HW3

7.6

(a) 增加available可以有效避免deadlock → state維持safe

(b) 減少available造成可使用的resource變少 → state有可能變為unsafe

(c) 增加MAX使得process能取得resource更多，造成available resource更難運用 → state有可能變為unsafe

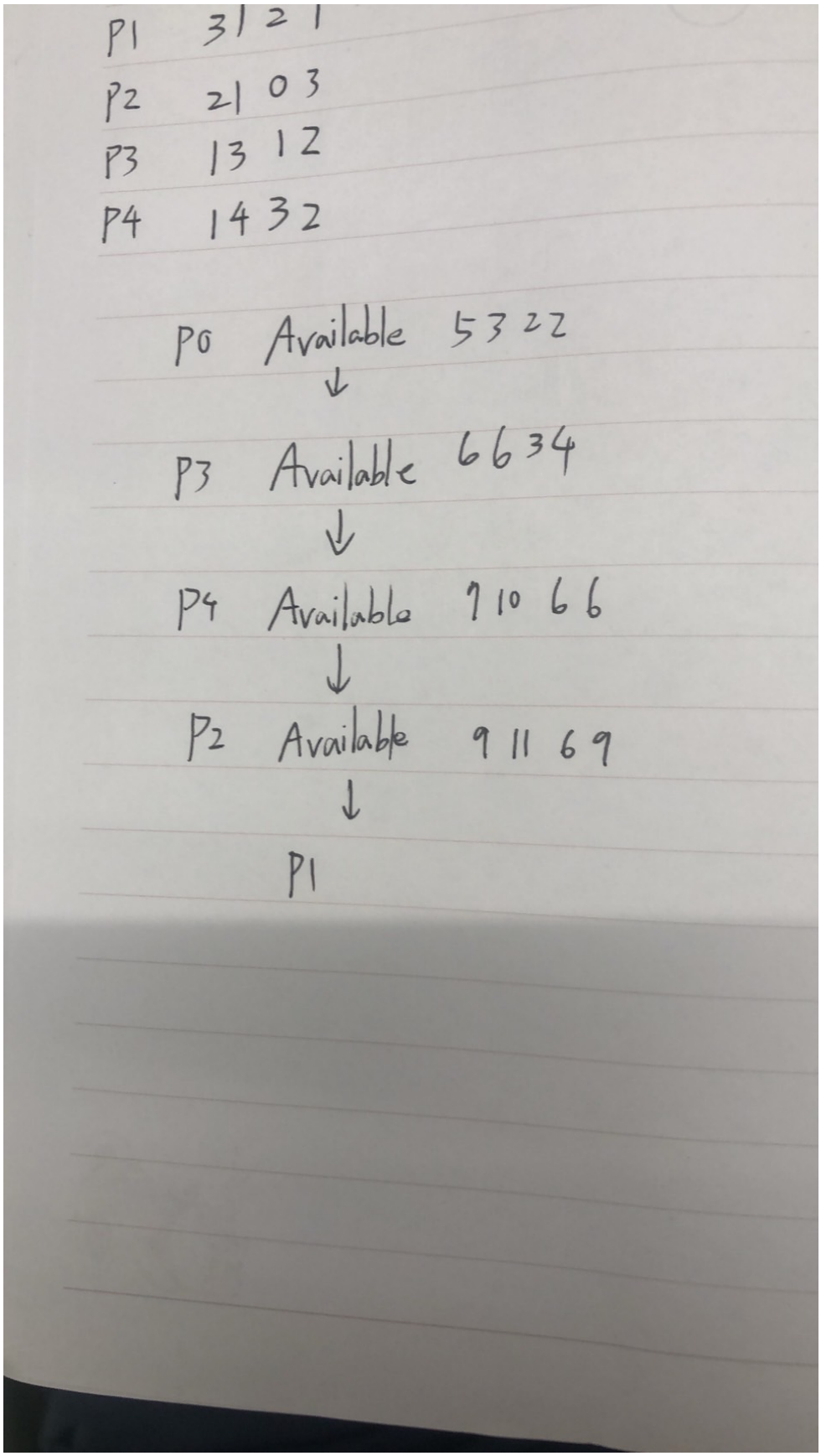
(d) 造成結果與(c)相反，有利於available resource的運用 → state維持safe

(e) 在增加process的數量，available resource沒有增加的情況下，會增加額外的MAX，且可能會減少原有的available resource → state 有可能會變為unsafe

(f) 減少process的數量與(e)的情況相反 → state 維持safe

7.13

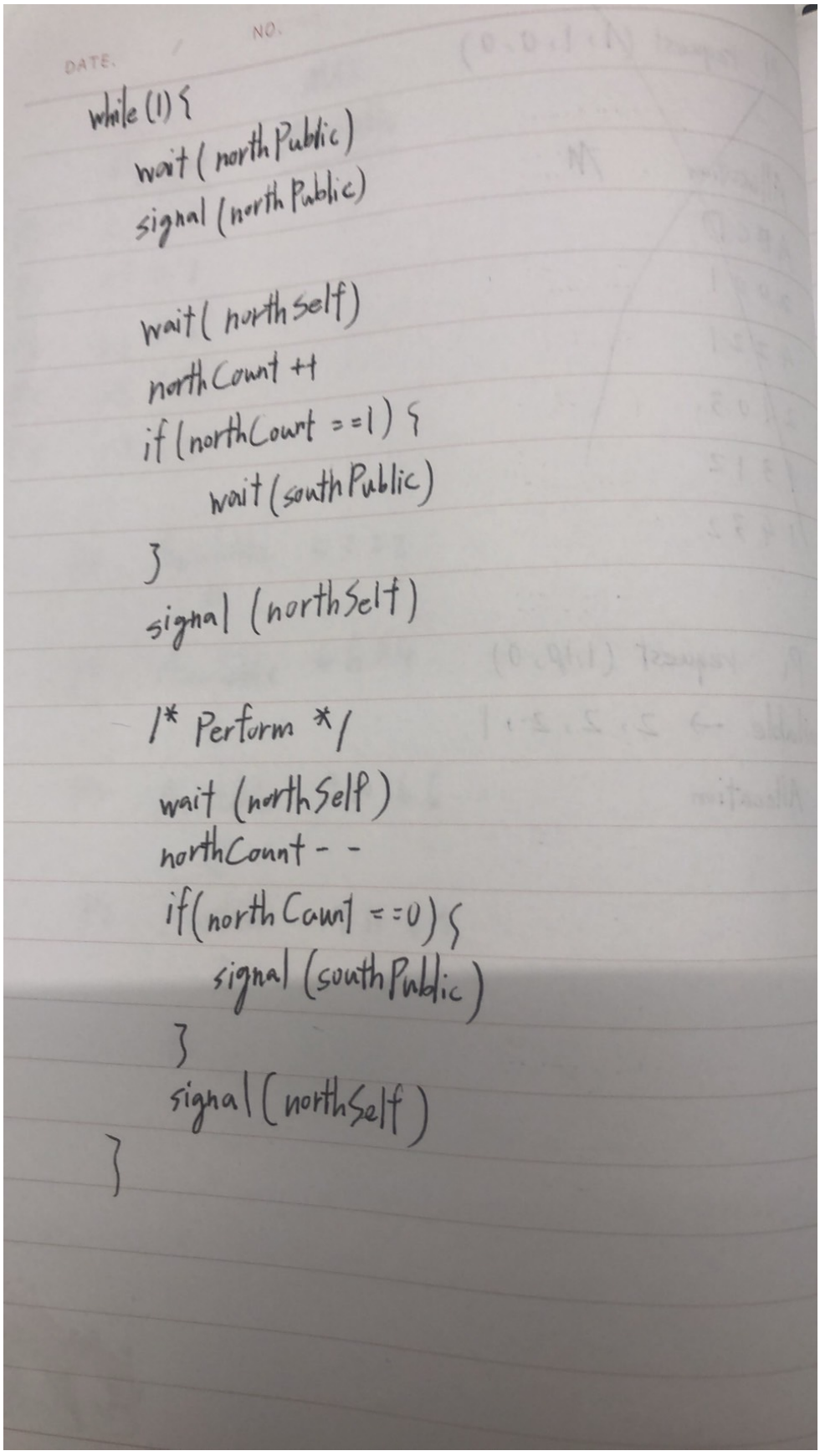
(a)


P1 3121
P2 2103
P3 1312
P4 1432
P0 Available 5322
P3 Available 6634
P4 Available 7 10 6 6
P2 Available 9 11 6 9
P1

(b) p1 request(1, 1, 0, 0)，p1 allocation 變為(4 ,2 ,2 ,1)，available變為(2, 2, 2, 1)，仍然可以找到一safe sequence<p0, p3, p1, p4, p2>

(c) p4 request(0, 0, 2, 0)，p4 allocation 變為(1, 4, 5, 2)，available變為(3, 3, 0, 1)，找不到safe sequence

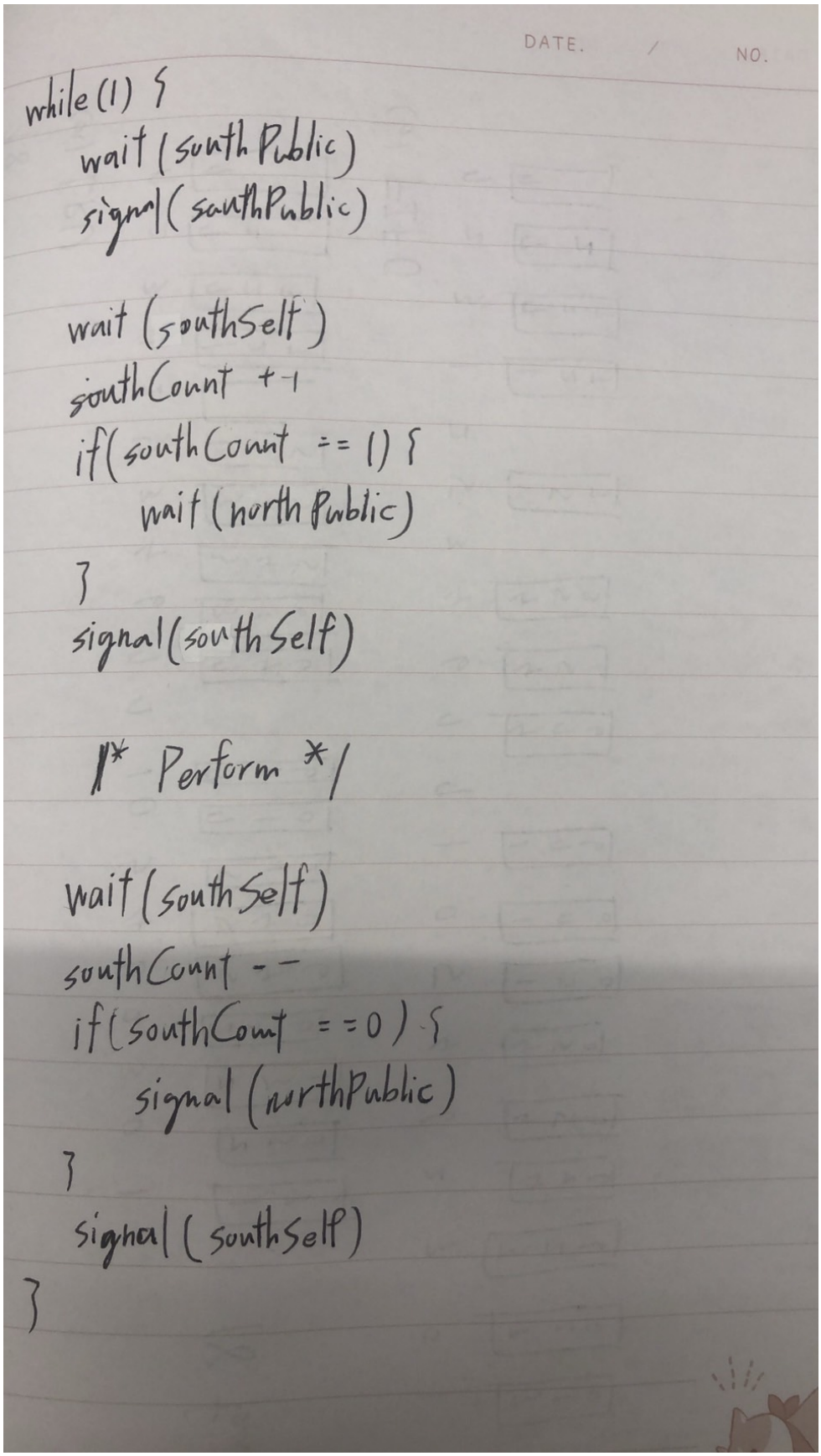
7.15



```
while (1) {
    wait (northPublic)
    signal (northPublic)

    wait (northSelf)
    northCount ++
    if (northCount == 1) {
        wait (southPublic)
    }
    signal (northSelf)

    /* Perform */
    wait (northSelf)
    northCount - -
    if (northCount == 0) {
        signal (southPublic)
    }
    signal (northSelf)
}
```



```
while(1) {
    wait(southPublic)
    signal(southPublic)

    wait(southSelf)
    southCount +1
    if(southCount == 1) {
        wait(northPublic)
    }
    signal(southSelf)

    /* Perform */

    wait(southSelf)
    southCount --
    if(southCount == 0) {
        signal(northPublic)
    }
    signal(southSelf)
}
```

8.1

- Internal fragmentation 分割出固定容量的memory，
- External fragmentation 分割出可變容量的memory

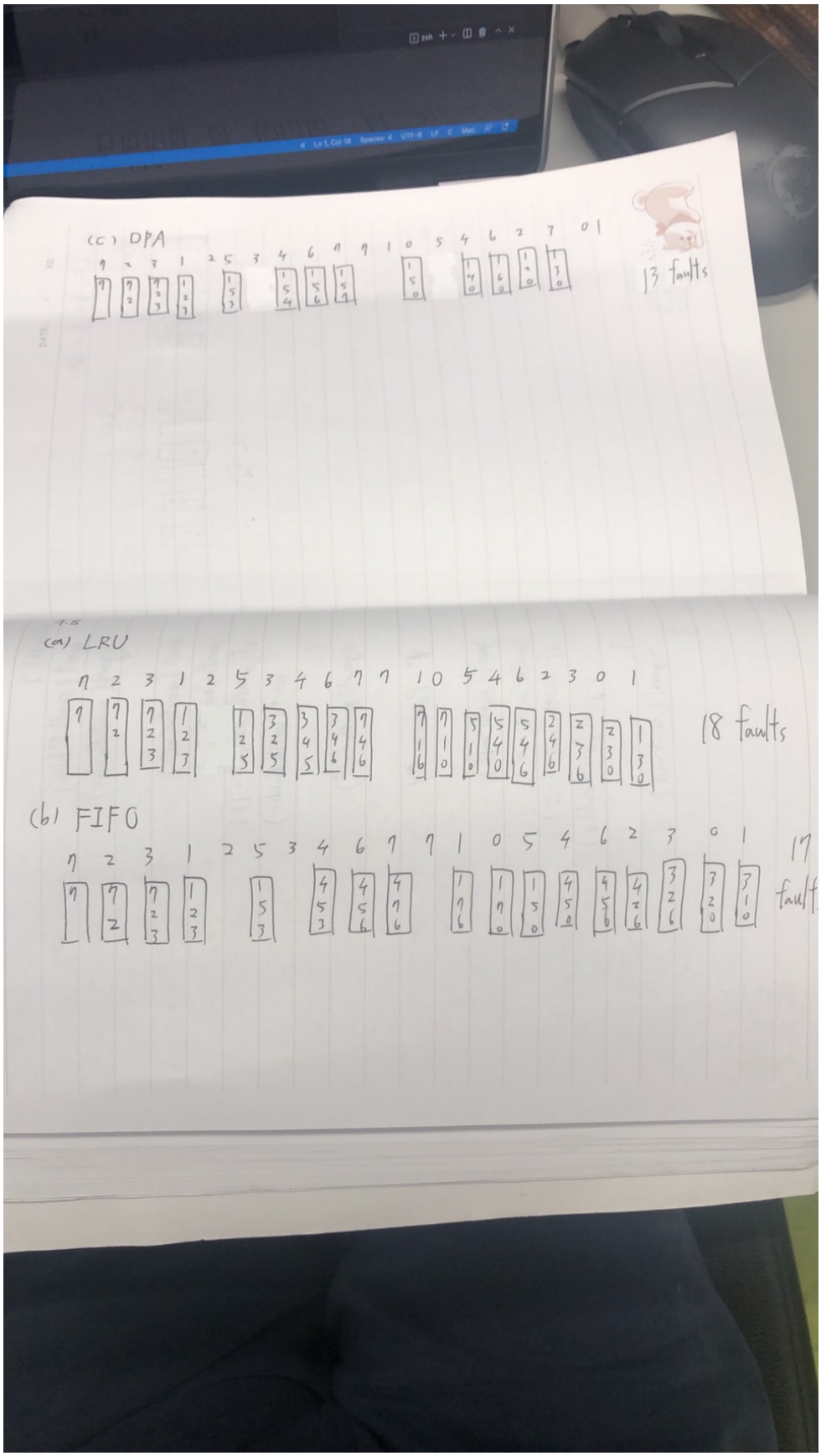
8.9

paging比segmentation還多，因為需要多的空間來存page table，而segmentation只需要一個segment就可以了。

8.16

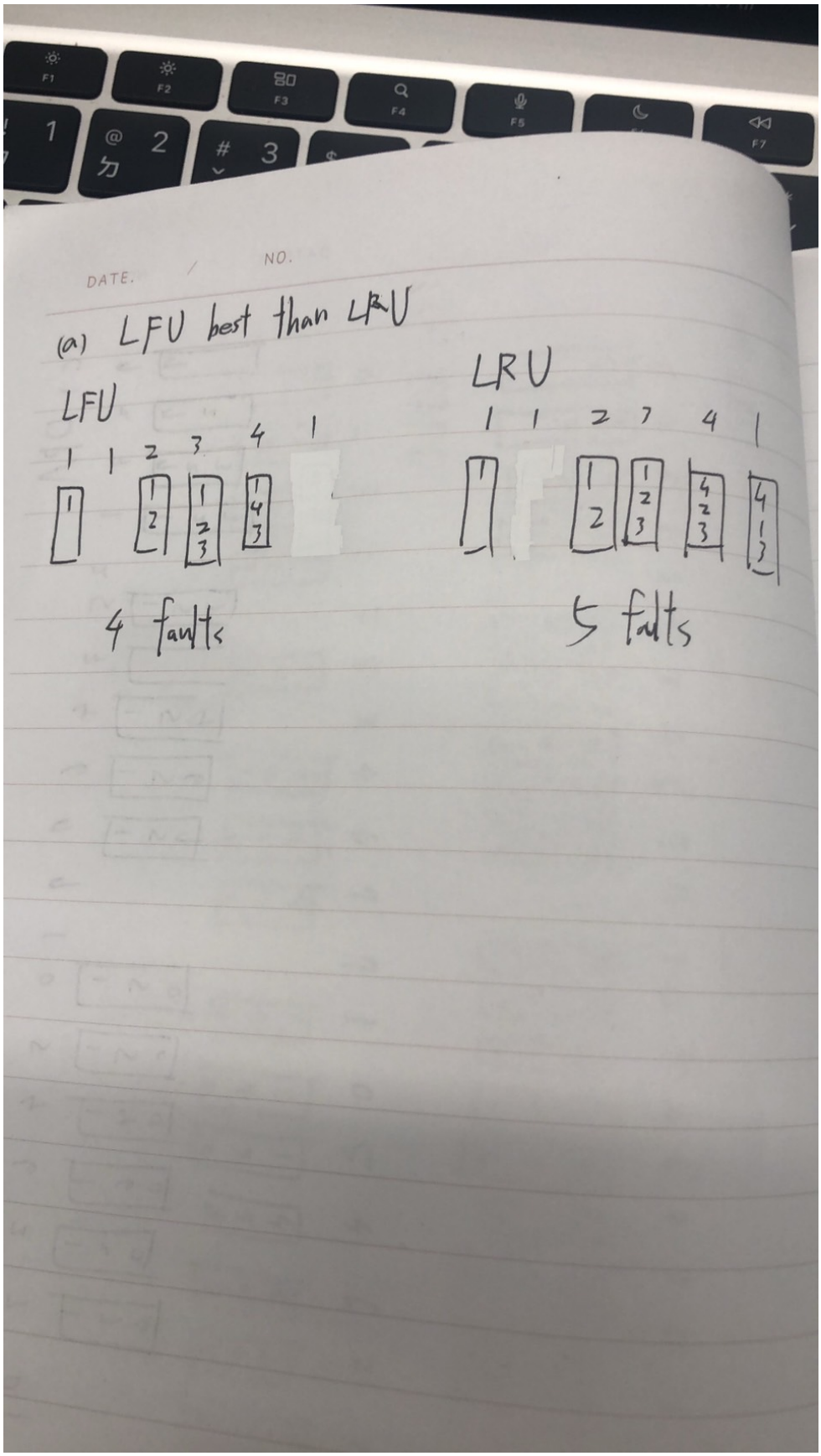
- page table 的entries為2^20個entries
- physical address size / page size ＝ 512MB / 4KB = 2^17

9.8


(c) OPA
13 faults
(a) LRU
18 faults
(b) FIFO

9.11

- LFU 是按照page使用的頻率，替換掉最不常用的page，而LRU為替換掉最長時間沒被用的
-

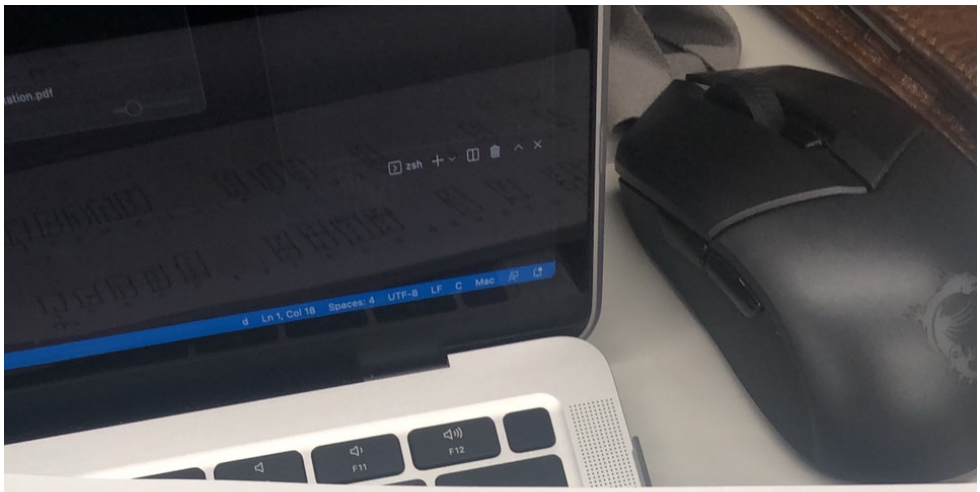

DATE. / NO.
(a) LFU best than LRU
LFU
1 1 2 3 4 1
4 faults
LRU
1 1 2 7 4 1
5 faults

9.17

(a)

- 先設為0
- 有新頁面的時候，counter+1
- 頁面不再有關聯的時候，counter-1
- replace時挑counter最小的，有重複可使用FIFO

(b)



(b)

1 2 3 4 5 3 4 1 6 7 8 7 8 9 7 8 9 5 4 5 4 2

14 faults

(c)

1 2 3 4 5 3 4 1 6 7 8 7 8 9 7 8 9 5 4 5 4 2

11 faults

5 faults

9.19

- 當page frames的數量沒達到process的最低要求時，就會導致page fault
- 檢測cpu跟process的使用率來偵測有沒有thrashing
- 由於frame的數量不能由系統控制，只能去減少level