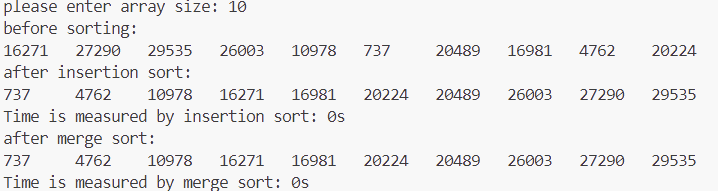
Homework 1

109610025陳品妍

執行結果(以size 10為例):

Size 10



比較insertion sort 以及merge sort之執行時間

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Size/time | 100 | 1,000 | 10,000 | 100,000 | 200,000 | 300,000 |
| Insertion sort | 0s | 0s | 0.02s | 2.5s | 10.382s | 22.717s |
| Merge sort | 0s | 0s | 0s | 0.016s | 0.034s | 0.057s |

|  |  |
| --- | --- |
|  |  |
| Size 100 | Size 1000 |
|  |  |
| Size 10000 | Size 100000 |
|  |  |
| Size 200000 | Size 300000 |

\*註一:因為merge sort的執行時間在小樣本中看不出其差異，所以多做了幾組數據來比較

\*註二:sorting 結果可供助教檢查其正確性，唯大樣本下解說放便起見先註解掉

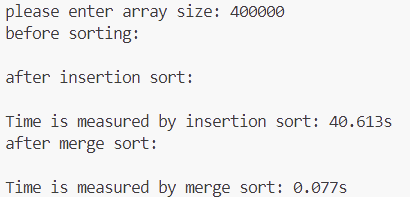
檢查insertion sort size時間複雜度:

10000以及100000:

100000以及200000:

100000以及300000:

100000以及400000:



說明:

從以上驗證可以看出的確insertion sort大致符合課堂所提到時間複雜度跟樣本的關係。其中的關係時間的比例約為樣本比例的平方。為worst case 而在n大過一個數之後，可以發現上界會越來越準確。另外還有再試n=400000的例子，可發現其越來越趨於穩定，沿著上升。

檢查merge sort size時間複雜度

100000以及200000:

100000以及300000:

100000以及400000:

說明:

與insertion sort比較之下，merge sort不像insertion sort緊貼著，而是都會比較差一點。可能是因為merge sort不是在main 函式裡，而是一直遞迴呼叫下一個函式，需要記錄離開函式的資訊，跳回來時在查找上一次離開的地方，因此執行上會比理論稍微差一點。

程式說明:

首先執行時，系統會提式輸入需要的array size大小，如果需要看到輸出結果是否正確，可以將註解消掉。