作業系統 課堂作業5 (Chapter 3) 2018/05/02

1. 請說明行程與程式的差異。

行程是主動的個體，程式是被動的個體

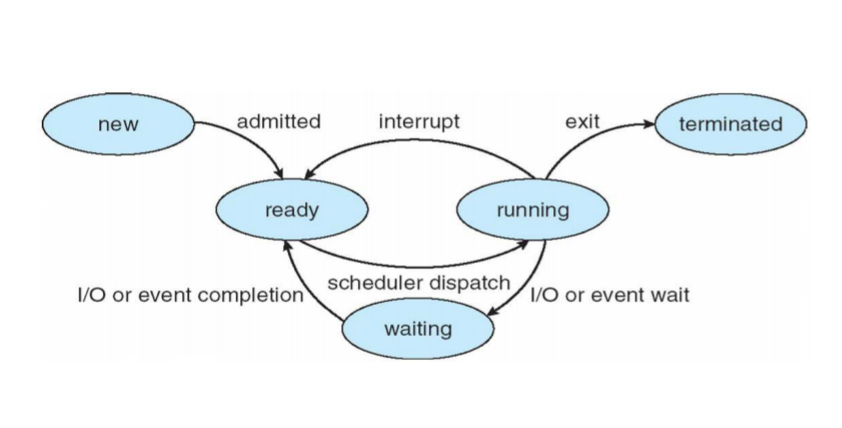
行程是暫時的，程式是長存的

行程與程式的組成內容不同

1. 一個行程的記憶體結構包含哪些項目？

程式碼(本文區text section)，堆疊(stack)，資料區間(data section)，堆積(heap)

1. 畫出行程狀態圖，並簡單說明每個狀態所代表的情形。



新建立(new)=行程剛產生

就緒(ready)=行程已準備好，可以被執行

執行(running)=行程指令正被執行

等待(waiting)=等待某件事發生，通常是等待I/O完成

結束(terminated)=行程結束

1. 請說明在那些情況下行程會離開執行的狀態。

I/O或事件等待、中斷、行程結束

1. 行程控制區塊 (PCB) 通常儲存哪些行程之重要資訊 ?

行程狀態、行程識別碼、程式計數器、暫存器、記憶體資訊、I/O狀態資訊、CPU排程資訊

1. 當一個執行緒被建立時，需要使用到什麼資源？與建立一個行程有什麼不同？

程式計數器、暫存器、堆疊空間

差別在執行緒能共享，行程不能

(不確定) 一個行程可以有一個以上的執行緒，建立行程需要較多的資源

和同一行程的其他執行緒共享相同的記憶體位址空間、程式區段 、資料區段，和一些系統資源。

1. 一個行程於何種情形下，會被移入工作佇列(job queue)？

當一個行程被允許進入系統時，會被移入工作佇列

1. 一個行程於何種情形下，會被移入就緒佇列(ready queue)？

當一個行程進入就緒狀態時，會被移入就緒佇列

1. 一個行程於何種情形下，會被移入等待佇列(waiting queue)？

當一個行程或事件需要作等待時，會被移入等待佇列

1. 一個行程於何種情形下，會被移入裝置佇列(device queue)？

等待I/O完成所產生的佇列，會被移入裝置佇列

1. 當一個行程分別在哪一個行程狀態下會在就緒佇列(ready queue)及裝置佇列(device queue)中排隊？

就緒狀態

等待(waiting)狀態

1. 請描述短程、中程及長程排程器的主要功能與差異。

長程-從行程池中選出行程並將他們載入記憶體放入就緒佇列內，以便執行，控制

多元程式規劃的程度，對I/O型行程和CPU型行程之間作行程組合與調配

中程-若多元程式規劃的程度太高時，中程排班會選擇一些行程自記憶體中置換

出去(swap out)到磁碟中，以便降低多程式程度

若多元程式規劃的程度很低時，中程排班器會把原被置換出去的行程再把

他們替換進來(swap in)到記憶體，並讓他們繼續執行

短程-從就緒佇列(ready queue)中選一個行程將CPU分配給他執行

差異->執行頻率不同，長程排班執行頻率較低，處理速度較慢

短程排班執行頻率較高，處理速度要很快

1. 說明I/O型行程(I/O-bound process)與CPU型行程(CPU-bound process)的差別。

I/O型行程-大部分時間在做I/O，只有少部份的時間在做計算

CPU型行程-大部分時間在做計算，只有少部份的時間在做I/O

1. 何謂全文切換 (Context Switch) ?

當CPU的使用權由一個行程切換另一個行程時，須將舊行程的狀態存起來並且把另一個行程之儲存狀態載回，以便將執行環境復原為後者當初被中斷時的狀態，此過程稱為全文切換或內容轉換

1. 說明fork()其傳回值如何區別父行程及子行程？

父行程-傳回值pid>0

子行程-傳回值pid=0

1. 當父行程尚未呼叫wait()而子行程已結束執行，此時子行程會暫時成為？當父行程未呼叫wait()且結束執行，子行程尚未結束則會成為？

殭屍行程

孤兒

1. 行程間通訊(IPC)所必需的基本運算有哪兩個？

(答案不確定) Send(message)

Receive(message)

(或是)兩種模式 共用記憶體

訊息傳遞

1. 藉由信箱(mailbox)來進行行程間通訊使屬於直接還是間接通訊？一個信箱可被幾個行程使用？

間接通訊

一個信箱可被多個行程使用

1. 說明等待式傳送(blocking send)與非等待式傳送(blocking send)的差別。

等待式傳送:傳送行程持續等待，直到訊息被接收為止

非等待式傳送:傳送端行程送出訊息之後，不必等訊息被接收，就可以繼續

執行後面的動作

1. 說明等待式接收(blocking receive)與非等待式接收(blocking receive)的差別。

等待式接收:接收行程持續等待，直到接收到訊息為止

非等待式接收:接收端行程在嘗試讀取訊息之後，不論是否有收到訊息，都可以

繼續執行後續的動作