作業十：

學習目標：

在 UNIX 中，parent 產生 child 通常會有某個特殊目的，例如：執行一個應用程式，或者是服務 client。因此通常 parent 需要知道 client 的最後執行狀況。

為了保留「最後執行狀況」，在 Linux 上是將作業系統核心（Linux kernel）中，用來描述 process 的資料結構保存下來，這個資料結構就是：task_struct。

如果一次伺服器：例如 web server 每次接收到一個新的 request 後產生一個新的 child 用以服務該 request，這個 web server 會產生大量的 child，如果每個 child 結束後都變成 zombie 則會消耗掉系統資源。

因此我們必須避免 zombie 的發生。

題目：

1. 研究在什麼情況下會產生 zombie。寫一支程式稱之為 zombie，這支程式會產生 10 個 child，每個 child 會執行「ls -alh /」，想辦法讓所有 child 都變成 zombie。
2. 研究在什麼情況下『不會』變成 zombie。寫一支程式稱之為 nozombie，nozombie 這支程式會產生 100 個 child，每個 child 分別執行「sleep 1」「sleep 2」…「sleep 100」。parent 至少必須等到所有 child 執行結束以後才結束。在這段期間「不能讓系統的 zombie 數量太多（例如：5 個）（請注意，即使正確的程式碼，也不一定保證 Linux 會馬上處理 zombie，這和 OS 的「政策」有關）

報告：

1. 產生 zombie 以後是否可以使用 kill 指令將 zombie 殺掉?
2. 請附上截圖證明『zombie』的確會產生 10 個 zombie。
3. 請附上截圖證明『nozombie』幾乎不會產生 zombie
4. 請說明你的系統中『task_struct』到底有多大，並附上截圖證明你的說法

Hint：可以使用 ps -aux |grep "defunct"

Hint：man slabinfo。sudo cat /proc/slabinfo | grep task_struct

繳交：

1. 程式碼和 makefile，助教執行『make』指令後，必須自動產生 nozombie、和 zombie 二個執行檔案。
2. 請將所有檔案壓縮成.tar.bz2。繳交到 ecourse2 上
3. 不能遲交
4. 再次提醒，助教會將所有人的作業於 dropbox 上公開
5. 如果真的不會寫，記得去請教朋友。在你的報告上寫你請教了誰即可。