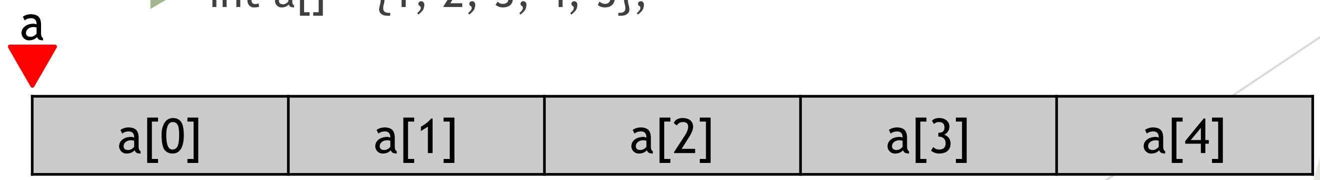
C/C++基礎程式設計 陣列與指標

講師：黃銀鵬

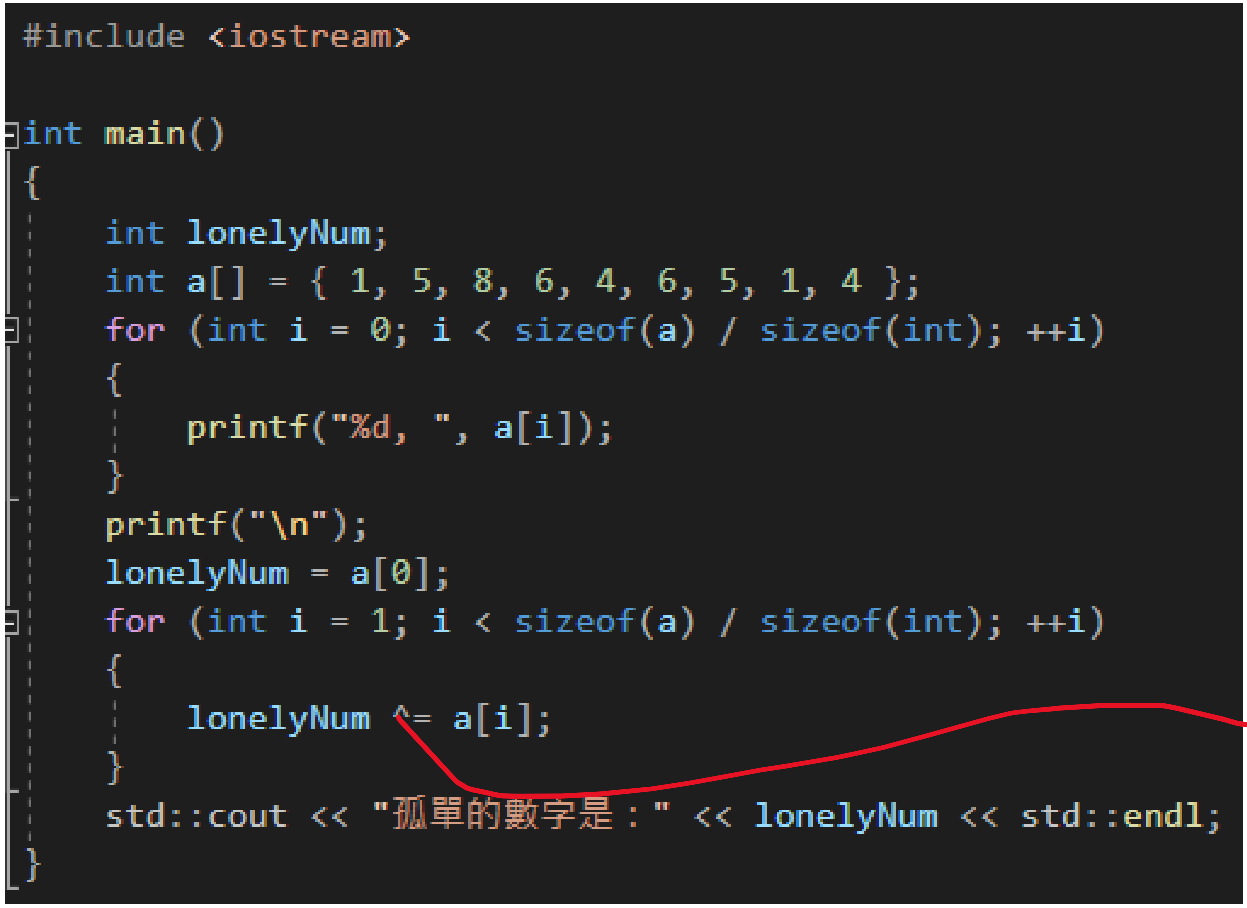
E-mail: yinpenghuang@gmail.com

陣列(算資料的容器)

- 為相同型別的變數集合。
- 陣列名稱代表著陣列起始位置。
- 長度為n的陣列，索引值從0開始，至n-1結束。
- 不可使用變數作為陣列長度。(C99版本原本支援，但C11後列為非必要，因此編譯器可能不支援)
- 一維陣列宣告方式： int a[5];
- 一維陣列初始化方式：
  - int a[5] = {0};
  - int a[5] = {1, 2, 3, 4, 5};
  - int a[] = {1, 2, 3, 4, 5};

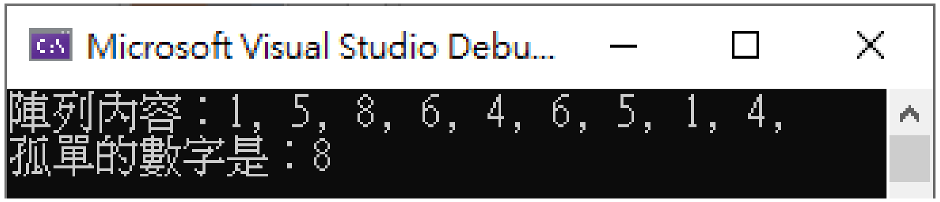


01- 孤單的數字



```cpp
#include <iostream>

int main()
{
    int lonelyNum;
    int a[] = { 1, 5, 8, 6, 4, 6, 5, 1, 4 };
    for (int i = 0; i < sizeof(a) / sizeof(int); ++i)
    {
        printf("%d, ", a[i]);
    }
    printf("\n");
    lonelyNum = a[0];
    for (int i = 1; i < sizeof(a) / sizeof(int); ++i)
    {
        lonelyNum ^= a[i];
    }
    std::cout << "孤單的數字是：" << lonelyNum << std::endl;
}
```

都有引響互相消除，重複出現，會互相消除



```
陣列內容：1, 5, 8, 6, 4, 6, 5, 1, 4,
孤單的數字是：8
```

練習題：計算總分與平均

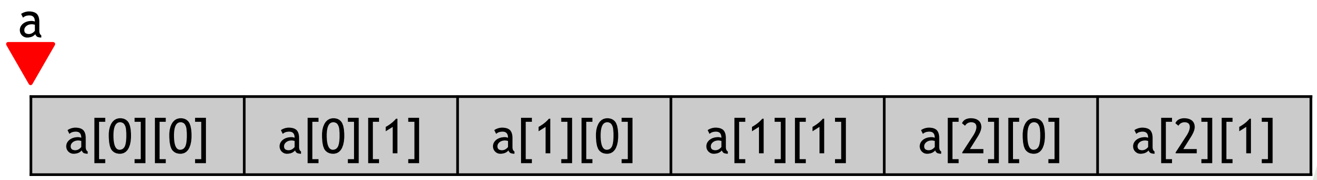
- 一個班級有10位學生，請寫一個程式輸入每位學生的成績，並計算班級總分與平均後顯示。
  - std::cin >> A[i];

二維陣列

電腦取代打字機，英文書寫從左上開始，電腦原點從左上開始

- 二維陣列宣告方式： int a[3][2];
- 二維陣列初始化方式：
  - int a[3][2] = {0};
  - int a[3][2] = {1, 2, 3, 4, 5, 6};
  - int a[3][2] = {{1, 2}, {3, 4}, {5, 6}};
  - int a[ ][2] = {{1, 2}, {3, 4}, {5, 6}};

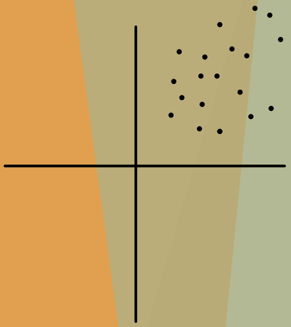
a[0][0]	a[0][1]
a[1][0]	a[1][1]
a[2][0]	a[2][1]



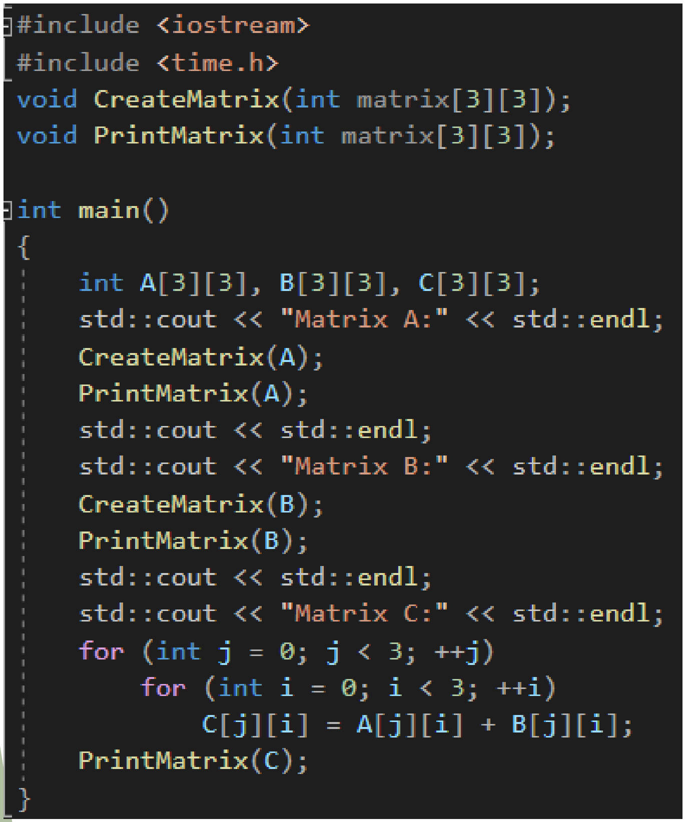
記憶體是線性的

2- 矩陣相加

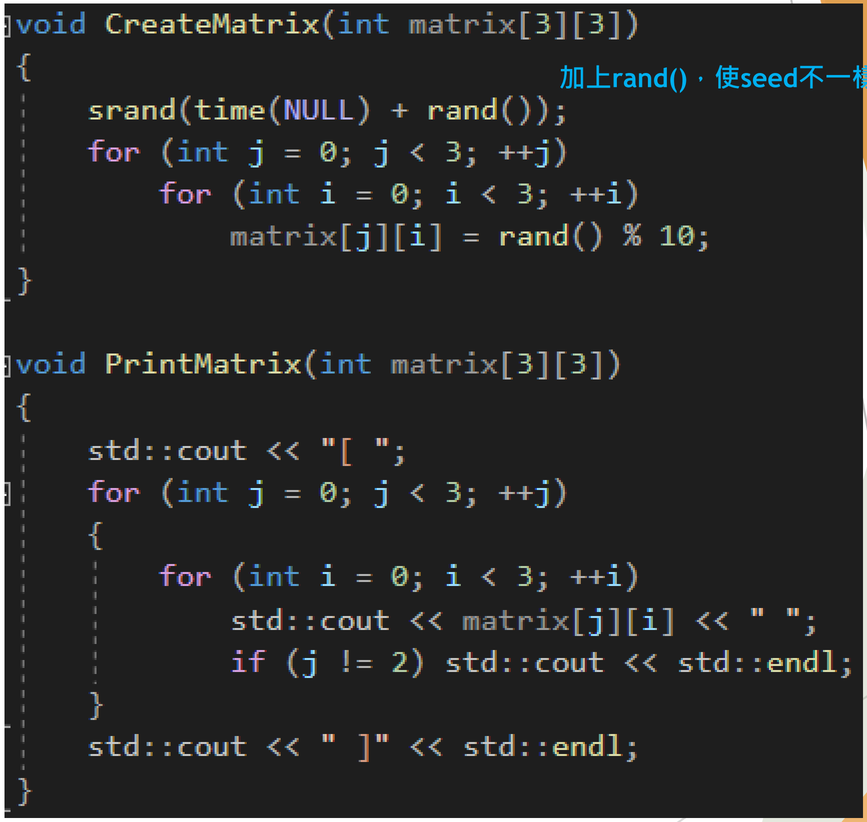
分佈在四個象限，無法形成圖形，
但時間還不夠亂，線性增長



數位電腦內的時間非連續的，而是離散的，快速取了兩個時間，送進產生亂數，會產生一樣的亂數，現在電腦越來越快

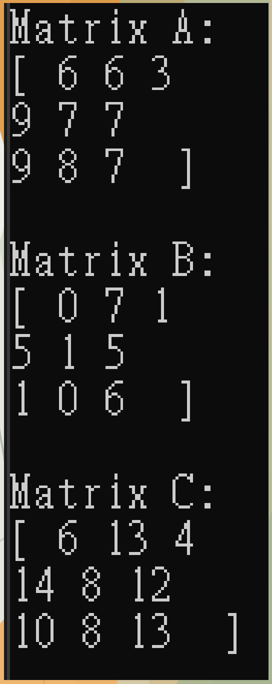


```cpp
#include <iostream>
#include <time.h>
void CreateMatrix(int matrix[3][3]);
void PrintMatrix(int matrix[3][3]);

int main()
{
    int A[3][3], B[3][3], C[3][3];
    std::cout << "Matrix A:" << std::endl;
    CreateMatrix(A);
    PrintMatrix(A);
    std::cout << std::endl;
    std::cout << "Matrix B:" << std::endl;
    CreateMatrix(B);
    PrintMatrix(B);
    std::cout << std::endl;
    std::cout << "Matrix C:" << std::endl;
    for (int j = 0; j < 3; ++j)
        for (int i = 0; i < 3; ++i)
            C[j][i] = A[j][i] + B[j][i];
    PrintMatrix(C);
}
```



加上rand()，使seed不一樣

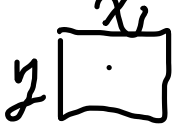
```cpp
void CreateMatrix(int matrix[3][3])
{
    srand(time(NULL) + rand());
    for (int j = 0; j < 3; ++j)
        for (int i = 0; i < 3; ++i)
            matrix[j][i] = rand() % 10;
}

void PrintMatrix(int matrix[3][3])
{
    std::cout << "[ ";
    for (int j = 0; j < 3; ++j)
    {
        for (int i = 0; i < 3; ++i)
            std::cout << matrix[j][i] << " ";
            if (j != 2) std::cout << std::endl;
    }
    std::cout << " ]" << std::endl;
}
```



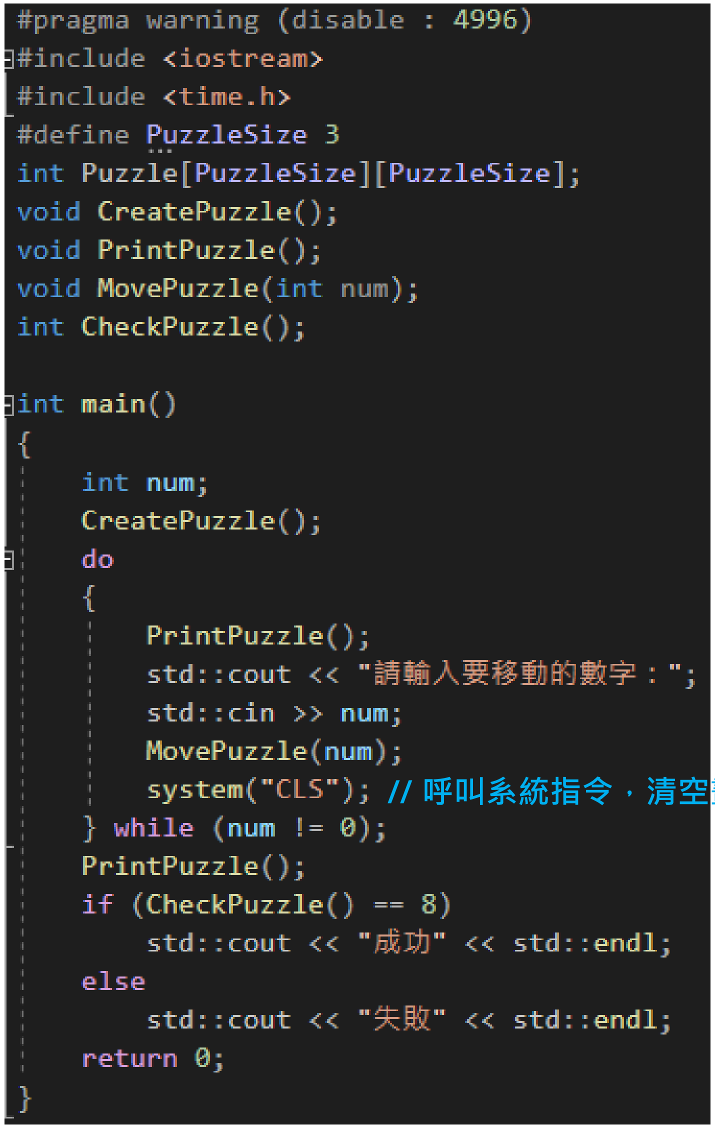
```
Matrix A:
[ 6 6 3
9 7 7
9 8 7   ]

Matrix B:
[ 0 7 1
5 1 5
1 0 6   ]

Matrix C:
[ 6 13 4
14 8 12
10 8 13   ]
```

3- 數字拼圖

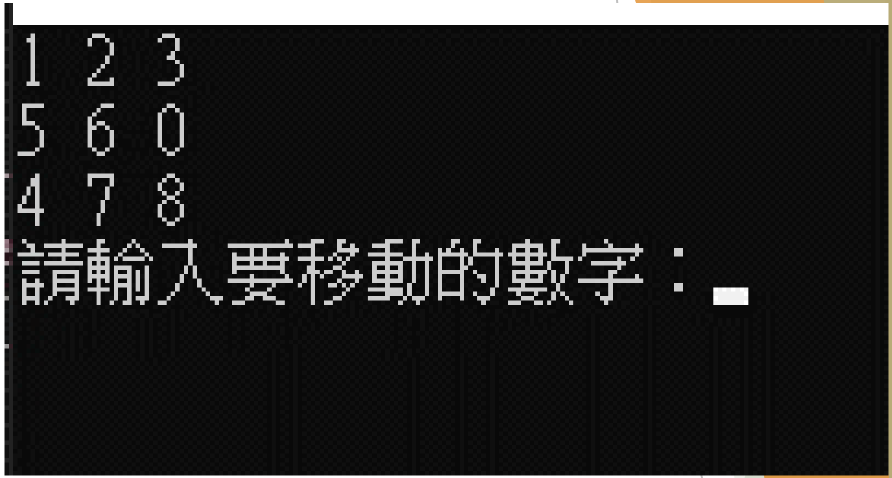
1D to 2D:
y_prime = i/x
x_prime = I mod(x);



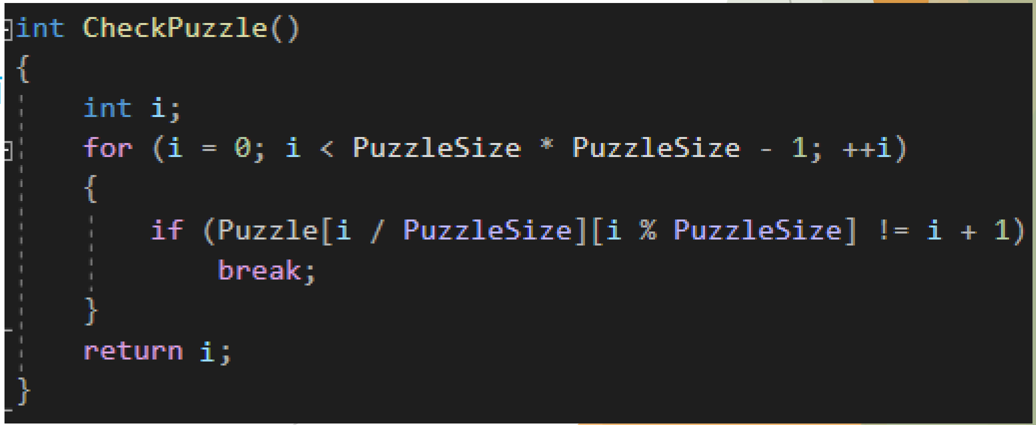
2D to 1D:
i = y_prime*x + x_prime



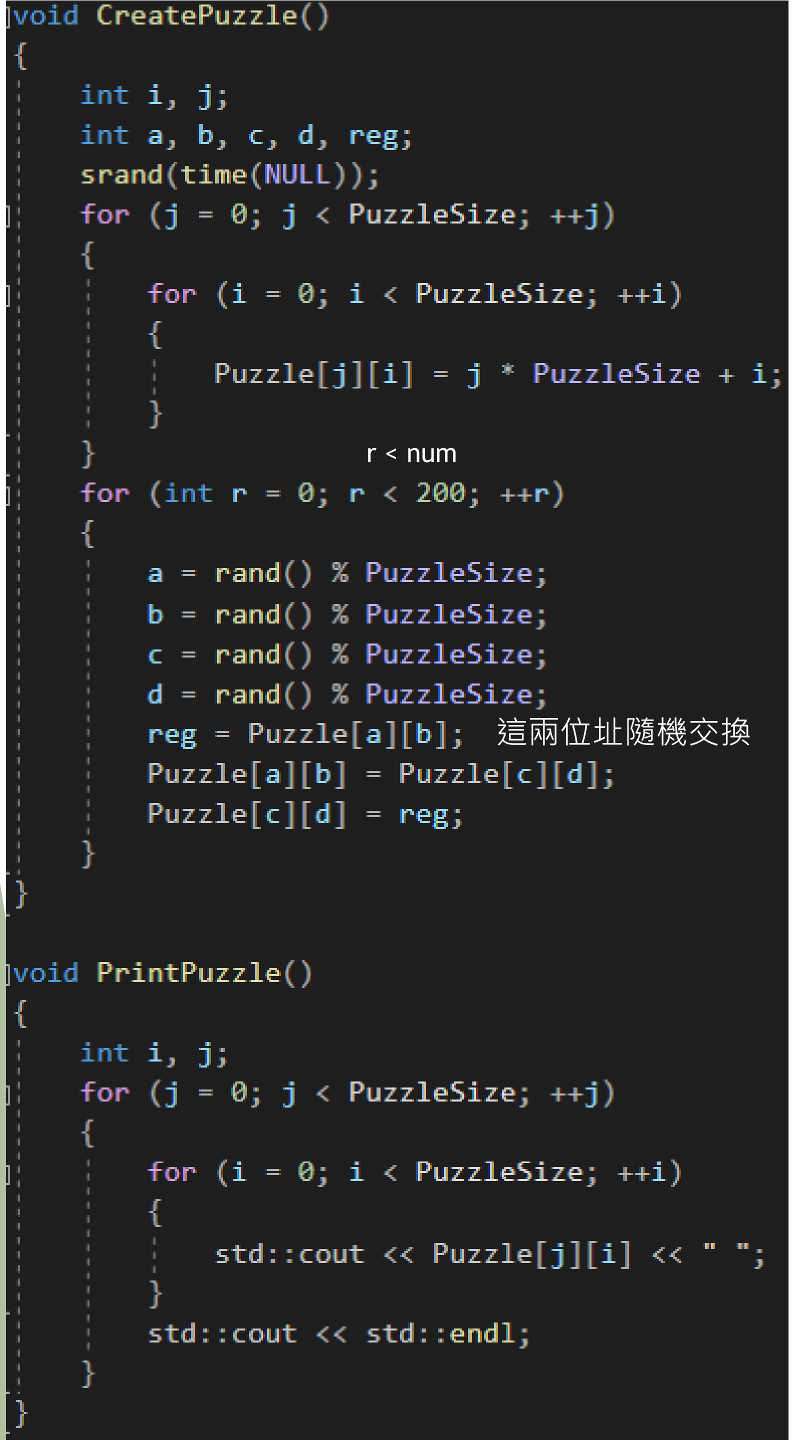
```
#pragma warning (disable : 4996)
#include <iostream>
#include <time.h>
#define PuzzleSize 3
int Puzzle[PuzzleSize][PuzzleSize];
void CreatePuzzle();
void PrintPuzzle();
void MovePuzzle(int num);
int CheckPuzzle();

int main()
{
    int num;
    CreatePuzzle();
    do
    {
        PrintPuzzle();
        std::cout << "請輸入要移動的數字：";
        std::cin >> num;
        MovePuzzle(num);
        system("CLS"); // 呼叫系統指令，清空畫面
    } while (num != 0);
    PrintPuzzle();
    if (CheckPuzzle() == 8)
        std::cout << "成功" << std::endl;
    else
        std::cout << "失敗" << std::endl;
    return 0;
}
```



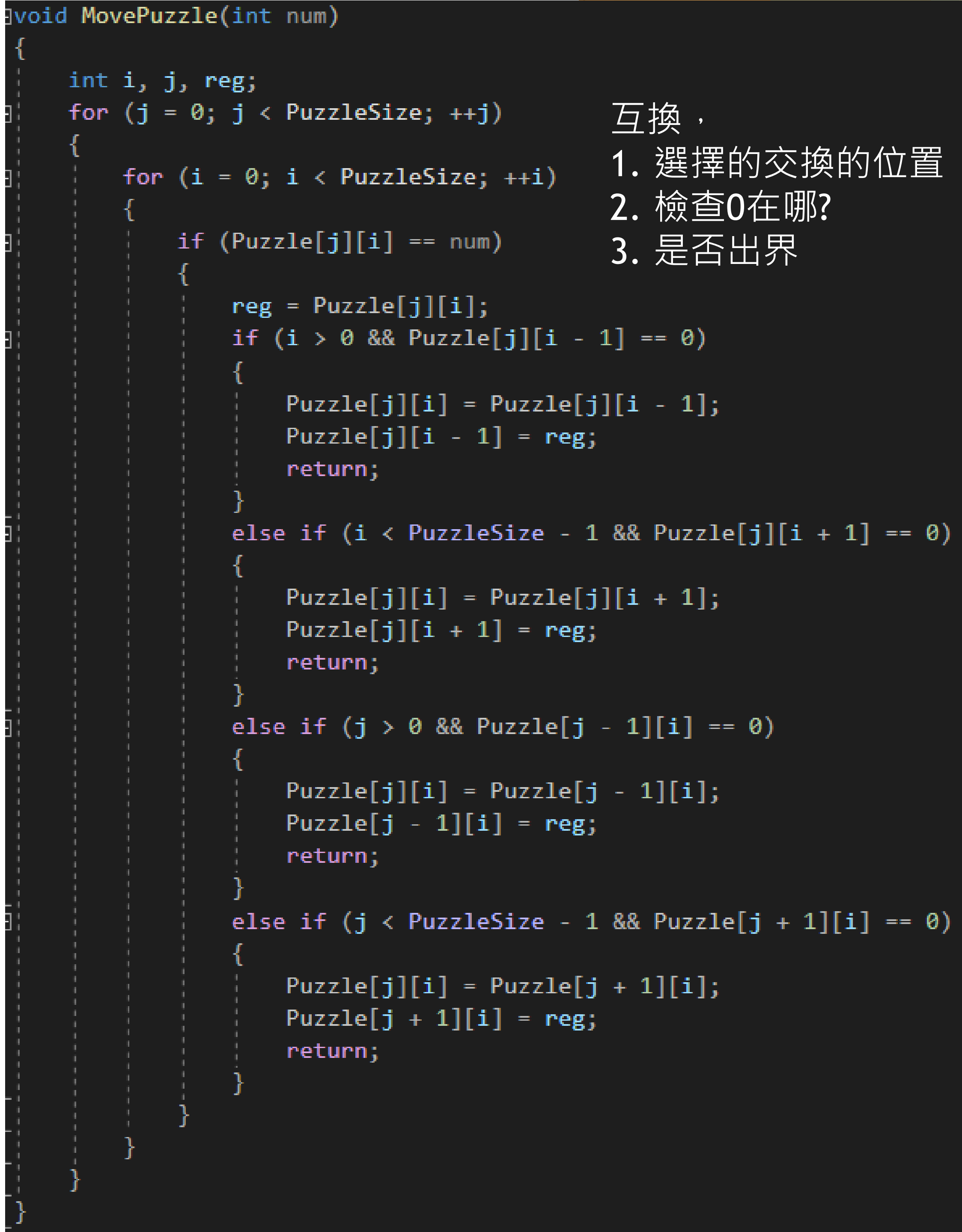
```
1 2 3
5 6 0
4 7 8
請輸入要移動的數字：_
```



```
int CheckPuzzle()
{
    int i;
    for (i = 0; i < PuzzleSize * PuzzleSize - 1; ++i)
    {
        if (Puzzle[i / PuzzleSize][i % PuzzleSize] != i + 1)
            break;
    }
    return i;
}
```



```
void CreatePuzzle()
{
    int i, j;
    int a, b, c, d, reg;
    srand(time(NULL));
    for (j = 0; j < PuzzleSize; ++j)
    {
        for (i = 0; i < PuzzleSize; ++i)
        {
            Puzzle[j][i] = j * PuzzleSize + i;
        }
    }                     r < num
    for (int r = 0; r < 200; ++r)
    {
        a = rand() % PuzzleSize;
        b = rand() % PuzzleSize;
        c = rand() % PuzzleSize;
        d = rand() % PuzzleSize;
        reg = Puzzle[a][b];  這兩位址隨機交換
        Puzzle[a][b] = Puzzle[c][d];
        Puzzle[c][d] = reg;
    }
}

void PrintPuzzle()
{
    int i, j;
    for (j = 0; j < PuzzleSize; ++j)
    {
        for (i = 0; i < PuzzleSize; ++i)
        {
            std::cout << Puzzle[j][i] << " ";
        }
        std::cout << std::endl;
    }
}
```



```
void MovePuzzle(int num)
{
    int i, j, reg;
    for (j = 0; j < PuzzleSize; ++j)
    {
        for (i = 0; i < PuzzleSize; ++i)
        {
            if (Puzzle[j][i] == num)
            {
                reg = Puzzle[j][i];
                if (i > 0 && Puzzle[j][i - 1] == 0)
                {
                    Puzzle[j][i] = Puzzle[j][i - 1];
                    Puzzle[j][i - 1] = reg;
                    return;
                }
                else if (i < PuzzleSize - 1 && Puzzle[j][i + 1] == 0)
                {
                    Puzzle[j][i] = Puzzle[j][i + 1];
                    Puzzle[j][i + 1] = reg;
                    return;
                }
                else if (j > 0 && Puzzle[j - 1][i] == 0)
                {
                    Puzzle[j][i] = Puzzle[j - 1][i];
                    Puzzle[j - 1][i] = reg;
                    return;
                }
                else if (j < PuzzleSize - 1 && Puzzle[j + 1][i] == 0)
                {
                    Puzzle[j][i] = Puzzle[j + 1][i];
                    Puzzle[j + 1][i] = reg;
                    return;
                }
            }
        }
    }
}
```

互換，
1. 選擇的交換的位置
2. 檢查0在哪?
3. 是否出界

洗牌法

2d to 1d
0 1 2
3 4 5
6 7 8
先填好0-8，再做交換

Rand $\boxed{\text{a, b location}}$(x, y 座標)

去互換

$\boxed{\text{c, d location}}$(x, y 座標)

這樣就可以確定執行次數了!(交換的次數)

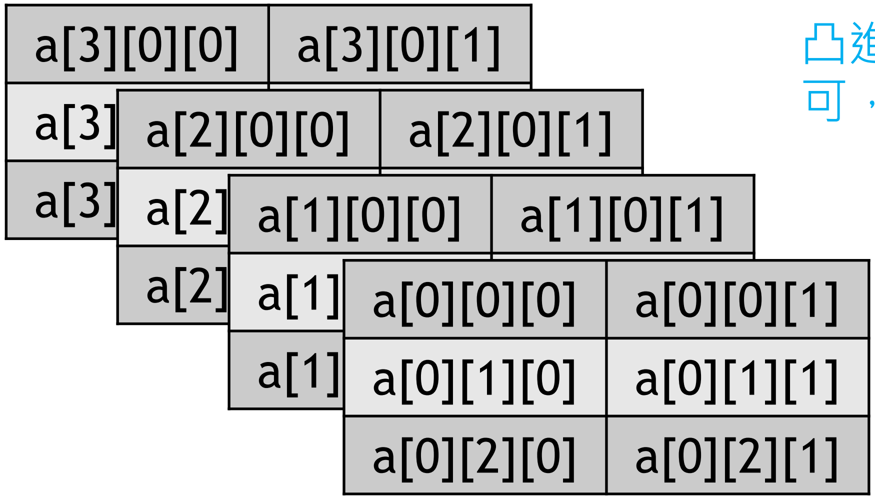
練習題：計算各班平均

- 現有3個班各5位學生，請寫一程式可以輸入每位學生成績，並計算出各班平均。
  - std::cin >> A[j][i];

三維陣列

讀檔，如果初始
化會初始不完，
一維比較會

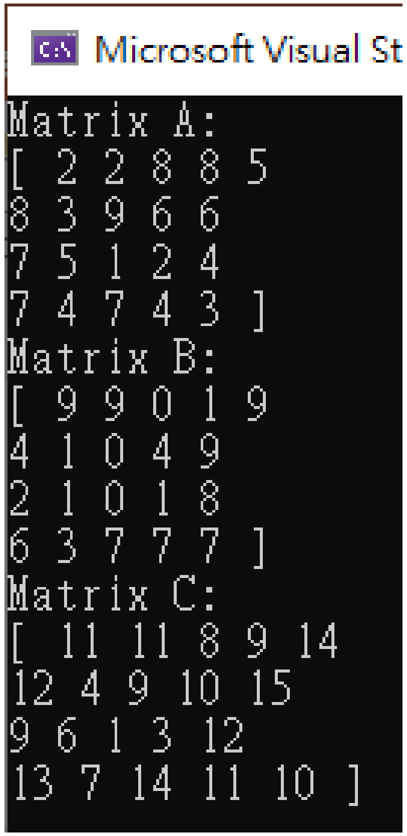
- 三維陣列宣告方式： int a[4][3][2];



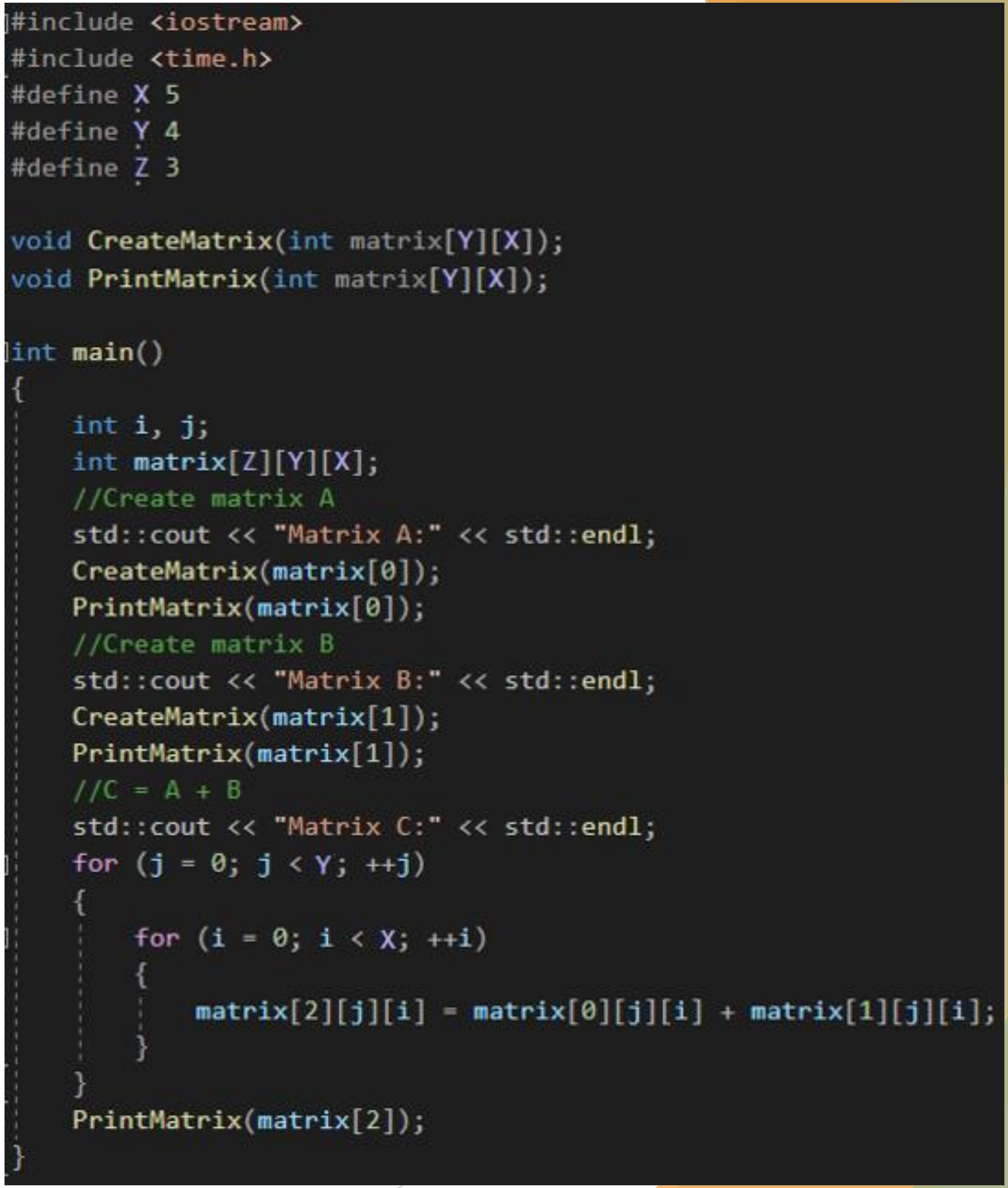
凸進來，或凹進去皆
可，根據需求

4- 矩陣相加

魔術方塊解題演算法

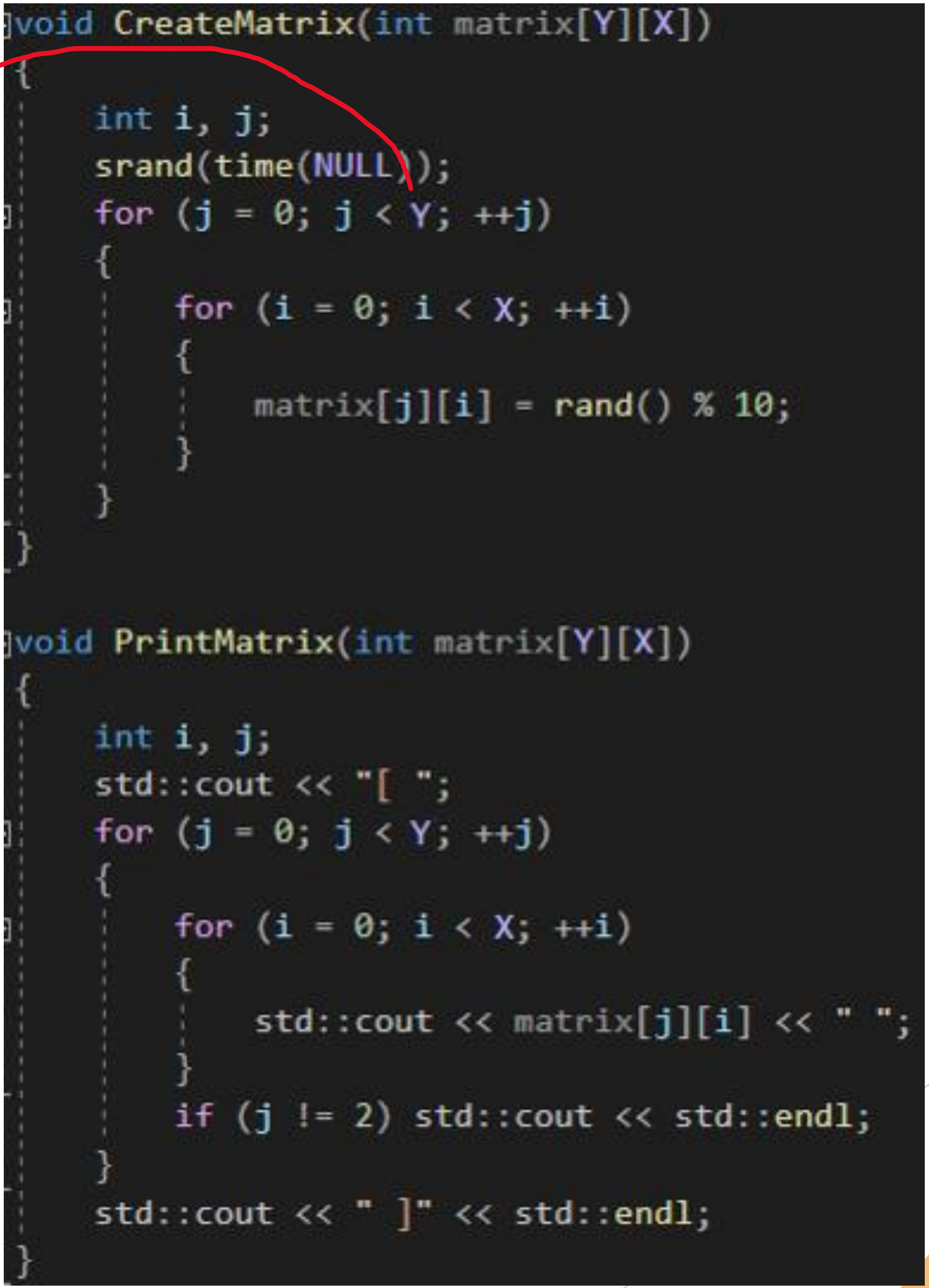


```
Matrix A:
[ 2 2 8 8 5
8 3 9 6 6
7 5 1 2 4
7 4 7 4 3 ]
Matrix B:
[ 9 9 0 1 9
4 1 0 4 9
2 1 0 1 8
6 3 7 7 7 ]
Matrix C:
[ 11 11 8 9 14
12 4 9 10 15
9 6 1 3 12
13 7 14 11 10 ]
```



```
#include <iostream>
#include <time.h>
#define X 5
#define Y 4
#define Z 3

void CreateMatrix(int matrix[Y][X]);
void PrintMatrix(int matrix[Y][X]);

int main()
{
    int i, j;
    int matrix[Z][Y][X];
    //Create matrix A
    std::cout << "Matrix A:" << std::endl;
    CreateMatrix(matrix[0]);
    PrintMatrix(matrix[0]);
    //Create matrix B
    std::cout << "Matrix B:" << std::endl;
    CreateMatrix(matrix[1]);
    PrintMatrix(matrix[1]);
    //C = A + B
    std::cout << "Matrix C:" << std::endl;
    for (j = 0; j < Y; ++j)
    {
        for (i = 0; i < X; ++i)
        {
            matrix[2][j][i] = matrix[0][j][i] + matrix[1][j][i];
        }
    }
    PrintMatrix(matrix[2]);
}
```

會有重複地要加rand()



```
void CreateMatrix(int matrix[Y][X])
{
    int i, j;
    srand(time(NULL));
    for (j = 0; j < Y; ++j)
    {
        for (i = 0; i < X; ++i)
        {
            matrix[j][i] = rand() % 10;
        }
    }
}

void PrintMatrix(int matrix[Y][X])
{
    int i, j;
    std::cout << "[ ";
    for (j = 0; j < Y; ++j)
    {
        for (i = 0; i < X; ++i)
        {
            std::cout << matrix[j][i] << " ";
        }
        if (j != 2) std::cout << std::endl;
    }
    std::cout << " ]" << std::endl;
}
```

練習題：計算每人每班平均

- 現有3個班各4位學生，每位學生有3個科目，請寫一程式可以輸入每位學生每科成績，並計算出每人及各班平均。
  - std::cin >> A[k][j][i];

指標-動態記憶體配置(一定要會)

- 配置記憶體使用new運算子。(看堆積區夠不夠，沒用到的堆疊區以外都是)
- 釋放記憶體使用delete運算子。
- 動態配置單一變數空間：

```
int* pa = new int;
delete pa;
```

- 動態配置陣列變數空間：

```
int* pa = new int[size];
delete[] pa;
```

配置物件(C++怪)
new(新增)物件(C++好)

動態記憶體配置公式

標記

int * P

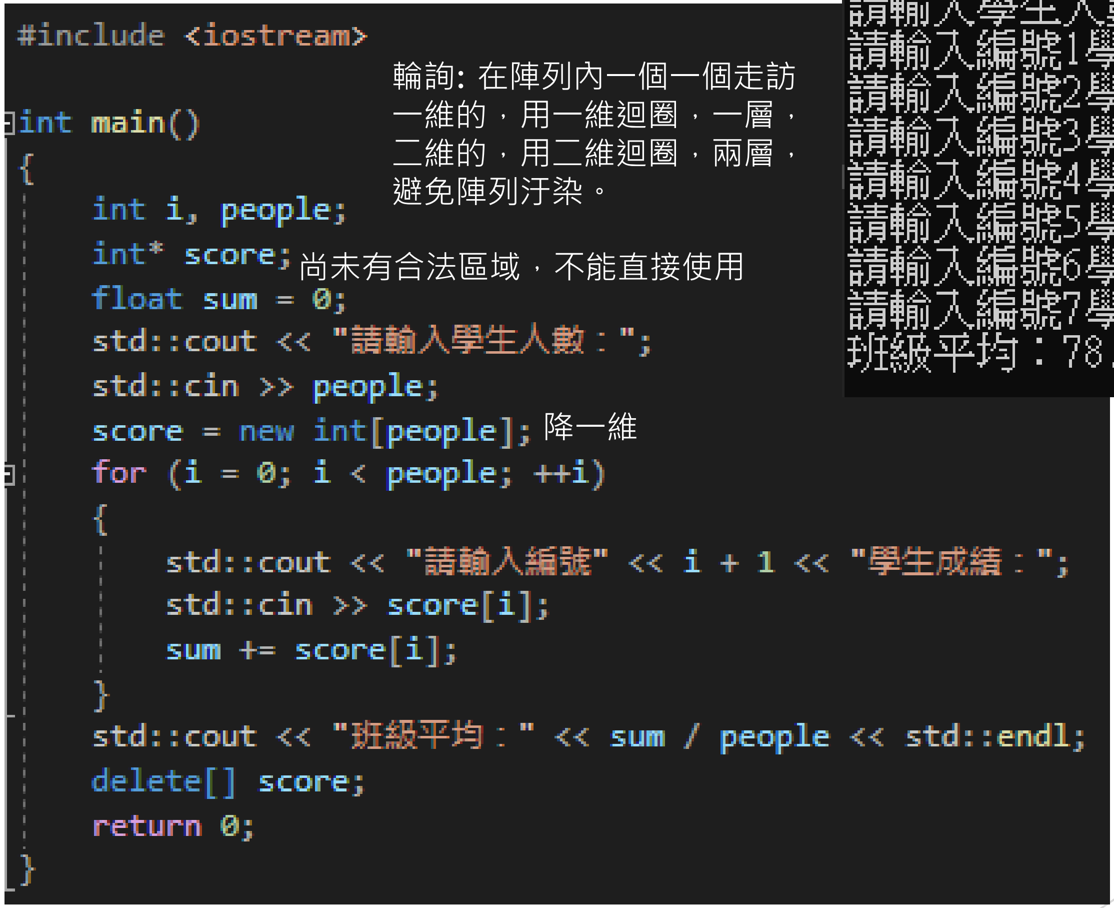
後面的為指標

- 降維配置，一次只降一維。(老師說的0維是變數，沒有負的，一定要降到0維，這樣才能存放東西)
- 每次配置是連續且線性空間。
- 次與次配置空間不保證其連續性。(老師說，把它當成不連續，空間是不能延伸的)
- 升維釋放，一次釋放一維。(高一維釋放，降低一維)
- 一組方括號[]，降一維指標。(跟陣列的用法相同)

學資料結構，適合練基礎語法

03- 一維指標記憶體配置 計算班平均

陣列進堆疊區，陣列是不能改變陣列大小，單純的動態記憶體配置不能延長，不能保證第一次配置跟第二次配置不保證連續

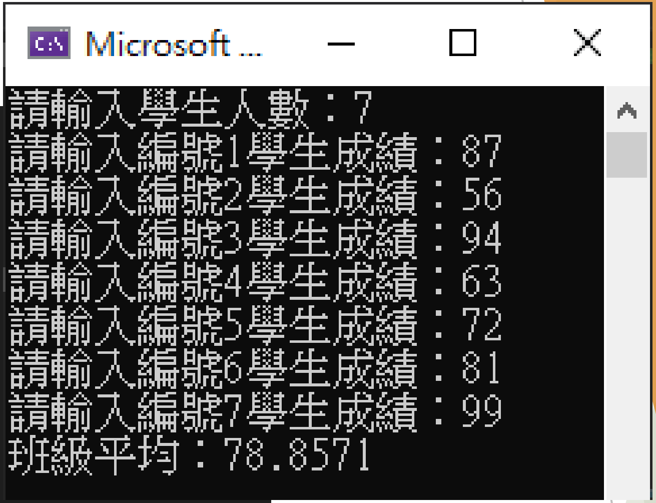


```
#include <iostream>

int main()
{
    int i, people;
    int* score;
    float sum = 0;
    std::cout << "請輸入學生人數：";
    std::cin >> people;
    score = new int[people];
    for (i = 0; i < people; ++i)
    {
        std::cout << "請輸入編號" << i + 1 << "學生成績：";
        std::cin >> score[i];
        sum += score[i];
    }
    std::cout << "班級平均：" << sum / people << std::endl;
    delete[] score;
    return 0;
}
```

輪詢: 在陣列內一個一個走訪
一維的，用一維迴圈，一層，
二維的，用二維迴圈，兩層，
避免陣列汙染。

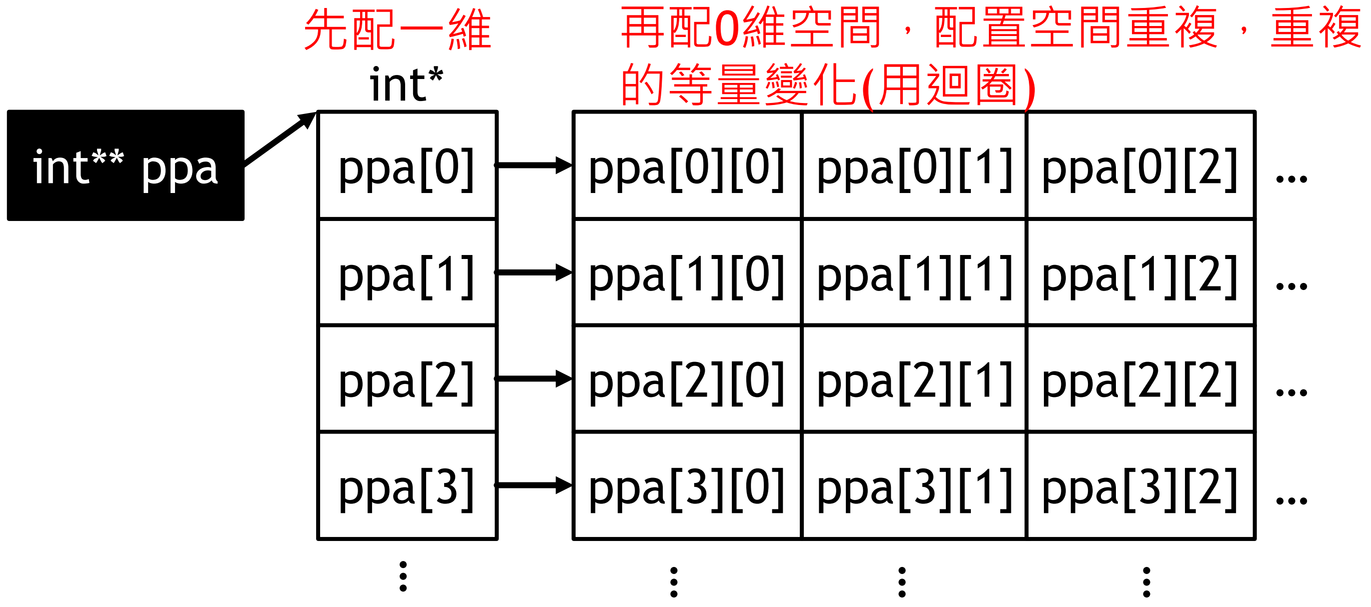
尚未有合法區域，不能直接使用

降一維



```
請輸入學生人數：7
請輸入編號1學生成績：87
請輸入編號2學生成績：56
請輸入編號3學生成績：94
請輸入編號4學生成績：63
請輸入編號5學生成績：72
請輸入編號6學生成績：81
請輸入編號7學生成績：99
班級平均：78.8571
```

二維指標記憶體配置



先配一維
int*
再配0維空間，配置空間重複，重複的等量變化(用迴圈)
int** ppa
ppa[0]
ppa[1]
ppa[2]
ppa[3]
ppa[0][0] ppa[0][1] ppa[0][2] …
ppa[1][0] ppa[1][1] ppa[1][2] …
ppa[2][0] ppa[2][1] ppa[2][2] …
ppa[3][0] ppa[3][1] ppa[3][2] …

每個橫條都是獨立的，不一定連續的

二維動態記憶體配置

```
int** ppa = new int*[y];
for(i=0; i<y; ++i)
    ppa[i] = new int[x];
```

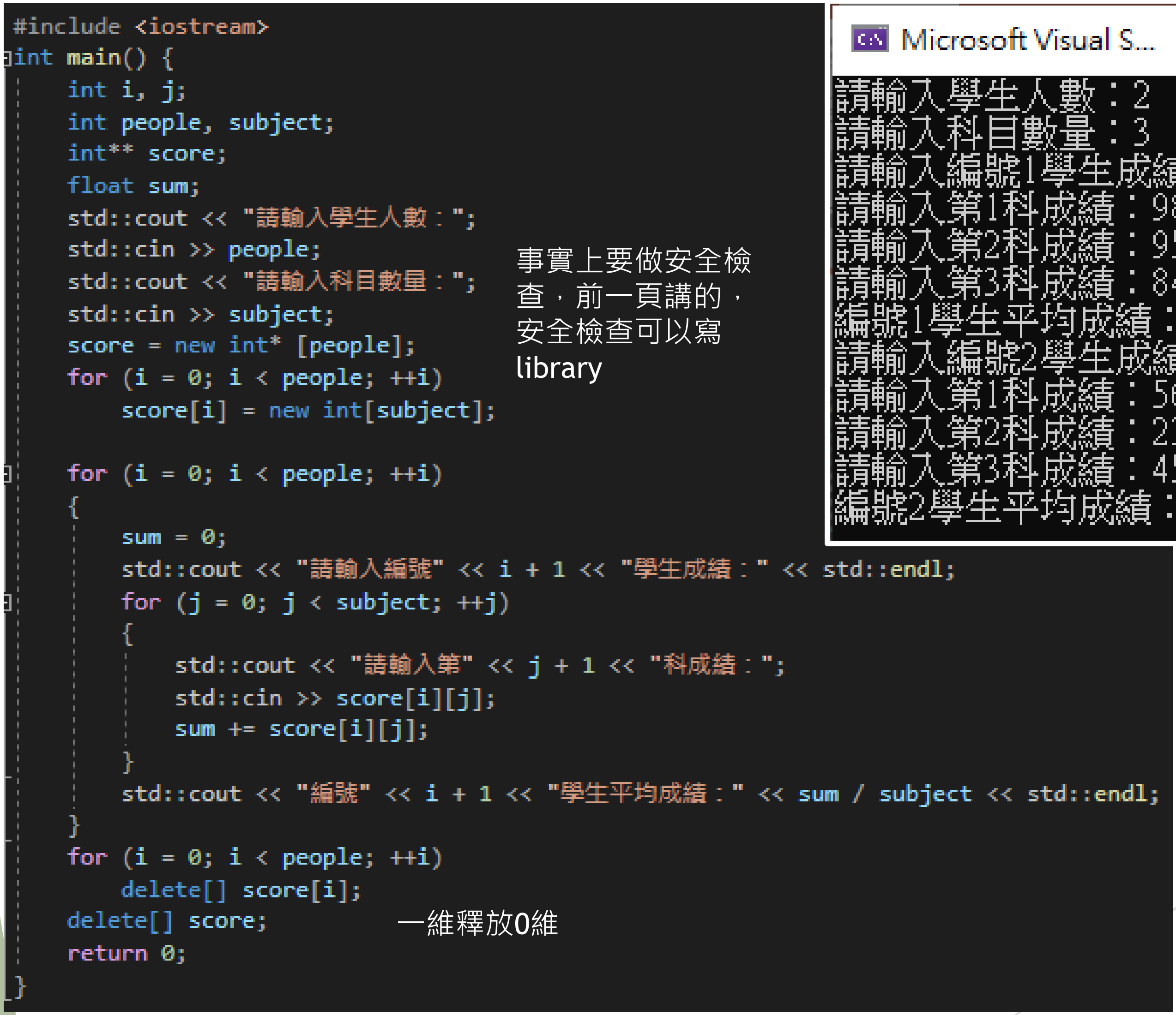
二維動態記憶體釋放

```
for(i=0; i<y; ++i)
    delete[] ppa[i];
delete[] ppa;
```

可以建立自己常用的Library，寫function，來配置空間

new失敗怎麼辦，後續要怎麼後續處理？前提:同步出現，前面都要刪除，成功的話有值，失敗是NULL

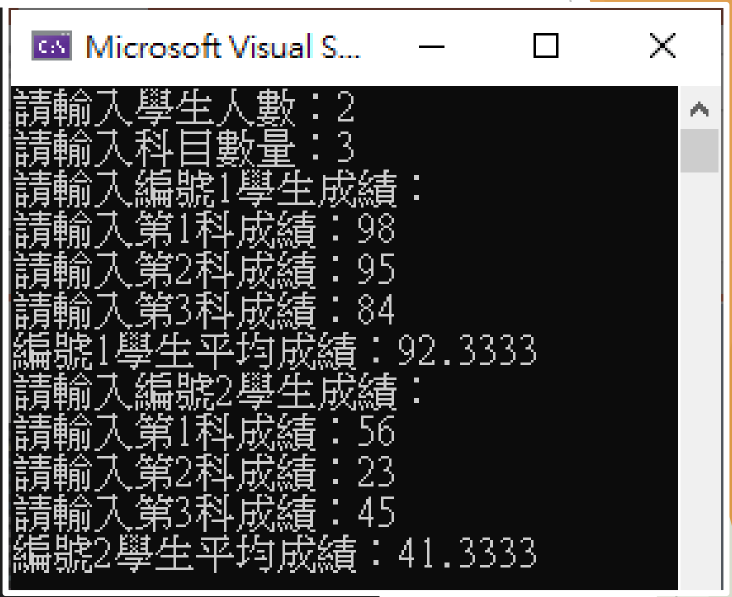
04- 計算個人平均



```cpp
#include <iostream>
int main() {
    int i, j;
    int people, subject;
    int** score;
    float sum;
    std::cout << "請輸入學生人數：";
    std::cin >> people;
    std::cout << "請輸入科目數量：";
    std::cin >> subject;
    score = new int* [people];
    for (i = 0; i < people; ++i)
        score[i] = new int[subject];

    for (i = 0; i < people; ++i)
    {
        sum = 0;
        std::cout << "請輸入編號" << i + 1 << "學生成績：" << std::endl;
        for (j = 0; j < subject; ++j)
        {
            std::cout << "請輸入第" << j + 1 << "科成績：";
            std::cin >> score[i][j];
            sum += score[i][j];
        }
        std::cout << "編號" << i + 1 << "學生平均成績：" << sum / subject << std::endl;
    }
    for (i = 0; i < people; ++i)
        delete[] score[i];
    delete[] score;
    return 0;
}
```

事實上要做安全檢查，前一頁講的，安全檢查可以寫library

一維釋放0維

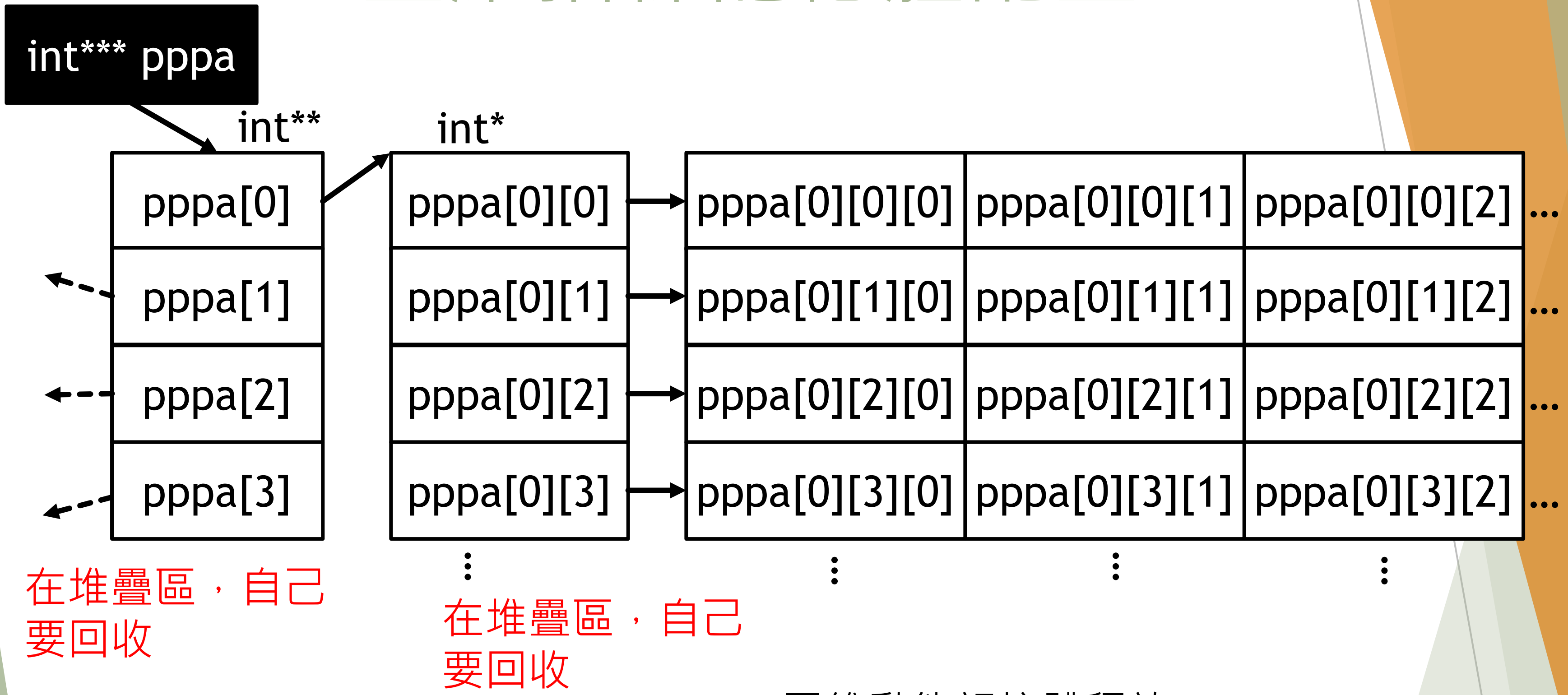


Microsoft Visual S...

```
請輸入學生人數：2
請輸入科目數量：3
請輸入編號1學生成績：
請輸入第1科成績：98
請輸入第2科成績：95
請輸入第3科成績：84
編號1學生平均成績：92.3333
請輸入編號2學生成績：
請輸入第1科成績：56
請輸入第2科成績：23
請輸入第3科成績：45
編號2學生平均成績：41.3333
```

在堆積區，生命週期
結束，自動刪除

三維指標記憶體配置



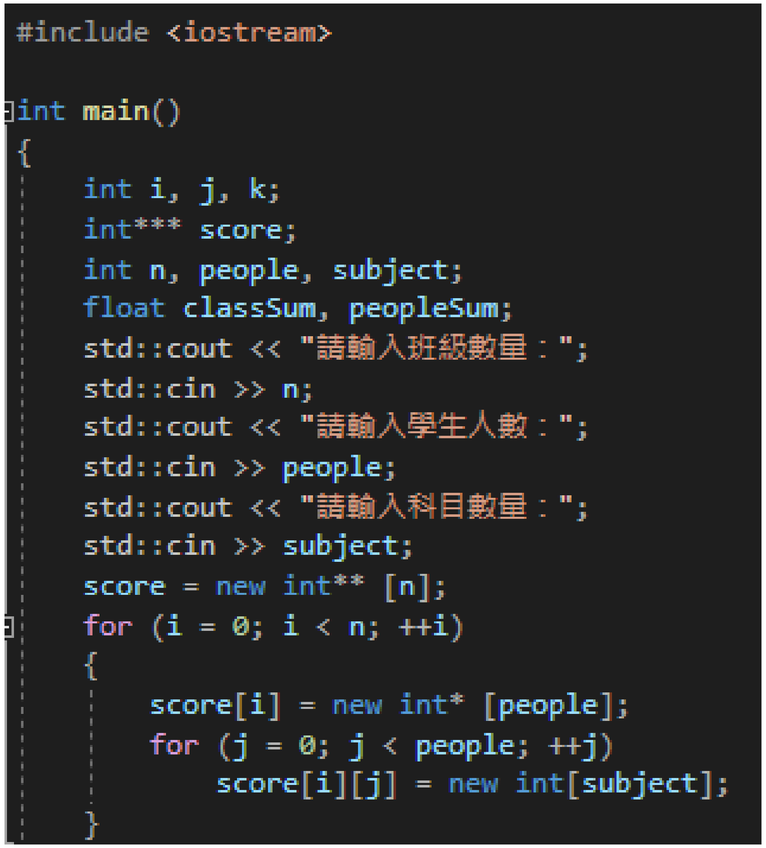
三維動態記憶體配置

```
int*** pppa = new int**[z];
for(j=0; j<z; ++j) {
    pppa[j] = new int*[y];
    for(i=0; i<y; ++i)
        pppa[j][i] = new int[x];
}
```

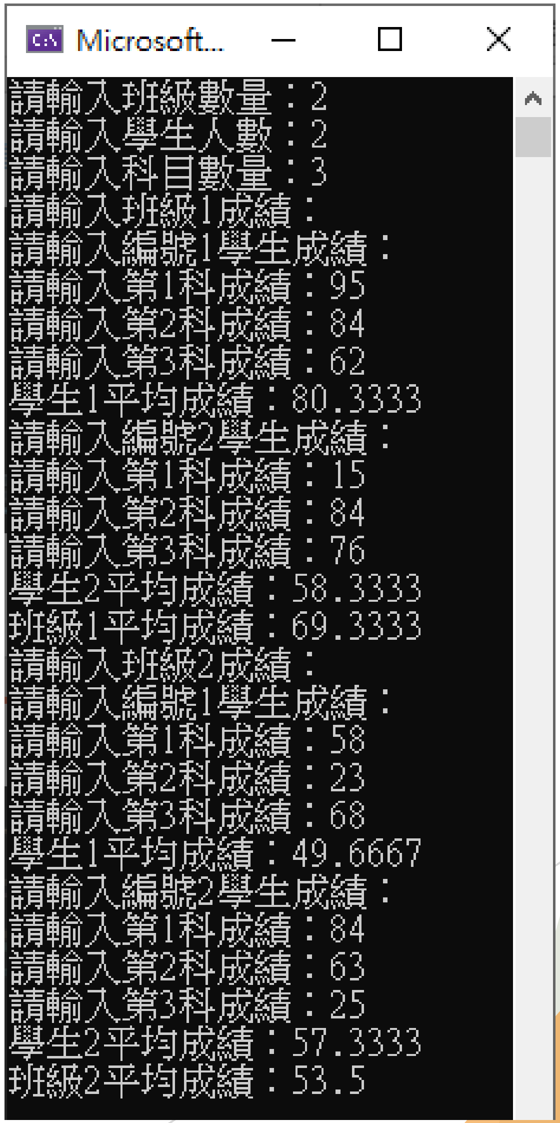
三維動態記憶體釋放

```
for(j=0; j<z; ++j) {
    for(i=0; i<y; ++i)
        delete[] pppa[j][i];
    delete[] pppa[j];
}
delete[] pppa;
```

05- 計算班、人平均

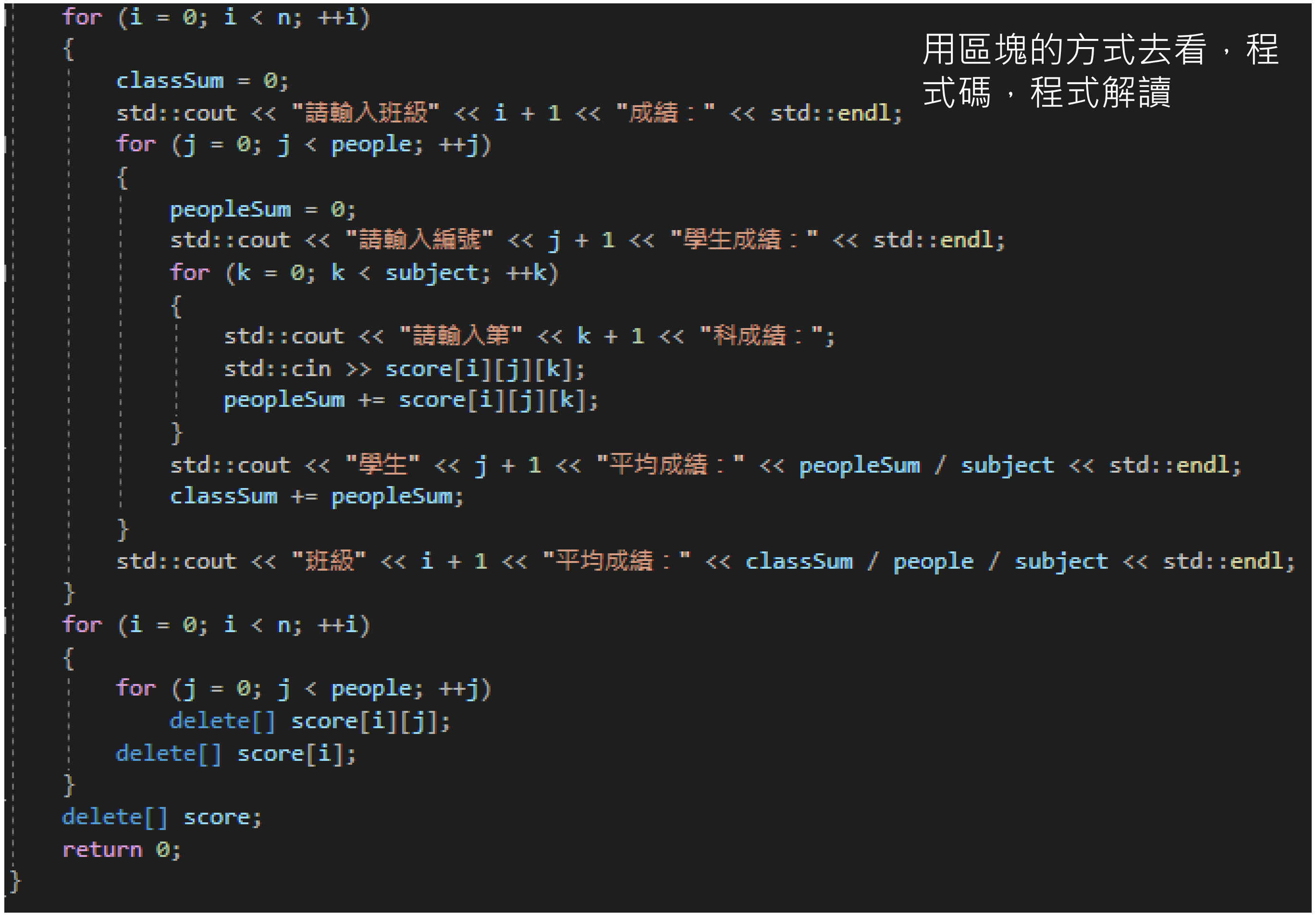


```cpp
#include <iostream>

int main()
{
    int i, j, k;
    int*** score;
    int n, people, subject;
    float classSum, peopleSum;
    std::cout << "請輸入班級數量：";
    std::cin >> n;
    std::cout << "請輸入學生人數：";
    std::cin >> people;
    std::cout << "請輸入科目數量：";
    std::cin >> subject;
    score = new int** [n];
    for (i = 0; i < n; ++i)
    {
        score[i] = new int* [people];
        for (j = 0; j < people; ++j)
            score[i][j] = new int[subject];
    }
```



Microsoft...

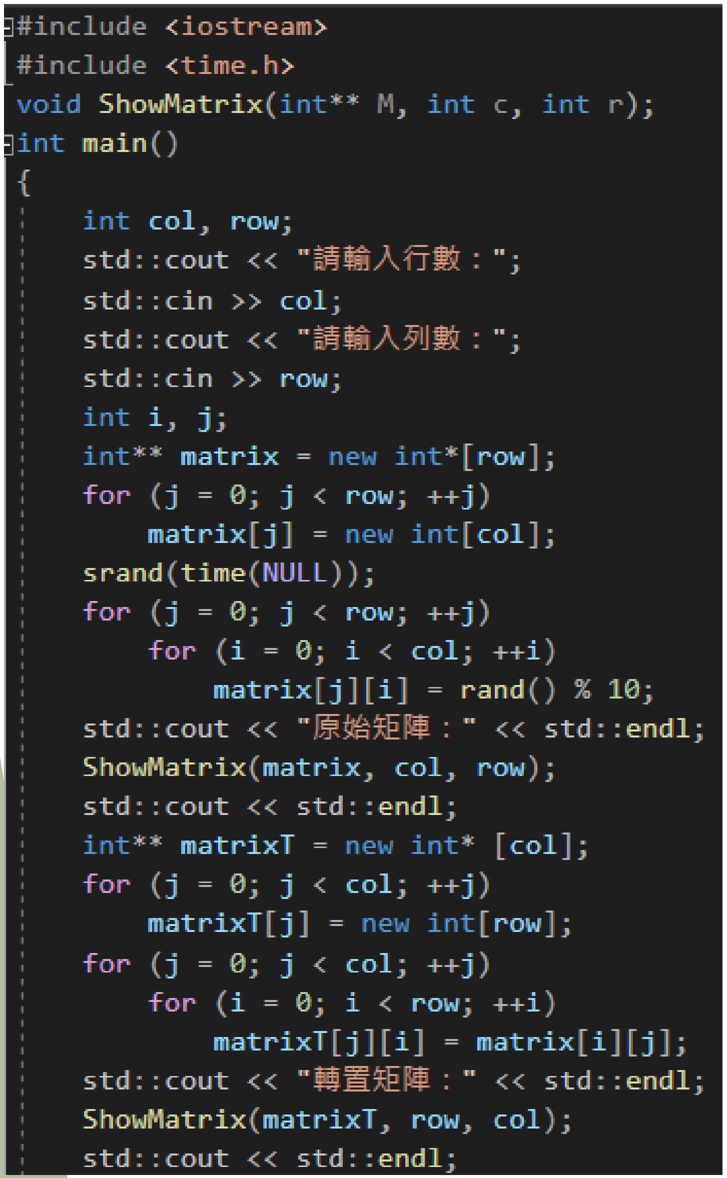
```
請輸入班級數量：2
請輸入學生人數：2
請輸入科目數量：3
請輸入班級1成績：
請輸入編號1學生成績：
請輸入第1科成績：95
請輸入第2科成績：84
請輸入第3科成績：62
學生1平均成績：80.3333
請輸入編號2學生成績：
請輸入第1科成績：15
請輸入第2科成績：84
請輸入第3科成績：76
學生2平均成績：58.3333
班級1平均成績：69.3333
請輸入班級2成績：
請輸入編號1學生成績：
請輸入第1科成績：58
請輸入第2科成績：23
請輸入第3科成績：68
學生1平均成績：49.6667
請輸入編號2學生成績：
請輸入第1科成績：84
請輸入第2科成績：63
請輸入第3科成績：25
學生2平均成績：57.3333
班級2平均成績：53.5
```



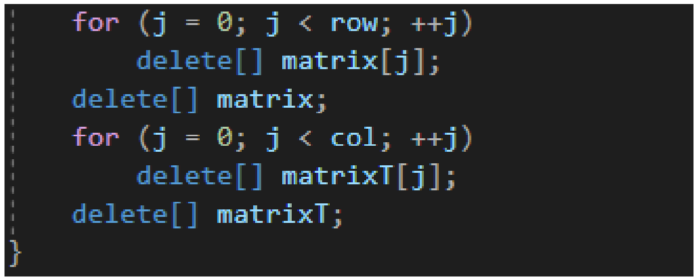
```
    for (i = 0; i < n; ++i)
    {
        classSum = 0;
        std::cout << "請輸入班級" << i + 1 << "成績：" << std::endl;
        for (j = 0; j < people; ++j)
        {
            peopleSum = 0;
            std::cout << "請輸入編號" << j + 1 << "學生成績：" << std::endl;
            for (k = 0; k < subject; ++k)
            {
                std::cout << "請輸入第" << k + 1 << "科成績：";
                std::cin >> score[i][j][k];
                peopleSum += score[i][j][k];
            }
            std::cout << "學生" << j + 1 << "平均成績：" << peopleSum / subject << std::endl;
            classSum += peopleSum;
        }
        std::cout << "班級" << i + 1 << "平均成績：" << classSum / people / subject << std::endl;
    }
    for (i = 0; i < n; ++i)
    {
        for (j = 0; j < people; ++j)
            delete[] score[i][j];
        delete[] score[i];
    }
    delete[] score;
    return 0;
}
```

用區塊的方式去看，程式碼，程式解讀

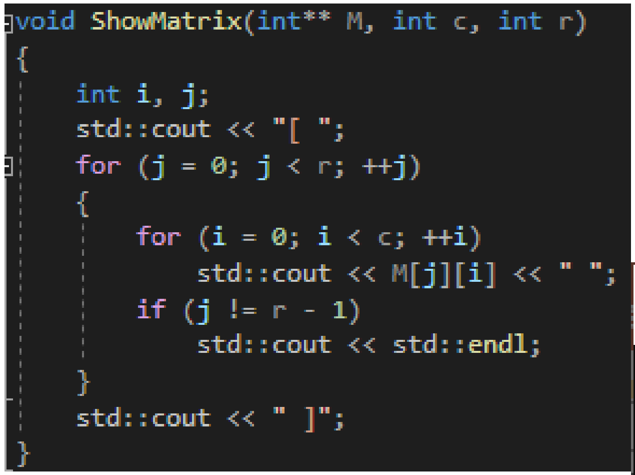
06- 轉置矩陣



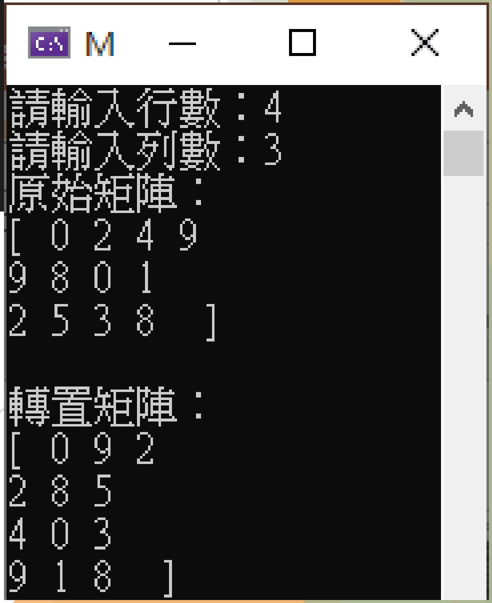
```cpp
#include <iostream>
#include <time.h>
void ShowMatrix(int** M, int c, int r);
int main()
{
    int col, row;
    std::cout << "請輸入行數：";
    std::cin >> col;
    std::cout << "請輸入列數：";
    std::cin >> row;
    int i, j;
    int** matrix = new int*[row];
    for (j = 0; j < row; ++j)
        matrix[j] = new int[col];
    srand(time(NULL));
    for (j = 0; j < row; ++j)
        for (i = 0; i < col; ++i)
            matrix[j][i] = rand() % 10;
    std::cout << "原始矩陣：" << std::endl;
    ShowMatrix(matrix, col, row);
    std::cout << std::endl;
    int** matrixT = new int* [col];
    for (j = 0; j < col; ++j)
        matrixT[j] = new int[row];
    for (j = 0; j < col; ++j)
        for (i = 0; i < row; ++i)
            matrixT[j][i] = matrix[i][j];
    std::cout << "轉置矩陣：" << std::endl;
    ShowMatrix(matrixT, row, col);
    std::cout << std::endl;
```



```cpp
    for (j = 0; j < row; ++j)
        delete[] matrix[j];
    delete[] matrix;
    for (j = 0; j < col; ++j)
        delete[] matrixT[j];
    delete[] matrixT;
}
```



```cpp
void ShowMatrix(int** M, int c, int r)
{
    int i, j;
    std::cout << "[ ";
    for (j = 0; j < r; ++j)
    {
        for (i = 0; i < c; ++i)
            std::cout << M[j][i] << " ";
        if (j != r - 1)
            std::cout << std::endl;
    }
    std::cout << " ]";
}
```



```
請輸入行數：4
請輸入列數：3
原始矩陣：
[ 0 2 4 9
9 8 0 1
2 5 3 8  ]

轉置矩陣：
[ 0 9 2
2 8 5
4 0 3
9 1 8  ]
```

想想看

- 陣列是最基本的資料結構，它的特性是什麼？它的特性可以帶來甚麼好處？
- 特性:
  - 1.隨機存取，雖機指定一個地方，再單位時間內存取這個空間，或使用這個空間，比較省時間 < -- > 循序存取
  - 2.為何可以隨機存取: 因為它是空間是連續的，所以才可以，數學換算即可，多項式算術可以再單位時間內完成