OS_HW3 程式說明文件

一、開發環境

1. 處理器：Intel(R) Core(TM) i7-10750H
2. 記憶體：8.00 GB
3. 系統類型：64 位元作業系統，x64 型處理器
4. Windows 規格：Windows 10 家用版
5. IDE：Visual Studio Code
6. 使用語言：C/C++

二、實作功能

讀取檔案內 Page Frame 的個數，以及各個 Page Reference 的次序，依這些資訊模擬各種指定的 Page Replacement 方法，須輸出每次 Page Reference 時，每一個 Page Frame 記錄的內容，並計算每種方法之 Page Fault 次數以及 Page Replace 次數。

三、資料結構

1. FIFO() :
   * vector< char > : buffer // 記錄 Page Frame 的內容
   * int : pageFault // 記錄 Page Fault 發生的次數，初始值為 0。
   * bool : isFault // 記錄 Page Fault 是否發生，若是，於輸出檔案寫入 F。
   * bool : stored // 記錄輪到的次序是否有在 Page Frame 中，若否，則發生 Page Fault。

2. LRU() :
   * vector< char > : buffer //記錄 Page Frame 的內容
   * int : pageFault // 記錄 Page Fault 發生的次數，初始值為 0。
   * bool : isFault // 記錄 Page Fault 是否發生，若是，於輸出檔案寫入 F。
   * bool : stored // 記錄輪到的次序是否有在 Page Frame 中，若否，則發生 Page Fault。
   * int : storedIndex // 記錄輪到的次序在 Page Frame 中的位置

3. LFU_FIFO() :

* vector< char > : buffer // 記錄 Page Frame 的內容
* int : pageFault // 記錄 Page Fault 發生的次數，初始值為 0。
* bool : isFault // 記錄 Page Fault 是否發生，若是，於輸出檔案寫入 F。
* bool : stored // 記錄輪到的次序是否有在 Page Frame 中，若否，則發生 Page Fault。
* int[100] : timeStamp // 記錄每個次序進入 Page Frame 的時間，初始值為 0。
* int[100] : counter // 記錄每個次序進入 Page Frame 的次數，初始值為 0。
* bool[100] : inBuffer // 記錄每個次序是否進入過 Page Frame
* bool : full // 記錄 Page Frame 是否已滿
* int : minIndex // 在 Page Replacement 時，可能要被交換次序的 index。
* int : minKey // 在 Page Replacement 時，可能要被交換次序的內容。
* int : minCounter // 在 Page Replacement 時，可能要被交換次序的次數。
* int : minTimestamp // 在 Page Replacement 時，可能要被交換次序的 timeStamp。

4. MFU_FIFO() :

* vector< char > : buffer // 記錄 Page Frame 的內容
* int : pageFault // 記錄 Page Fault 發生的次數，初始值為 0。
* bool : isFault // 記錄 Page Fault 是否發生，若是，於輸出檔案寫入 F。
* bool : stored // 記錄輪到的次序是否有在 Page Frame 中，若否，則發生 Page Fault。
* int[100] : timeStamp // 記錄每個次序進入 Page Frame 的時間，初始值為 0。
* int[100] : counter // 記錄每個次序進入 Page Frame 的次數，初始值為 0。
* bool[100] : inBuffer // 記錄每個次序是否進入過 Page Frame
* bool : full // 記錄 Page Frame 是否已滿
* int : maxIndex // 在 Page Replacement 時，可能要被交換次序的 index。

* int : maxKey // 在 Page Replacement 時，可能要被交換次序的內容。
* int : miaxCounter // 在 Page Replacement 時，可能要被交換次序的次數。
* int : maxTimestamp // 在 Page Replacement 時，可能要被交換次序的 timeStamp。

5. LFU_LRU() :
   資料結構跟 LFU_FIFO()相同。

6. MFU_FIFO() :
   資料結構跟 MFU_FIFO()相同。

四、 運作流程

1. 請使用者輸入檔案名稱，進行開檔，讀入 Page Frame 個數及次序，並依序呼叫指定方法的 function。

2. FIFO :
   首先將 buffer 內用'-'填滿，目的是之後判斷到是'-'字元，就表示還沒滿，可以直接放入，其次先判斷輪到的次序有沒有重複在頁框中，若沒有，代表發生 Page Fault，替換時將最早進來的次序移除，最後計算總共發生 Page Fault 的次數以及 Page Replace 的次數。

3. LRU :
   首先將 buffer 內用'-'填滿，目的是之後判斷到是'-'字元，就表示還沒滿，可以直接放入，其次先判斷輪到的次序有沒有重複在頁框中，若沒有，代表發生 Page Fault，替換時先找出過去最久不被使用到的頁作移除，並將剩下次序依序往前提，再把 vector 的第 0 個設成目前的次序，完成置換的動作，最後計算總共發生 Page Fault 的次數以及 Page Replace 的次數。

4. LFU_FIFO :

首先將 buffer 內用'-'填滿，目的是之後判斷到是'-'字元，就表示還沒滿，可以直接放入，其次先判斷輪到的次序有沒有重複在頁框中，若沒有，代表發生 Page Fault，替換的話要先找出誰的 counter 值最小，代表最不常使用，因為我有一個記錄 counter 的陣列，就利用一個迴圈找出，那如果 counter 相同，就比較誰的 timeStamp 最小，而且這個 timeStamp 是設定過一次就不會再更改的，因為要符合 FIFO 的精神，最後計算總共發生 Page Fault 的次數以及 Page Replace 的次數。

5. MFU_FIFO :

運作流程與 LFU_FIFO 大略相同，差別在於是要先找出誰的 counter 值最大，代表最常使用，可以先請他離開了，而當 counter 相同時，也是比較誰的 timeStamp 最小，代表最先來過，最後計算總共發生 Page Fault 的次數以及 Page Replace 的次數。

6. LFU_LRU :

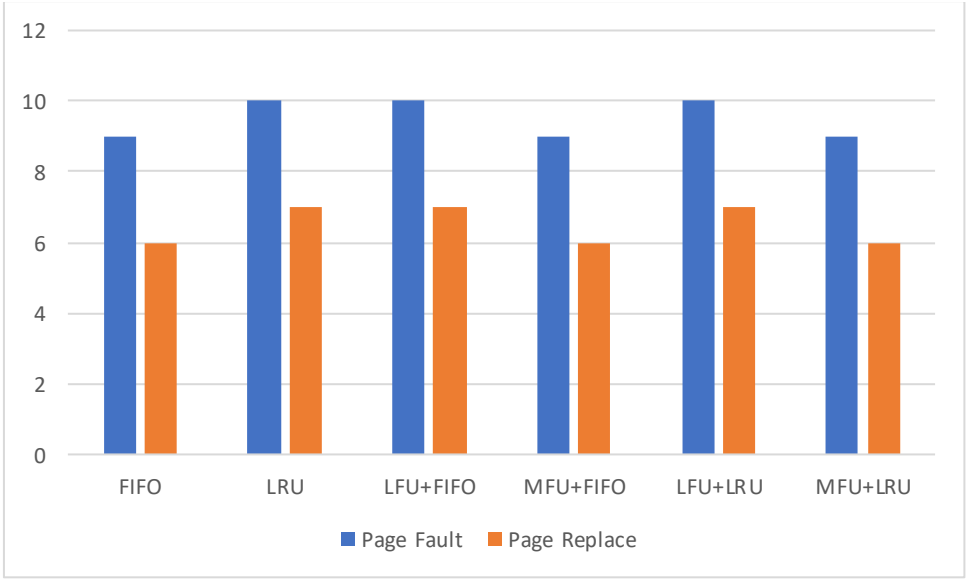
這部分也是以 LFU_FIFO 作延伸，先找出誰的 counter 值最小，代表最不常使用，如果 counter 相同，就再找出過去最久不被使用到作替換，最後計算總共發生 Page Fault 的次數以及 Page Replace 的次數。

7. MFU_LRU :

這個 function 我是先複製 MFU_FIFO 作雛形，再去更改當 counter 相同時要作的處理，因為是使用 LRU，所以要找出過去最久不被使用到作替換，這個利用我的 timeStamp 陣列可以完成，最後計算總共發生 Page Fault 的次數以及 Page Replace 的次數。

五、 方法比較

1. input1 : n = 12, Page Frame = 3



2. input2 : n = 20, Page Frame = 3

六、結果與討論

1. 在 input1 中比較 Page Fault 的次數，FIFO 比 LRU 更多一點，而 LFU 跟 MFU 沒有太大差異，甚至 MFU 的次數跟 FIFO 相同，另外 LFU+FIFO 跟 LFU+LRU 結果是相同的，MFU+FIFO 跟 MFU+LRU 也看不出差異，整體來說，可能是資料數不足，看起來較無感。

2. 在 input2 的結果中可以比較明顯的感受到差異，FIFO 的 Page Fault 次數就比 LRU 多，LFU 跟 MFU 也都比 LRU 多，LFU 跟自己比較的情況下，FIFO 演算法顯得比較笨重，造成次數較 LRU 高，MFU 也是相同情形，整體而言有測出差異性，個人覺得 LRU 可能是最佳的演算法，不管是程式的計算量或是執行效率上，都有不錯的成效。

3. FIFO 是將存在於實體記憶體頁框中最久的分頁給取代掉，實作起來最為容易。

4. LRU 是將存在於頁框中最久沒用到的分頁給取代掉，實作起來比 FIFO 稍微困難，需儲存每個在頁框內的分頁使用後閒置的時間。

5. LFU 是讓次數最少的那一頁被替換掉，因為要比較哪一個頁的次數最少，多了一次迴圈找尋的動作，所以實作上個人覺得比 LRU 更難一些。

6. MFU 可以說就是 LFU 的分身，基本上換湯不換藥。