Chapter 9 ： 函式之參數傳遞

C/C++所提供的函式參數傳遞方式有，傳值法(call by value)，傳位址法(call by address)，與傳參考法(call by reference)三種。首先，我們先來看傳值法。

9.1 傳值法(call by value)

格式:

函式定義/原型宣告: `void function(int x, int y);` //x, y 為 參數(parameters)
函式呼叫: `function(a, b);` //a, b 為 引數(arguments)

呼叫時，函式參數與輸入變數(引數)的關係，相當於：`int x = a; int y = b;`

說明:

傳值法的定義是函式的接收參數(parameter)，其型態為變數或物件，函式呼叫時所對應的引數(argument)/傳入值，即是該參數的初始值。

當引數為「變數」時，傳值法的函數呼叫只是將呼叫端(下例之主程式)的輸入變數(引數)的值，「拷貝」給函式中對應的參數，以後兩個變數間就不再有任何關係(二者有各自獨立的記憶體空間)。對函式內的變數或參數做任何運算都不會改變主程式中的變數值，此為傳值法的特點。

例:

```
int addbyone(int x )
{
    x++;
    cout<<x<<endl;          //執行後,x=101
    return x;
}

void main()
{
    int x = 100;
    int y = addbyone(x);
    cout<<x<<endl;          //函式呼叫後,x 仍保持為 100
}
```

程式範例: cpp_ex40.cpp

注意要點:

如欲由函式傳回值，需使用 return 指令。但一函式只能執行 return 一次，故只能有一個傳回值，更改函式呼叫等號右邊的變數值。如欲使用函式來改變主程式中的多個變數值(傳回兩個以上的值)，則需使用傳位址法或傳參考法。

9.2 傳位址法(call by address)

格式:

```
函式定義/原型宣告:    void function(int *x, int *y);   //x, y 為 parameters
函式呼叫:             function(&a, &b);                //&a, &b 為 arguments
```

呼叫時，函式參數與輸入變數的關係，相當於：int* x = &a; int* y = &b;

說明:

傳位址法的定義是函式的接收參數(parameter)，其型態為變數或物件指標，函式呼叫時的引數(argument)/傳入值，是輸入變數或物件的記憶體位址。

傳位址法的做法是將傳值法中接收某輸入變數之值的拷貝，改用指標做參數來「指到」呼叫時之輸入變數，由於指標儲存的是記憶體位址，故呼叫時傳入的需是輸入變數或物件的記憶體位址(可使用「&」取址)。換句話說，函式呼叫時不拷貝給函式某變數或物件之值，而是傳其記憶體位址，告訴函式此資料之所在，相對的函式中需使用指標作為參數來接收傳入的位址(二者間相當於使用等號相連)。

如此一來，函式中的指標參數即定位到輸入變數或物件的記憶體位址，之後於函式內便可用依址取值(deference)的運算來直接存取其值。

例:

```
int addbyone(int* xptr )
{
    (*xptr)++;
    cout<<*xptr<<endl;          //執行後,*xptr=101
    return *xptr;
```

```
}

void main()
{
    int x = 100;
    int y = addbyone(&x);
    cout<<x<<endl;          //函式呼叫後,x 為 101
}
```

程式範例: cpp_ex41.cpp　cpp_ex42.cpp

注意要點:

1. 使用傳位址法可使函式傳回兩個以上的值(或需用傳參考法)。其做法是可透過指標參數，將值存入呼叫時之輸入變數中，一函式可有一個以上的指標參數。
2. 如欲避免函式更改輸入變數之值，或使指標參數定址，可使用 const 關鍵字。

程式範例: cpp_ex43.cpp

9.3　傳參考法(call by reference)

格式:

```
函式定義/原型宣告:    void function(int& x, int& y);   //x, y 為 parameters
函式呼叫:             function(a, b);                  //a, b 為 arguments

呼叫時，函式參數與輸入變數的關係，相當於： int& x = a;   int& y = b;
```

說明:

傳參考法的定義是函式的接收參數(parameter)，其型態為變數或物件的參考，函式呼叫時的引數(argument)，即為參考參數所參照的輸入變數或物件。

使用傳參考法，是將某函式之參數，宣告為參考型態，呼叫時直接輸入某變數或物件，函式之呼叫相當於二者間使用等號相連，如此一來，函式之呼叫方式雖與傳值法相同，但其作用已不是輸入變數或物件的值之拷貝，而是使得函式中

的參數成為輸入變數或物件的參考(別名)，二者均代表相同的一塊記憶體空間，改變函式中參考參數的值，也等於改變了主程式中其所對應的輸入變數或物件的值。

例:

```
int addbyone(int& xref )
{
    xref++;
    cout<<xref<<endl;        //執行後,xref=101
    return xref;
}

void main()
{
    int x = 100;
    int y = addbyone(x);
    cout<<x<<endl;           //函式呼叫後,x 為 101
}
```

程式範例: cpp_ex44.cpp　cpp_ex45.cpp

注意要點:

1. 使用傳參考法與傳位址法有同樣的效果，唯形式有所不同，C++提出傳參考法主要是為了簡化傳位址法中必需要做的取址以及依址取值等運算。
2. 使用傳參考法或傳位址法來做參數傳遞，可節省拷貝傳遞參數之時間及記憶體空間，尤其是當傳遞的參數為多筆資料構成的物件。
3. 如欲避免函式更改輸入變數之值，可使用 const 關鍵字。

程式範例: cpp_ex46.cpp　cpp_ex47.cpp

9.4 案例研究

函式使用傳值法，傳位址法及傳參考法，來嘗試對其之兩個輸入變數的值做對調。

程式範例: cpp_ex48.cpp