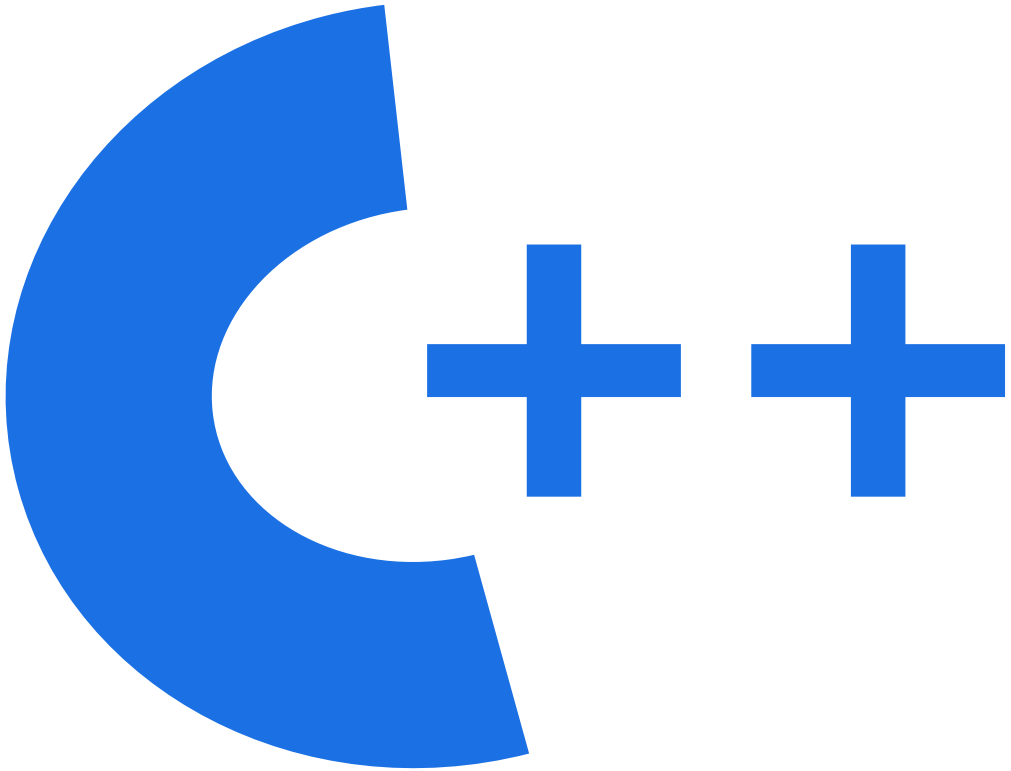
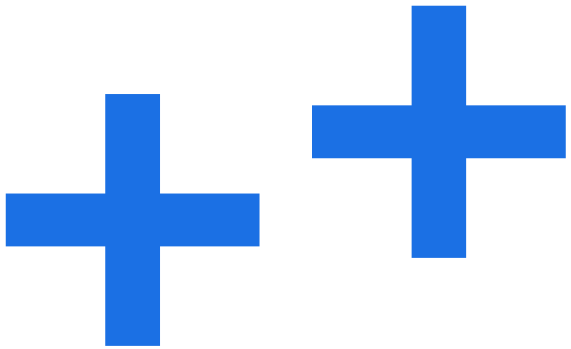


陣列-
一維陣列

為什麼要使用陣列



當資料量變大時

- 前面幾章的學習，我們已經可以處理許多複雜的問題。
- 只是當資料量變多時，依靠前面的知識是不夠的。
- 即使簡單的問題，也可能需要比較不一樣的方法。

輸入5個整數，
並且印出5個整數。

輸入50個整數，
並且印出50個整數。

定義50個變數是不能解決問題的

```
#include <iostream>
using namespace std;
int main(){
int
a1,a2,a3,a4,a5,a6,a7,a8,a9,a10,a11,a12,a13,a14,a15,a16,a17,a18,a19,a20,a21
,a22,a23,a24,a25,a26,a27,a28,a29,a30,a31,a32,a33,a34,a35,a36,a37,a38,a39,
a40,a41,a42,a43,a44,a45,a46,a47,a48,a49,a50;
return 0;
}
```

輸入50個整數，並且輸出50個整數。

- 使用迴圈，改善了輸出入的效率
- 但是，
  - 在記憶體中，並沒有紀錄每一個資料

```
#include<iostream>
int main(){
        int a[50],i;
        for(i=0;i<50;i++){
            cin>>a[i];
            cout<<a[i]<<endl;
        }
        return 0;
}
```

陣列才是輸入大量相同類型變數的正解

```
#include<iostream>
int main(){
   int a[50],i;
    for(i=0;i<50;i++){
        cin>>a[i];
        cout<<a[i]<<endl;
    }
    return 0;
}
```

- 使用陣列元素
  num[i]來代替num0, num2, num3, ...num49
- 當迴圈變量i=0時
  num[i]就是num[0]
- 當迴圈變量i=1時
  num[i]就是num[1]
- 當迴圈變量i=49時
  num[i]就是num[49]

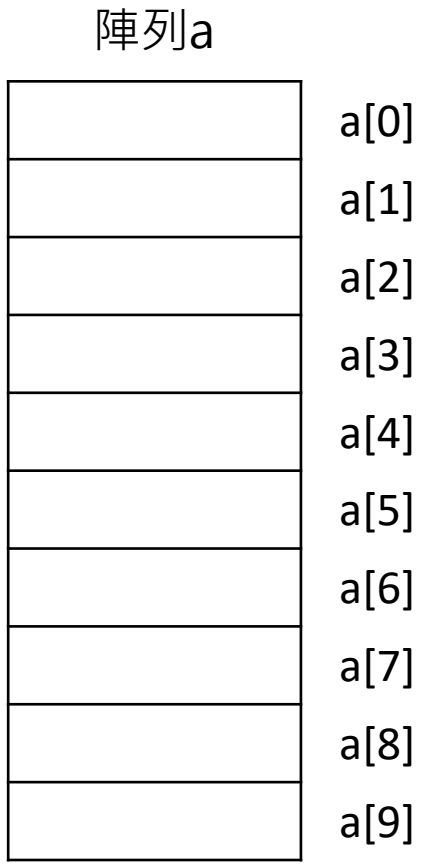
一維陣列的定義

一維陣列

- 有限個相同資料型態的元素所組成
- 這些元素儲存於連續的記憶體中
- 共用一個的陣列名稱
- 每一個元素經由索引 (index，或稱註標) 來識別
- 陣列可解釋為一組索引與資料的對映。

a[10]

- [ ]表示需要定義變數的個數
- 這裡定義10個
- 這10個變量，分別用a[0]、a[1]、a[2]、a[3]、a[4]、a[5]、a[6]、a[7]、a[8]、a[9]來表示。



一維陣列的引用

一維陣列引用的方法

- 陣列名[下標]：
  - a[5]

- 如果i與j都是整數變數，以下都是合法的：
  - a[i+j], a[i++]

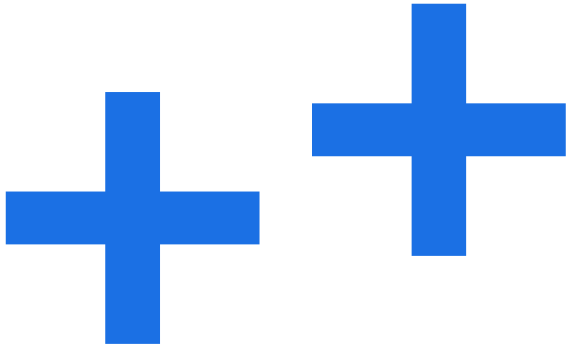
一維陣列引用說明

- 下標可以是值為整數的表達式，該表達式可以包含變數和函式呼叫。
- 引用時，下標值必須在陣列定義的下標值範圍內。
- 陣列的精妙在於下標可以是變數，通過對下標變數值的靈活控制，達到靈活處理陣列元素的目的。
- C/C++語言只能逐個引用陣列元素，而不能一次引用整個陣列。例如：int a[100], b[100]; a=b; 這樣的寫法是非法的。
- 陣列元素有點類似同類型的普通變數，對其進行賦值和運算的操作和普通變數是一樣的。例如：int c[10]=34; 實現了c[10]賦值為34

一維陣列的初始化

在初始列表中定義全部陣列的值

- **格式**：
  資料型態 陣列名[常數表達式]={值1, 值2, ...}
- **例如**：
  int a[5]={1, 2, 3, 4, 5}
- 在初始列表中可以寫出全部陣列的值，也可以寫出部分。沒寫的部分會被列為0。
  int x[10]={0, 1, 2, 3, 4 }
- 全部初始化為0
  int a[5]={}



陣列的越界

使用陣列時，要注意

- 陣列下標值為正整數。
- 在定義元素個數的下標範圍內使用，例如：

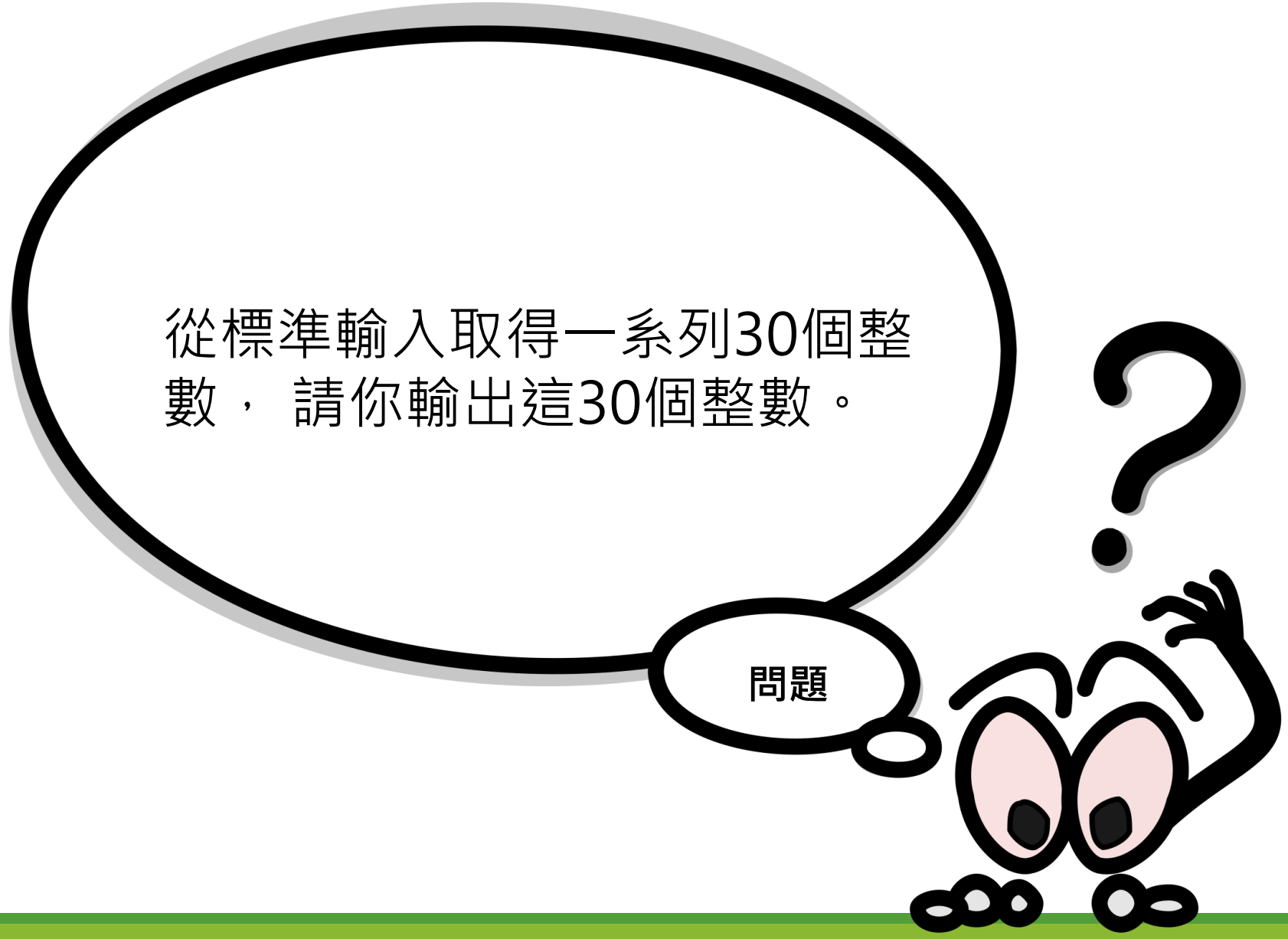
  int a[5]={1, 2, 3, 4, 5}
- 如果下標寫成負數或者大於陣列元素的個數時，成為陣列越界，程式編譯結果是不會錯的，例如：

  int a[10]; a[-3]=5; a[20]=15; a[10]=20;

陣列越界造成的後果

- 這種類型的錯誤，常是難以捕捉。
- 因為越界本身並不一定會讓程式立即出錯，可能遇到某些數據時才造成錯誤。有時由於越界，意外的改變了變數值或指令，導致在調試器裡調試的時候，程式不按照應當的次序運行的怪現象。

從標準輸入取得一系列30個整數， 請你輸出這30個整數。



```cpp
#include<iostream>
using namespace std;

int main(){
    int array[30]; /*宣告一個長度為30的陣列*/
    int i;
    for(i=0;i<30;i++){
        cin>>array[i];
    }
    cout<<"Array Value\n";
    for(i=0;i<30;i++){
        cout<<"array["<<i<<"] "<<array[i]<<endl;
    }
    return 0;
}
```

使用迴圈與陣列輸入資料

使用迴圈與陣列輸出資料