Homework1 書面報告

- 開發環境:

  Visual Code

  編譯器 Anaconda

- 實作方法與流程

1. 建立 Process & Thread 方法

   ✡ Process

   使用 python 的 multiprocessing 模組(使用函數):

   from multiprocessing import Process, Manager

   p = Process(target=函數名稱, args=引用參數) # 創建一個 Process

   p.start() # Process 啟動

   p.join() # 阻止主 Process 執行，等待子 Process 結束後再繼續往下執行
   （用於 Process 間同步）

   共用資源方法:

   使用 multiprocessing 模組的 Manager 函數:

   (在 Process 共享資料使用特殊的 list)

   lsttmp = Manager().list() #可以使全部的 Process 共用此 list

   ✡ Thread

   使用 python 的 threading 模組(使用函數):

   import threading

   p = Process(target=函數名稱, args=引用參數) # 創建一個 thread

   p.start() # thread 啟動

   p.join() #阻止主 thread 執行，等待子 thread 結束後再繼續往下執行
   （用於 thread 間同步）

   共用資源方法:

   因為多個 thread 在同一個 Process，所以 threads 會共用程式碼、資料段、處理器內所用到的資源，想將資料在不同 threads 之間傳遞，只要設一個 list 作為共享資源即可

2. Merge 方法

   將已經排序的兩筆資料(left, right)做比較，比較 left 的第一筆資料與 right 資料，將較小的數值排入另一個新的陣列，並將此數值在原陣列中刪除，持續做相同判斷直到其中一個陣列(left, right)為空，再將剩下(left, right)的加入新陣列。

3. 流程

   任務一: 讀檔->做 BubbleSort->寫檔

   任務二: 讀檔->切 K 份->做 BubbleSort->mergeSort->寫檔

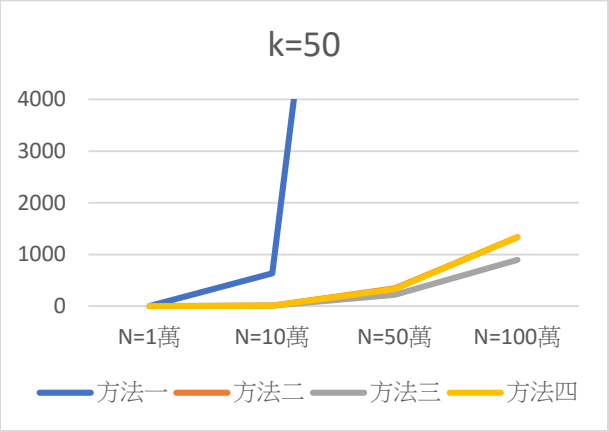
   任務三: 讀檔->切 K 份->用 K 個 Processes 做同時做 BubbleSort->用 K-1 個 Processes 做 mergeSort->寫檔

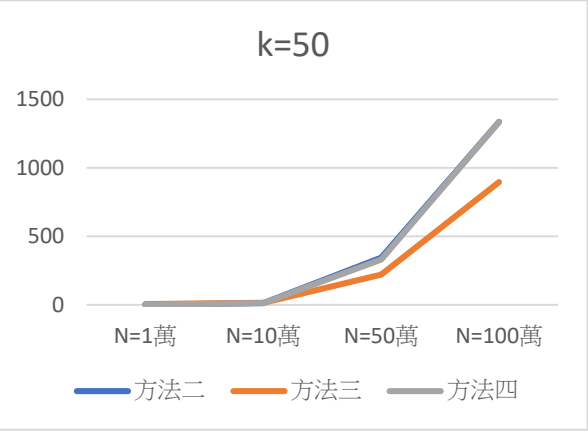
任務三: 讀檔->切 K 份->用 K 個 threads 做 BubbleSort-> 用 K-1 個 threads 做 mergeSort->寫檔

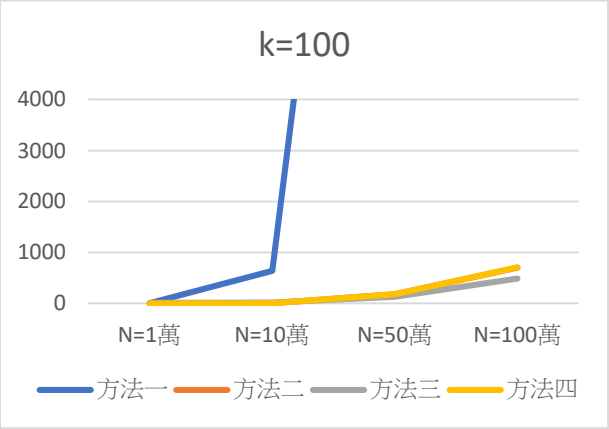
- 探討結果和原因

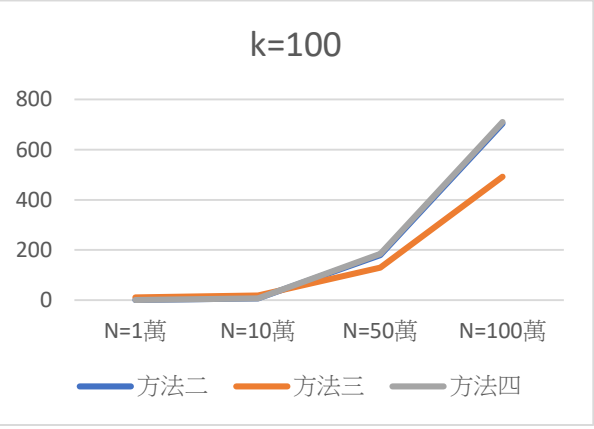
1. 不同 N 值 VS 執行時間（單位：s）

K = { 50,100,200,450 }	N = 1 萬	N = 10 萬	N = 50 萬	N = 100 萬
方法一	6.3125	638.46875	19753.54839	108,000
方法二	0.376436949	13.20214224	345.1881266	1335.072877
	0.297391415	7.011094093	179.3901484	703.7073672
	0.286216974	3.918684244	104.3911498	398.6548502
	0.270279408	2.288167	57.55715299	221.4547598
方法三	5.655389786	14.70824075	221.2755265	895.5896845
	10.58078837	18.71613312	130.1632857	492.6206949
	23.10771918	29.13227367	92.76657176	398.6453958
	51.71209359	57.7041328	93.11238265	215.0064909
方法四	0.143601894	12.70714331	330.9372067	1334.89278
	0.095906734	6.649906635	184.3409414	710.5370305
	0.088033676	3.874405861	110.1219122	401.6195512
	0.115698814	2.197887897	61.05302358	225.0186396

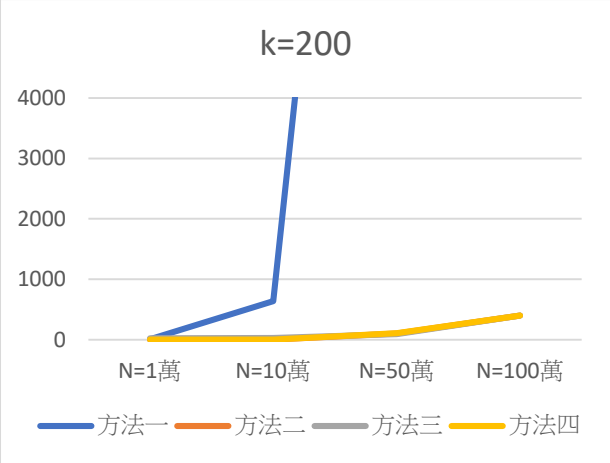


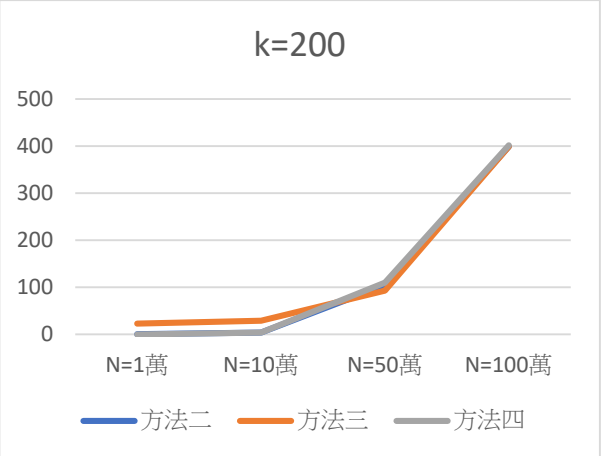


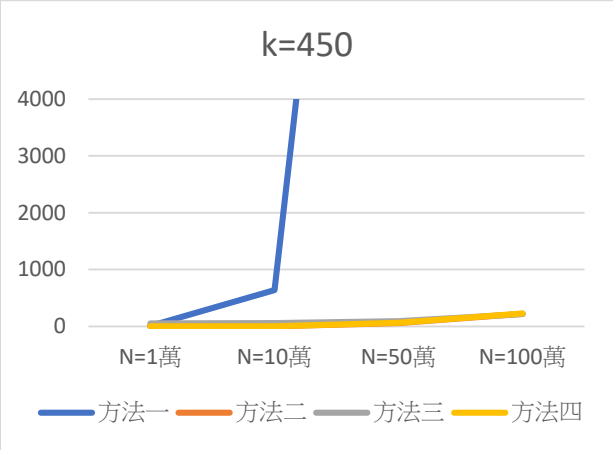


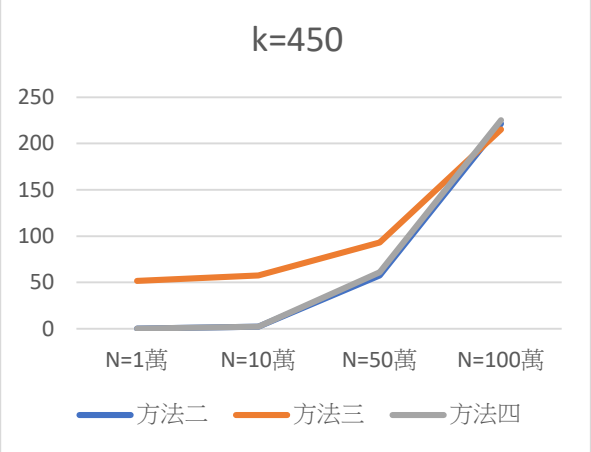


在相同 K 值我們可以看到 CPU 時間快慢分別是: 方法三 > 方法四 > 方法二 > 方法一
當資料筆數不多時切 K 份，對 CPU 快慢效果不明顯。







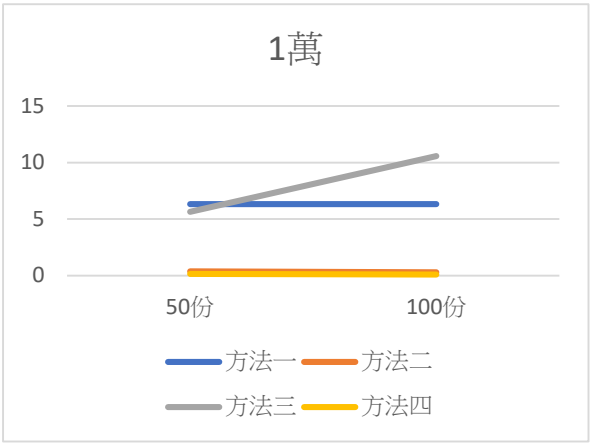


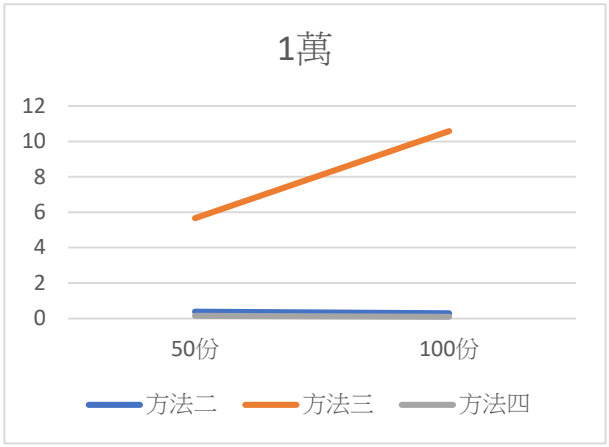
在相同 K 值我們可以看到 CPU 時間快慢分別是: 方法三 > 方法二 > 方法四 > 方法一

相同 K 值，方法三是多個 Process 同時執行所以較其他都快，在 Process 內做 threads 沒有比直接做排序快

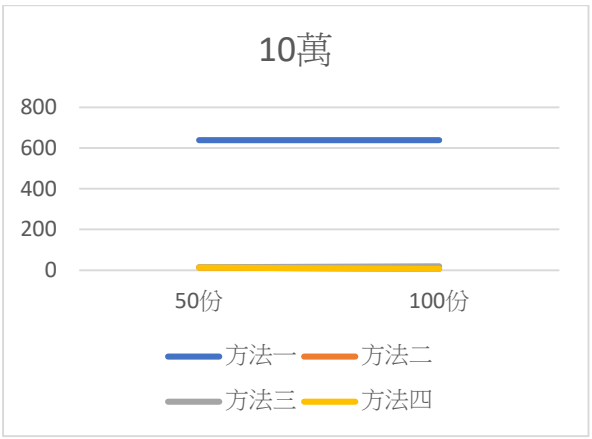
2. 不同 K 值 VS 執行時間（單位：s）

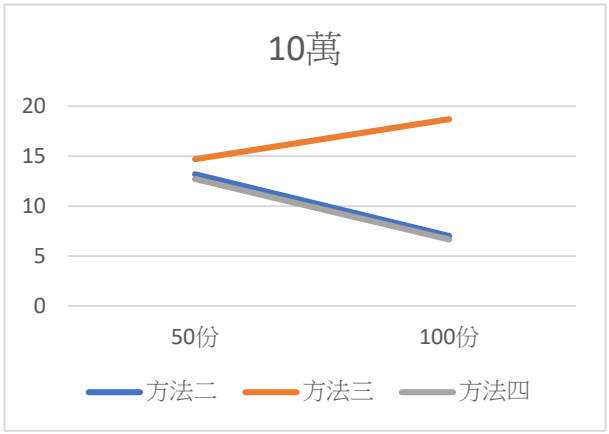
N = {1萬, 10萬, 50萬, 100萬}	K = 50份	K = 100份
方法一	6.3125 638.46875 19753.54839 108,000	6.3125 638.46875 19753.54839 108,000
方法二	0.376436949 13.20214224 345.1881266 1335.072877	0.297391415 7.011094093 179.3901484 703.7073672
方法三	5.655389786 14.70824075 221.2755265 895.5896845	10.58078837 18.71613312 130.1632857 492.6206949
方法四	0.143601894 12.70714331 330.9372067 1334.89278	0.095906734 6.649906635 184.3409414 710.5370305





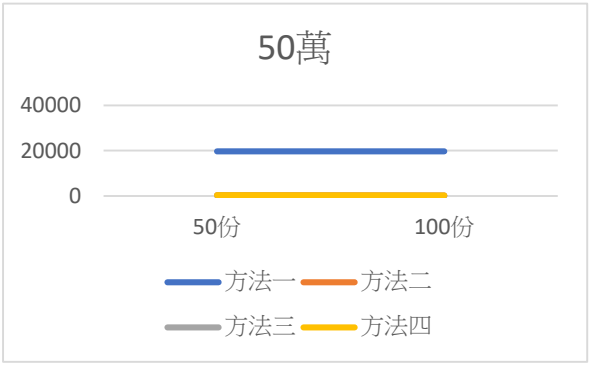
切 100 份時方法三最慢可能是因為做很多 context switch

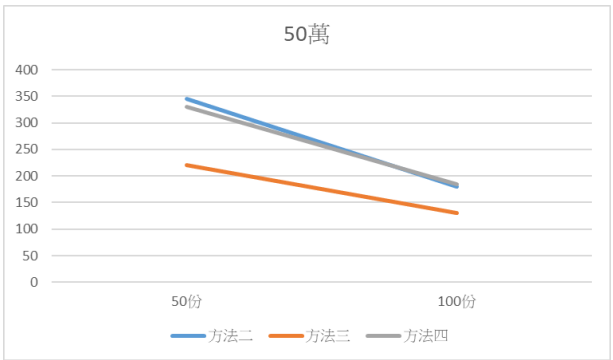


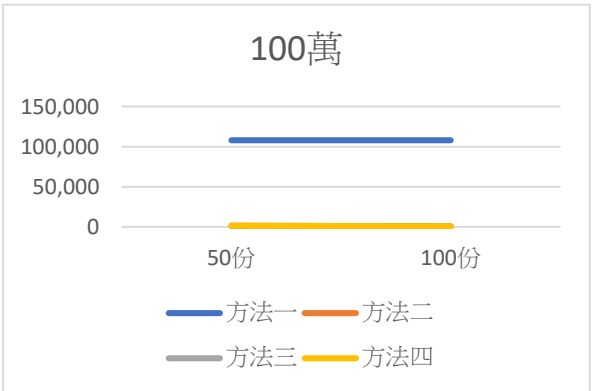


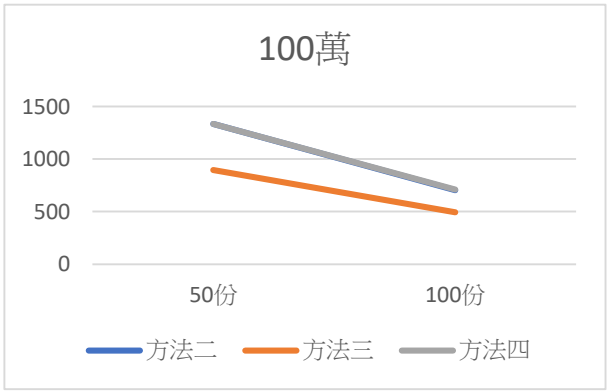
在相同 N 值我們可以看到 CPU 時間快慢分別是: 方法四 > 方法二 > 方法三 > 方法一

切 100 份時方法三較方法二、四慢可能是因為做很多 context switch









在相同 N 值我們可以看到 CPU 時間快慢分別是: 方法三 > 方法二 > 方法四 > 方法一

資料筆數越多，分越多段 CPU 時間越短，方法一到三都一樣。

- 遇到的困難

在跑方法三的時候，我有遇到一個問題當切 K 份超過 478 時就會跳出以下錯誤，但是我用 Mac 的電腦跑還有用桌機跑都不會有這個問題，後我推測可能是因為我的記憶體空間不足。所以解決的方式有兩種，第一種是不要將 K 設定超過 478，第二種是該買更高規格的新電腦了(記憶體要插好插滿)。這是我遇到的比較不常見的錯誤值得紀念一下，所以也在文件中補充。

```
477
請輸入方法名稱(0[quit], 1, 2, 3, 4):
3
請輸入檔案名稱：
input_1w
請輸入要切成幾份[1~陣列長度]:
478
Process Process-5689:
Traceback (most recent call last):
  File "C:\Users\user\anaconda3\lib\multiprocessing\managers.py", line 802, in _callmethod
    conn = self._tls.connection
AttributeError: 'ForkAwareLocal' object has no attribute 'connection'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "C:\Users\user\anaconda3\lib\multiprocessing\process.py", line 315, in _bootstrap
    self.run()
  File "C:\Users\user\anaconda3\lib\multiprocessing\process.py", line 108, in run
    self._target(*self._args, **self._kwargs)
  File "d:\三下\系統程式\HW1\HW1公告測資\input\WH01OS_2.0.py", line 21, in BubbleSort
    result.append(lst)
  File "<string>", line 2, in append
  File "C:\Users\user\anaconda3\lib\multiprocessing\managers.py", line 806, in _callmethod
    self._connect()
  File "C:\Users\user\anaconda3\lib\multiprocessing\managers.py", line 793, in _connect
    conn = self._Client(self._token.address, authkey=self._authkey)
  File "C:\Users\user\anaconda3\lib\multiprocessing\connection.py", line 505, in Client
    c = PipeClient(address)
  File "C:\Users\user\anaconda3\lib\multiprocessing\connection.py", line 708, in PipeClient
    h = _winapi.CreateFile(
OSError: [WinError 231] 所有的管道例項都在使用中。
```