# Assignment6

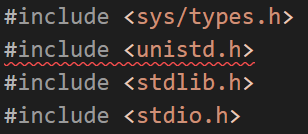
**Team18**

112062519廖思愷

112062636游竣量

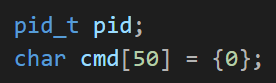
111065547游述宇

* **Explanations of implemented code**

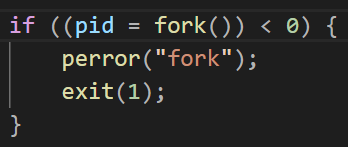


引入必要的標頭文件：

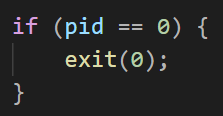
* sys/types.h 和 unistd.h 提供了 fork() 函數和許多UNIX系統服務的原型。
* stdlib.h 包含了各種通用工具函數，如 exit。
* stdio.h 是處理輸入輸出的函數。



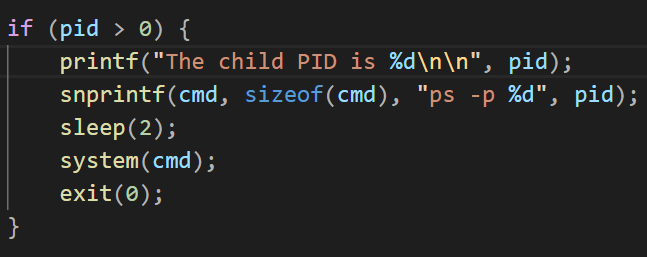
* 宣告一個 pid\_t 類型的變數 pid，它用於儲存 fork 函數返回的process ID。
* char cmd[50] 是一個字符陣列，用來儲存將要執行的命令，並且用 {0} 初始化為全零。



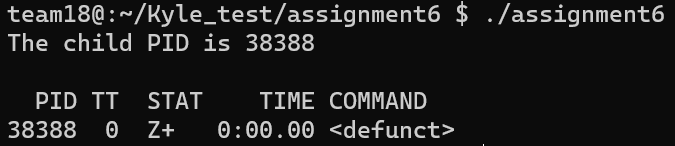
* fork() 函數被調用來創建一個child process。如果 fork() 返回一個小於0的值，這表示process創建失敗，程式將通過 perror 函數打印錯誤信息，並通過 exit(1) 結束程序。



* 這是在child process中運行的代碼。如果 fork() 返回0，代表這是child process，child process將立即通過 exit(0) 結束，在parent process通過 wait() 或 waitpid() 函數來回收child process的結束狀態之前，child process為 zombie process。



* 如果 fork() 返回大於0的值，這代表現在是parent process，並且這個值是child process的PID。
* parent process將打印child process的PID。
* snprintf 函數用於格式化字符串並將其保存在 cmd 字符陣列中。這行代碼將生成一個字符串，該字符串是 ps 命令用於查詢 child process 的信息。
* sleep(2) 函數會讓 parent process 睡眠2秒，確保 child process 已變成 zombie process。
* system(cmd) 函數將執行 cmd 字符串中的命令，這樣就會執行之前格式化好的 ps 命令來顯示 child process 信息。
* **screenshot of result**

****

* 由結果可知 child process 的 PID 為38388，透過 ps 指令的輸出可知：
  + STAT為Z+，表示進程已經終止，但仍然殘留在系統進程表中，因為進程的父進程尚未讀取子進程的終止狀態，所以此 child process 為 zombie process。
  + COMMAND 為 <defunct> ，也表示進程已經終止，但尚未從進程表中清除。所以此 child process 為 zombie process。