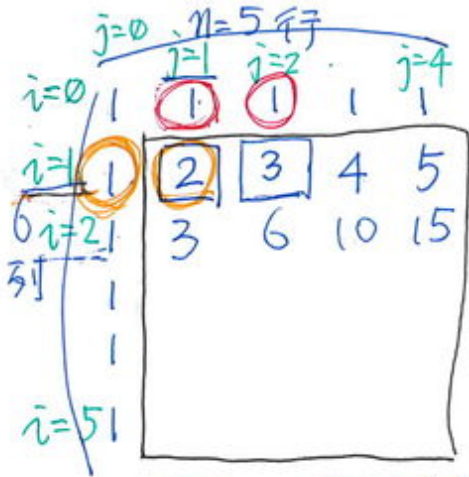


Dynamic Memory Allocation

$m=6$ $n=5$ Matrix: 6×5
($m \times n$)



$$f(i, j) = f(i-1, j) + f(i, j-1)$$

下方 左側

$$f(1, 2) = f(0, 2) + f(1, 1) = 1 + 2 = 3$$

初始: 第一列 & 第一行皆為 1

* 2 維陣列的表示形式

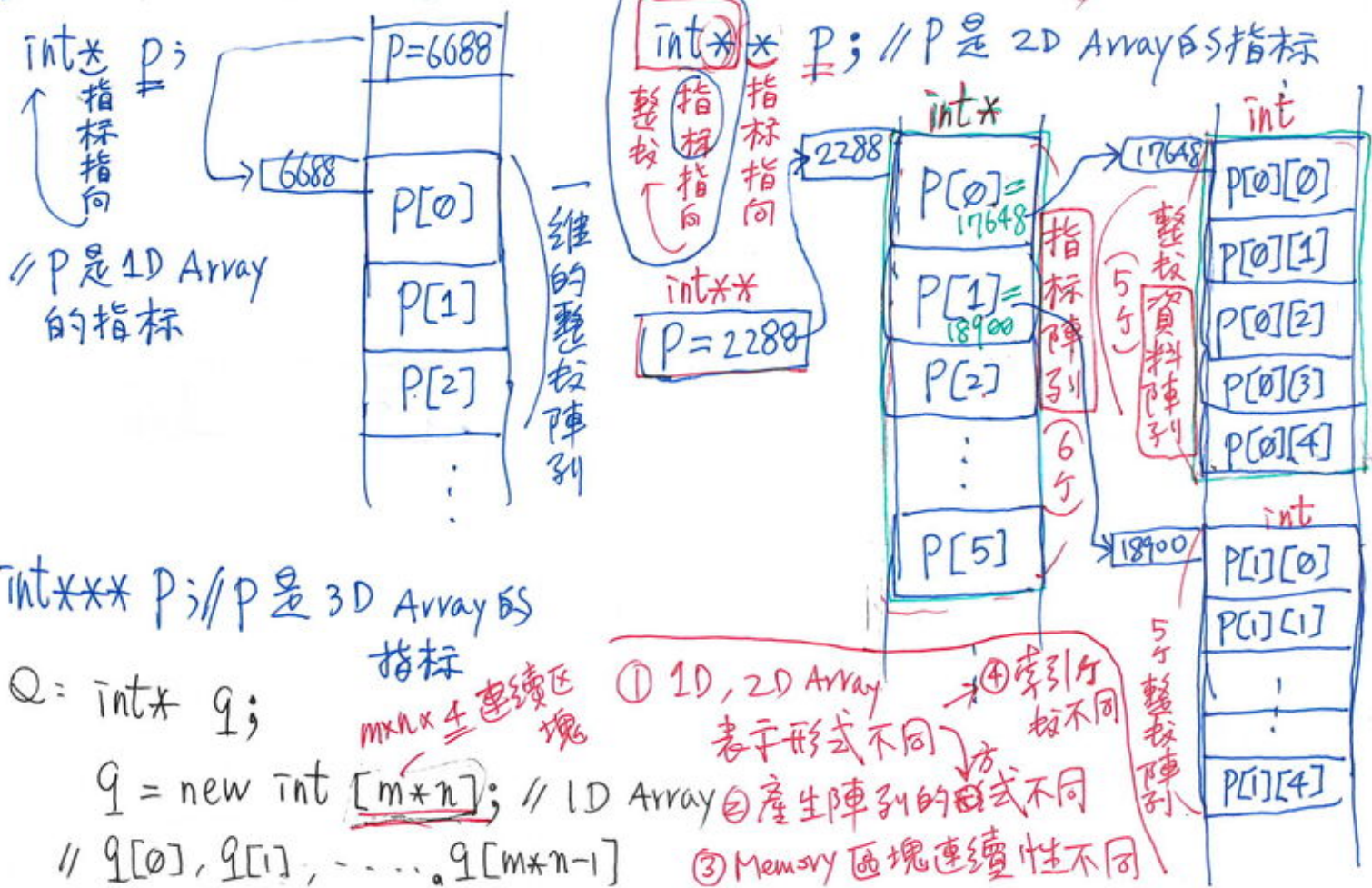
2014-May-27

資工-甲

* 2 維陣列如何產生 ... new/delete

* 2 維陣列填入數值 ... 遞迴公式

* 2 維陣列記憶體配置出錯 ... Memory 不足
 ... 缺少足夠的連續區段
 => C++ Exception Handling
 意外; 例外 處理



```
int** p = NULL;
```

建立 2D 陣列

```
P = new int* [m]; // 產生指標陣列
    指標
for (int i=0; i<m; i++) {
    P[i] = new int [n]; // 產生資料陣列
}
```

刪除 2D 陣列

```
for (int i=0; i<m; i++) {
    delete [] P[i]; // 刪除每個資料陣列
}
delete [] P; // 刪除指標陣列
P = NULL;
```

Exception Handling 意外處理

C: malloc/free, C++: new/delete

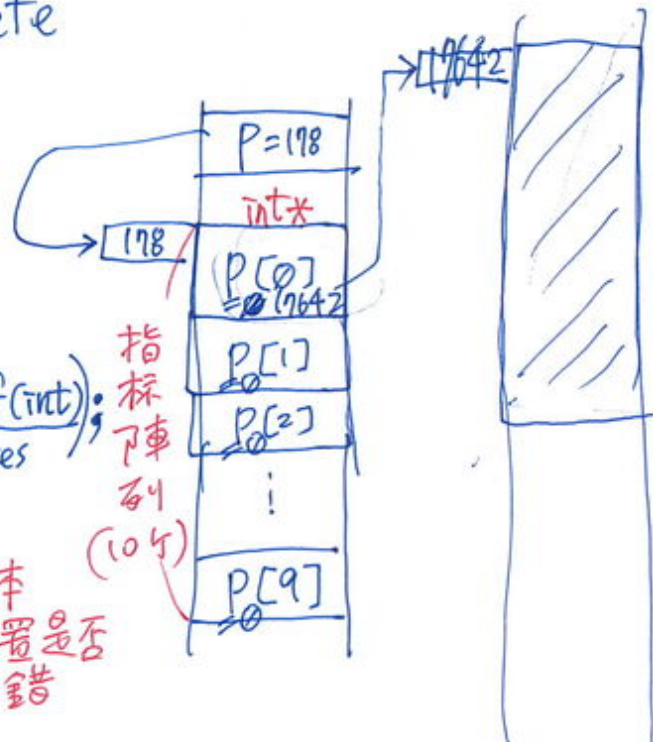
```
int* P[10] = {NULL}
```

指標
10個
指標陣列

```
for (int i=0; i<10; i++) {
    P[i] = (int*) malloc (102400000 * sizeof(int));
    (int*) 轉型 返回 (17642) addr. 4 bytes
    (void*)
    if (P[i] == NULL) // 檢查記憶體配置是否出錯
        cout << "Memory Error\n";
}
```

錯誤處理

```
for (int i=0; i<10; i++) {
    if (P[i] != NULL) { // P[i] 有成功配置 memory
        free (P[i]);
        P[i] = NULL;
    }
}
```



C++: new/delete

Unhandle Exception

```
int* p[10] = {NULL};
```

未處理 意外
丟出 訊息

```
for (int i=0; i<10; i++) {  
    p[i] = new int [102400000];  
    if (p[i] == NULL)  
        cout << i << ": mem error\n";  
}
```

改寫 \rightarrow `p[i] = new() int [102400000]`
nothrow
不丟意外訊息

```
for (int i=0; i<10; i++) {  
    delete [] p[i];  
    p[i] = NULL;  
}
```

刪除陣列 依 p[i] 的 Addr.

改寫: 攔截意外訊息

```
for (int i=0; i<10; i++) {  
    p[i] = new int [102400000];  
    try { // 試  
        p[i] = new int [102400000];  
    } catch (...) { // 各式各樣的意外  
        // 攔截 意外訊息  
        // 意外錯誤處理  
        cout << i << ": Memory Error\n";  
    }  
}
```