指數成長(O(n2))。可是資料量小時(在此次測試時為資料<100項時)，效率反而與能與其他二者匹敵，有時甚至更好。

Heap Sort and Quick Sort則在資料量大時有較好的處理效能，且雖然二者的趨勢線目前都成指數性成長，但是在跟Insertion Sort之後可以發現其實也十分平緩，因為他們的效率為O(n log n)所以會比Insertion Sort的O(n2)好。

Heap使用Max Heap的結構來加速排序，及當完成Heap Tree時，僅須找到root即可吝由大排到小，而使用Adjust()可以使她在創建Heap Tree時只用O(n)，而移除root後找出新root也僅和樹的深度有關，即O(log n)，而排序n項即最後成為O(n log n)。

Quick Sort 則會是最快是因為雖然他在worse case的表現不是最好，但通常遇到的都是平均情況，而他的平均情況其實趨近於最佳情況，所以能比其他二者快。再加上他訪問的資料通常都是較為鄰近的，而訪問鄰近的資料所耗費的時間會比訪問距離遠的快些許時間，所以藉著此為小差距的累積在資料量大時能有較好的表現