1. Introduction:

本次實驗需要實作泡沫排序、插入排序、快速排序、合併排序，並且需要測試他的穩定度和執行時間。

1. Implement Details:

泡沫排序: 利用兩個迴圈檢查每一個數字，並利用swap() 函式，把每一次迴圈中，最大的數字移到最右，然後不讓它參與下次比較，重複多次，即可完成排序。

插入排序: 將數列分成尚未排列過的數列和已排列過的數列，將尚未被排列的數列，一個一個地插入已排列過的數列，即可完成排序。

快速排序: 先在數列中選一個數字當作pivot，然後讓數字跟pivot 比大小，如果比pivot大，往右移，比pivot小，則往左移，當全部的數字都被檢查過後，再將pivot與 最左邊比pivot大的數交換，即可將數列分成，比pivot小和pivot大的數字，再針對這兩個數列，重複做此操作，即可完成排序。

合併排序: 將數列一分為二，分成左數列和右數列，再將左數列和右數列做同樣的操作，當每個數列只剩兩個數字時，再對其做比大小的排序，然後再將排序好的數列合併起來，然後利用先前排序的結果，加速比大小的速度，重複此操作，即可完成排序。

1. Experiments & Results:

輸入測資的方法，我採用I/O讀檔的格式，將人為誤差降至最低，並且不把測試結果印出，因為測資數量太過龐大，怕影響實際實驗結果(排序時間)。

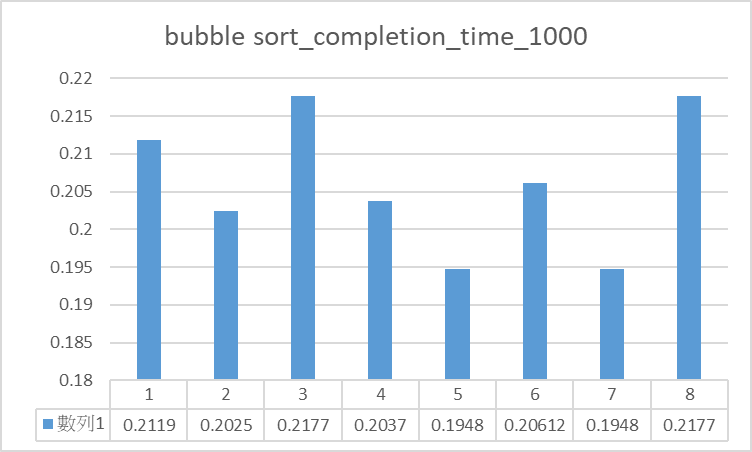
時間單位為秒，測試方法為Dev C++ 所內建的功能。

Completion time:

Bubble sort:

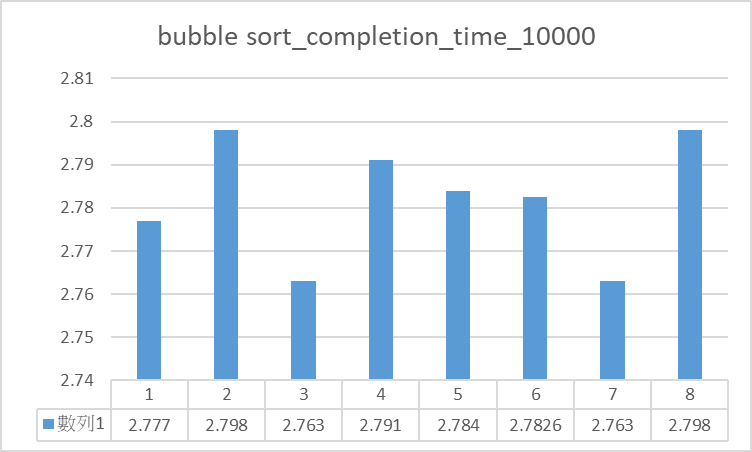
completion\_time\_1000:

1~5 為實驗結果，6 為平均值，7為最快時間，8為最慢時間



completion\_time\_10000:

1~5 為實驗結果，6 為平均值，7為最快時間，8為最慢時間



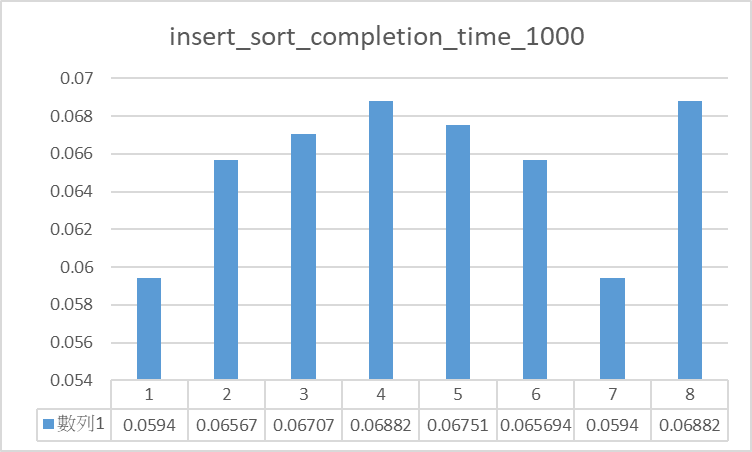
completion\_time\_10000、completion\_time\_1000000:

因為測資太大，而無法正常執行。

Insertion sort:

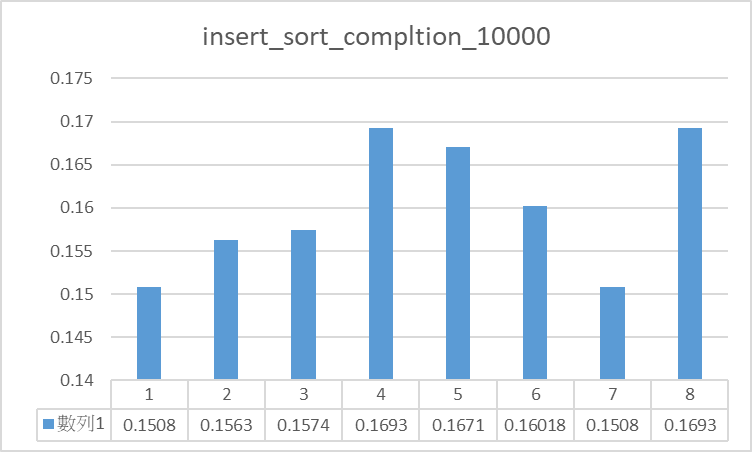
completion\_time\_1000:

1~5 為實驗結果，6 為平均值，7為最快時間，8為最慢時間



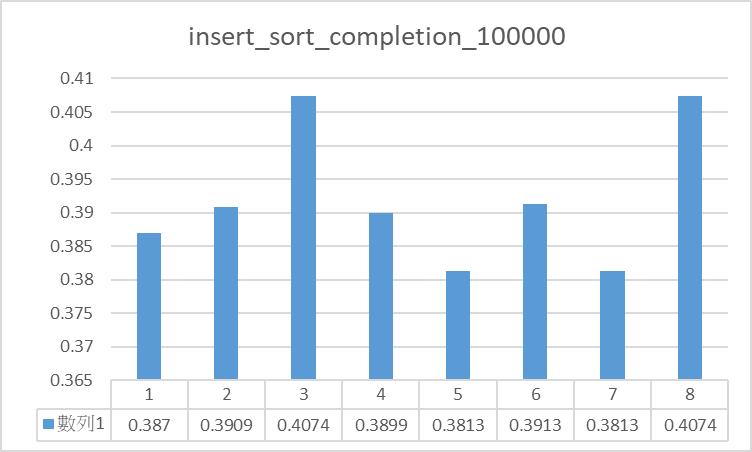
completion\_time\_10000:

1~5 為實驗結果，6 為平均值，7為最快時間，8為最慢時間



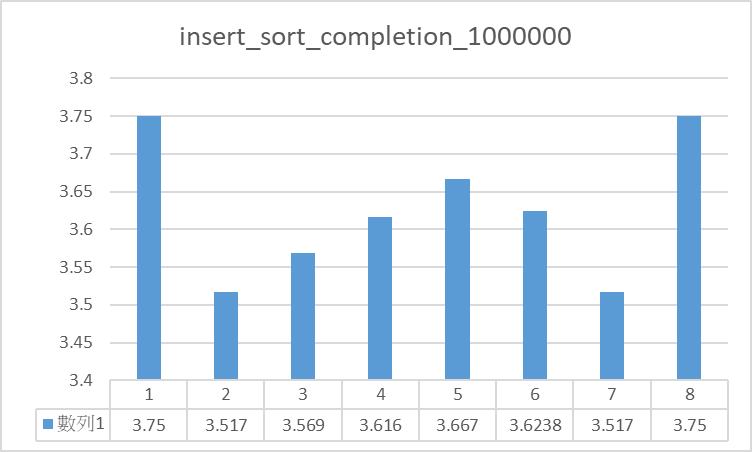
completion\_time\_100000:

1~5 為實驗結果，6 為平均值，7為最快時間，8為最慢時間



completion\_time\_1000000

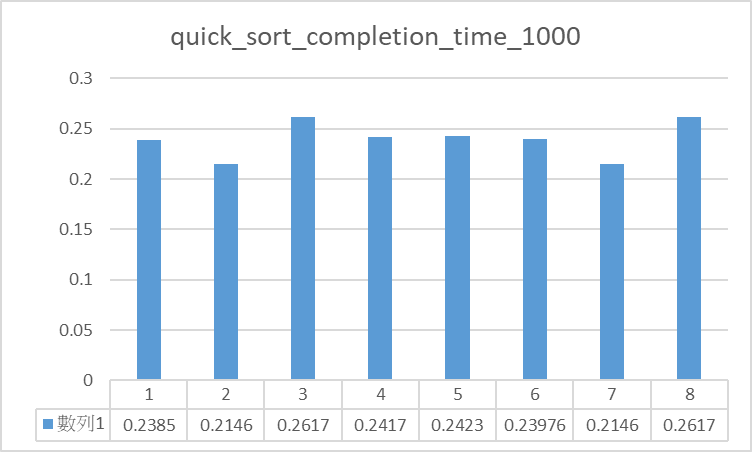
因測資太大而無法進行運算。



Quick sort:

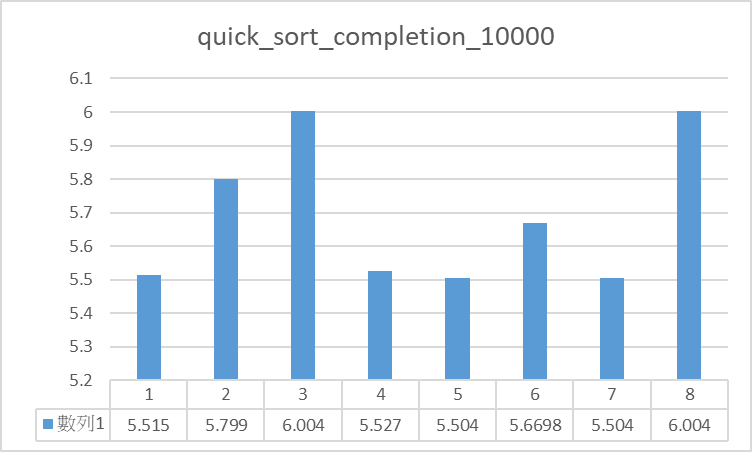
completion\_time\_1000:

1~5 為實驗結果，6 為平均值，7為最快時間，8為最慢時間



completion\_time\_10000:

1~5 為實驗結果，6 為平均值，7為最快時間，8為最慢時間

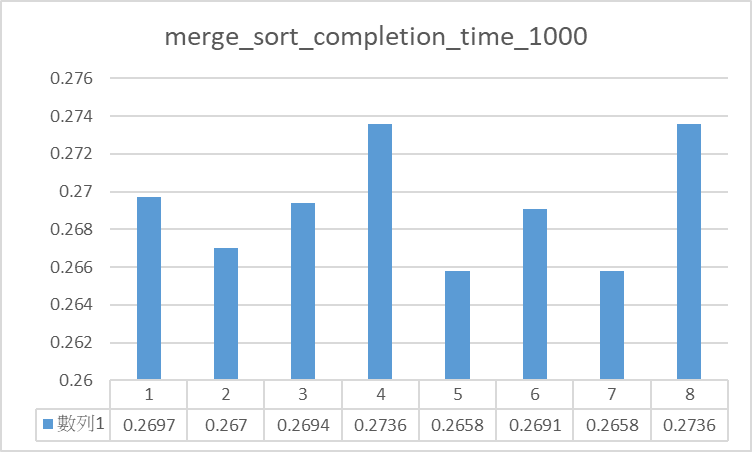


completion\_time\_100000、completion\_time\_1000000，因為測資太大，所以無法測試。

Merge sort:

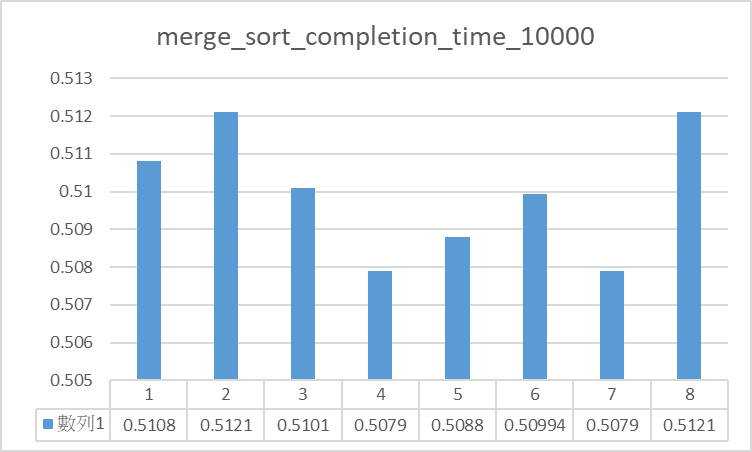
completion\_time\_1000:

1~5 為實驗結果，6 為平均值，7為最快時間，8為最慢時間



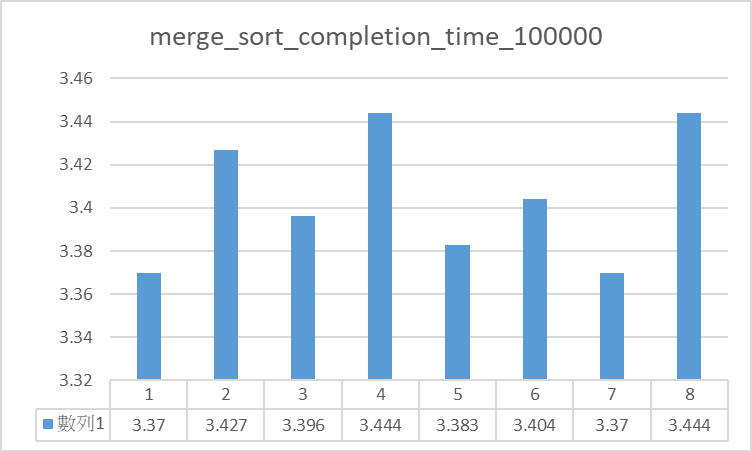
completion\_time\_10000:

1~5 為實驗結果，6 為平均值，7為最快時間，8為最慢時間



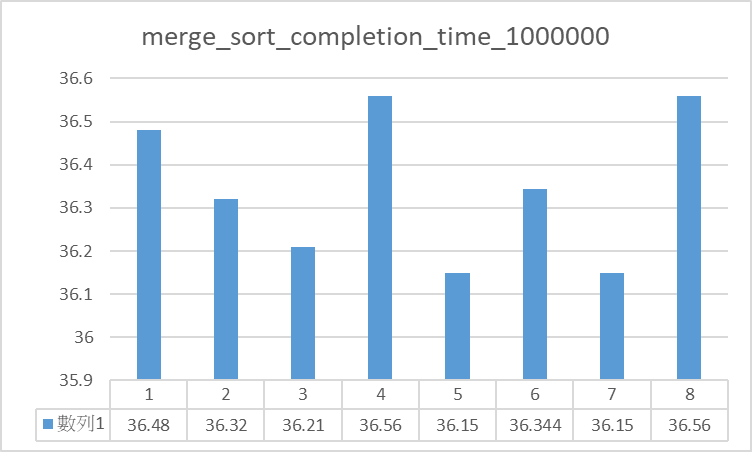
completion\_time\_100000:

1~5 為實驗結果，6 為平均值，7為最快時間，8為最慢時間



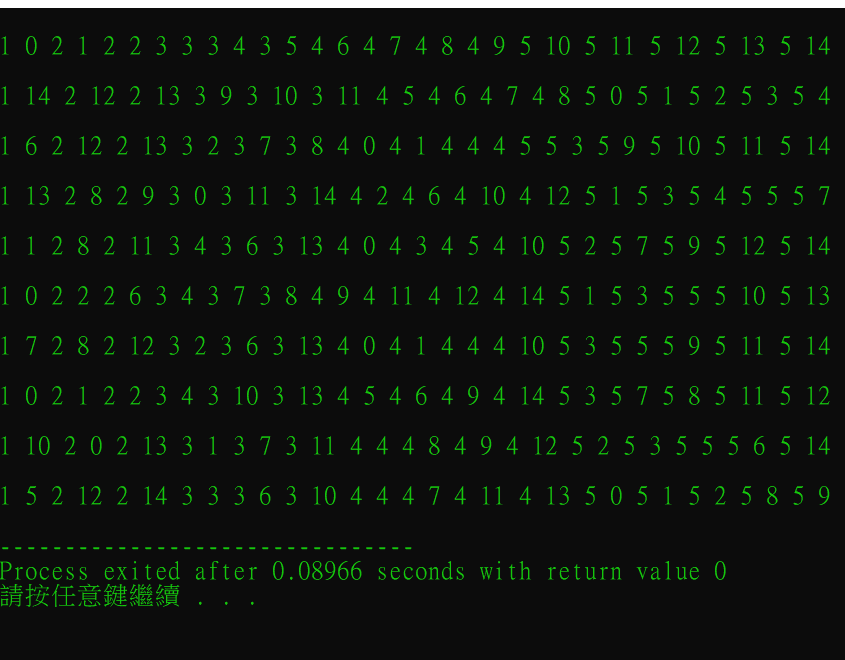
completion\_time\_1000000:

1~5 為實驗結果，6 為平均值，7為最快時間，8為最慢時間

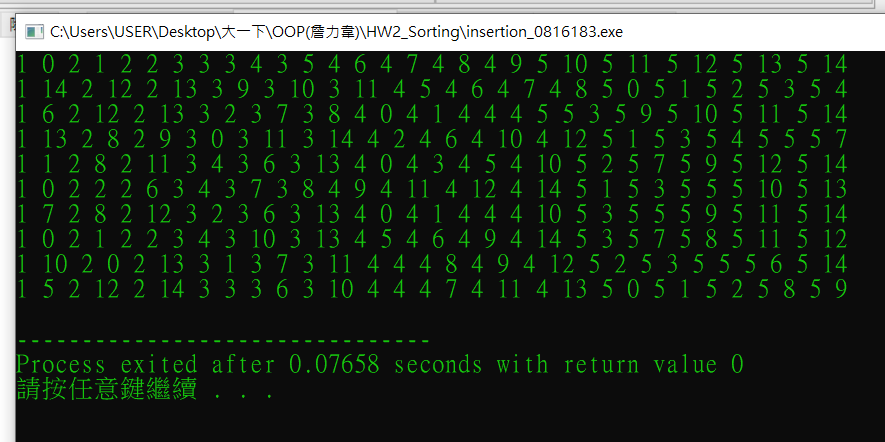


Stability:

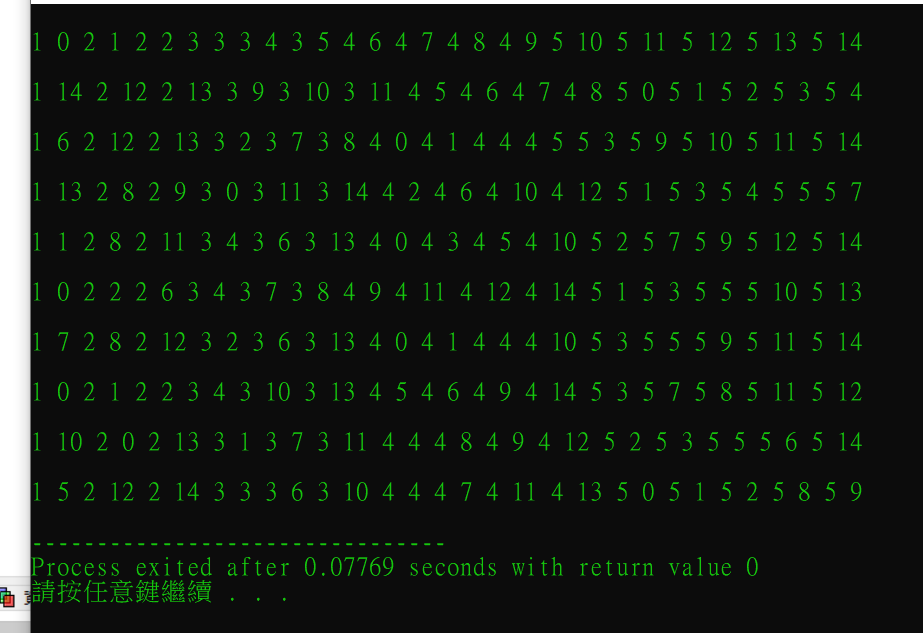
Bubble sort:



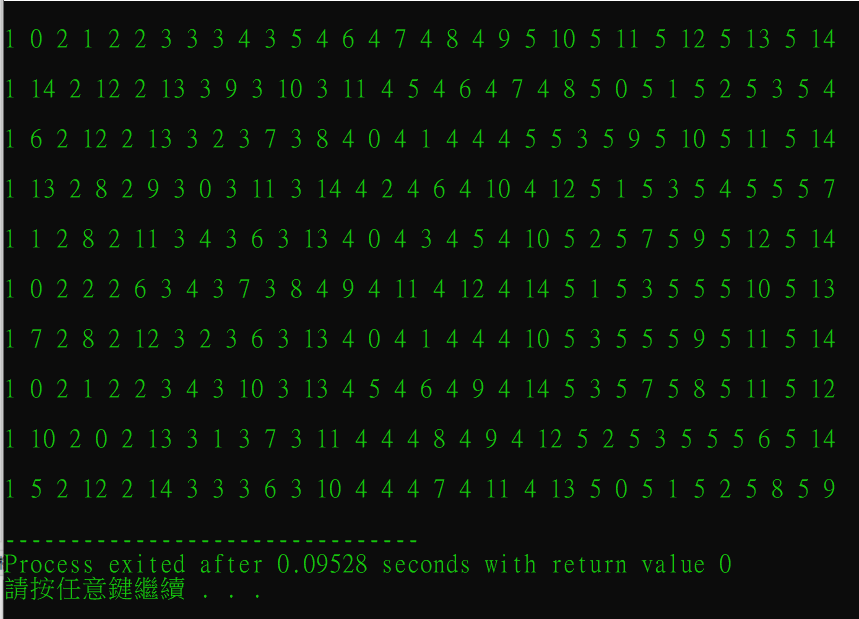
Insert sort:



Quick sort:



Merge sort:



在數字後面的排序是他原本的index，我利用structure完成實作，因為四種排序中，同一種數字的相對位置不變，故四種排序都是穩定排序。

Discussion:

實作這四種排序其實不難，但是要完成助教所提供的測資卻有些挑戰，必須要結合之前所學才能克服，至於有一個令我困惑的問題是，插入排序法竟然在某些情況下執行效率能贏過快速排序 ?

Conclusion:

這份作業幫助我結合之前所學，並且以不同的角度切入程式執行，

這是一次很新奇的體驗，可以自己確認自己程式的執行效率，希望以後有更方便的方法，達到相同的目的。。