**# OS Project 1 Report**

**資工二 謝心默 B06505017**

**Design**

**General**

- main process和 child processes分別使用兩顆CPU來跑以防止互搶資源

- main process維護一個local clock，並在相對應的時間執行、block、結束child processes

- 在child processes的那顆CPU上，只有當下正在執行的process會有較高的priority

- 利用share memory 來達到讓child processes照著schedule的時間terminate

**FIFO**

First-Come, First-Served Scheduling

原理：

對於每個process，先ready的先執行

實作方式：

- 將所有process按照ready time由小到大排序

- 對於每個process，若其ready time為當前的時間，則將其標記為ready的狀態，當其ready time為當前ready狀態最小的process，執行直到結束

**RR**

原理

利用計時器固定週期，讓正在執行的process在執行一個週期後進入queue中，並從queue中選定下一個執行的process。

執行方式

- 將所有process按照ready time由小到大排序

- 對於每個process，若其ready time為當前的時間，則將該process放入queue中

- 如果目前沒有在執行process，並且queue中有process，則從queue中取出process執行，若在執行一個Time Quantum後沒有結束，則將其execution time減少一個Time Quantum，並將該process放回queue中

**SJF**

原理

選擇執行時間最短的process來執行，此處應用的SJF是non-preemptive的，不會中斷仍在執行的process

執行方式

- 將所有process照ready time由小到大排序，若ready time相同，再按照execution time由小到大排序

- 對於每個process，若其ready time為當前的時間，則將其標記為ready的狀態，當其為當前ready狀態最小execution time的process，執行直到結束

**PSJF**

原理

每次選擇剩餘執行時間最短的 process 來執行，當新產生的 process 執行時間更短時，就可以插隊（preemption）

執行方式

- 將所有process照ready time由小到大排序，若ready time相同，再按照execution time由小到大排序

- 對於每個process，若其ready time為當前的時間，則將其標記為ready的狀態，當其為當前ready狀態的process中最小execution time的process，則將原本在執行的process替換，並讓原本執行的process回到ready的狀態中。

- 與SJF相似，唯一不同點在於main process在fork新process後，如果新的process執行時間較短，則之後轉移cpu時，不是轉移回去，而是轉給新process

**kernel version**

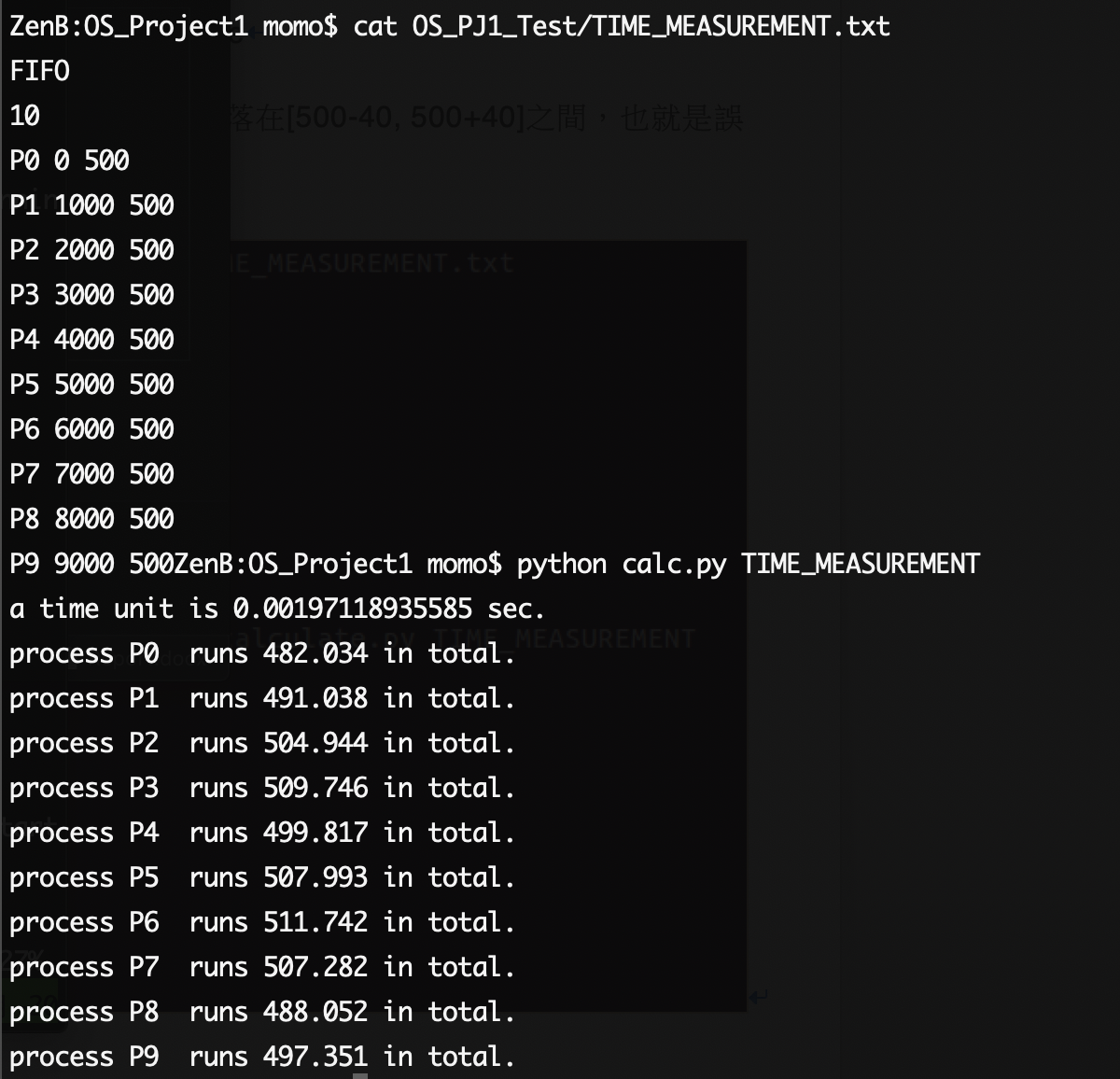
Environment: linux ubuntu 16.04 LTS

kernel: linux-4.14.25

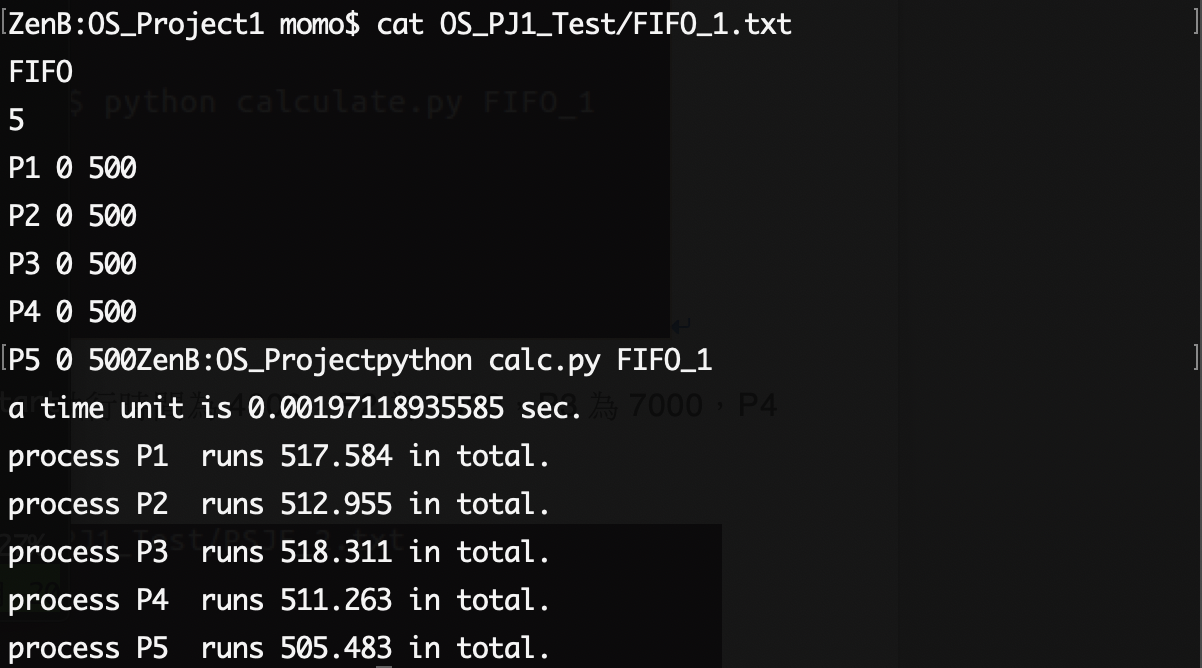
**Comparison**

使用程式(calc.py)計算每支process用了幾個unit time

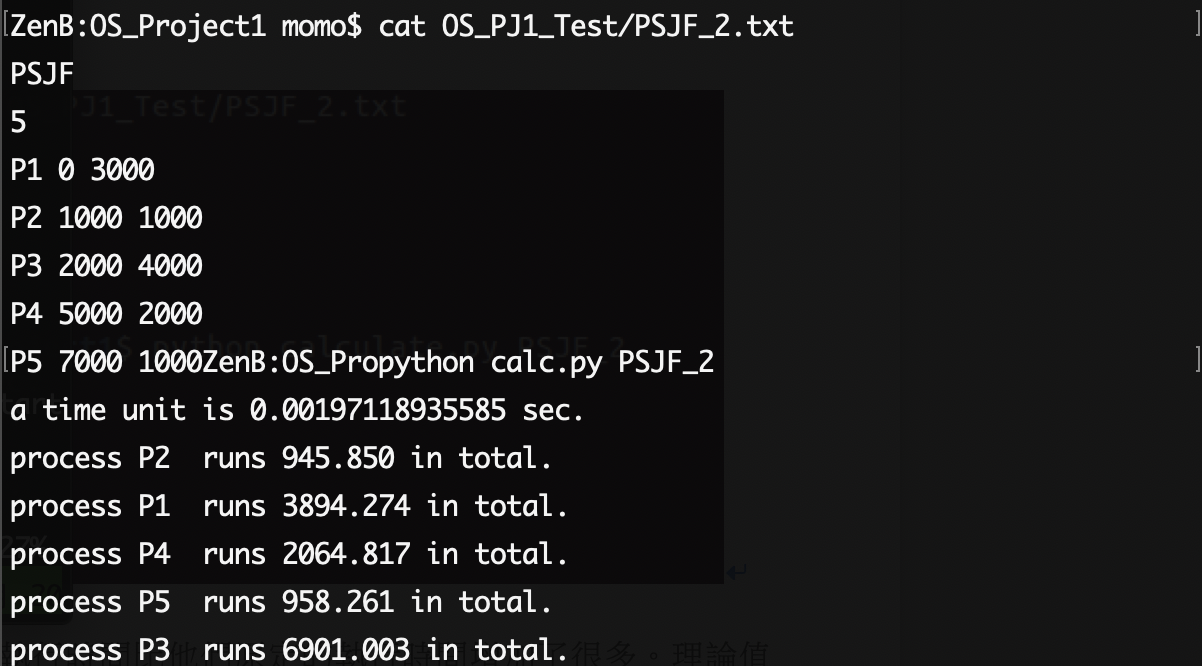
1. TIME\_MEASUREMENT：每支process的範圍在[482, 512]之間，誤差大約3.6%



1. FIFO\_1: 每支process的範圍在[505, 519]之間，誤差大約3.8%



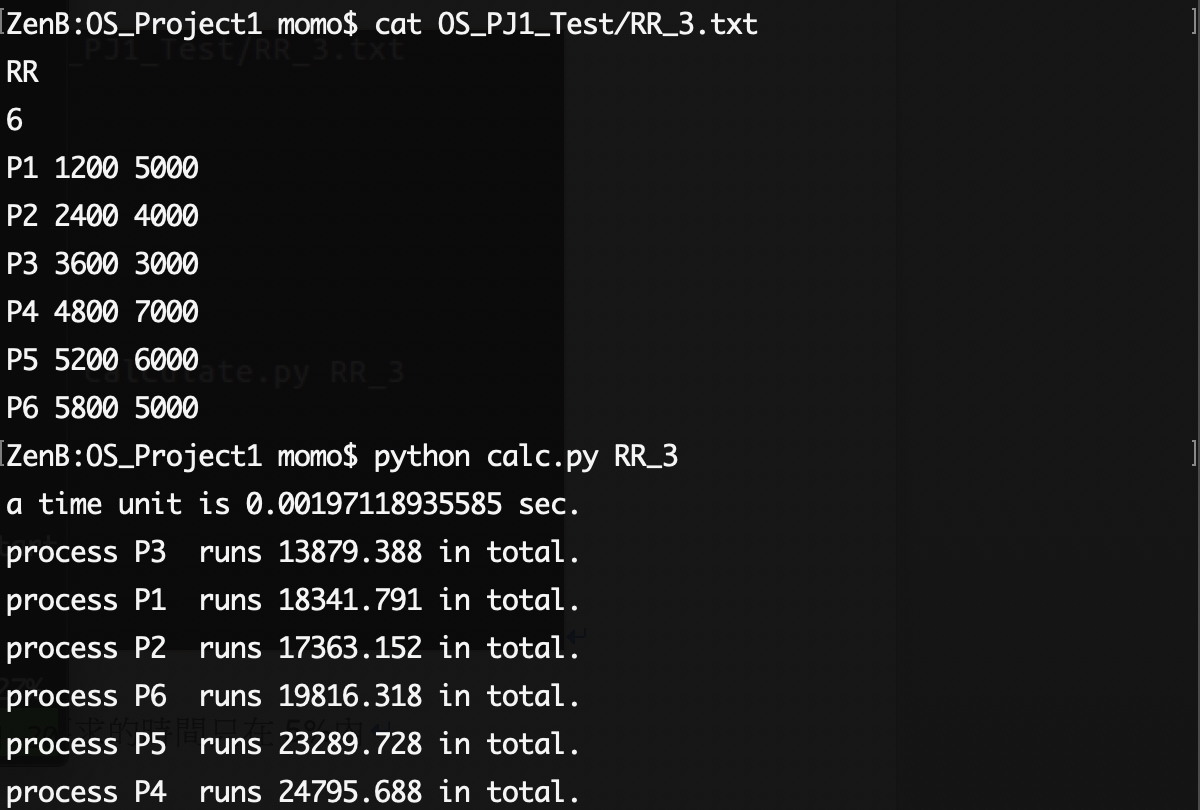
1. PSJF\_2: 因為有preemption，因此P1的理論總執行時間為4000，P2為1000，P3為7000，P4為2000，P5為1000，實際誤差最大約5.6%



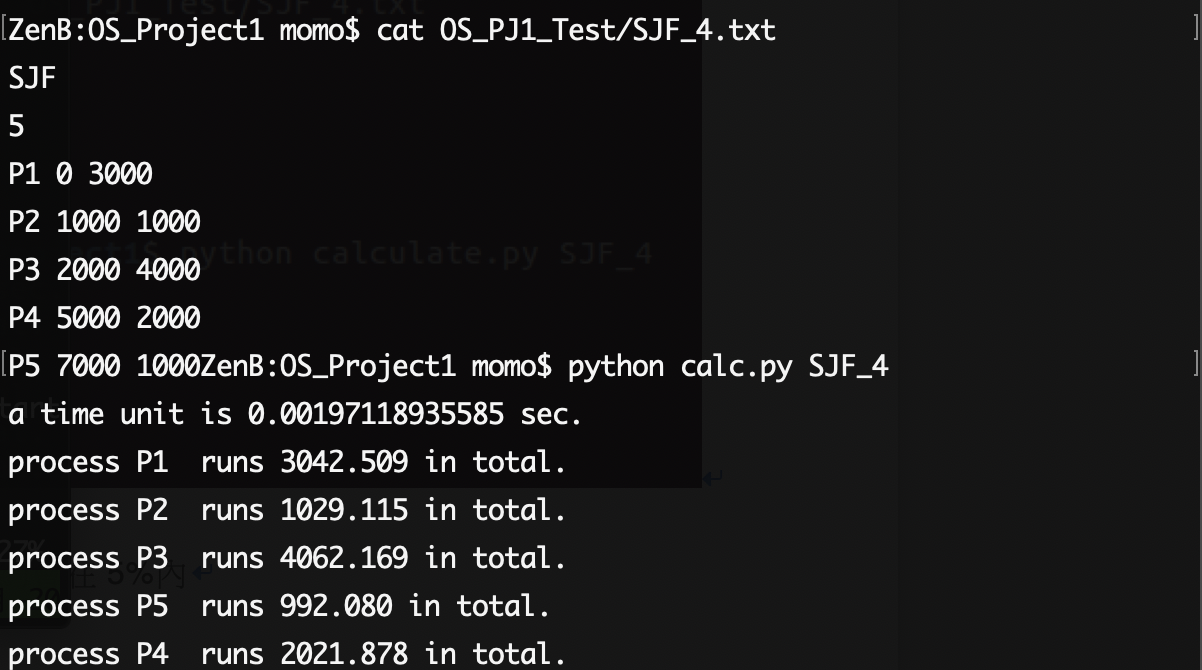
1. RR\_3: 因為會一直context switch，因此總執行時間比原定的執行時間增加了很多。

理論值應為: P1:18500，P2:17500，P3:14000，P4:25000，P5:23500，P6:20000

實際誤差約為0.9%



1. SJF\_4: 誤差約1.5%



**Discussion**

由以上比較可看出排程的順序是正確的，執行時間誤差也相當小。造成誤差的原因可能為以下：

1. scheduler 除了排程外，還有其他執行所花費的時間，例如動態調整各process的priority、執行新的process與資料結構的操作等。
2. 同樣執行時間的兩支process，理論上會跑相同次數的迴圈數，但實際跑的時間也不一樣，可能是因為CPU本身造成差異。
3. 電腦上不只有我們的程序，我們的process可能因為context switch而被中斷，然而我們的計時不會跟著中斷，因此造成程序執行時間增加。