C++

C++ is a statically typed, free­form, multi­paradigm, compiled, general­purpose programming language. It was developed by Bjarne Stroustrup starting in 1979 at Bell Labs as an enhancement to the C language

**目錄**

程式設計

第一章 認識C++ ……………. 3

第二章 基礎+運算式介紹 …………….. 7

第三章 迴圈 ………………….. 12

第四章 函數 ………………….. 22

第五章 陣列 ..................... 24

第六章 字串 ………………….. 27

附錄 內建函數 ………………….. 33

第一章 認識C++

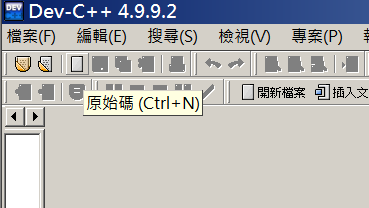
**§1-­1**初識DEV C++ & zerojudge

DEV C++是一套用於開發C/C++的自由的整合性開發環境(IDE)，允許任何目的執行 此程式和再發行複製。

目前最新版本為5.8.3版。不過在比賽時，主辦單位(還有學校)大多提供4.9.9.2版。

如果家中電腦是win8或win10的要載5版的

主畫面->原始碼就是你寫程式的地方。

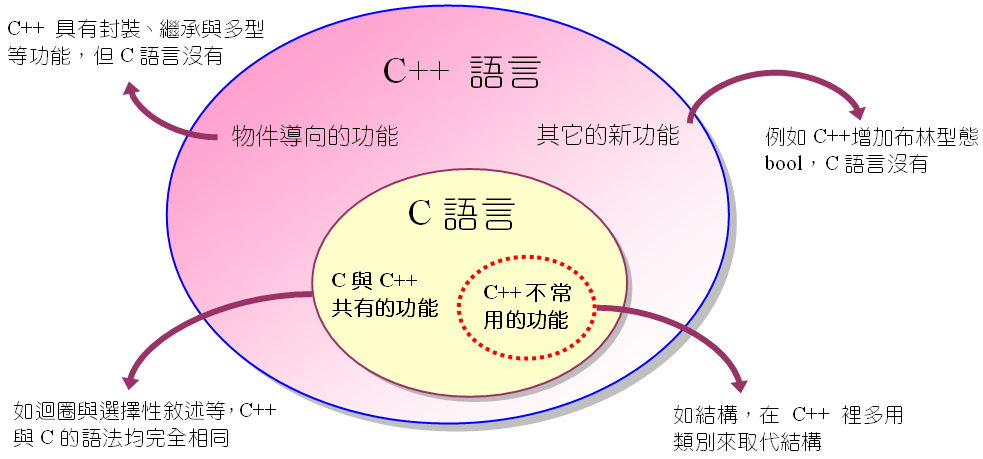


Zerojudge是一個專門為程式設計的初學者(尤其是高中生)設計的，裡面有各種不同難度的題目。除了檢驗自己的學習成果，也能增強自己的實力喔！

分類題庫：可以用題號搜尋自己想做的題目。

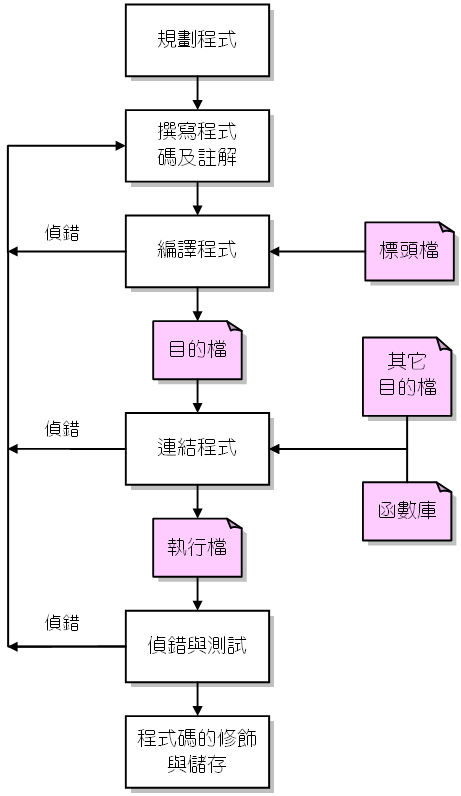
**§1-­2** C++的特色&歷史

* + 高效率且可攜性佳的語言
  + C++在執行前必須先經過編譯，執行效率遠高於直譯式語言
  + 可在修改少許程式碼的情況下，便能在不同的作業系統執行
  + 程式碼的再利用
  + 可依據原有類別所提供的功能，再加上新的需求，進而設計出新的類別，便是程式碼的「再利用」
  + 程式碼的再利用可減少程式碼的開發時間與費用
* C語言是於1972年所發展出來的語言
* C++ 於1980年開始發展
  + 將物件導向（object oriented）加入C語言中
  + 稱為「C with classes」程式語言
* C++具有下列的幾項特色：
  + 向下相容C語言



**§1-­3** 規劃與實作

* 1.規劃程式
* 2. 撰寫程式碼及註解
* 3. 編譯程式
* 4. 連結程式
* 5. 偵錯與測試
  + 語意錯誤（semantic error）
  + 語法錯誤（syntax error）
* 6. 程式碼的修飾與儲存



**§1-­4** 程式碼架構

以下為跑出my first C++的程碼

#include<iostream>

using namespace std;

int main()

{

cout<<"my first C++"; *//記得要用” “把字包起來喔*

system("pause");

return 0;

}

**#include<iostream>**

在程式碼中，include就是引入，將所需的標頭檔(範例為iostream)引入，以使用各功能(範例中為cout)。

C++中輸入輸出是由iostream提供，這也就是為何要在程式的一開頭總是加上它

**using namespace std;**

意義是使用std這個「命名空間」，若省略此行，則cout須寫為std::cout。故務必每次

寫程式都加上本行程式碼，記得喔。

**int main()**

此為程式起始點，是為「主函式」，使用時需使用大括號{}來包住中間的程式碼。

**system("pause");**

若省略這行，則程式執行後會像fireflower一樣一閃而逝，pause意指暫停執行(停住頁面)，以便查看結果。(限Windows使用，如果是5版的不用加他自己就會停了)

**return 0;**

指回傳值為0，通知系統程式執行完畢。

第二章 基礎

**§2-­1** 變數型別

宣告變數的基本方法：**型別名稱 變數名稱**;

註：[命名時，它不可以使用數字作為開頭，也不可以使用一些特殊字元，像是!@#$](mailto:像是!@#$之)之類的字元，更不可以和關鍵字同名，例如int、float、class等等。

基本資料型態共有九種，如下表

|  |  |  |  |
| --- | --- | --- | --- |
| 型態 | 型別名稱 | 大小 | 資料範圍 |
| 布林 | bool | 1 byte | True或False |
| 字元 | char | 1 byte | 0~255 |
| 字串 | string | 10 bytes  加字串長度 | 0 到大約 20 億 |
| 整數 | short | 2 bytes | ­2^15 ~ 2^15 ­1 (2^15=32768) |
| 整數 | int | 4 bytes | ­2^31 ~ 2^31 ­1 (2^31=2147483648) |
| 整數 | long long | 8 bytes | ­2^63 ~ 2^63 ­1 (2^63=18446744073709551616) |
| 浮點數 | float | 4 bytes | 精確至7位 |
| 浮點數 | double | 8 bytes | 精確至8位 |

除了使用以上的型別名稱外，還可以使用修飾詞修改型別的一些屬性。例如常用的unsigned可以使變數不得儲存負數，但儲存範圍放大兩倍，以後會再提到。

**§2-­2** 範例程式**I**—標準輸入與輸出

建議先把範例程式碼打出來，然後再看本頁說明

範例：

#include<iostream>

using namespace std;

int main()

{

int a;

*// int是「整數」的意思，即前一頁的「變數型態」*

*// a則是「變數名稱」，只是個名字，想怎樣打都可以，可不只一字*

*//變數名稱可以有數字，但開頭需是英文，例如：abcd123*

*//所謂的變數，就是一塊記憶體空間，用於記憶一些資料*

*//「整數變數」就是指一塊用於儲存一個整數的空間*

cout<<"請輸入一數：";  *//輸出""內的字*

cin>>a; *//執行到本行會停下來要求輸入，並將輸入傳給a*

cout<<"您輸入的數是 "<<a<<endl; *//endl是輸出換行*

system("pause");

}

編者常把 >>"讀流進" <<讀流出。你照這樣念念看上面程式碼。

輸出時非變數要用" "框起來，而特殊字元呢? 我要怎輸出 "，難不成是" " "

當然不是，要用所謂的跳脫自元

如: cout<<"我是\"學生\"";

會輸出: 我是**”**學生**”**

其他跳脫字元：

|  |  |
| --- | --- |
| \n | 換行 |
| \t | Tab |
| \a | 嗶嗶聲 |
| \f | 換頁 |
| \’ | 單引號 |
| \" | 雙引號 |
| \\ | 反斜線 |
| \b | 指標倒退一格 |

**§2-­3** 算術運算子 & 賦值運算子

|  |  |
| --- | --- |
| 運算子名稱 | 語法 |
| 加法 | a + b |
| 減法 | a - b |
| 乘法 | a \* b |
| 除法(去小數) | a / b |
| 餘數 | a % b |

Ex1: int a=5 , b=6 , c ; c=a+b; c就是11

但是請看看 a = a+6 ; 這個等式不成立吧。但執行結果a就是11。在程式語言裡面， ”=” 代表 ”將右邊的值存進左邊的變數” 可能要花點時間接受

因此你現在知道a=a+1;會由甚麼結果了吧，在程式里左邊的程式碼會以後常用到， 有一種簡單的寫法 a++;

Ex2: a=a+1; 等同於 a++;

|  |  |
| --- | --- |
| 用途 | 符號 |
| 加法後賦值 | += , ++ |
| 減法後賦值 | -= , -- |
| 乘法後賦值 | \*= |
| 除法後賦值 | /= |
| 餘數後賦值 | %= |
|  |  |

**§2-­4** 範例程式**II**—數值輸出

#include <iostream>

using namespace std;

int main()

{

int a,b;

b=5; *//將b變數指定為整數5，注意左右不可以反過來*

a=b\*2+3; *//將a變數指定為b變數乘以2再加3，也就是13*

cout<<"a="<<a<<endl

<<"b="<<b<<endl;

*//箭頭裡直接放上變數名稱，就會輸出變數的值*

*//a=跟b=是字串，就是單純地把這兩個字給輸出*

*//這裡用上了C++習慣編寫風格之一：一行輸出一行程式碼*

*//注意第一行並沒有分號，而第二行沒有cout，因為這兩行其實本來是一行*

*//這一句程式碼會依序輸出a=兩個字、a變數值、換行、b=兩字、b變數值、換行，然後才結束*

*//為了方便閱讀，把實際輸出時的兩行字在程式碼裡分兩行寫*

cout<<"a+b="<<a+b<<endl;

*//輸出"a+b="四個字之後輸出a b兩變數相加的值，然後換行*

system("pause");

return 0;

**Zerojudge**實作練習：

d483：hello, world

}

**§2-­5** 字元與ASCII碼

一個英文字母、數字或其他的符號，我們稱它為字元。要表示一個字元，我們可以用一對單引號 ' 把該字元夾起來，例如：

**char c='a';**

事實上，字元在電腦中是以一個八位元的整數來儲存(即 1 Byte)，而這個符號與數字的對應關係我們稱為 ASCII 碼 (American Standard Code for Information Interchange)，例如：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 數字 | 符號 | 數字 | 符號 | 數字 | 符號 |
| 48 | **0** | 65 | **A** | 97 | **a** |
| 49 | **1** | 66 | **B** | 98 | **b** |
| 50 | **2** | 67 | **C** | 99 | **c** |
| 51 | **3** | 68 | **D** | 100 | **d** |
| 52 | **4** | 69 | **E** | 101 | **e** |

也就是說，其實字元也是一個數字，因此也可以拿來做加減乘除等四則運算。

例如:

**char c='a';**

**c=c+3;**

**cout<<c;**

上面的例子會印出字元 d。

詳細可查關鍵字 ascii，至於中文比較複雜，我只能說他占兩個字元。

第三章 迴圈

**§3-­1** 關係運算式

|  |  |  |
| --- | --- | --- |
| 運算子 | 說明 | 範例 |
| == | 等於 | a == b |
| != | 不等於 | a != b |
| < | 小於 | a < b |
| > | 大於 | a > b |
| <= | 小於等於 | a <= b |
| >= | 大於等於 | a >= b |
| ! | 邏輯取反 | !a |
| && | 邏輯AND | a && b |
| || | 邏輯 OR | a || b |

注意：

|| 跟 && 一次要打兩個，只打一個另有意義

**§3-­2** While迴圈

C++提供while迴圈式，它根據您所指定的條件式來判斷是否執行迴圈本體，語法如下所示：

**while (條件式) {   
    陳述句一;   
    陳述句二;   
}**  
在迴圈後如果有兩個以上陳述句，稱之為「複合陳述句」（compound statement），此時您必須使用{ }將複合陳述句包括起來。While通常用於重複性的動作，而停止條件未知何時發生的情況，例如一個使用者輸入介面，使用者可能輸入10次，也可能輸入20次，這時迴圈停止的時機是未知的，您可以使用while迴圈來作這個事。

**§3-­3** 範例程式**III**—While迴圈

#include <iostream>

using namespace std;

int main()

{

char c;

cout<<"Press a Key:";

cin>>c;

while(c!='E' && c!='e')

*//運算子 && 叫做and，****3-1****有提到過*

*//接收兩布林參數，若皆為真則傳回真，否則傳回假*

*//也就是說 如果c不等於大E也不等於小e 引數就為真*

*//while就是所謂的迴圈*

*//跟if一樣接收一個布林引數*

*//若為真，則執行下一句並再次判斷引數，直到判斷為假為止*

*//如果程式邏輯錯誤，就會造成while迴圈永遠不會判斷為假*

*//這就稱為「無窮迴圈」，會讓程式卡死*

{

*//有時候，判斷條件後要執行的敘述句不只一句*

*//這時可以用大括號 { } 把一句以上的敘述句框起來，會視為一個敘述句*

cout<<"Press E or e to exit:";

cin>>c;

} *//這是while的大括號*

system("pause");

**Zerojudge**實作練習：

a001：哈囉

a002：簡易加法

d461：班際籃球賽

d559：班號

return 0;

}

**§3-­4** If條件判斷

為了應付程式可能遇到的各種狀況，C++提供了if 條件判斷陳述，它的語法如下：

**if (條件式一)   
    陳述句一;   
else if (條件式二)   
    陳述句二;**

注意：**(條件式)** 之後不需要加分號，不然分號會被視為一個空白敘述句

如果你需要的條件很多個

else if可以有很多個

**else**

**陳述句三;**

這個語法的意思，白話來說，就是當條件式一成立時，則執行陳述句一；當條件式二成立時，則執行陳述句二，要不然就執行陳述句三；如果條件式一二皆不成立時並不想作任何事，則else可以省略。   
  
和While一樣，如果有兩個以上陳述句，也要使用{ }將複合陳述句包起來，例如：

**if (條件式) {   
    陳述句一;   
    陳述句二;   
}   
else {   
    陳述句三;   
    陳述句四;   
}**

**※延伸概念※**

一組if...else...判斷式，在程式裡可以視同一個敘述句。

也就是可以放在另一組if的後面，判斷前組if為真時才判斷後面那組if是真是假。

也就是在多組的if...else當中，在if敘述句寫入普通敘述句，在else之後緊鄰下一組if...else。

造成的效果，就是依序判斷每個if的引數。

當判斷為真時，執行該if的敘述句並離開整組else if。

因為從該敘述句結束開始，就都屬於else的範圍，所以會全部跳過，直到else if結束。

**§3-­5** 範例程式**IV**—If條件判斷

#include <iostream>

using namespace std;

int main()

{

int a,b;

char c;

cin>>a>>c>>b;

if(c=='+')

*//當引數 (c=='+') 判斷為〈true〉時，會執行下一個敘述句並一直執行下去*

*//若引數判斷為「假」〈false〉，則會跳過下一個敘述句，從下下句繼續執行*

*//這裡的下一句就是指cout這行，要判斷為真才會執行輸出*

*//一般習慣把下一句寫在『下一層』，也就是前面多打幾個空白，便於閱讀*

cout<<a<<"+"<<b<<"="<<a+b<<endl;

else if(c=='-')

cout<<a<<"-"<<b<<"="<<a-b<<endl;

else if(c=='\*'||c=='X'||c=='x')

*//接收兩個布林型態參數〈是或否〉，若其中有一個為是，則回傳是，兩者皆否則回傳否*

cout<<a<<"\*"<<b<<"="<<a\*b<<endl;

else if(c=='/')

cout<<a<<"/"<<b<<"="<<a/b<<endl;

**Zerojudge**實作練習：

a003：兩光法師占卜術

a004：文文的求婚

a053：Sagit's 計分程式

d460：山六九之旅

d064：ㄑㄧˊ數？

else

cout<<"Wrong Input."<<endl;

system("pause");

return 0;

}

**§3-­6** For迴圈

for迴圈跟while有著異曲同工之妙，基本上可以使用while取代所有的for

一般是用在執行特定次數的迴圈，並為了美觀而自while獨立出來

在C++中所提供的重複性計算陳述之一是 for 迴圈式，它的基本語法如下：

**for (初始變數; 判斷式; 遞增式) {   
    陳述句一;   
    陳述句二;   
}**  
如果陳述句只有一個，也就是非複合陳述句，{ } 可以省略不寫；for 迴圈的第一個初始變數陳述句只會執行一次，之後每次重新進行迴圈時，都會根據判斷式來判斷是否執行下一個迴圈，而每次執行完迴圈之後，都會執行遞增式一次。

**for (**變數 **=** 起始直 **;** 判斷式**(**不打則視為**1) ;** 變數增減 **)**

變數可直接宣告於for內，變數增減可寫為 a += 1，請參考2-3說明。

從0到10，包括10，每次加1

for( int i = 0; i <= 10; i++ )

從5到1，不包括0，每次減1

for( int j = 5; j > 0; j­­ )

從10到0，包括0，每次減2

for( int k = 10; k >= 0; k ­= 2 )

**§3-­7** 範例程式**V**—For迴圈

#include <iostream>

using namespace std;

int main()

{

for(int i=10;i>0;i--)

*//C++允許變數宣告在程式碼中間〈C不可，一定要在最前面〉*

*//所以也可以把變數宣告在for參數列裡面*

*//這樣一來就不用多一行來宣告 i*

*//不過這樣一來， i 就不可以在迴圈以外的地方使用*

*//雖然一般的for都是從0開始，條件為小於次數*

*//但有些情況也可以玩『倒數』，像是現在這樣〈但大多數還是正數〉*

*//要注意的是要改成大於跟--，不然會變無限迴圈或不執行*

*//對應於++ ， i-- , --I , i-=1 , i=i-1 都是讓i值減少1*

*//不過沒有\*\*i , //i ，因為沒有意義*

cout<<i<<" ";

**Zerojudge**實作練習：

a147：Print It All

a244：新手訓練~ for + if

d532：文文的求婚 (三)

d498：我不說髒話

*//這裡輸出的第二個參數不同於以往的字串*

*//用的是單引號，也就是字元*

cout<<endl;

system("pause");

return 0;

}

**§3-­8** 範例程式**VI**—(break,continue,goto )

break可以離開目前switch、for、while、do while的區塊，但是只能出現在switch、for、while中，否則會發生編譯錯誤。

#include<iostream>

using namespace std;

int main(){

int a; cout<<"1+1=";

while(cin>>a){

if(a==2){

break; *//如果a=2則跳出while*

}

cout<<"錯誤，請重新輸入"<<endl;

}

cout<<"正確";

system("pause");

}

continue與break相似，只是他是跳回迴圈區塊的開頭繼續下一個迴圈，而不是離開迴圈

#include<iostream>

using namespace std;

int main(){

for(int i=0;i<=10;i++){

if(i==5){

continue; *//如果i=5就會直接重跑，所以5沒有輸出*

}

cout<<i<<endl;

}

system("pause");

}

goto是個很方便，但是最不被建議使用的語法，濫用它的話會破壞程式的架構。

goto可以在程式中任意跳躍，跳躍前必須先設定好目的地，跳躍時必須指定目的 地。

#include<iostream>

using namespace std;

int main(){

int a;

begin: cout<<"1+1=";

cin>>a;

if(a!=2){

goto error;  *//如果輸入不是2就跳到error*

}

cout<<"正確";

system("pause");

return 0;

error:

cout<<"錯誤，請重新輸入"<<endl;

goto begin; *//跳到begin*

}

**§3-­9** 選擇敘述( switch )

選擇敘述switch同等於VB的Select Case，以下為語法說明：

**switch(已宣告的變數) {**

**case '整數常數':**

可以有很多個case

而陳述句和break也隨之增加

**陳述句;**

**break;   
}**  
不同於之前是需要時才要打，switch一定要有大括號

case 空白後接一個整數常數並加上冒號，注意不使用括號跟分號

所謂的常數，就是一個直接打上來的數值，例如： 0 'a' 0x2B "Hey!!"

程式會一一比對switch的參數跟哪一個case的常數值一樣，就從該case開始執行

case若輸入變數、數值〈算式〉、重複的常數都會發生編譯錯誤

因為下一個case並不會影響這個case的執行，若是不加break就會往下執行到別的狀況。

**§3-­10** 範例程式**VII**—選擇敘述( switch )

#include <iostream>

#include <cmath>

using namespace std;

*//今天多了一行include*

*//這是為了使用「非標準函式庫」所需要的*

*//在寫程式語言時，會用到各式各樣的「函式」*

*//其中常用到的一些會被訂為「標準函式」，像是cout等等*

*//標準函式強制每個編譯器編譯出來都會是一樣作用*

*//而非標準函式則有可能會因編譯器而有所不同*

*//因此，一般並不建議使用*

int main()

{

int a,b;

char c;

while(1) *//無窮回圈的應用之一*

*//為了讓程式可以不斷執行，而故意讓while的參數為常數1*

*//但是要確保有正常的結束方式，不然就得強制關閉*

{

cin>>a>>c>>b;

*//依序讀入一整數、一字元、一整數*

*//也就是一個基本的算式*

switch(c) {

*//switch()接收一個整數參數〈在此把char當成整數來用〉*

*//會比較其值並從指定處繼續執行程式*

case '+':

cout<<'='<<a+b<<endl;

break;

*//break會對switch生效，跳到switch結束，而continue則會穿透switch*

case '-': *//注意case後面除了冒號外，不需要再一層的大括號*

cout<<'='<<a-b<<endl;

break;

case '\*':

case 'X':

*//因為不會互相干擾，可以像這樣一次指定多種狀況*

case 'x':

*// '\*'和'X'的case會先執行完自己本身〈空的〉，再往下執行到'x'的case*

cout<<'='<<a\*b<<endl;

break;

case '/':

cout<<'='<<(double)a/b<<endl;

*// (double)是轉換型態語法*

*//在小括號內輸入一個型態，可以強制將後方的數值轉換成這個型態*

*//不過也有些型態間的轉換會遺失資訊或無法轉換*

*//例如double〈倍精浮點〉轉int會捨小數，字串無法轉換成其他型態*

*//這裡要轉換，是因為直接用int相除會自動去小數*

break;

*//雖然是最後一個case，但是打上break還是個好習慣，也許哪天下面會再多一個case出來，這個缺少的break就變成bug了*

}*//switch的大括號*

}*//while的大括號*

*//由於這裡不可能會執行到，所以無需暫停指令*

return 0;

}

第四章 函數

**§4-­1** 了解函數

每次寫程式都有 int main() ←這就是一個函數，而我們把程式碼放在主函數main() 裡面。就像變數一樣，一個函式使用前，必須先「宣告〈declare〉」它

函式的宣告基本上分為三部分：回傳值型態、函式名稱、以及參數列

**宣告：**

回傳值型態 函式名稱(參數列)

{

　　程式碼

}

**定義：**

回傳值型態顧名思義，就是這個函式的回傳值是什麼型態

函式名稱就是這個函式的名字，注意不可以跟變數名稱撞名

參數列就是這個函式將接收幾個、什麼型態的數值作為參數

若有多數個參數以 , 分開，格式跟變數宣告很像，參數型態後面加上名稱

在函式宣告時可以將名稱省略，因為這不重要，就算打了名稱也會被編譯器忽略

函式宣告跟變數宣告一樣，後面要加上分號

|  |  |  |  |
| --- | --- | --- | --- |
|  | 回傳值型態 | 函式名稱 | 參數列 |
| Ex1 | int | main | () |
| Ex2 | bool | putDigit | (int,bool) |

當程式變大了會發現，有些常用到的區域常出現，這時我們就可以把這段程式碼寫進一個函數裡。簡單來說，就是你丟一些變數進函數，函數跑一跑，丟出一個值來。(或之後對陣列or全域變數做更動)

多說無益，請看隔壁頁範例

**§4-­2** 範例程式**VIII**—自訂函數

目標：(sum是把1到n加總起來)

#include<iostream>

using namespace std;

int sum(int); *//宣告函數sum，類型是int*

int maim(){

int n,ans;

cin>>n;

ans = sum(n);  *// n丟進sum這個函數的結果*

cout<<ans<<endl;

system("pause");

return 0;

}

int sum(int n){ *//宣告n項於此函數的計算方式*

int s=0;

for(int i=0;i=n;i++){

s += i;

}

return s; *//s這個值被回傳，也就是n丟進sum這個函數的結果*

}

**§4­3** 內建函數

c內建很多好用的函數，詳見32頁。

第五章 陣列

**§5-1** 一維陣列

是時後了解宣告變數是怎一回事了。int a;就像跟hotel(記憶體)開個房間(空間)，這個空間的大小由變數型態決定。

至於陣列就是相鄰的房間，用法相當方便。有時會有多份相關聯的資料，需要用到很多個變數儲存，使用a1,a2,a3...這樣的宣告法會很不好用，因為資料量隨便都有可能破百。這時就可以使用**陣列**。陣列是一連串的變數，緊鄰在一起，並共用一個名稱〈陣列的名稱〉 。讀取陣列要在變數名稱後面加上中括號 [ ] 並在裡面寫上「索引」值 〈Index〉。不同的索引值會對應到陣列中不同的「元素」〈也就是單一個變數〉索引是一個整數值，可以經由計算得出。

宣告變數**:**

資料型態 陣列名[陣列大小];

例：int a[100]; //宣告一個整數陣列，可以儲存100個整數。

float b[50]; //宣告一個浮點數陣列，可以儲存50個浮點數。

int c[]={1,2}; //如果你沒有打陣列大小，就會依照你指定的陣列內容數開房間(像這個就會開2個房間)

索引值：在陣列每個元素都有的一個編號，稱之為索引值，從0到陣列大小­1。切 記！不是從1到陣列大小。

宣告陣列 陣列可以利用for迴圈進行處理。

**§5-­2** 範例程式**IX**—陣列

#include <iostream>

using namespace std;

int main()

{

int a[10];

*//宣告一個整數陣列，內含10個元素*

*//C系列裡陣列的索引值，從0開始*

*//也就是a[0] a[1] a[2] ... a[8] a[9] 一共十個元素，a[10]並不存在*

*//索引值可以是任一個整數，可以超過預設大小，也可以是負數*

*//但是當所存取的索引值超出實際範圍，就會存取到毫不相關的資料*

*//這會造成無法預期的結果，因為沒人知道記憶體中陣列旁邊放著什麼資料*

for(int i=0;i<10;i++)

*//for迴圈習慣從0開始計數，就是為了配合陣列索引從0開始的設定*

a[i]=i\*i;

*//存取第i個元素，並指定為i平方*

a[20/4-5]+=10;

*//可以在索引裡打上算式，這裡算出來會是0*

*// += 就是讓該變數值增加多少*

for(int i=0;i<10;i++)

switch(i)

{

case 0:

cout<<"The 1st value is: "<<a[0]<<endl;

break;

case 1:

cout<<"The 2nd value is: "<<a[1]<<endl;

break;

case 2:

cout<<"The 3rd value is: "<<a[2]<<endl;

break;

default: *//default是為「預設值」，指跟所有的case都不合的情況*

cout<<"The "<<i+1<<"th value is: "<<a[i]<<endl;

break;

}

system("pause");

return 0;

}

**§5-­3 二維陣列**

二維陣列表示法與一維陣列類似，是由兩個中括號所組成的

宣告變數**:**

資料型態 陣列名[陣列大小] [陣列大小];

例：int a[100][100]; //宣告一個整數陣列，可以儲存100x100個整數。

float b[50][50]; //宣告一個浮點數陣列，可以儲存50x50個浮點數。

int c[]= {{1, 2, 3},{4, 5, 6}}; //如果你沒有打陣列大小，就會依照你指定的陣列內容數開房間(像這個就會開2個大房間，每個大房間下面再開3個小房間)

以下由圖說明(a[3][3])

|  |  |  |
| --- | --- | --- |
| a[0][0] | a[0][1] | a[0][2] |
| a[1][0] | a[1][1] | a[1][2] |
| a[2][0] | a[2][1] | a[2][2] |

如上圖,a[i][j]代表所開出的二維陣列有i列,j行

第六章 字串

**§6-1** 字串

在C++中字串的本質是由字元所組成的陣列，並在最後加上一個空字元，下面這個程式就是一個"hello"字串的宣告：

**Char str[ ]= {'h', 'e', 'l', 'l', 'o', '\0'};**  
之後您可以直接使用str來代表該字串，例如在文字模式下輸出str字串：

**cout << str << endl;**   
C++是使用空字元來識別一個字元陣列是否表示字串，像上例就可以用來表示一個字串.

字串是字元陣列，所以您可以使用陣列的存取方式取出每一個字元，在指定"hello"時表面上雖然只有5個字元，但是最後會加上一個空字元'\0'，所以 str[]共使用了6個字元，由於字串的最後一個字元為空字元，所以您可以檢驗這個字元來看看字串是否為空，例如：  
**if(str[0] == '\0') {  
    cout << "字串為空";  
}**

空字元在作條件判斷時，會被視為0，所以上式還可以這麼寫：  
**if( ! str[0] ) {  
    cout << "字串為空";  
}**  
字串一但指定，它的長度就固定了，也就是字元陣列的長度，所以您由使用者輸入取得字串值時，需注意不要超過字串的長度；從使用者輸入取得字串值只要這麼作 就可以了：   
**char str[80];   
cout << "輸入字串：";   
cin >> str;   
cout << "您輸入的字串為 " << str << endl;**

這個程式片段可以取得使用者的字串輸入，所輸入的字串長度不得超過80個字元，80個字元的上限包括空白字元，所以實際上可以輸入79個字元；如果輸入的字元超出所宣告的上限，會發生不可預期的錯誤。

**a020: 身分證檢驗**

例題練習→

我國的身分證字號有底下這樣的規則，因此對於任意輸入的身分證字號可以有一些基本的判斷原則，請您來判斷一個身分證字號是否是正常的號碼(不代表確有此號、此人)。

(1) 英文代號以下表轉換成數字  
  
      A=10 台北市     J=18 新竹縣     S=26 高雄縣  
      B=11 台中市     K=19 苗栗縣     T=27 屏東縣  
      C=12 基隆市     L=20 台中縣     U=28 花蓮縣  
      D=13 台南市     M=21 南投縣    V=29 台東縣  
      E=14 高雄市     N=22 彰化縣     W=32 金門縣  
      F=15 台北縣     O=35 新竹市     X=30 澎湖縣  
      G=16 宜蘭縣     P=23 雲林縣     Y=31 陽明山  
      H=17 桃園縣     Q=24 嘉義縣     Z=33 連江縣  
      I=34 嘉義市     R=25 台南縣  
  
  (2) 英文轉成的數字, 個位數乘９再加上十位數的數字  
  
  (3) 各數字從右到左依次乘１、２、３、４．．．．８  
  
  (4) 求出(2),(3) 及最後一碼的和

  (5) (4)除10 若整除，則為 real，否則為 fake

輸入說明 ：

一組身分證號碼

輸出說明 ：

輸出 real or fake

範例輸入 ：

T112663836

S154287863

範例輸出 ：

real

fake

#include<iostream>

using namespace std;

int main(){

char A;

char num[9]; *//開一個字元陣列有9個空格*

while(cin>>A>>num){ *//輸入英文字母和9個數字*

int ans=0; *//要初始化 這樣才不會影響到下一次的運算值*

if(A>='A'&&A<='H'||A=='W') A= A- 'A '+ 10;

*//如果英文字母的值介於A~H，輸出題目要的值*

else if(A>='J'&&A<='N') A = A - 'A' + 9;

*//如果英文字母的值介於J~N，輸出題目要的值*

else if(A>='P'&&A<='V') A = A -'A' + 8;

*//如果英文字母的值介於P~V，輸出題目要的值*

else if(A=='I') A=A-'A' + 26; *//如果英文字母的值等於I，輸出題目要的值*

else if(A=='O') A=A-'A '+ 21; *//如果英文字母的值等於O，輸出題目要的值*

else if(A=='X') A=A-'A '+ 7; *//如果英文字母的值等於 X，輸出題目要的值*

else if(A=='Y') A=A-'A' + 7; *//如果英文字母的值等於Y，輸出題目要的值*

else if(A=='Z') A=A-'A' + 8; *//如果英文字母的值等於 Z，輸出題目要的值*

int k=0; *//宣告變數k*

for(int i=0;i<9;i++){

k+=(8-i)\*(num[i]-'0'); *//讓他做題目要的運算*

}

ans=(A%10)\*9+A/10+k+num[8]-'0';  *//輸出答案*

if(ans%10==0)

cout<<"real"<<endl;

else

cout<<"fake"<<endl;

}

system("pause");

return 0;

**Zerojudge**實作練習：

b041:密碼安全設定

(可用內建函數寫，但最好不要)

//這題如果你自己解的出來，那麼你字元和字串幾乎無敵了!!!

}

**§6-2 string**

陣列本身對自己的長度沒有意識，所以無法判斷自己是否為空字串，而陣列也不能直接指定給另一個陣列，所以您無法直接將字串指定給另一個字串，您也無法對兩個字串直接進行串連的動作

char str1[] = "text1";

char str2[] = "text2";

str1 = str2; // error

cout << str1 + str2 << endl; *// error*

C++標準函式庫提供了string類別 **#include <string>**

您可以使用三種方式來建立一個string類別的物件（object），例如：

string str1; // 建立一個string物件，內容為空字串

string str2("caterpillar"); // 以字串常量建立字串

string str3(str2); // 以string實例建立字串

#include<iostream>

#include<string>

using namespace std;

您也可以使用size()來測試字串長度，使用empty()來測試字串是否為空字串

int main(){

string a,b;

string c("test");

cin>>a;

b=c;

cout<<"a:"<<a<<endl;

cout <<"b:"<<b<<endl;

cout <<"c:"<<c<<endl<<"a:"<<endl;

for(int i=0 ; i<a.size() ; i++){

cout<<a[i]<<endl;

}

system("pause");

}

**§6­3 ( cin.getline() , getline() )**

一般的cin是以空白、換行、tab鍵來做資料的結束，所以使用cin來輸入會無法取得 空白。

cin.getline()跟getline()可以解決這個問題。

這兩個輸入用的函數只用換行來當作資料 的結束，所以可以接收空白和tab鍵的資料。

cin.getline()是給字元陣列使用，getline()則是給string使用。以下為兩種函數的使用方法：

#include<iostream>

#include<string>

using namespace std;

int main(){

char a[10];

string b;

cin.getline(a,10); *//第一個值是字元陣列名稱，第二個值則是決定取前幾個字元*

*//注意:上次提過最後一個字元是\0，所以這裡實際只取前9個字*

getline(cin,b);

*//第一個值就打cin，第二個值則是string的名稱*

cout<<"a:"<<a<<endl;

cout<<"b:"<<b<<endl;

system("pause");

}

**<附錄>**

**處理字元陣列的函數 以下a,b皆為字元陣列**

**#include<cstring>**

內建函數：

strcpy (a,b) ; 將b複製給a

strncpy (a,b,5) ; 將b複製前5個字元給a

strcat (a,b) ; 在a後面加上b

strncat (a,b,5) ; 在a後面加上b前面5個字元

strcmp (a,b) ; 如果兩個字元陣列都一樣回傳0

strchr (a,’s’) ; 在a裡找s第一個出現的位址

strlen (a) ; a有幾個字，不包含’\0’

**字串**

**#include<string>**

以下ans皆為字串string，a,b皆為數字(a,b可省略不打)

ans.size() ans有幾個字元

ans.empty() ans是否為空字元

ans.assign(string,a,b)

從string的第a個字元開始取出後b個字元來指定給此字串物件。

ans.append(string,a,b)

從string的第a個字元開始取出後b個字元來附加至此字串物件後。

ans.find(string,a)

在std的第a個字元後檢查第幾個字元開始是string的子字元(沒有則回傳­1)。

**檢查字元 以下a皆為字元**

**#include<cctype>**

isalnum(a) 是否是字母或數字

isalpha(a) 是否是字母

isdigit(a) 是否是數字

isgraph(a) 是否是數字、字母或標點

isupper(a) 是否是大寫

islower(a) 是否是小寫

ispunct(a) 是否是標點

isprint(a) 是否可輸出字元(可視字元)

isxdigit(a) 是否是十六進位數

tolower(a) 轉換為小寫

toupper(a) 轉換為大寫

**C語言的特殊字元函數庫(C++缺此函數庫)**

#include<conio.h>(這是c的非標準函式庫，並非所有支援c的平台都可使用)

getche()

鍵盤取得一個字元存入變數，不必等待按Enter鍵。

getch()

鍵盤取得一個字元存入變數，不必等待按Enter鍵，也不會輸出。

**數學**

參數可為數字或變數，回傳答案

**#include<cmath>**

sqrt() 計算開方值

pow(x,y) 計算x的y次方值

ceil() 計算無條件進位值

floor() 計算無條件捨去值(高斯函數)

cos() 餘弦函數

sin() 正弦函數

tan() 正切函數

log10() 對數函數以10為底)

max(x,y) x,y取出較大的數

**其他的函數庫**

**#include<cstdlib>**

atof() 轉換字串至倍精度浮點數

atoi() 轉換字串至整數

atol() 轉換字串至長整數

rand() 取得亂數

srand() 刷新亂數表(以免取得相同亂數)

system() 系統語言

**塗鴉牆**

版權所有，翻印必究

您的指正，我們由衷感謝 若您對本書的內容，編排有任何意見

歡迎至 http://ssinrc.org/feedback留言 或Email[至：admin@ssinrc.](mailto:admin@ssinrc.org)org

書 名/ 程式語言學習資料

出 版/ 臺北市立松山高級中學資訊研究社

主編/江承翰

編輯群/ 崧資第26屆幹部

封 面/ 陳詠萱

參考網站[**/**http://www.cplusplus.com/reference](http://www.cplusplus.com/reference/)/

參考書籍：C++教學手冊

編 輯/ 松山高中資訊研究社

日 期/ 民國104年9月

網 址/ http://ssinrc.org