



# C/C++基礎程式設計

陣列

講師：張傑帆  
CSIE, NTU

你必須要有一個自己想解決的問題，一個自己想糾正的錯誤。

*You've got to have a problem that you want to solve; a wrong that you want to right. -Steve Jobs*

# 課程大綱

- 一維陣列
- 二維陣列
- 其他多維陣列



# 一維陣列的宣告

- 用途：在記憶體中找出一塊連續的空間來存放多個相同資料型態的內容

- 一維陣列的宣告

- 語法：資料型態 陣列名稱[長度];

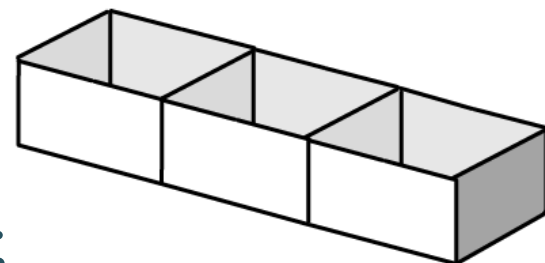
- 例如：int a[N];

- (N為一個整數常數 不為變數)

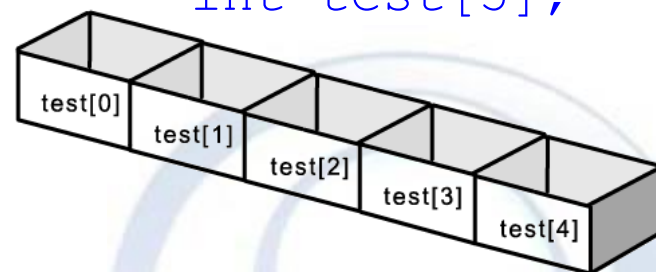
位置: a



N個欄位



int test[5];



# 一維陣列的使用

- 在使用宣告好的陣列時，要使用一個陣列中的元素可以表示成：**陣列名稱[索引]**。

- 「**陣列名稱**」則是用來表示一塊位置緊密相鄰的記憶體空間的**起始位址**。

- 「**索引**」的功能用來表示該陣列元素是在記憶體空間的**第幾號位置**

- **範例：**

- `int a[3];` // 宣告陣列a有3個整數
- `a[0]=1;` // 把a[0]設定為1
- `a[1]=2;` // 把a[1]設定為2
- `a[2]=3;` // 把a[2]設定為3
- **a[3]是不存在的。**

`int test[5];` ← 宣告陣列

`test[0] = 80;`

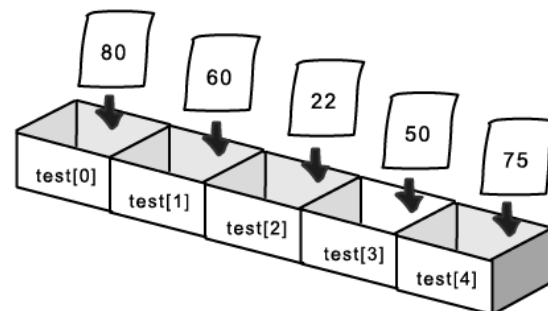
`test[1] = 60;`

`test[2] = 22;`

`test[3] = 50;`

`test[4] = 75;`

逐一將值指定  
給陣列元素



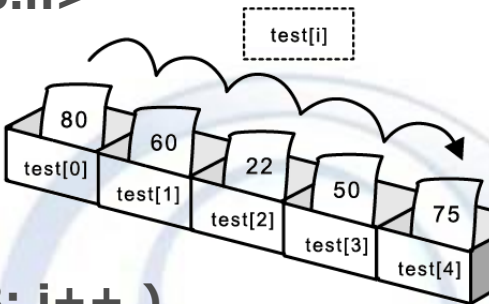
# 一維陣列的使用

- 範例：輸入全班3位同學的成績並輸出
  - 比較下列兩個版本程式
  - 思考：如果班上有50位同學的話呢？

```
#include <stdio.h>
int main()
{
    int score1, score2, score3;
    scanf("%d",&score1);
    printf("I get %d\n",score1);
    scanf("%d",&score2);
    printf("I get %d\n",score2);
    scanf("%d",&score3);
    printf("I get %d\n",score3);
    return 0;
}
```

```
#include <stdio.h>
int main()
{
    int i;
    int score[3];

    for ( i=0; i<3; i++ )
    {
        scanf("%d", &score[i]);
        printf("I get %d\n", score[i]);
    }
    return 0;
}
```



# 一維陣列的使用

- 使用陣列常搭配使用 **#define** 用來定義一個常數，方便做程式修改
- 範例：輸入全班50位同學的成績並輸出

```
#include <stdio.h>
#define STUDENT 50

int main()
{
    int i;
    int score[STUDENT];

    for ( i=0; i < STUDENT; i++ ) {
        scanf("%d", &score[i]);
        printf("I get %d\n",score[i]);
    }
    return 0;
}
```

# 一維陣列的初始化

- 宣告陣列並給初始內容的語法：
  - 資料型態 陣列名稱[長度]={內容0,內容1,內容2,...,內容N-1};
  - 資料型態 陣列名稱[]={內容0,內容1,內容2,...,內容N-1};
  - 資料型態 陣列名稱[長度]={0}; //所有資料設為0

```
int main()
{
    int a[5]={1,2,3,4,5};
    // 設定 a[0]=1, a[1]=2, a[2]=3, a[3]=4, a[4]=5

    int b[]={1,2,3};
    // 若size沒指定, 因為給了3個數字, 在此會自動設定為b[3];

    int c[5]={0};
    // 設定 c[0]=0, c[1]=0, c[2]=0, c[3]=0, c[4]=0

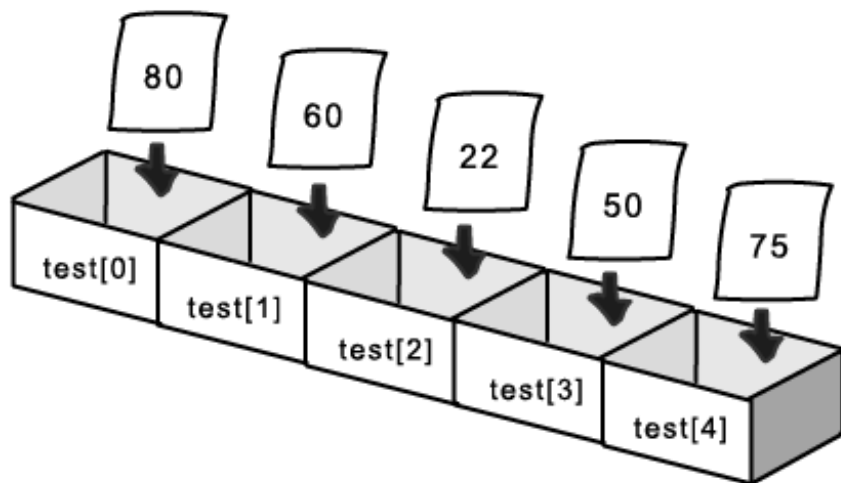
    return 0;
}
```



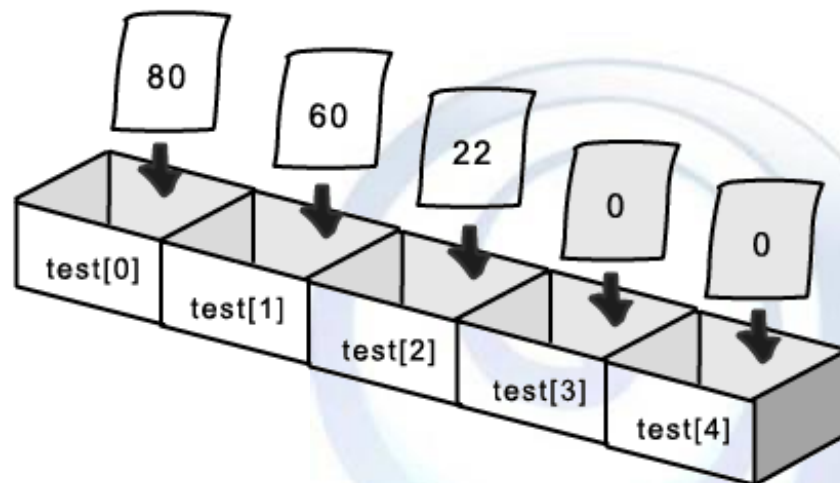
## 陣列的初始化

- 當初始化子不足時，多的元素的值會自動儲存成0。

```
int test[5] = {80, 60, 22, 50, 75};
```



```
int test[5] = {80, 60, 22};
```





# 陣列的複製

- 因為宣告陣列後如果沒有初始化，之後只能夠一個一個索引值做設定，如果想要複製整個陣列可以用memcpy函式做資料複製

**#include<string.h>**

- memcpy(目標陣列, 來源陣列, sizeof(型態)\*個數);

```
int main()
{
    int a[5] = {1,2,3,4,5};
    int b[5];

    memcpy(b, a, sizeof(int)*5);

    return 0;
}
```

# 一維陣列的應用

- 範例:輸入班上3位同學分數, 求出它們的平均值.

```
#include <stdio.h>
#define STUDENT 3
int main()
{
    int i;
    double sum=0;
    double aver;
    int score[STUDENT];

    // 分別讀入3個數值
    for ( i=0; i < STUDENT; i++ ) {
        scanf("%d", &score[i]);
    }
    // 計算總和
    for ( i=0; i < STUDENT; i++ ) {
        sum+=score[i];
    }
    // 求平均值
    aver=sum/STUDENT;
    printf("average=%lf\n",aver);
    return 0;
}
```

## 練習

- 延續上頁範例，輸入3位同學成績，印出平均分數後，印出不及格的同學之號碼與分數

- 輸入 `40 60 50`

- 輸出 

```
avg = 50.00
fail:
1: 40
3: 50
```

## 練習：找出數列中最大值

- 延續上頁範例，輸入3位同學成績，印出平均分數與不及格的同學之號碼與分數後，印出其中最高分同學之分數與座號
- 輸入 `40 60 50`

- 輸出

```
avg = 50.00
fail:
1: 40
3: 50
highest:
2: 60
```

# 陣列中資料的排序

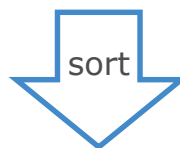
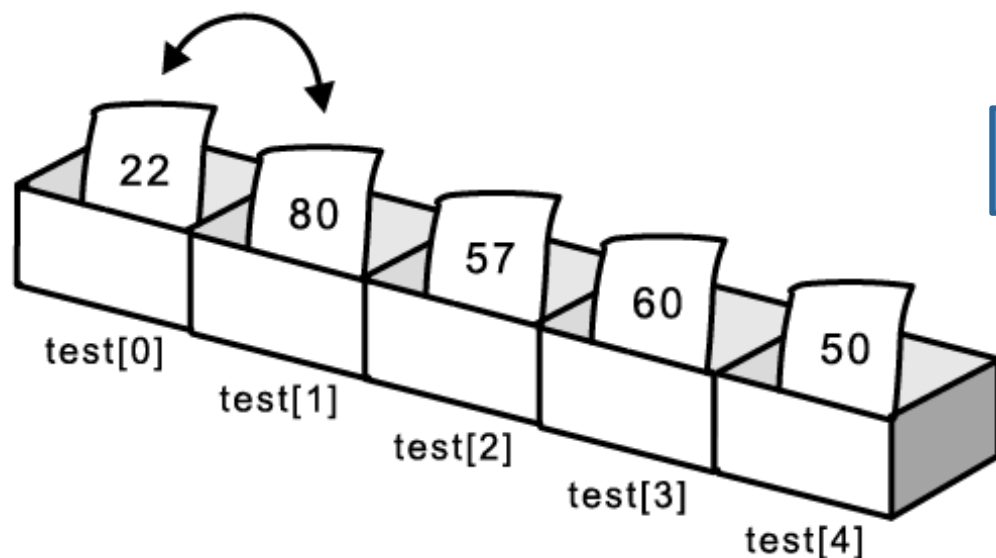
▶ 輸入五個整數, 印出從小到大的結果

▶ 例如：

■ 輸入：34 12 5 66 1

■ 輸出：1 5 12 34 66

(做法如下頁所示)



# 陣列中資料的排序

- 使用迴圈將最大值換到陣列最後

兩數交換範例: 將a,b兩變數內容互換

```
int a=10, b=5,  
temp;
```

```
temp = a;  
a = b;  
b = temp;
```

- 將上面的功能重覆做4次即可

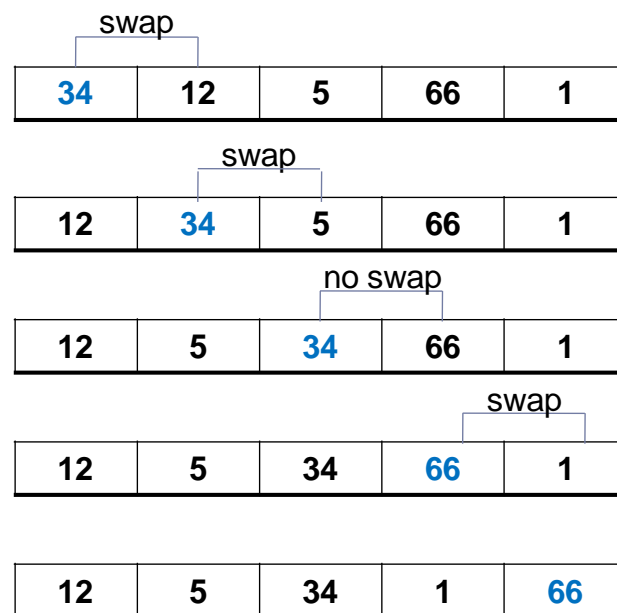
34	12	5	66	1
----	----	---	----	---

12	5	34	1	66
----	---	----	---	----

5	12	1	34	66
---	----	---	----	----

5	1	12	34	66
---	---	----	----	----

1	5	12	34	66
---	---	----	----	----



# 陣列中資料的排序

- 範例：

```
#include <stdio.h>

int main()
{
    int data[5] = {34,12,5,66,1}; // 欲排序的資料
    int temp;
    int i, j;

    for(j=5; j>1; j--) { // 需要找四次最大值
        for(i=0; i<j-1; i++) { // 每次要比對j-1次
            if(data[i] > data[i+1]) { // 若比後面的大就交換
                temp = data[i];
                data[i] = data[i+1];
                data[i+1] = temp;
            }
        }
    }
    return 0;
}
```



## 動動腦

- 如何改變排序的順序，由原本的小到大變成大到小？



# 課程大綱

- 一維陣列
- 二維陣列
- 其他多維陣列



# 二維陣列的使用

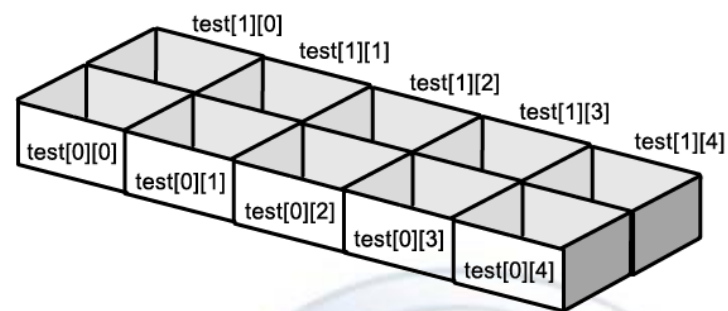
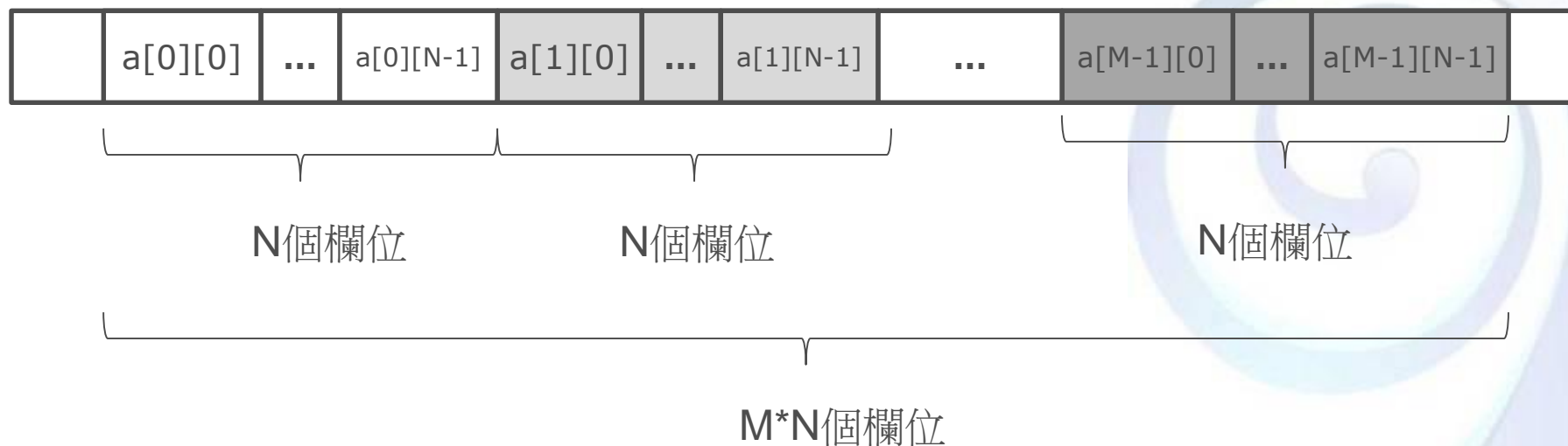
- 二維陣列

- 資料型態 陣列名稱[長度1][長度2];

- Example: `int a[M][N];`**

- (N,M為整數常數, 不為變數)

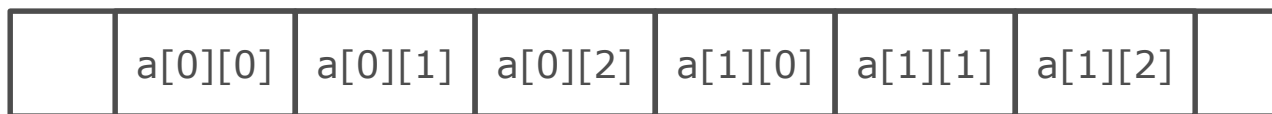
位置: a



## 二維陣列的使用

- 利用連續的記憶體空間，製作行列的效果!
- **EX:** `int a[2][3];`

位置: a



	第0行	第1行	第2行
第0列	a[0][0]	a[0][1]	a[0][2]
第1列	a[1][0]	a[1][1]	a[1][2]

# 二維陣列

- 設定二維陣列初值的方法為

```
int main()
{
    int a[2][3]={ {1, 2, 3}, {4, 5, 6} };
    int b[2][2]={ 1,2,3,4 };
    int c[2][2]={0}; //所有資料設為0
    return 0;
}
```

- 陣列宣告後，要設定陣列時，就必須一個一個欄位做設定

```
int main()
{
    int a[2][2];
    a[0][0] = 1;
    a[0][1] = 2;
    a[1][0] = 3;
    a[1][1] = 4;
    return 0;
}
```

## 二維陣列與巢狀迴圈

- 使用 巢狀迴圈 輸出二維陣列內容:

```
#include <stdio.h>
int main()
{
    int a[3][3]={ {1, 2, 3}, {4, 5, 6}, {7, 8, 9} };
    int i,j;
    for ( j=0; j<3; j++ ) {
        for ( i=0; i<3; i++ ) {
            printf("%d ", a[j][i]);
        }
        printf("\n");
    }
    return 0;
}
```

## 二維陣列與巢狀迴圈

- 範例:輸入兩班上3位同學分數,然後再次輸出成績.

```
#include <stdio.h>
#define CLASS 2
#define STUDENT 3

int main()
{
    int i, j;
    int score[CLASS][STUDENT];
    // 分別讀入兩班各個數值
    for ( j=0; j < CLASS; j++ ) {
        printf("班級%d:\n", j+1);
        for ( i=0; i < STUDENT; i++ ) {
            printf(" 學生%d: ", i+1);
            scanf("%d", &score[j][i]);
        }
    }
    // 分別輸出兩班各個數值
    for ( j=0; j < CLASS; j++ ) {
        printf("班級%d:\n", j+1);
        for ( i=0; i < STUDENT; i++ )
        {
            printf(" 學生%d: %d分\n", i+1, score[j][i]);
        }
    }
    return 0;
}
```

```
40 50 60
70 80 90
班級 1:
學生 1: 40 分
學生 2: 50 分
學生 3: 60 分
班級 2:
學生 1: 70 分
學生 2: 80 分
學生 3: 90 分
```



## 二維陣列與巢狀迴圈

- 範例:輸入兩班上3位同學分數，求出它們的平均值

```
#include <stdio.h>
#define CLASS 2
#define STUDENT 3
int main()
{
    int i, j;
    double sum=0;
    double aver;
    int score[CLASS][STUDENT];
    // 分別讀入兩班各3個數值
    for ( j=0; j < CLASS; j++ ) {
        printf("班級%d:\n", j+1);
        for ( i=0; i < STUDENT; i++ ) {
            printf("學生%d: ", i+1);
            scanf("%d", &score[j][i]);
        }
    }
    // 計算總和
    for ( j=0; j < CLASS; j++ )
        for ( i=0; i < STUDENT; i++ )
            sum+=score[j][i];

    // 求平均值
    aver=sum/(STUDENT*CLASS);
    printf("average=%lf\n",aver);
    return 0;
}
```

## 練習

- 修改上頁範例，輸入兩班3位同學分數後，印出各班總分與平均

- 輸入

```
70 80 90
50 60 70
```

- 輸出

```
class 1
1: 70
2: 80
3: 90
sum: 240
avg: 80.00
class 2
1: 50
2: 60
3: 70
sum: 180
avg: 60.00
total: 420, avg: 70.00
```

# 課程大綱

- 一維陣列
- 二維陣列
- 其他多維陣列



## 其他多維陣列的使用

- 三維陣列

- 資料型態 陣列名稱[長度1][長度2][長度3];

- 四維陣列

- 資料型態 陣列名稱[長度1][長度2][長度3][長度4];

- ...



## 其他多維陣列的應用

- 例如：儲存4班同學，每班3人，2科考試成績
- 宣告要用的資料
  - `int score[4][3][2];`
- 第一班,第二號同學,第一科100分
  - `score[0][1][0] = 100;`
- 例如：5間學校，4班同學，每班3人，2科考試成績
- 宣告要用的資料
  - `int score[5][4][3][2];`

# 其他多維陣列的應用

- 範例:輸出2班2位同學2科考試成績

```
#include <stdio.h>
#define CLASS 2
#define STUDENT 2
#define SUBJECT 2
int main()
{
    int i, j, k;
    int score[CLASS][STUDENT][SUBJECT] = {20,30,40,50,60,70,80,90};

    // 分別輸出兩班各個數值
    for ( j=0; j < CLASS; j++ ) {
        printf("班級%d:\n", j+1);
        for ( i=0; i < STUDENT; i++ ) {
            printf("學生%d:\n", i+1);
            for ( k=0; k < SUBJECT; k++ ) {
                printf("第%d科: %d分\n", k+1, score[j][i][k]);
            }
        }
        printf("\n");
    }
    return 0;
}
```

## 回家作業

- 假定某班有5位學生，每位學生各修3門科目，請利用二維陣列的方式儲存學生的各科成績，並將每位學生的各科成績、總分及平均列印出來，並找出班上最平均高分的學生。

座號	科目A	科目B	科目C
1	76	73	85
2	88	84	76
3	92	82	92
4	82	91	85
5	72	74	73

```
student 1
1: 76
2: 73
3: 85
sum: 234
avg: 78.00
student 2
1: 88
2: 84
3: 76
sum: 248
avg: 82.67
....
student 5
1: 72
2: 74
3: 73
sum: 219
avg: 73.00
total: 1225, avg: 81.67
highest avg: student 3: 88.67
```