

國立臺灣科技大學答案卷

科目 作業系統 任課教師 石鴻英 系所班組別 四信二
學號 B9415013 姓名 吳偉誠 考試日期 97 年 5 月 1 日

評	分	教師簽名或蓋章
	85	

記分欄 從此處開始寫起。試卷用紙務須節用，非經主試認可不得續用其他紙張作答。

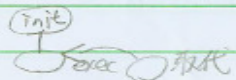
19. 不一定，multi-programmed 包含了兩種，一種是先佔式多工，另一種則是 time-sharing，time-sharing 確保一個 process 在使用了 CPU 一段時間後，須交出 CPU 控制權，也就是每一個 process 都能夠被分配到一段小時間，但是另一種就必須等到 process 有中斷或 system call 才能交出 CPU 控制權。所以 multi-programmed 不是只有一種 time-sharing 而已。
11. Dual mode，分 0 跟 1，執行 I/O 的指令是一種特權指令，目的讓 user 不能去修改 I/O 的中斷向量與 ISR 起始位址。
20. 軟體中斷的產生 - ① 發生 ÷ 0 的情況或是非法參考記憶體，溢位...
② 組合語言的指令或 C 語言的 ex. scanf()
26. 要做 context switch，呼叫一個 system call 表示產生中斷，將現在執行到一半的 process 的 register, Program counter 等資料記錄在 PCB 中，並由其他的 PCB 取出資料執行所要跑的 process，執行完畢再從第一個 PCB 取回剛剛執行一半的 process 資料。假使不做 context switch，那表示一個 process 沒有任何的 system call，當然也不會做任何交換。

2. **microkernel**: 將 kernel mode 裡的一些程式移到 user mode, 可以剩下基本的程式就好, ex: process 管理, memory allocate, process communicate, 這樣 kernel 變得很小, 而很多的應用程式在 user mode 可以隨意的使用, 包括先前屬於 kernel mode 的程式, 而不會影響到 kernel mode 的運行, 所以安全性和可靠性相對提升。

3. **wait** → **ready**
I/O 結束時, 由 wait 回到 ready queue, 準備繼續執行 (preemptive)

5. **memory management**: base register, limit register (起始的地址, 可用的 range)
accounting: 紀錄 memory, CPU 的使用量與那個 process 使用, 並使用了多久時間

4a. **fork()**: 建立一個一樣的 process.

5. **exec()**: 將 process 以新的 process 取代之。


4b. **shared memory** 比 **message passing** 快, 因為 shared memory 是由指標去存取位於共享記憶體體的資料, 但是 **message passing** 是將資料發到 kernel, 再由 kernel 傳遞參數過去, shared memory 因為使用指標加上不用 copy 資料, 所以快了許多, 但是使用 shared memory 方式, Bug 會比 message passing 多, 因為 OS 只提供一塊共享記憶體, 兩個 process 間的參數要由 programmer 自己去 check, 所以需要進行協調, 而 message passing 是藉由 OS 傳遞參數, 所以不會有協調問題, 所以相對 Bug 少。

記分欄

請頁從此開始寫起。

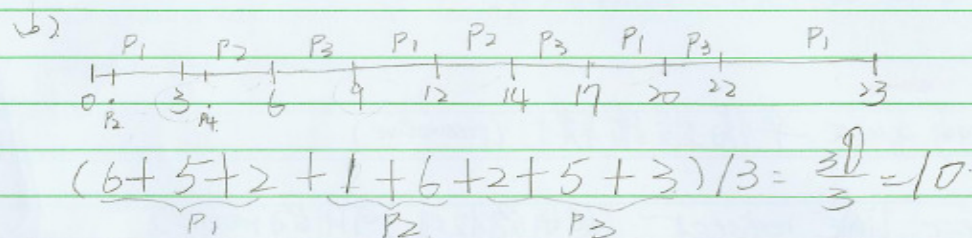
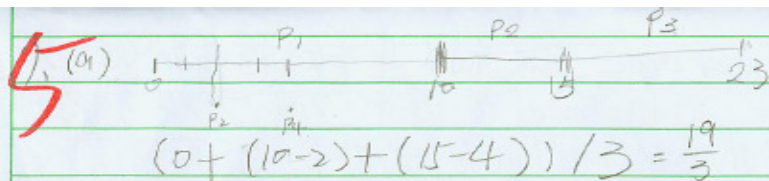
5a. code, data

5b. No

6a. non-preemptive, 表示不會搶奪 CPU。假使有 1 個 process 沒有 system call, 現在有 1 個 process 被載入, 它會一直等, 甚至過了很久, 它仍然未被執行到。所以在 non-preemptive, 如果有無限迴圈或沒有系統呼叫, 它將會一直被執行, 以致於後面的 process 無法享用資源。

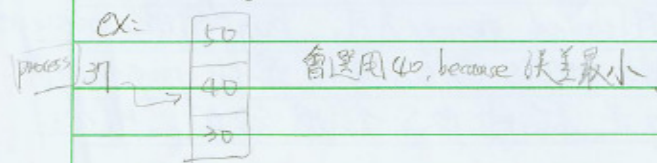
6b. throughput: CPU 在固定的時間 (單位時間) 內, 所能執行 process 數, 越大效能越高。
response-time: process 丟進去給 CPU 執行, 第一次回應的時間在 time-sharing 中, 越小越好。

32-16
6c. Windows XP 給予優先權較高的 process, 較短的時間量 (數字越大, 優先權越低)
Linux 則給予優先權較高的 process, 較長的時間量 (數字越小, 優先權越高)
0~99



8, 5) compiler time, load time.
 2) load time, 因為 relocation 需要 dynamic

9a 5 每次從 list 中找出可以用的 hole 缺差最小的, 會有內部分裂與外部分裂, 但是那塊被浪費的 memory 區塊會比其他兩種小。記憶體使用率也高於 worst-fit



記分欄

轉頁從此開始寫起。

96. 用來記錄所選取到的TLB位址，還有相對應的大小。

96. 90%找9次mem

10%找1次mem

TLB = 10ns, Mem = 120ns

$$0.9 \times (20 + 120) + 0.1 \times (20 + 120 + 120) = 0.9 \times 140 + 0.1 \times 260 = 26 + 26 = 152 \text{ ns}$$

在TLB找到所花的time

由mem找到所花的time

$$\frac{152 - 120}{120} = \frac{32}{120} = \frac{4}{15} \approx 26\%$$