HW1 Document 資訊三乙 10727219 許志仲

1. 開發環境

平台:Windows 10

使用語言:C/C++

使用Libiary: iostream, fstream, string, vector

1. 實作方法和流程
2. 將指定文件之資料全部讀入，先讀page frame，再來讀入所有page reference，並且存入char vector備用。
3. 讀完資料後，依序執行FIFO(), LRU(), LFU(), MFU(), LFULRU(), MFULRU()等六項page replacement方法。
4. 六種page replacement方法的實作方式
5. FIFO

一開始先初始化指定長度的page frame名為temp，為char的array，接者使用迴圈依序取出page reference中的char進行操作。

首先先進temp查找指定ch，如果找不到就會發生page fault，紀錄page fault數量後將ch推入temp的第一個，如果temp滿了就犧牲刪除掉最後一個，所有page查找完後，印出此次迴圈page fault, page replaces, page frames的相關數據。

1. LRU

一開始先初始化指定長度的page frame名為temp，為char的array，接者使用迴圈依序取出page reference中的char進行操作。

首先先進temp查找指定ch，如果找不到就會發生page fault，紀錄page fault數量後將ch推入temp的第一個，如果temp滿了就犧牲刪除掉最舊(後)一個，而如果找到的話，就將其ch移動到temp的最前，滿足least recently used的條件，所有page查找完後，印出此次迴圈page fault, page replaces, page frames的相關數據。

1. LFU+FIFO

一開始先初始化指定長度的page frame名為temp，為char的array，以及一個相同大小的int array名為freq，用來記錄所有page頻率，接者使用迴圈依序取出page reference中的char進行操作。

首先先進temp查找指定ch，如果找不到就會發生page fault，紀錄page fault數量，先檢查目前存放的page，如果已滿就將頻率最少且最舊的page移出，將ch放入第一個，如果未滿就直接放入，而如果找到的話，就將對應的freq欄位+1，以此紀錄使用頻率，所有page查找完後，印出此次迴圈page fault, page replaces, page frames的相關數據。

1. MFU+FIFO

一開始先初始化指定長度的page frame名為temp，為char的array，以及一個相同大小的int array名為freq，用來記錄所有page頻率，接者使用迴圈依序取出page reference中的char進行操作。

首先先進temp查找指定ch，如果找不到就會發生page fault，紀錄page fault數量，先檢查目前存放的page，如果已滿就將頻率最高且最舊的page移出，將ch放入第一個，如果未滿就直接放入，而如果找到的話，就將對應的freq欄位+1，以此紀錄使用頻率，所有page查找完後，印出此次迴圈page fault, page replaces, page frames的相關數據。

1. LFU+LRU

一開始先初始化指定長度的page frame名為temp，為char的array，以及一個相同大小的int array名為freq，用來記錄所有page頻率，接者使用迴圈依序取出page reference中的char進行操作。

首先先進temp查找指定ch，如果找不到就會發生page fault，紀錄page fault數量，先檢查目前存放的page，如果已滿就將頻率最少且最舊的page移出，將ch放入第一個，如果未滿就直接放入，而如果找到的話，就將對應的freq欄位+1，以此紀錄使用頻率，並且移置最前面當成最新的，所有page查找完後，印出此次迴圈page fault, page replaces, page frames的相關數據。

1. MFU+LRU

一開始先初始化指定長度的page frame名為temp，為char的array，以及一個相同大小的int array名為freq，用來記錄所有page頻率，接者使用迴圈依序取出page reference中的char進行操作。

首先先進temp查找指定ch，如果找不到就會發生page fault，紀錄page fault數量，先檢查目前存放的page，如果已滿就將頻率最多且最舊的page移出，將ch放入第一個，如果未滿就直接放入，而如果找到的話，就將對應的freq欄位+1，以此紀錄使用頻率，並且移置最前面當成最新的，所有page查找完後，印出此次迴圈page fault, page replaces, page frames的相關數據。

1. 結果討論

FIFO VS LRU

相較於FIFO，LRU會把查找到的page移動到frame的最前，在正常使用的狀況下，因為相同的page比較可能再次出現，選擇捨棄最久沒看到的page可以讓page fault相對減少，page replace也會相對減少。

LFU VS MFU

LFU和MFU的差別在於，一個是選擇頻率最少的page丟掉，理由是最少用到的以後應該不會常用到，另一個則是頻率最多的，理由是猜測這個page可能已經用完了，若是這兩個方法都能在符合理想狀況的環境下執行，就能有效減少page fault，但反之，如果LFU遇到又使用之前很少用的page，MFU繼續遇到常用的page，就會有較多的page fault。