練習：印出九九乘法表

題號：206

http://2015.sprout.csie.org/oj/pro/206/

提示：改一下我們剛剛玩耍的Code

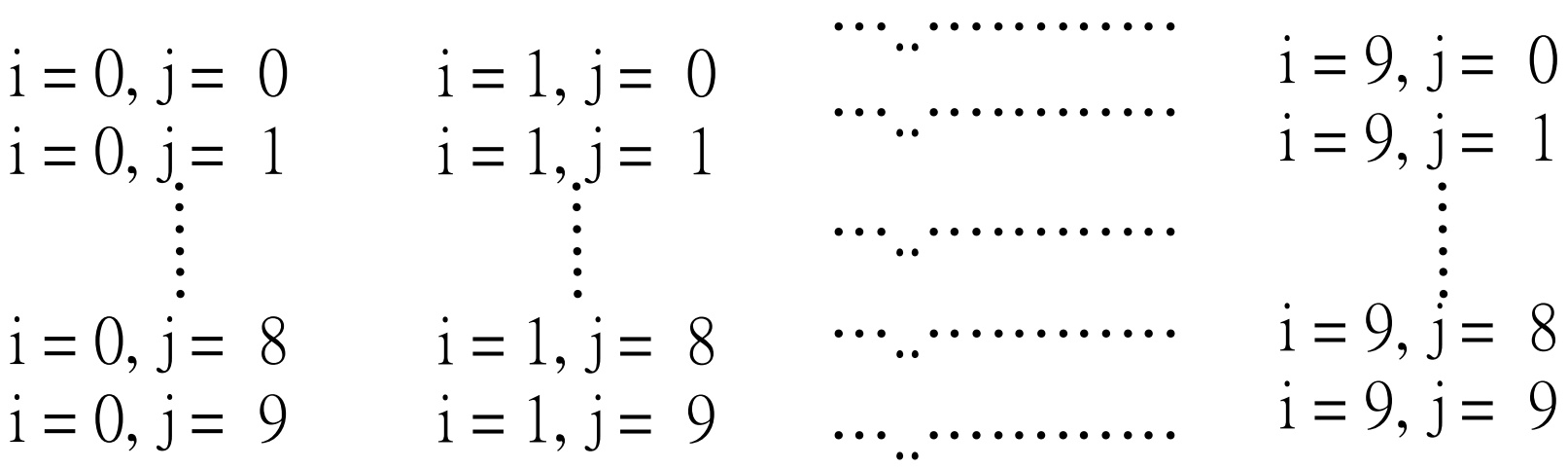
還記得上禮拜...

```
 1 #include <iostream>
 2
 3 int main(){
 4
 5     for(int i = 0; i < 10; i++){
 6         for(int j = 0; j < 10; j++){
 7             std::cout << i << " " << j << std::endl;
 8         }
 9     }
10     return 0;
11 }
```

```cpp
#include <iostream>

int main(){

    for(int i = 0; i < 10; i++){
        for(int j = 0; j < 10; j++){
            std::cout << i << " " << j << std::endl;
        }
    }
    return 0;
}
```



就稍微改一下.....

一下就好...

想想九九乘法表的起點...

想想九九乘法表的起點...

起點：1 x 1

想想九九乘法表的起點...

起點：1 x 1, 1 x 2, ……..

終點：....9 x 8, 9 x 9

嘿嘿～～～

B O N U S

樹大招風改一(210)

- 輸入說明
  - 輸入第一行是正整數 n (1≤n≤500)，接著輸入 n 個正整數 a0,…,an−1
- 輸出說明
  - 請輸出它們的最大值 (記得換行)

樹大招風改一(210)

此題採用的祕技：錯了再說

樹大招風改一(210)

- 前置動作：先把所有的數字儲存到陣列

樹大招風改一(210)

- 思路：

1. 先把第一個數字當作最大值，也就是當作答案

樹大招風改一(210)

- 思路：
  1. 先把第一個數字當作最大值，也就是當作答案
  2. 從第二個數字開始掃描，如果發現有數字比當前的答案還大，就去更新答案的值。
     （簡單來說就是發現答案錯了之後偷改答案....）

樹大招風改一(210)

- 思路：
  1. 先把第一個數字當作最大值，也就是當作答案
  2. 從第二個數字開始掃描，如果發現有數字比第二個數字還大，就去更新答案的值
  3. 掃描完全部的數字後，就會發現答案就是最大值了

     （因為答案已經沒辦法再大了....）

樹大招風改二(211)

- 輸入說明
  - 輸入第一行是正整數 n (1≤n≤500)，接著
  - 輸入 n 個正整數 a0,…,an−1
- 輸出說明
  - 請把所有等於最大值的項目都改成 −1。例如底下的最大值是 5，就把所有出現的 5 都改成 −1

樹大招風改二(211)

祕技：運用樹大招風改一

畢竟是改二

樹大招風改二(211)

- 思路：
  1. 先把所有的數字存在陣列中
  2. 找出陣列中的最大值
  3. 掃描儲存在陣列中的值

     如果陣列中的值等於最大值，就把這個值改成-1

樹大招風改三(212)

- 題目說明
  - 輸入第一行是正整數 n (1≤n≤500)，接著輸入 n 個正整數 a0,…,an−1，重複以下的動作 n 次，每次時把當前的最大值都設成 −1
  - 印出一行當前的 a0,…,an−1，數字間用一個空白隔開
- 輸出詳細說明
  - 對於每一次迴圈，都會輸出一行共 n 個數字，總共有 n 次迴圈。每次迴圈時，**輸出的 n 個數字都要用一個空白隔開，但行末沒有空白**。

樹大招風改三(212)

偷吃步：樹大招風改二

樹大招風改三(212)

偷吃步：樹大招風改二

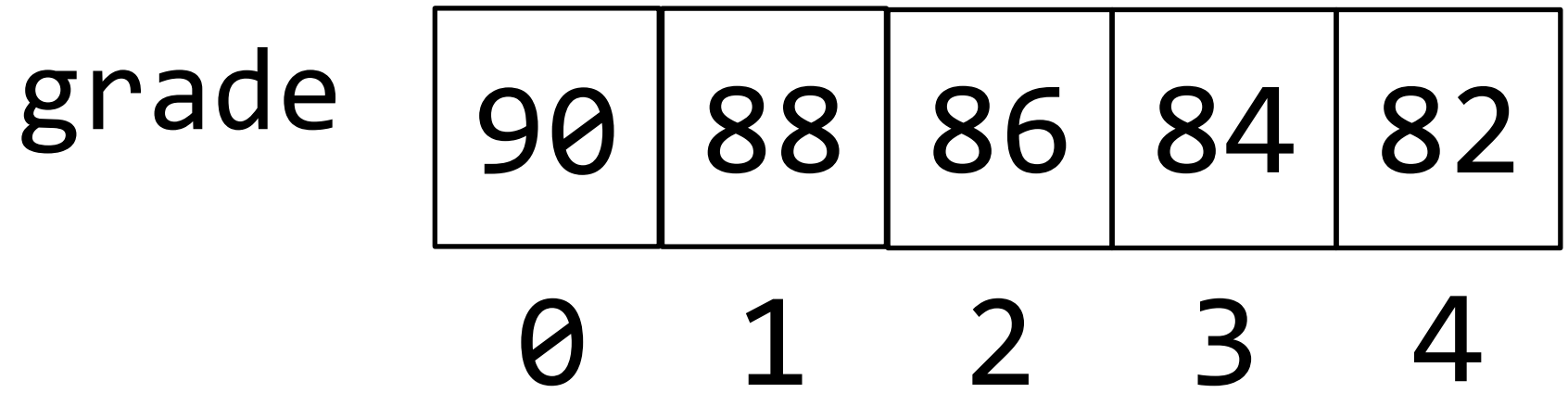
每改一次就輸出一次！

複習

```
int grade[5]={90,88,86,84,82};
```

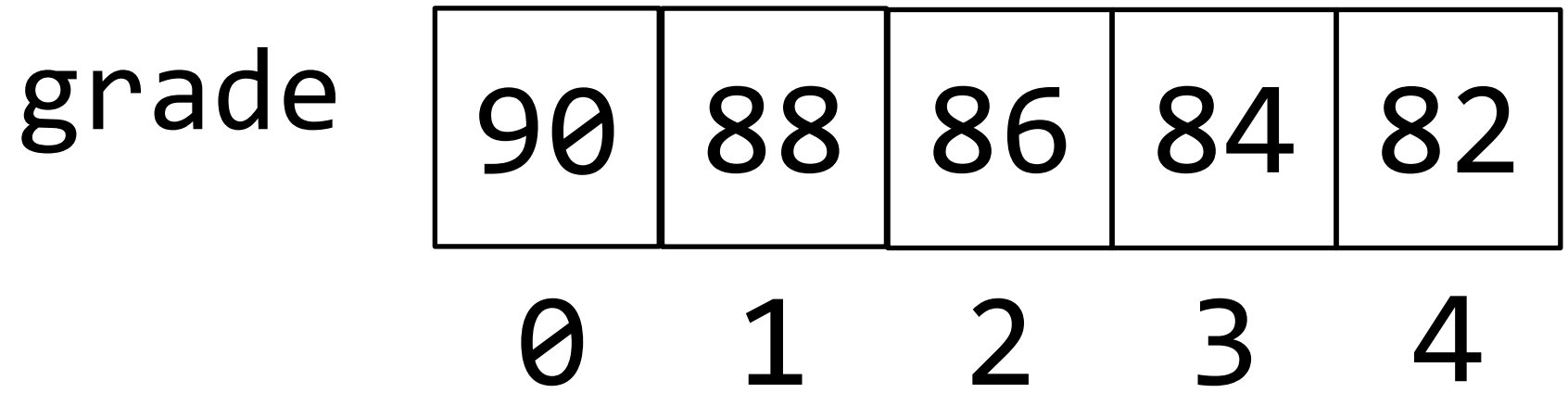
複習

```
int grade[5]={90,88,86,84,82};
```



複習

```
int grade[5]={90,88,86,84,82};
```




```
grade[0]=90; grade[3]=84;
grade[1]=88; grade[4]=82;
grade[2]=86;
```

複習(cont)

```
for(int i=0; i<5; i++){
    std::cout<<grade[i]<<" ";
}
```

複習(cont)

初始化陣列

```
int array[5] = {0};
```

陣列的宣告

```
int array[] = {2,4,6,8,10};
int array[5] = {2,4,6,8,10};
```

一維陣列

```
int array[5] = {2,4,6};
```

2	4	6	0	0

二維陣列

```
int grade[3][5]={0};
```

有3列,每一列有5格(行),存的是整數

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

二維陣列

```
int grade[3][5] =
                {{98,97,96,95,94},
                 {93,92,91,90,89},
                {88,87,86,85,84}};
```

98	97	96	95	94
93	92	91	90	89
88	87	86	85	84

二維陣列

```
int grade[3][5] =
{98,97,96,95,94,93,92,91,90,89,
88,87,86,85,84};
```

98	97	96	95	94
93	92	91	90	89
88	87	86	85	84

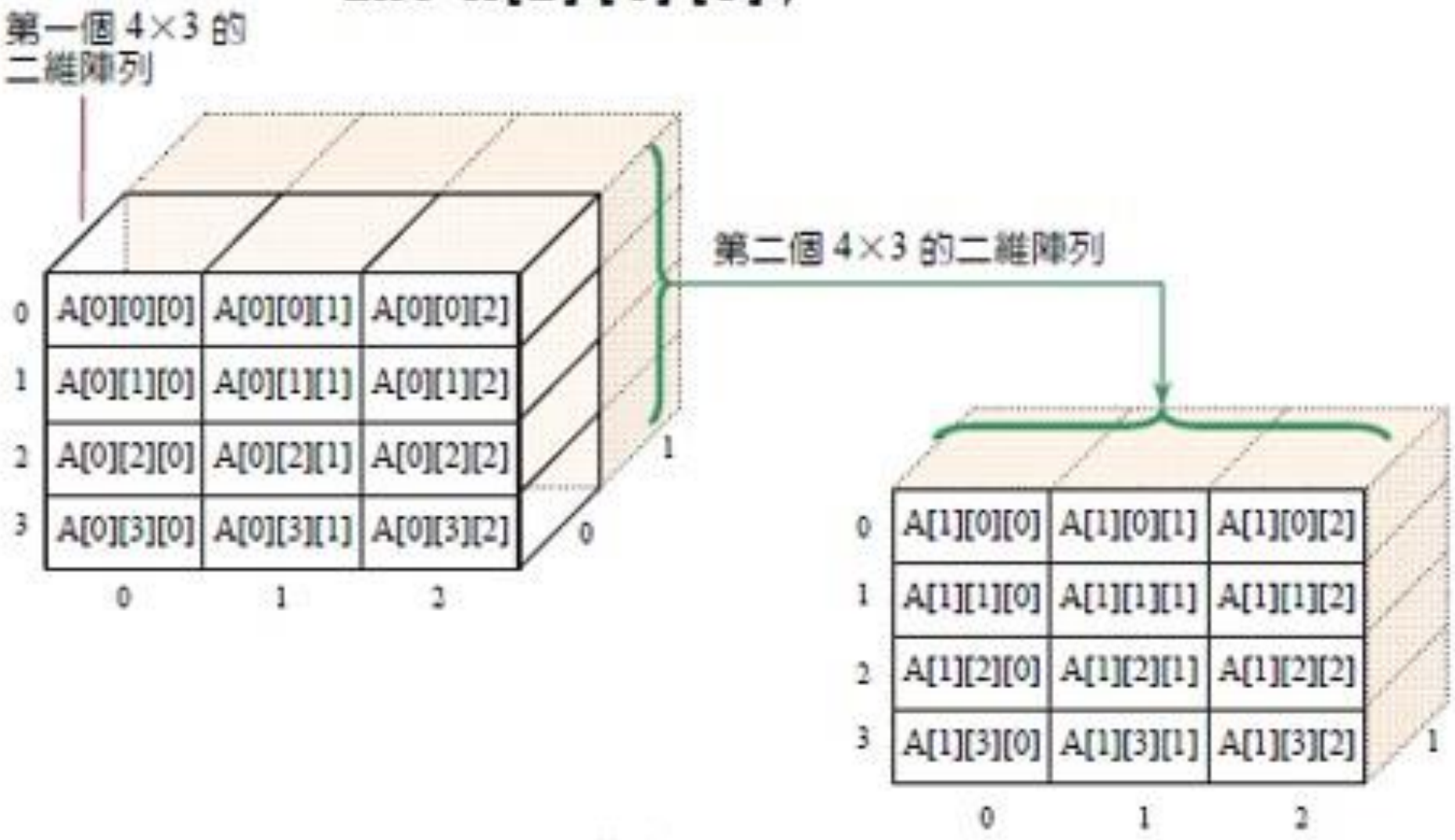
二維陣列

```
int grade[3][5] =
{98,97,96,95,94,93,92,91,90,89,
88,87,86};
```

98	97	96	95	94
93	92	91	90	89
88	87	86	0	0

三維陣列

```
int A[2][4][3];
```



Example

Input

第1行有兩個正整數m,n(1≤m,*n*≤100)，代表矩陣的大小(m*n)；第2行到第m+1行，每行會有n個整數，代表矩陣裡面的元素。

Output

此矩陣

Sample Input

2 3

3 5 6

1 3 5

Sample Output

3 5 6

1 3 5

Example(cont)

```
#include <iostream>
int main(){
	int m,n;
	int in[100][100]={0};
	std::cin>>m>>n;
	for (int i=0; i<m; i++){		//讀取input
		for (int j=0; j<n; j++){
			std::cin>>in[i][j];
		}
	}
	for (int i=0; i<m; i++){		//輸出
		for (int j=0;j<n;j++){
			std::cout<<in[i][j]<<" ";
		}
		std::cout<<std::endl;
	}
	return 0;
}
```