**嵌入式作業系統Embedded Operating Systems** (**Lab 4 Assignment Report)**

學號:A041533 姓名:林子翔

1. **Lab 4 作業一：說明如何修改 Lab 3 的程式，使其可以套用於 multithread 的架構中**

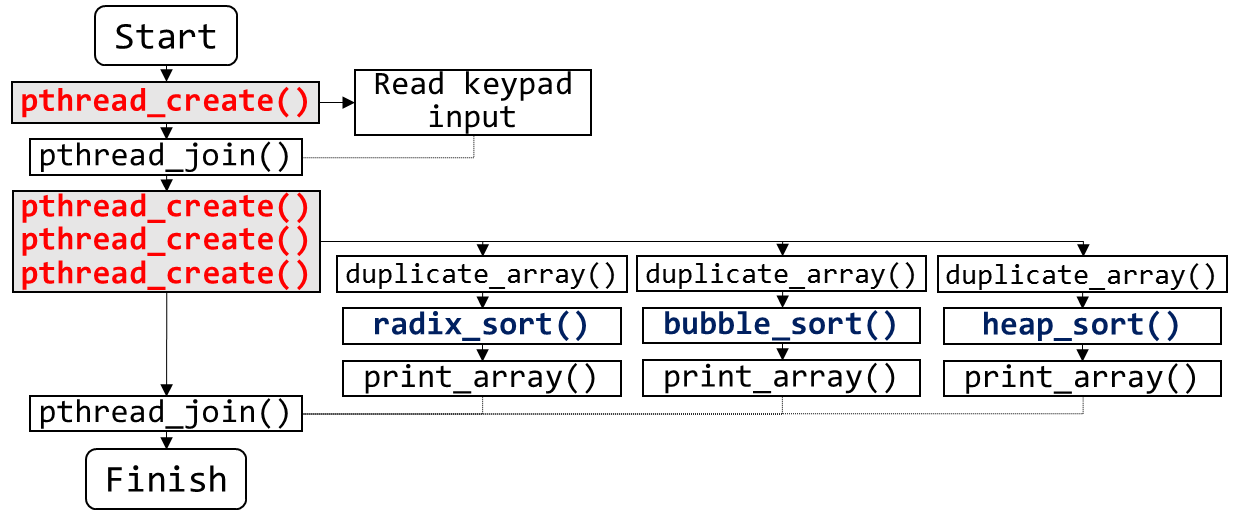
在Lab3的作業中，其程式流程如下**圖一**所示，首先以ioctl系統呼叫使用creator-pxa270-lcd.ko驅動程式模組讀取keypad輸入的按鍵值，重複讀取迴圈直到輸入為'#'後進入sorting階段，而sorting部分基本上為三大步驟: 複製輸入array、使用排序演算法進行排序、輸出排序後結果，排序演算法則依題目規定使用基數排序法(radix sort)、氣泡排序法(bubble sort)與堆積排序法(heap sort)。

|  |  |
| --- | --- |
|  |  |
| **圖一 Lab3作業的程式流程圖** | **圖二Lab4A使用Process流程圖** |

1a) Rewirte the first program in Lab 3 Assignment using 3-4 processes

在**圖一**的程式流程中，radix sort、bubble sort、heap sort三者為data independent的關係，故如**圖二**在main()中完成keypad讀取後，即可讓三個排序演算法獨立於各自的process中執行，最後以waitpid等待所有process結束返回，其中最值得注意的是無須執行duplicate\_array()函式，因為fork出的process is copied in memory from the parent，故於各自process中進行排序時不會互相影響，此外，在讀取keypad輸入的部分，我採用的是malloc的方式依據使用者輸入值的數量動態分配記憶體，因此在每一process結束時，都必須free()屬於自己的陣列。

1b) Rewirte the first program in Lab 3 Assignment using 3-4 threads

如**圖三**所示為使用thread進行平行的版本，由於thread之間可以輕易地共享資料(相較於Process需IPC)，故於讀取keypad輸入部分就開執行緒，在join後開始sorting，最後main()在完成join所有sorting的thread後，結束並釋放動態分配的記憶體。

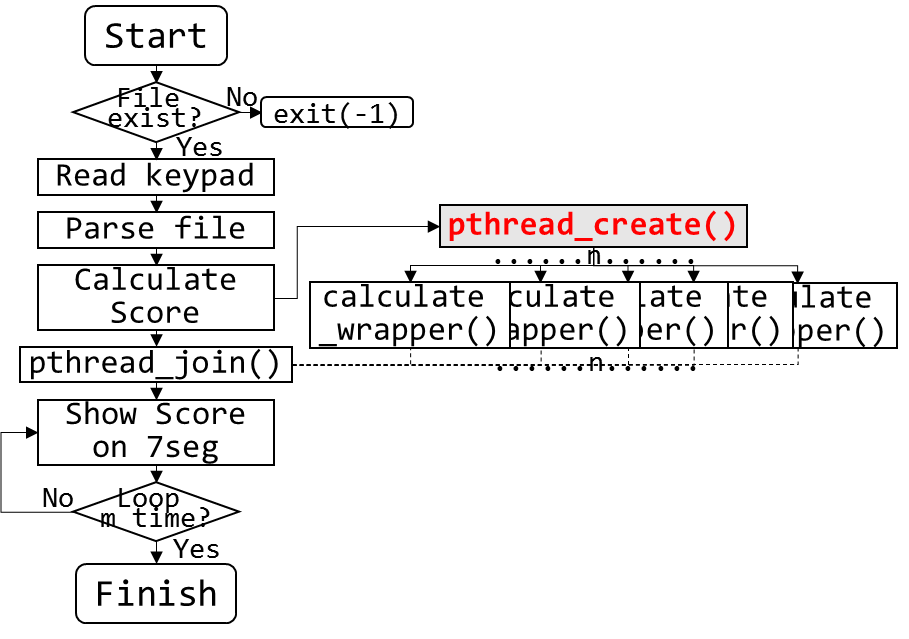
**圖三Lab4B使用Thread流程圖**

1. **Lab 4 作業二：說明整個程式的架構，各執行緒的設計、工作流程等**

作業二的程式架構如**圖四**所示，首先以fopen檢查預定的小考成績檔案是否存在於檔案系統，接著同part 1的操作由keypad讀取n與m，惟輸入結果必須由字元轉換為數字，若轉換失敗則要求使用者重新輸入；接著是parse file，此部分為了可以適用任何長度的檔案內容輸入，我使用了malloc/realloc機制、getc自行判斷'\n'或EOF，並且以strtok進行字串分割，將最後結果以struct subject裝載；進入計算成績部分，這個部分將n筆成績分別各使用一條thread執行，每一條thread依照分派的struct資料內容不同，進行不同的計分方式，使用strcmp比較subject->name並將計算結果存在subject->final\_score；主程式main()在完成joinn筆成績計算的執行緒後，最後進行結果顯示於七段顯示器部分，由於驅動程式模組在\_7SEG\_MODE\_HEX\_VALUE模式下，使用十六進制方式表示數值，故需要對於十位數、個位數與小數位數第一位數進行轉換，其轉換程式碼如下:

1. \_7seg\_info\_t data;
2. data.Mode = \_7SEG\_MODE\_HEX\_VALUE;
3. data.Which = \_7SEG\_ALL;
4. unsigned **long** h1 = (((**int**)scores)/10);
5. unsigned **long** h2 = ((**int**)scores%10);
6. data.Value = h1 << 12 | h2 << 8 |
7. (**int**)((scores-(**int**)scores)\*10);
8. printf("%X\n", data.Value);

完成顯示後依使用者輸入的m次判斷是否需要重複執行或完成整支程式的執行。

**圖四Lab4作業二流程圖**