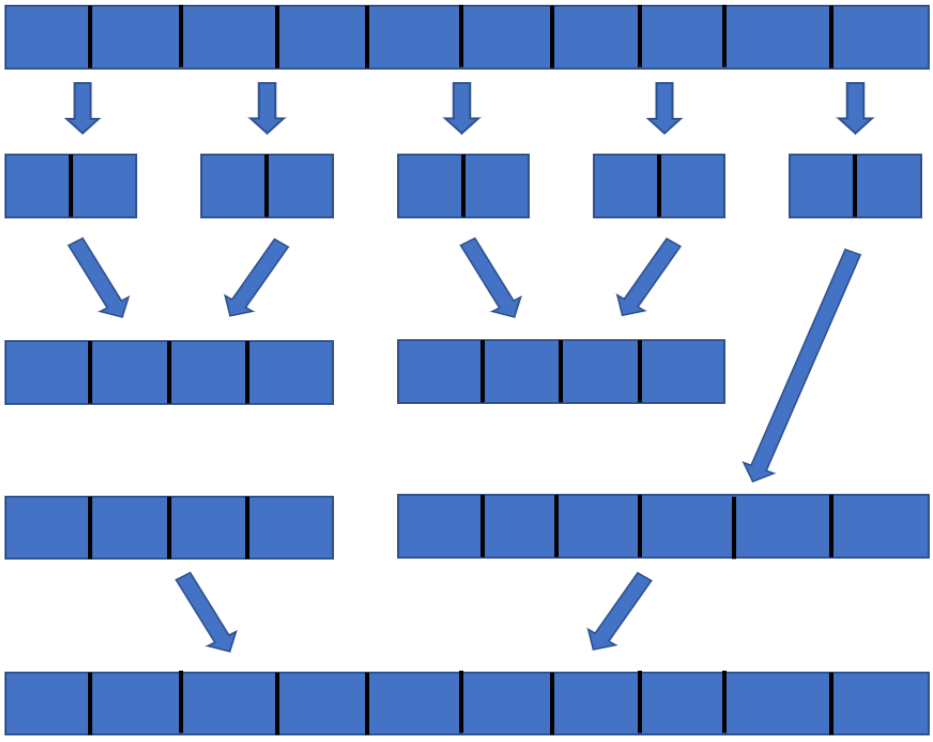
電機系 資料結構緒論 作業四

姓名：郭庭維
系級：會計 112
學號：H14086030

1. 實作並簡述(1) ~ (6) sort 的運作方式
   (1) Insertion sort：從數列前面開始排序，確保前 n 個數已經排好序後，將第 n+1 個數加入數列對應的位子。
   (2) Selection sort：掃描整個陣列把第一小的數字跟數列第一個數字交換，接著把第二小的跟數列第二個數字交換
   (3) Quick sort：
      i. 以某個數字為分界點（e.g.第一個數）
      ii. 左邊與右邊各有個 index，左邊確保每個數都小於分界點，右邊則確保都大於分界點，如果有左邊有不符合的就和也不符合右邊的交換位子。
      iii. 重複以上步驟直到左 index 在右 index 的右邊（交換位置）
      iv. 左 index 左側的所有數字都小於分界點，右側則都大於分界點
      v. 將小於與大於的部分重新做一次 quick sort
   (4) Merge sort
      i. 把數列拆成兩個數一組
      ii. 將這兩個數字排序
      iii. 與旁邊其他數組合併，合併成由小到大的數列
      iv. 持續步驟三直到所有都被合併回一個數列



(5) Heap sort

i. 把數列建構成 max heap tree

ii. Pop a[0]，把它移到末端，例如 max heap tree 長度為 8，則把 a[0]，移動到 a[8]

iii. 重新建構 max heap tree，重複步驟一二

(6) Radix sort

i. 把數列中個位數字為 0 1 2 3…依序放到名稱 0 1 2 3…的 array 暫存

ii. 依序把 0 1 2 3 Array 拿出來（個位數字是 1 的會比 2 還要前面）

iii. 重複步驟一、二 依序用 個十百千萬位數做為排序標準

iv. 執行到最高位數的後，最後排列結果即為排序後的結果

2. 紀錄每次的 CPU 運行時間並輸出

(1) 測試環境說明：

作業系統：win 10 pro

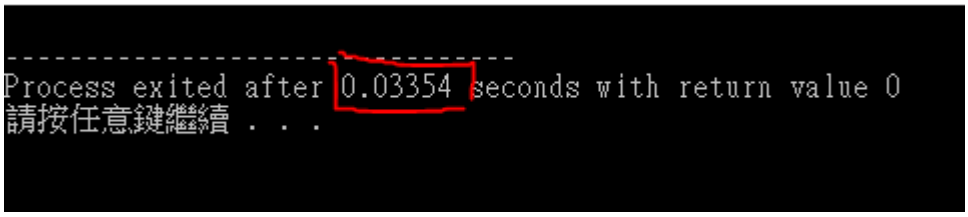
CPU：i5-9400F

(2) 測試方法：

i. 寫好各種 sort function 後，把透過其他資源建立的隨機、不重複、大小相同的十個數列放在 main 中（數字範圍為 1-200 萬）

ii. 呼叫對應的 sort function 紀錄執行十次所需要的時間

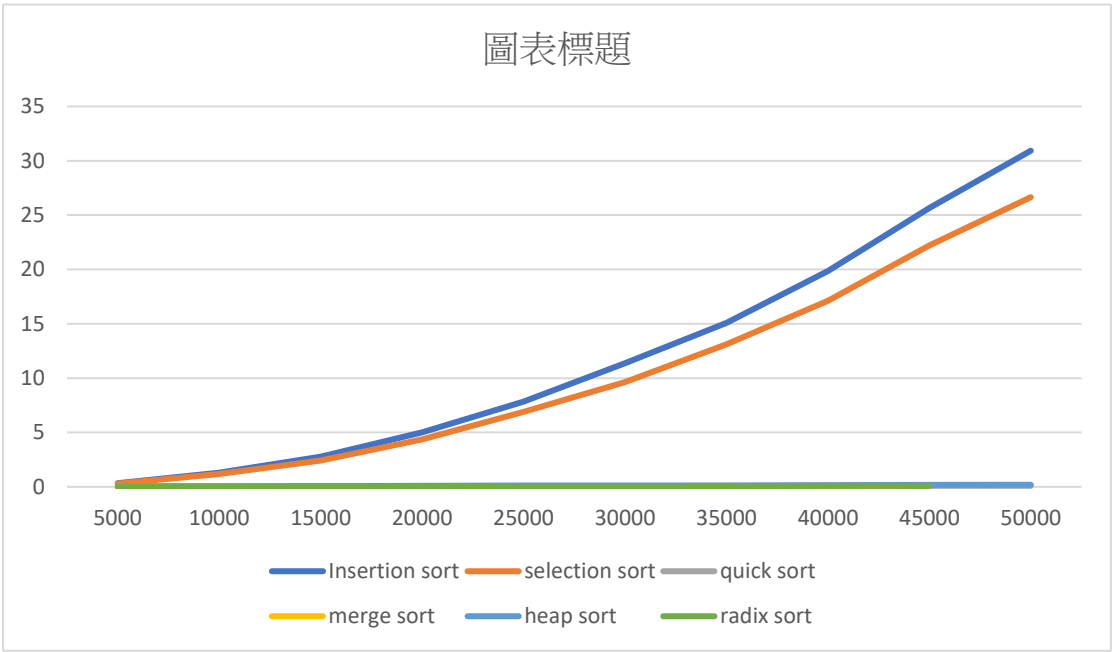
iii. 時間則使用這個時間做為執行時間，因為沒有使用 print function 印出排列後的數列，所以我相信這個時間還算精確。



(3) 執行時間統計結果如下表，備註：merge sort 與 radix 在資料量 50000 筆時會"with return value 3221225725"，推測是遞迴過多的緣故，所以資料從缺，因為認為九組數據也足夠了，因此沒有再測試下去。下面的時間都是執行十次排序的總時間，沒有另外除 10 求平均。

size	Insertion sort	selection sort	quick sort	merge sort	heap sort	radix sort
5000	0.3365	0.2942	0.02766	0.03126	0.03742	0.03023
10000	1.3	1.17	0.03299	0.03379	0.04652	0.03771
15000	2.78	2.42	0.04337	0.04779	0.06745	0.04687
20000	5.01	4.351	0.0491	0.05402	0.08545	0.05398
25000	7.83	6.912	0.05724	0.0644	0.1159	0.05706
30000	11.38	9.632	0.06739	0.07177	0.1202	0.06637
35000	15.1	13.1	0.06921	0.08348	0.1358	0.07753
40000	19.85	17.12	0.07732	0.09199	0.1619	0.08202
45000	25.66	22.23	0.08562	0.1009	0.178	0.08365
50000	30.93	26.65	0.09306		0.192	

3. 建立圖表



4. (1) ~ (6) sort 的優缺點
   (1) 從上面圖上可以看到 selection sort 跟 insertion sort 很明顯是效率最差的兩種排序方法，這也跟我們在課本上學到的複雜度$O(n^2)$是吻合的，20000 筆資料所需的時間幾乎是 10000 的 4 倍，四萬筆資料也是兩萬筆的 4 倍左右
   (2) 相較之下其他種算法對於資料量大小的影響還是有，但少了很多，資料量十倍之下，大概只有三倍左右的差距
   (3) 優缺點：
      i. Selection sort／Insertion sort：
         優點：直觀、實作容易
         缺點：但是效率最差
      ii. Quick sort：
         優點：這次幾種算法中，是第二快的排序算法
         缺點：unstable sort、worse case O(n^2)
      iii. Merge sort
         優點：stable sort，效率也不錯，可以支援 external sort
         缺點：可能會需要額外空間，這次測試中發現過大的資料會出現錯誤，可能要改成 external sort
      iv. Heap Sort
         優點：如果用 Array 來建構 heap，不需要額外空間，效率不差
         缺點：nonstable sort。效率不及其他排序法
      v. Radix sort
         優點：這次測試中，屬於效率最高的算法
         缺點：需要大量的空間來存放，而且測試過程會發現無法應付過大的資料
5. 心得與討論：

   我覺得有這次的機會可以實際測試每一種排序算法的效率是一件很有趣的事情，看到複雜度$O(n^2)$的算法跟其他排序法比起來效率簡直是天壤之別，如此具體的差異讓我以後在寫程式的時候會更加重視演算法複雜度的問題。但是比較可惜的是這次的作業時間上有點趕，加上自己同時還有 3-4 份期末報告要交沒有辦法花太多時間在處理程式上的 bug。有的排序法如果時間到沒有寫出來，卡住的部分只能先參考課本。暑假後會自己重新地把這些算法再練習一遍打好自己的基礎。

   另外看中山大學的影片時候有提到 radix sort 的效率應該要是最高的，但我自己實驗看來，在前面幾次資料量比較小的時候，quick sort 會比 radix sort 還快，但是 25,000 筆資料後，確實 radix sort 會比較快。