

資料結構與C++進階班 前言

講師：黃銀鵬

E-mail: yinpenghuang@gmail.com

什麼是資料結構

- ▶ 資料的容器。
- ▶ 不同的容器有不同的特性。
- ▶ 利用容器的特性簡化程式設計。
- ▶ 使用正確的容器可以大幅度提升程式執行效率。

證書發給規定

- ▶ 課程評量可分作業、出席狀況、課堂表現或課堂總測驗等，由講師依各課程類別，於開課日告知同學評量方式。
- ▶ 結業成績七十分(含)以上即可授予本校推廣教育結業證書；此證書經本校校長、電資學院院長、資訊工程學系暨研究所所長，及訓練班班主任核章後，送校核定正式關防及鋼印。
- ▶ 請注意！證書一旦開課後即無法變更；請於報名時填寫正確之中、英文姓名及出生年月日。
- ▶ 證書遺失欲重新申請證書：因原證書已開立過，故補發之證書將會在證書編號加註R字，以便與原證書作區別。中文或英文證書重新製作的工本費為各為新台幣一百元整。訓練班之證書係用以評量學員在學習時刻之程式設計能力，由於程式設計技能可能與時俱進或退步，因此超過課程結業日起兩年的證書不會再被補發。(例如：結業日是2018/02/23，申請證書補發截止日為2020/02/23。)
- ▶ 結業證書如有相關問題請諮詢聯絡電話：(02)3366-4888轉總機9

證書領取方式

- ▶ 發放證書時間：課程結束日後一個半月，將公告通過名單於本班官網，請同學自行上網查詢是否通過和證書製作進度，同學上網查詢為合格名單及個人證書編號後，即可準備領取證書。
- ▶ 證書領取期限：自課程結束日起兩年內，逾期將予以銷毀。
- ▶ 證書領取方式：
 - ▶ 現場領取：請於上班日一到五 早上**9:00**-下午**17:00**前攜帶學員本人證件或影本前往領取證書
 - ▶ 郵寄：請將身分證或健保卡正面影本、**36元A4**回郵信封，寄至：**106**台北市羅斯福路四段一號資訊工程系【資訊系統訓練班】收。我們收到後，會儘快幫您掛號寄出。(若要連同收據一起領取請另外於信件中說明。)
- ▶ 收據領取方式
 - ▶ 現場領取：請於上班日一到五 早上**9:00**-下午**17:00**前攜帶學員本人證件或影本前往領取
 - ▶ 郵寄：請將身分證或健保卡正面影本、**36元A4**回郵信封，寄至：**106**台北市羅斯福路四段一號資訊工程系【資訊系統訓練班】收。我們收到後，會儘快幫您掛號寄出。(單領收據或是要連同證書一起領取，皆請於回郵信件中說明。)
 - ▶ 若需提前領取收據，請於開課後先**email**告知(開立收據需經學校審核，審核天數為**1-3**天，確認無誤後才會寄出。若需報帳者請自行評估時間並提前告知)
*欲領取收據者，一律都要在信封內註明清楚，避免行政人員進行回傳作業時遺漏資料!
 - ▶ 課程開立收據時間
實體課程：開課第四堂課以後。
線上課程：開課第三周以後。
★報名年底課程需報帳者，請自行留意公司報帳時間，學校年度結算後無法更改收據開立時間。(EX:2021.12.05已開立的收據；無法再更改為跨年度的時間：2022.01.01)

計分方式& 計分方式

▶ 作業繳交：

- ▶ 以繳交電子檔為主，每個題目一個檔案。(用word繳交)
- ▶ 文字或截圖+黑視窗也要!
- ▶ 第八天課程會公布google表單統一收作業。
- ▶ 作業內容包含程式碼及程式執行結果截圖。

▶ 計分方式：

- ▶ 作業：50%
- ▶ 期末考：30%
- ▶ 出席率：20%

- ▶ 本課程以實務為導向，演算理論討論不深，不適合碩士入學等考試。

參考書目

- ▶ C 語言程式設計(2/e), The C Programming Language, 2/e
 - ▶ 作者：Brian W. Kernighan, Dennis M. Ritchie 著, 蔡文能 譯
 - ▶ 出版商：培生
 - ▶ ISBN：9861541713
- ▶ 資料結構 -- 使用C, 4/e
 - ▶ 作者：蔡明志
 - ▶ 出版商：碁峰資訊
 - ▶ ISBN：9864765795



參考書目

- ▶ C++ 程式設計(3/e)
 - ▶ 作者：張耀仁
 - ▶ 出版商：碁峯
 - ▶ ISBN：986476053X
- ▶ 資料結構 -- 使用 C++, 4/e buy?
 - ▶ 作者：蔡明志
 - ▶ 出版商：碁峰資訊
 - ▶ ISBN：9864765973



參考書目

- ▶ C++ 標準庫 - 學習教本與參考工具(2/e)
 - ▶ 作者：Nicolai M. Josuttis 著、侯捷 譯
 - ▶ 出版商：碁峯
 - ▶ ISBN：9863474398



使用軟體

- ▶ Visual Studio 2019 (講義或在家練習，Windows作業系統使用)
- ▶ Visual Studio 2017 (課堂教室已安裝，Windows作業系統使用)
- ▶ Dev C++ (課堂教室已安裝，Windows作業系統使用)
- ▶ gcc or g++ (Linux 作業系統使用)
- ▶ Xcode (Mac作業系統使用)
- ▶ Code::Blocks(支援Windows、Linux及Mac OS X數種平台)
- ▶ 請勿使用Visual Studio Code，因為設定編譯器過程太繁瑣，不適合新手使用。

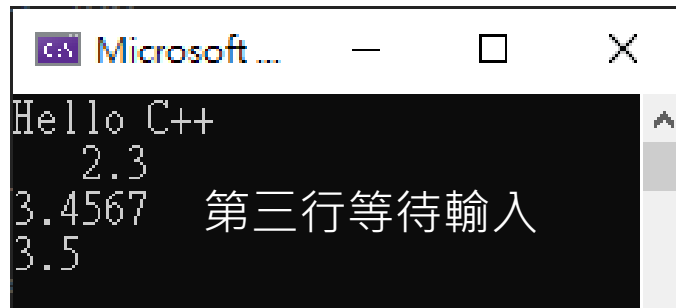
Visual Studio Community

- ▶ 微軟公司所開發，為免費版本，另有付費專業版。
- ▶ 支援多種程式語言，如：**C++**、**C#**、**Visual Basic**、**HTML**、**JavaScript**及**Python** 等程式碼。
- ▶ 目前最新版本為**2022**，近年版本變革較多，介面改變較大，但實際操作大同小異。
- ▶ 本講義內容均使用**2019**版本進行操作練習。

01-Hello C++

命名衝突浮現，把包裝拿掉，程式碼可讀性，有很多種函式庫，你哪知道你用了哪種函式庫，加::，code更容易懂，馬上懂，寫註解！

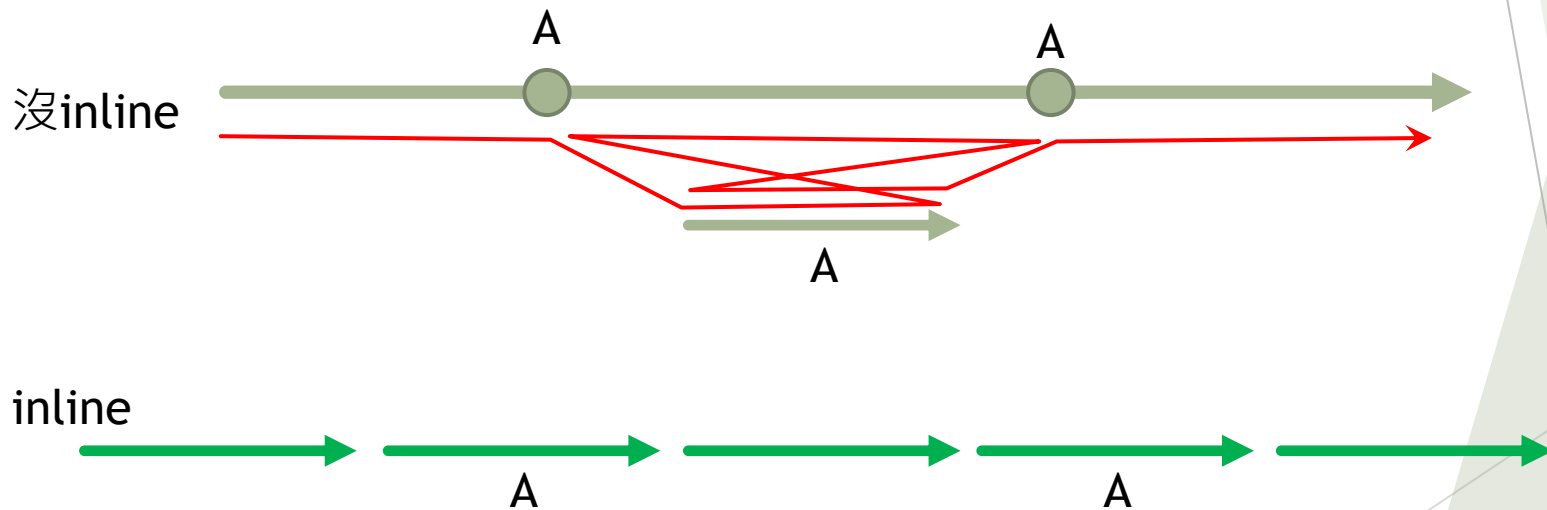
```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
    float f = 2.33333;
    cout << "Hello C++" << endl;
    std::cout << std::setprecision(2) << std::setw(6) << f << std::endl;
    std::cin >> f;
    cout << f << std::endl;
    return 0;
}
```



```
Microsoft ...
Hello C++
    2.3
3.4567  第三行等待輸入
3.5
```

inline函式

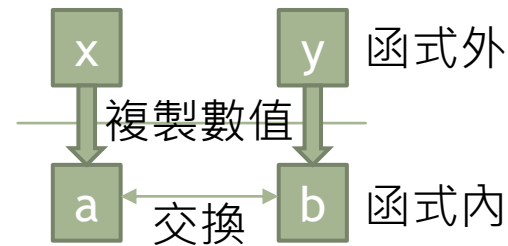
- 使用**inline**可以將函式展開至程式碼中，以減少呼叫函式時間，但會增加程式執行檔大小。



函式-傳值、傳址、傳參考

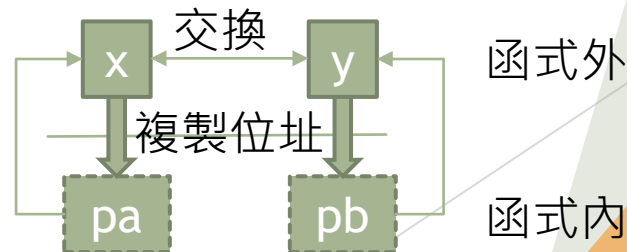
- ▶ 傳值：將數值複製一份給函式參數，不論函式內進行任何運算，皆不會影響原本的數值。

- ▶ `int swap(int a, int b);`



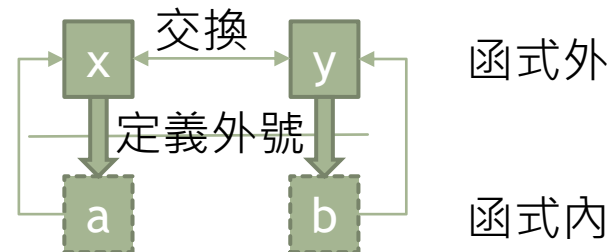
- ▶ 傳址：將位置複製一份給函式參數，透過位置間接對變數進行運算，會修改到函式外的值。

- ▶ `int swap(int* pa, int* pb);`



函式-傳值、傳址、傳參考

- ▶ 傳參考：將變數建立一個參考(外號)機制，透過參考直接對變數進行運算，會修改到函式外的值。
 - ▶ `int swap(int &a, int &b);`



函式-函式重載

- ▶ 同名異式：相同的名稱，進行相同的抽象概念的工作，但可能因為需求回傳型別、參數數量、參數型別的不同，而製作多個函式。
 - ▶ `int swap(int a, int b);`
 - ▶ `int swap(int* pa, int* pb);`
 - ▶ `int swap(int &a, int &b);`
- ▶ 理論上傳值呼叫與傳參考可以同時存在，但因為呼叫的歧義會造成實際無法同時存在。

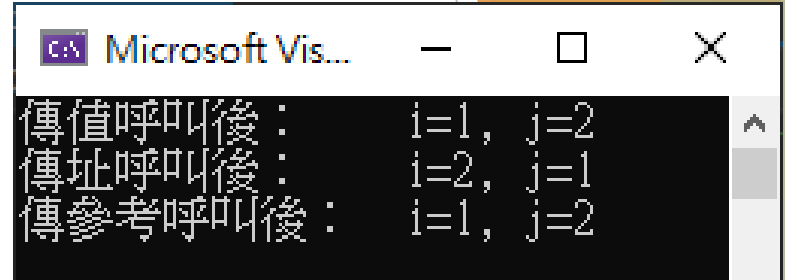
02- 兩數互換

```
#include <iostream>
void swap(int a, int b);
void swap(int* pa, int* pb);
void swap_ref(int& a, int& b);
int main() {
    int i = 1, j = 2;
    std::cout << "傳值呼叫後：\t";
    swap(i, j);
    std::cout << "i=" << i << ", j=" << j << std::endl;
    std::cout << "傳址呼叫後：\t";
    swap(&i, &j);
    std::cout << "i=" << i << ", j=" << j << std::endl;
    std::cout << "傳參考呼叫後：\t";
    swap_ref(i, j);
    std::cout << "i=" << i << ", j=" << j << std::endl;
    return 0;
}

void swap(int a, int b) {
    int c;    c = a;
    a = b;    b = c;
}

void swap(int* pa, int* pb) {
    int c;    c = *pa;
    *pa = *pb; *pb = c;
}

void swap_ref(int& a, int& b) {
    int c;    c = a;
    a = b;    b = c;
}
```



```
Microsoft Vis...
傳值呼叫後： i=1, j=2
傳址呼叫後： i=2, j=1
傳參考呼叫後： i=1, j=2
```

呼叫的歧異性，因為call by reference 跟在call 這個func. 時，會與call by value的func. Name 搞混。