HW3

第3組

104201025 張立欣

104303205 歐金榮

104303206 黃筱晴

104303542 林亦寧

105503512 趙德昊

105503516 游秉中

106503014 張秉洋

1.

在課本4.1節中(p.68)作者將stock buying problem轉成maximum-subarray problem再用Divide-and-conquer的概念來解此問題，請給一個*O(n)*執行時間的Pseudo code algorithm直接解stock buying problem，毋需再做任何transformation。

int max=0 , min=0, max\_r=-1, min\_r=-1

int a[]; \*/stock/\*

for i in range(1, len(a)):

if a[i] > a[max]: max=i

else if a[i] < a[min]:

if a[max]-a[min] > a[max\_r]-a[min\_r]: max\_r, min\_r=max, min

max=min=i

if a[max]-a[min] > a[max\_r]-a[min\_r]:

return a[max]-a[min], max, min

else

return min\_r, max\_r, a[max\_r]-a[min\_r]

2.

a.

列出陣列<2,3,8,6,1>中的5對inversions。

(0,4)//<2,1>

(1,4)//<3,1>

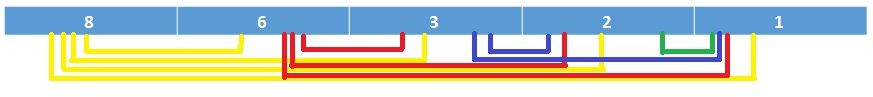
(2,3)//<8,6>

(2,4)//<8,1>

(3,4)//<6,1>

b.

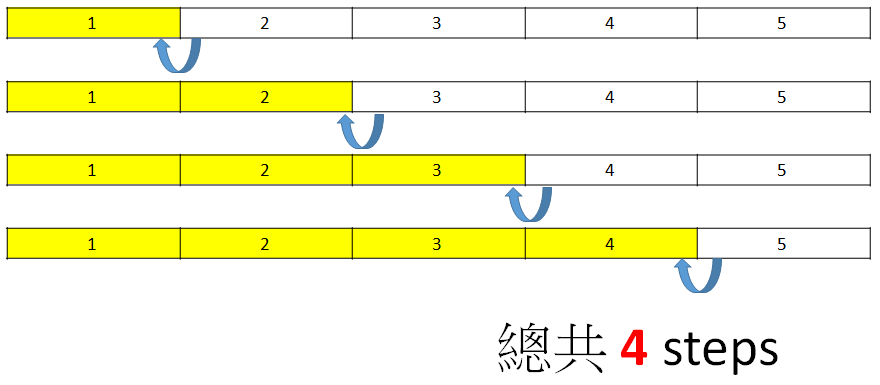
將集合元素由大到小排序後，凡任取兩個元素都是inversions。共有=對inversions。



c.

考慮集合{1,2,3,4,5}，使用insertion sort由小到大排序，比較以下兩種情況：

情況一：Array1=<1,2,3,4,5>有0對inversions。



情況二：Array2=<1,2,3,5,4>有1對inversions。

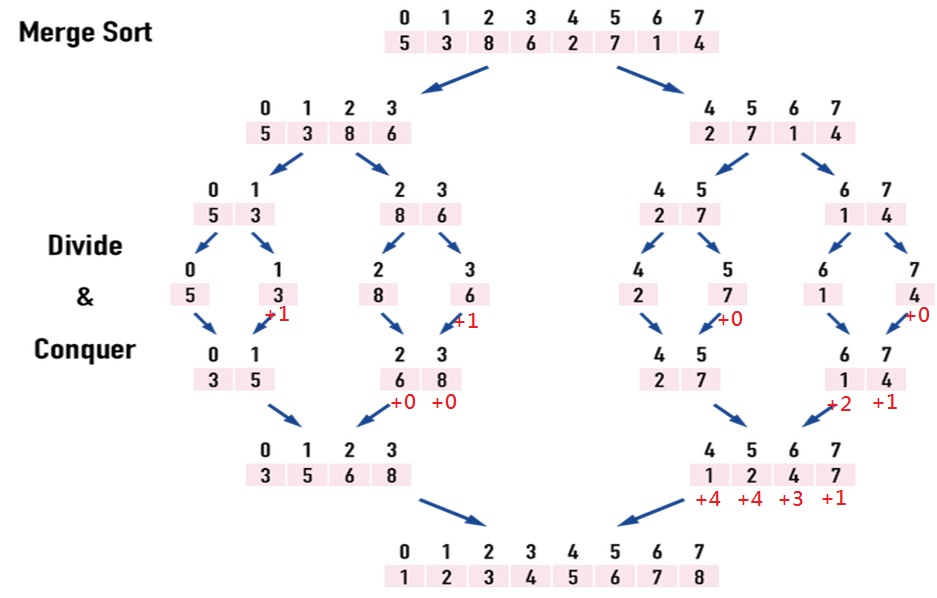


結論：使用insertion sort由小到大排序時，陣列中inversions越少，running time越短。

d.

使用修改的merge sort演算法求出陣列的inversions數量。

在進行RightSub[]和LeftSub[]合併的過程中，當準備丟出RightSub[i]時，若LeftSub[]中仍有n個未丟出元素，則inversions數量的計數值+=n。



T(n)=2T()+θ(n)，使用Master Theorem 得T(n)=θ(n lg n)。

3.

如果從每個遞迴的引數來看，可以發現每一次都會是 2/3n去跑，又總共執行了三次遞迴，而第2&3 行的運行時間為 ，所以綜合起來可以得：

再透過 Master Theorem，令 a = 3, b = , k = 0 (即nk = n0 =1)，可以知道：

因此可以得知符合 Theorem 的第一種情況，即得：

接著由於其迴圈的終止條件是看 arrays’ size，所以不論 sorting 的狀況如何，所花費的時間會是一樣的，因此 best-case, average-case, and worst-case 的時間複雜度皆相同。

而由 pass by address 可以知道其 swap 會直接在原地執行，且 swap 也不會用到多餘的空間，因此為 in-place。

最後我們用反例來證明其為 unstable：

Original: [5, 2a, 2b, 3] → [3, 2a, 2b, 5] → [2b, 2a, 3, 5] ⇒ 2a, 2b 的位置交換了！

4.

a.

|  |  |  |  |
| --- | --- | --- | --- |
| 2 | 3 | 8 | 9 |
| 4 | 14 | 16 | **∞** |
| 15 | **∞** | **∞** | **∞** |
| 12 | **∞** | **∞** | **∞** |

b.

i.

Since [1,1] is the min in the Young tableau , if [1,1] is **∞ ,** then other element have to larger then [1,1] , so oher element have to be **∞.**so the matrix is empty.

ii.

Since [m,n] is the max in the Young tableau , if [m,n] < **∞ ,** then other element have to smaller then [m,n] , so oher element have to < **∞.**  
so the matrix is full.

c.

EXTRACT-MIN (A[m][n]) {

i=1 , j=1 , A[i,j]= **∞** ; //把MIN用**∞**代換

while ( A[i,j]>A[i+1,j] or A[i,j]>A[i,j+1] ) {

//直到右方和下方都大於等於A[i,j]

if ( A[i+1,j]<A[i,j] ) { //選右方或下方小的交換

SWAP (A[i+1,j] , A[i,j]) ;

i++;

}else {

SWAP (A[i,j+1] , A[i,j]) ;

j++;

}

}

}

p=m+n

T(p)=T(p-1)+O(1) ( Since原矩陣=子矩陣+交換 )

=T(p-2)+2\*O(1)

=……

=O(p)

=O(m+n)

d.

Same like (c) , insert in A[m][n] (最右下角) , choose MAX( left , top) and swap until left and top both smaller then the insert element .

p=m+n

T(p)=T(p-1)+O(1) ( Since原矩陣=子矩陣+交換 )

=T(p-2)+2\*O(1)

=……

=O(p)

=O(m+n)

e.

Step1: insertion the numbers into empty tableau

Step2: do EXTRACT-MIN to array until the tableau is empty

Time:

Insertion -> n^2\*O(n+n) = O(n^3)

EXTRACT-MIN -> n^2\*O(n+n) = O(n^3)

Hence , time complexity is O(n^3)+ O(n^3) = O(n^3).

f.

SEARCH (A[m][n] , k) {

i=m , j=1 ; //選左下角當起點

while (i>0 and j<=n) { //還在矩陣內時執行

if (A[i,j] > k) { //當該格 > 要找的值就往上一格

i-- ;

}else if (A[i,j] < k) { //當該格 < 要找的值就往右一格

j++ ;

}else {

return EXIST ; //當該格 = 要找的值就回傳值存在並結束

break ;

}

}

if (i < 1 or j > n) { //如果超出矩陣範圍還沒找到則回傳找不到

return Not Found ;

}

}

Worse case : when k is at A[1,n] , need the most step to search ,

hence total step to search is (m-1)+(n-1)=m+n-2

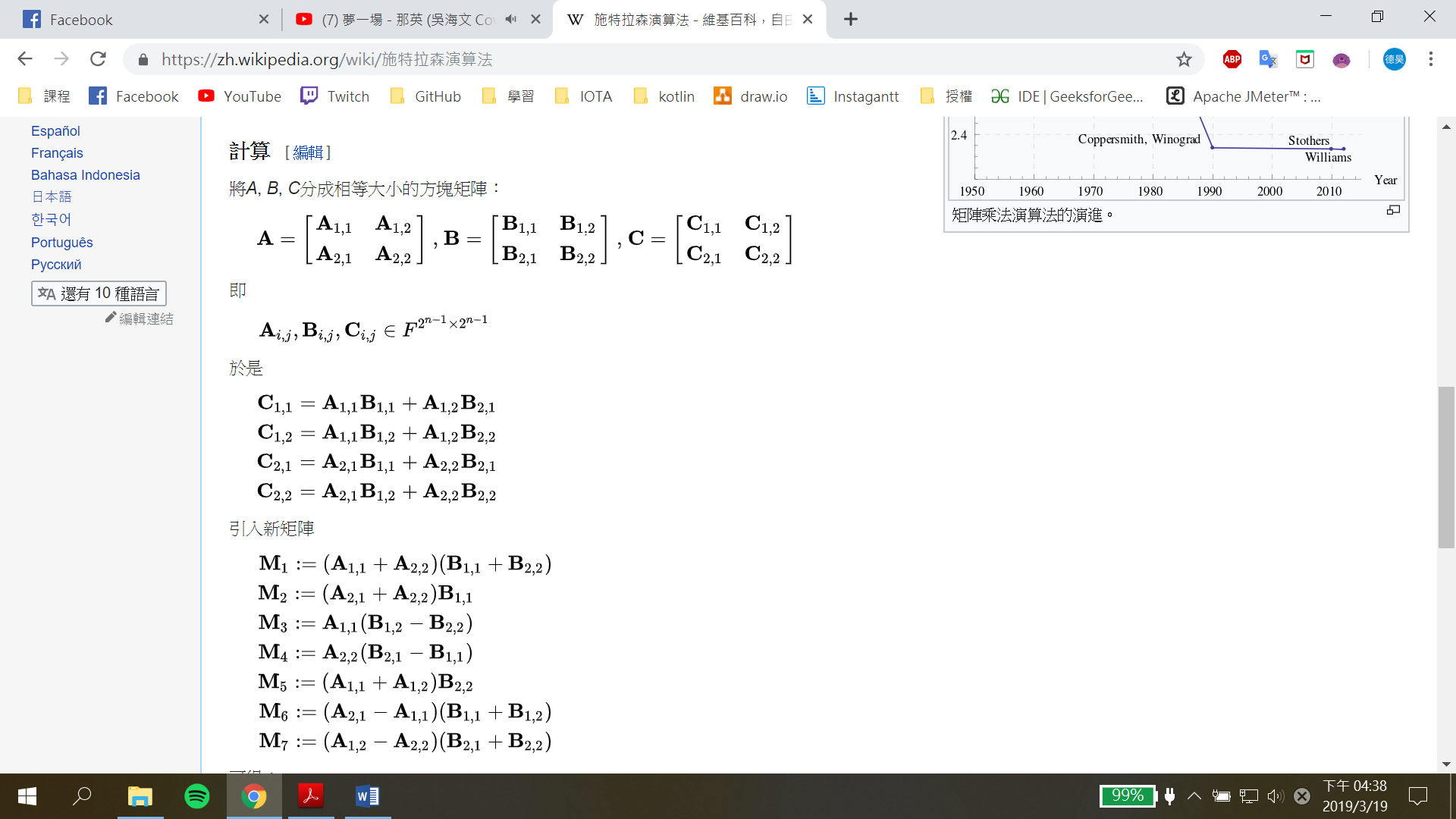
Therefore , time complexity is O(m+n-2) = O(m+n) .

5.

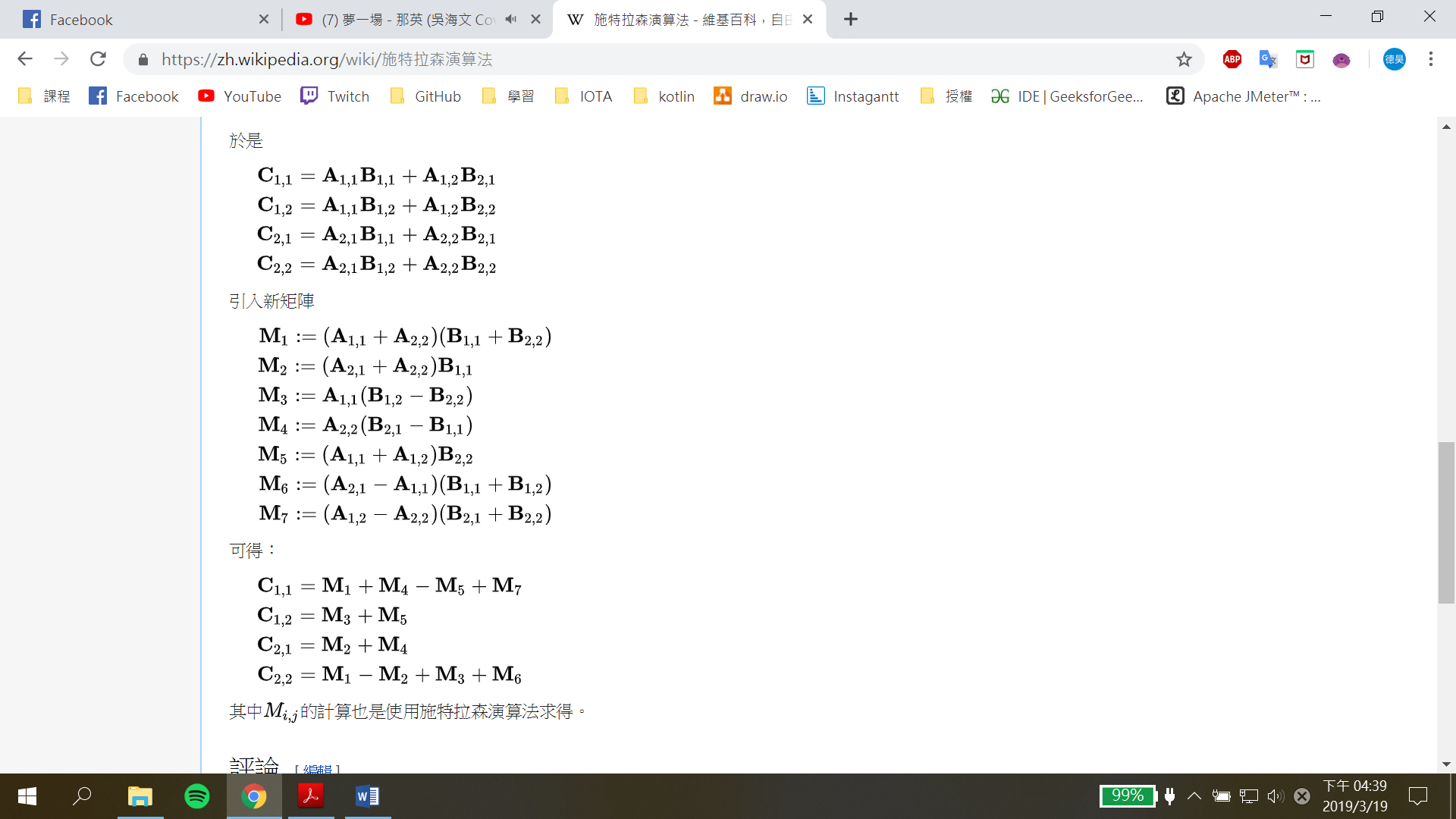
(1)

如果n不等於2k，利用補0的方式將矩陣做成2k x 2k。

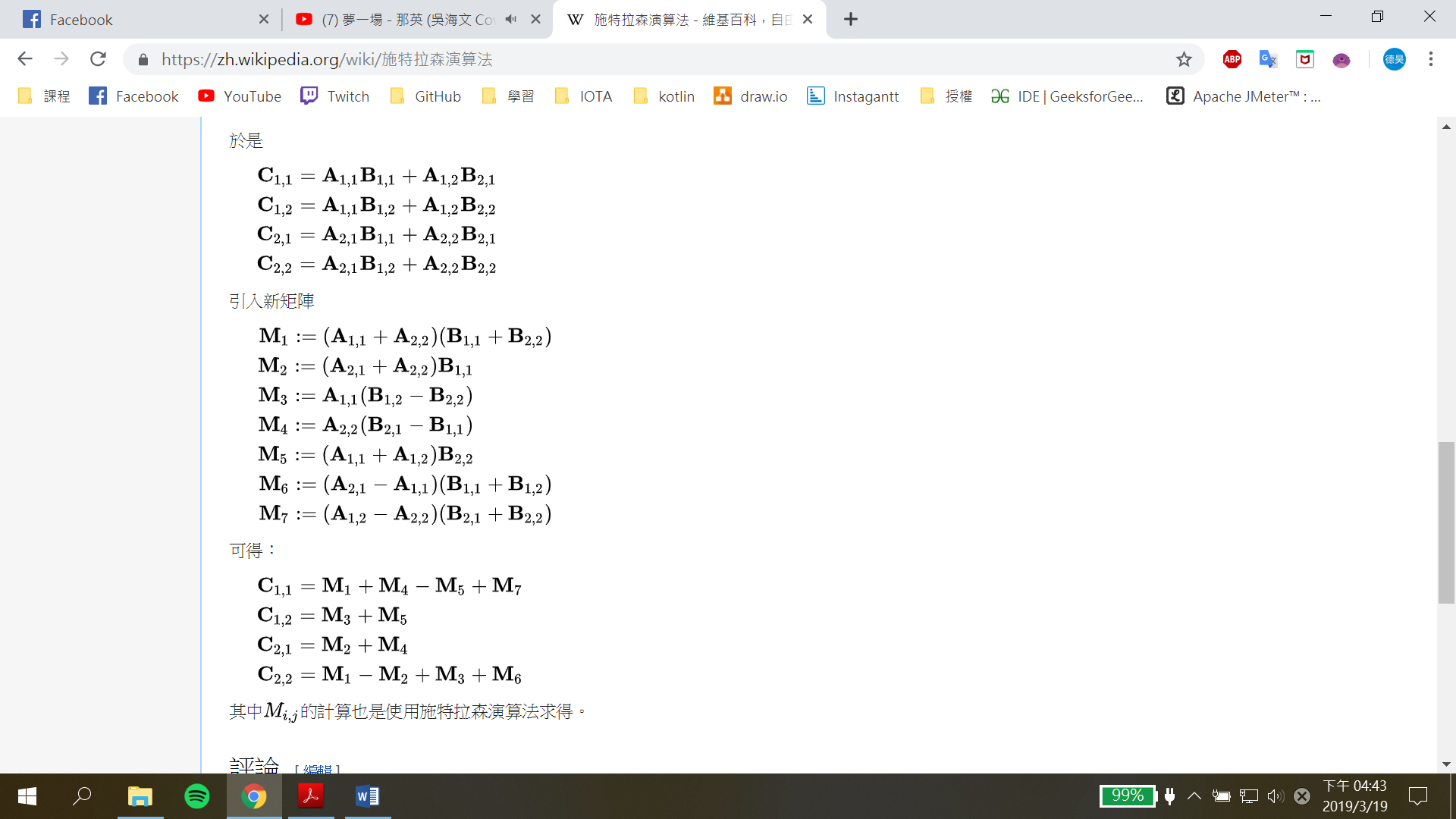
(2)



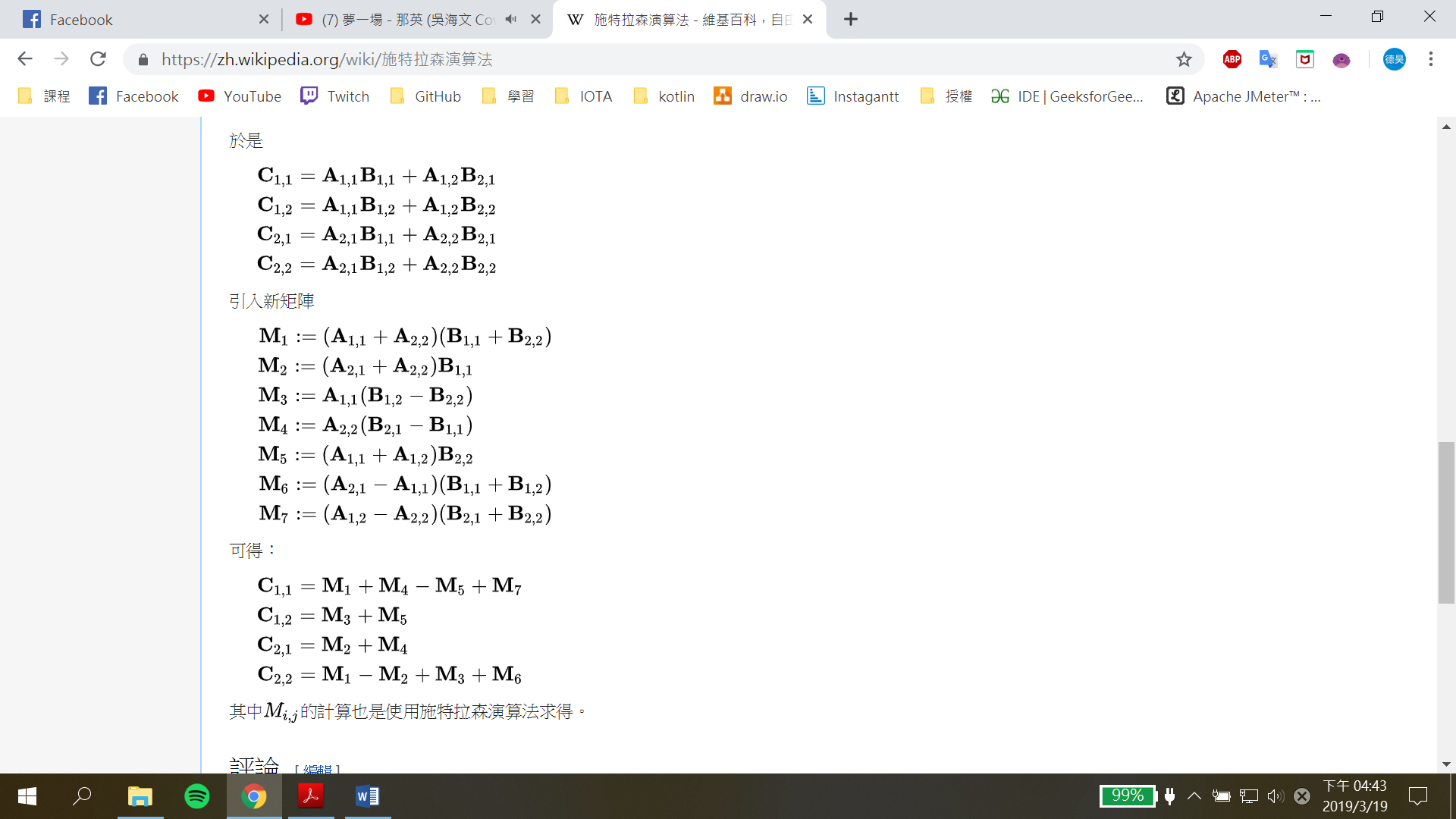
假設C=AB，根據一般矩陣乘法，需要做8次乘法，如下圖，



而Strassen’s algorithm利用多做加減法(計算時間較乘法短)，將乘法次數由8減少至7，



將原本的C矩陣由上圖M值代換，得到以下C矩陣



因為只考慮乘法次數，列出遞迴式 ，利用 Master theorem，得到結果為 ，由此可知 Strassen’s algorithm 需要時間。

6.

Assume Where

By Master theorem

7.

假設有個 n x n 的矩陣，將它分割成68 x 68、70 x 70 和 72 x 72，可得到以下三條式子，

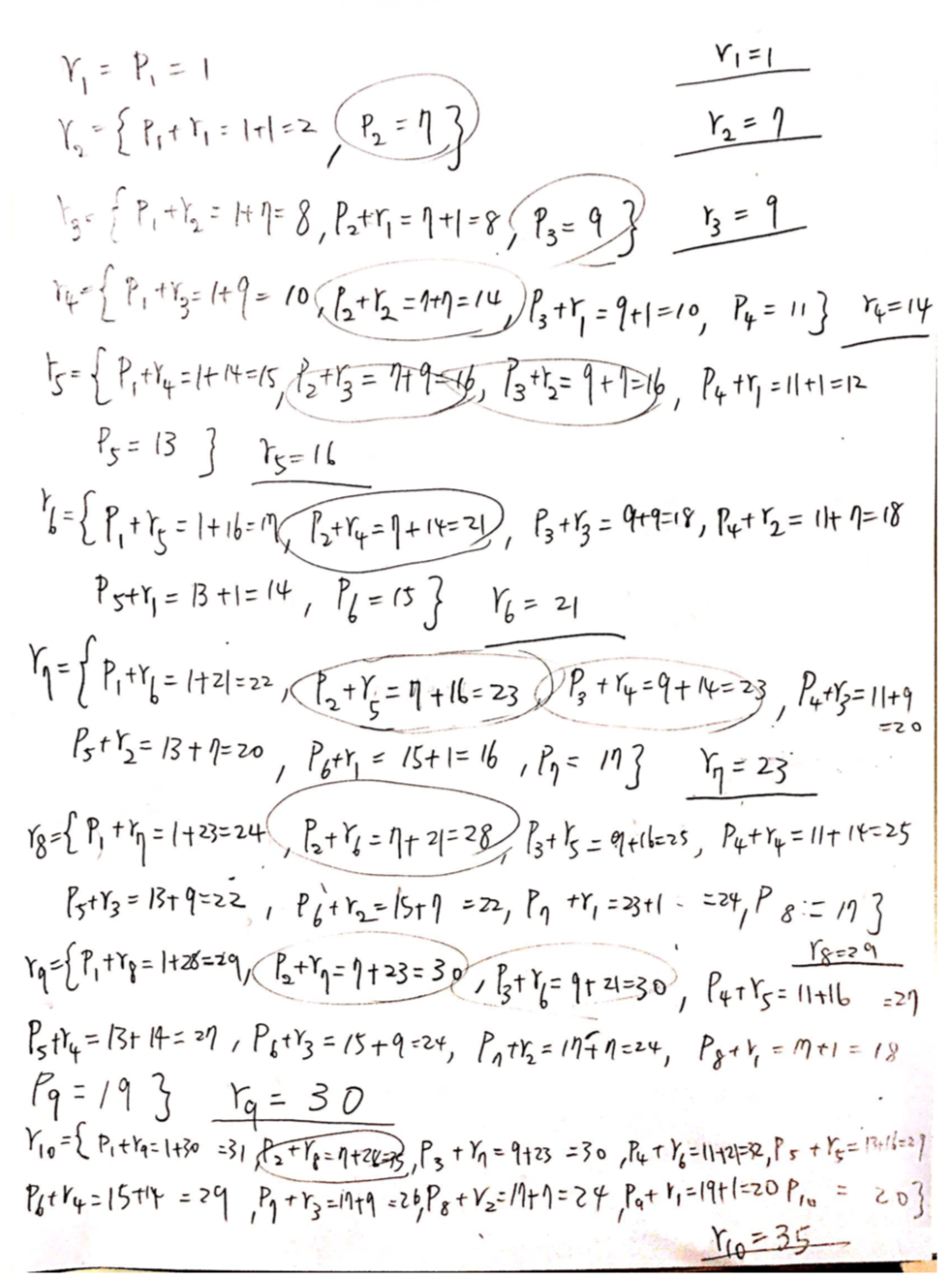
接著利用 Master theorem，分別得到

由上式可知，因此第二種分割的方式是最快的。

Strassen’s algorithm 的時間需要，因此此分割方式比Strassen’s algorithm快。

8.

這一題為dynamic programming的一個例子，參考上課投影片的遞迴函數，可以列出以下



上圖中圈起來的值就代表各個長度切割的最大利潤，利用bottom up的方式可求得r10=35，再利用遞迴關係式可以回推是由5個長度為2的鐵桿售價和。

