Chapter 7 ： 指標(pointer)與參考(reference)

7.1 指標(pointer)

7.1.1 指標的宣告

格式：

資料型態 * 指標名稱; //使用「*」

說明：

對於某資料型態 T，T* 之資料型態為「指到 T 型態資料之指標」，指標如變數亦是代表某儲存資料之空間，不同的是它所能存放的內容是「記憶體位址」，某一資料型態之「變數」或「陣列元素」在記憶體中的位址(記憶體巢編號)，而非程式中欲直接運算之資料。

使用指標可存取其所指的記憶體巢內的資料，此為除了使用變數或陣列元素外，另一種功能強大(但較複雜)的操作資料的方式。

指標與變數或陣列不同的是，指標可以任意改變其所指(參照/對應)之記憶體巢(變數或陣列一旦宣告產生，於其生命週期內，其所對應之記憶體巢固定不變)，故可由單一指標所指位址之自由移動，存取一群資料，如一陣列。

指標的功能有：1.定位資料 2.存取/操作資料 3.機動地改變其所指(參照)之資料，詳見 7.1.2。

就指標的宣告而言，指標變數代表儲存記憶體位址的某記憶體空間(2 bytes)；所指定的資料型態(例：int)則規定了被此指標所指的記憶體巢的空間的大小(4 bytes)，以及存取時如何解讀其所指記憶體巢中資料(binary data->integer)。

例： 同時宣告一個整數變數(a)與一個整數指標(b)

```
int a, *b;          或      int* b, a;
```

宣告二浮點數指標變數:

```
double *x, *y;      或      double* x;   double* y;
```

注意要點：

任何資料型態之指標變數所佔記憶體大小均為 2 bytes。

7.1.2 指標的初值設定及運算

1. **取址運算** -- 將指標「指到(point to)」某一變數 ： 指標所存資料為位址，某變數的位址可由定址運算子(&)得到，之後將此記憶體位址值以指定運算子(=)指定(copy)給指標。變數之型態，需與指標宣告時指定之所指型態相同。

   ```
   指標= &變數;   // or &陣列元素;
   ```

2. **依址取值** -- 使用取值運算子(*)，可存取指標所指記憶體中的資料，對其做指定，讀取，或運算。Note：指標前有「*」，則代表其所指的記憶體巢。

   ```
   *指標;
   ```

3. **算數(改址)運算** --指標可以遞增(++)或遞減(--)，整數(n)也可與指標相加(+或+=)或相減(-或-=)，這些運算的作用是改變指標所指之記憶體位址，於記憶體中前後移動，移動的單位是一個(++/--)或 n 個指標所指之資料型態(T)所佔記憶體空間大小，這些運算通常用於操作一陣列。

   ```
   指標++;
   指標--;
   指標  += n;          *(指標  + n)
   指標  -= n;          *(指標  - n)
   ```

例：

```
int y = 5;
int *yPtr;
yPtr = &y;
cout<<"*yPtr ="<< *yPtr++ <<endl;
```

程式範例： **cpp_ex31.cpp** **cpp_ex32.cpp**

注意要點:

1. 區別依址取值與指標的宣告。(difference : statement 有無資料型態指定)
2. 注意依址取值運算是否有加( )之不同： *ptr++ vs. (*ptr)++
3. 一維陣列的陣列名實為一(定址)指標，其固定指到陣列的第一個元素，不能以運算改變其所指的位址。

程式範例： **cpp_ex33.cpp** 指標陣列

程式範例： **cpp_ex34.cpp** 指標與 const 關鍵字

注意要點：

1. 如 const 設於變數型態之前，則此指標為「唯讀」指標，無法使用其來更改其所指記憶體空間之儲存值。
2. 如 const 設於*之後，則此指標為「定址」指標，其所指的位址固定不可改變。

程式範例： **cpp_ex35.cpp** void 指標

注意要點：

1. void 指標可指向任何型態的變數，一般指標只能指到指定型態之變數。
2. void 指標只有記錄所指資料的起始位址，而無記錄資料的型態(記憶體大小/如何解讀，故對其依址取值時需強制轉換其型別。

5.2 參考(reference)

格式:

```
資料型態 & 參考 = 某變數; //使用「&」
```

說明:

C++除了一般 C 的變數型態及指標外，另提供了參考型態，參考可視為某被參考變數的「別名」，使某記憶體巢有多個名稱/變數代表它。

參考之宣告為在資料型態與欲產生的參考間加上參考運算子(&)，且宣告時必須要同時指定其所參考到的變數，不可單獨存在。

宣告參考並不會為其產生新的記憶體空間，而是指定給被參考變數的記憶體空間，它們都代表/對應到相同的一塊記憶體巢，對參考作處理即是對被參考之變數作處理，反之亦然。

例: 宣告整數變數 a 之參考

```
int a;
int& a_ref = a;
```

程式範例: **cpp_ex36.cpp**

注意要點:

1. 一參考變數所占記憶體大小及其運算方法與其參考到的變數相同。
2. 指標與參考的主要作用，在於函式(function)之參數傳遞。