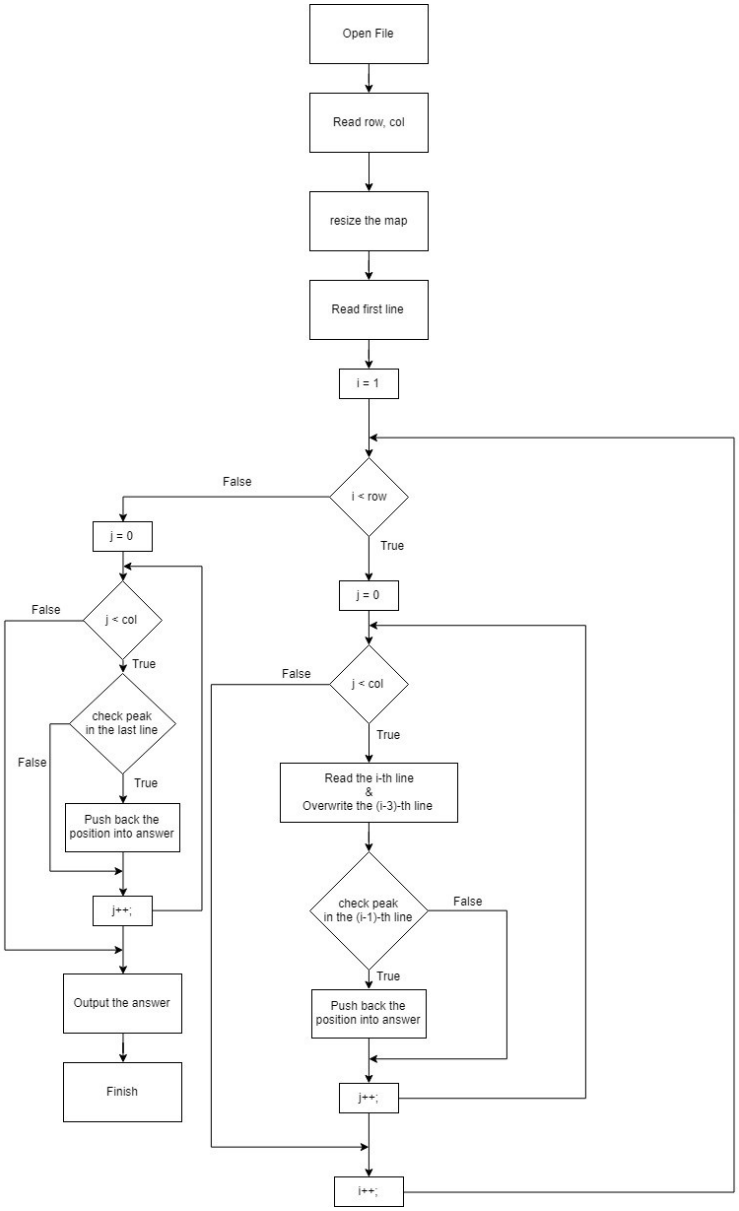
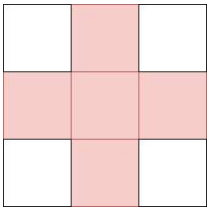
Project 1 Peak Finder Report

106062130 邱詠智

1. Project Description
   a. Flow chart



   b. Detailed Description



檢查一個點是否為 peak，至少需要上下左右跟自己共五個點，row 來看，至少要 3 個 rows。如右圖，紅色為必要情報。

第 i-th row 需要 (i-1)-th / (i+1)-th 的資料，同時進行讀入跟檢查，至少要同時儲存 2 row + 1 的資料。

實作上，我儲存了 3 rows，i-th row 不會影響 (i+3)-th 之後的檢查，因此讀入 i-th rows (i>3)，可以覆寫資料至 (i-3)-th row 以減少記憶體。

Peak 檢查：以下為 code

```
inline bool peak(short x, short y){
    bool valid = true;
    if( x > 0 )     valid &= test[x%3][y] >= test[(x+2)%3][y];
    if( x < m-1 )   valid &= test[x%3][y] >= test[(x+1)%3][y];
    if( y > 0 )     valid &= test[x%3][y] >= test[x%3][y-1];
    if( y < n-1 )   valid &= test[x%3][y] >= test[x%3][y+1];
    return valid;
}
```

(x, y) 代表 (row, col) 因為只儲存了 3 排， x%3 才是 x-th row 的資料。

(x+1) %3 = (x+1)-th row, (x+2)%3 = (x-1)-th row.

為了避免負數，不使用 (x-1)%3。

使用 and 運算子的原因在於，不想寫很多層的 if else。

Ex:

```
if( x == 0 && y == 0)          return ...;
if( x == row-1 && y == 0 )     return ...;
```

資料儲存： O(3*row + size of ans)

變數	資料型態
row size	short
col size	short
map	Vector< long int >
ans	Vector < pair< short, short> >

運算時間：

input : O(row*col)

檢查一個點是否為峰點：O(1)

output：O(size of ans)

總複雜度：O(row*col)

2. Test case Design

我的 Test case 為 1 * 712 的 matrix，內容基本上是亂數。

如果使用 3 排的寫法，有可能會忘記要檢查最後一排，因此只出 1 排。

中間有插入連續的 -777777，用來測試 >= ， 以及是否有檢查負數。

3. 感想

這次 project，只要小心謹慎，基本上不會出問題。
不論如何改良速度 big-O = O (row*col) 是沒辦法加速的。
空間複雜度也是相同的情況，我的方法 O (3 row + size of ans)，如果要繼續優化，最少也是到 O (2 row + 1 + size of ans)。
(0 <= size of ans <= row*col)

我個人認為 執行時間/記憶體 慢最快的 2 倍就算 0 分，有點過於苛刻了，因為這次 project 沒有能明顯改良效率的演算法/資料結構。