PREFACE

序

在一些場合和教 C++ 的老師交換意見，也和學生討論有關書上的內容，大概皆有不同的需求，因此基於老師的教學與學生的自習之角度，可否有本可以兼顧這兩者需求的書呢？以期達到兩全其美的目標，這也是本書誕生的主因。

本書以日常生活化的題目來講授 C++ 的主題，讓授課老師以整體概念解說每一主題的核心，如猜猜你的生日或是猜出你心中 1 到 100 的數字來貫穿選擇敘述、迴圈敘述、函式與陣列，以大樂透的電腦選號來說明陣列的建立與應用。同時在每一小節皆附有練習題供自習者練習，並有練習題輔助你是否做對。

本書的重點在物件導向程式設計，所以對封裝、繼承和多型有相當多的著墨，同時也利用範例解說物件導向程式設計可以減少維護成本。除此之外，對函式更加深入，計有預設函式參值、多載函式、樣板函式等等，同時也論及樣板類別的好處及其應用，本書也討論一些標準樣板函式庫的主題使用，讓你可以很快看懂其他的主題，可參考第 15 章。

引以自豪的是每一章的習題皆是精心設計的，可讓授課老師出作業用，也可以讓自習者測試自已對本章的了解狀況及其實力。本書共分兩部分，一為非物件導向部分，這在本書的第 1 章到第 9 章；二是物件導向的主題，這在第 10 章到第 13 章，同時也對異常處理和檔案的輸出與輸入有所探討，這分別是第 14 章和第 16 章。

從目前的 TOIBE 所公佈的受歡迎程式語言排行榜，以目前 2024 年六月來看，C++ 已是第 2 名了，聰明的你，是否也要想想加入學習 C++ 程式語言的行列，相信這本書是你最佳的選擇。祝福你能有一技之長，找工作沒煩惱。

蔡明志
mjtsai168@gmail.com

CHAPTER

3

選擇敘述

C++ 是一種程序性語言（procedure language），表示它是一行接一行敘述執行的，不過有些敘述在某些條件才會執行的，因此，需要選擇敘述（selection statement）來輔助之。

C++ 的選擇述計有 if，if...else，else if，switch...case。以下我們將一一的探討之。在未進入主題前，先來談幾個選擇敘述與下一章的迴圈敘述皆會用到的一些觀念，

3-1 bool 型態

bool 型態（或稱布林型態）表示不是 true 就是 false，除了 0 以外，其餘的數字皆為真，亦即只有 0 的數值是假。C++ 的真，是以 true 表示；而假，是使用 false 表示。注意，都是小寫喔！而 bool 是 Boolean（布林）的縮寫。

範例程式：boolType.cpp

```
01  #include <iostream>
02  using namespace std;
03  int main()
04  {
05      bool yesOrNo = true;
06      cout << "#1: " << yesOrNo << endl;
07
08      yesOrNo = false;
09      cout << "#2: " << yesOrNo << endl;
10
11      yesOrNo = 1;
```

```
12      cout << "#3: " << yesOrNo << endl;
13
14      yesOrNo = 0;
15      cout << "#4: " << yesOrNo << endl;
16
17      yesOrNo = -1;
18      cout << "#5: " << yesOrNo << endl;
19
20      yesOrNo = 21;
21      cout << "#6: " << yesOrNo << endl;
22      return 0;
23  }
```

```
#1: 1
#2: 0
#3: 1
#4: 0
#5: 1
#6: 1
```

從輸出結果發現，true 值是以 1 印出，而 false 則以 0 印出。

3-2 關係運算子

在判斷條件式時，其答案不是 true，就是 false。而這些條件式是以關係運算子（relational operator）和運算元所組成的。

關係運算子又稱比較運算子（comparative operator）。表 3.1 是 C++ 的關係運算子。運算子是一個符號，用以表示某種功能罷了。

表 3.1　C++ 的關係運算子

關係運算子	功能	範例（int x=100;）	結果
<	小於	x < 100	false
<=	小於等於	x <= 100	true
>	大於	x > 100	false
>=	大於等於	x >= 100	true
==	等於	x == 100	true
!=	不等於	x != 100	false

表中要注意的是，等於運算子（==），是由兩個 = 所組成，若只有一個 = ，則是指定或設定運算子。表 3.1 中的範例可以輔助你了解其功能。

練習題

3.1 假設 x 是 1，試問以下的運算式何者為 true？

(a) x > 0　　(d) x >= 0

(b) x < 0　　(e) x != 1

(c) x != 0　　(f) x <= 1

3.2 試問以下片段程式的輸出結果。

```
#include <iostream>
using namespace std;
int main()
{
    bool b = true;
    int i = b;
    cout << b << endl;
    cout << i << endl;

    return 0;
}
```

3-3 if 敘述

當你只想在某一條件為真時，執行其對應的敘述時，而條件為假時，不予以理會，此時將會以 if 敘述為之，其語法如下：

```
if (條件判斷式) {
    條件判斷式為真時，執行的敘述
}
```

注意，條件判斷式要以左、右小括號括起來。其對應的流程圖，如圖 3-1 所示：

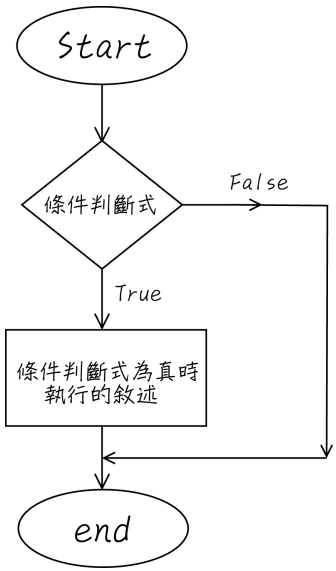

圖 3-1 if 敘述流程圖

我們來看一些範例及其說明。

範例程式：if-1.cpp

```
01  #include <iostream>
02  using namespace std;
03  int main()
04  {
05      int i;
06      cout << "Enter an integer: ";
07      cin >> i;
08      if (i > 0) {
09          cout << i << " 是正整數"<< endl;
10      }
11      cout << "Over\n";
12      return 0;
13  }
40
```

```
Enter an integer: 50
50 是正整數
Over
```

```
Enter an integer: -3
Over
```

此程式的 if 判斷式，若為真時，所要執行的敘述只有一個時，是可以省略左、右大括號，但若執行的敘述有二個或以上時，則必須加上左、右大括號，將其括起來，否則，它只會執行一個敘述而已，如以下的 if-2.cpp 程式。

範例程式：if-2.cpp

```
01  #include <iostream>
02  using namespace std;
03  int main()
04  {
05      int i;
06      cout << "Enter an integer: ";
07      cin >> i;
08      if (i > 0) {
09          cout << i;
10          cout << " 是正整數"<< endl;
11      }
12      cout << "Over\n";
13      return 0;
14  }
```

```
Enter an integer: 50
50 是正整數
Over
```

```
Enter an integer: -3
Over
```

此程式將 if-1.cpp 程式中，if 判斷式若為真時，所執行的敘述以二個敘述來表示，因此要加上左、右大括號。我的處理方法是，不管是執行一個或多個敘述，都加上左、右大括號，這樣子在下次再維護時，可能會加上判斷式為真時的執行敘述，就不會出現錯誤的訊息。

再來看一範例，請你輸入一半徑，若半徑大於 0，則計算它的面積，若小於等於 0，則不予以理會。

範例程式：circleArea.cpp

```
01  include <iostream>
02  using namespace std;
03  int main()
```

```
{
    int radius;
    double area;
    cout << "輸入一半徑: ";
    cin >> radius;
    if (radius > 0) {
        area = M_PI * radius * radius;
        cout << "area = " << area << endl;
    }
    cout << "Over\n";
    return 0;
}
```

```
輸入一半徑: 10
area = 314.159
Over
```

```
輸入一半徑: -10
Over
```

練習題

3.3 試問下一程式，若輸入 100 和 -100 時，其輸出結果分別為何？

```
#include <iostream>
using namespace std;
int main()
{
    int i;
    cout << "Enter an integer: ";
    cin >> i;
    if (i > 0)
        cout << i;
        cout << " 是正整數"<< endl;
    cout << "Over\n";
    return 0;
}
```

3.4 請撰寫一個 if 敘述，使其在 x 大於 0 時，指定數值 1 給 y。

3.5 請撰寫一個 if 敘述，使其在考績（score）大於 85 時，將薪資（salary）提高 2%。薪資初始值為 36000，考績由主管輸入。

3-12 個案討論：猜猜你的生日

先從十進位轉換為二進位、八進位和十六進位說起。將十進位的數值轉換為二進位，只要將它除以 2，取其商，再將此商視為被除數，除以 2，重複上述的動作，直到商小於 2 就結束。每次除以 2 皆會記錄其餘數，如將 100 轉換為二進位，請參見圖 3-5：

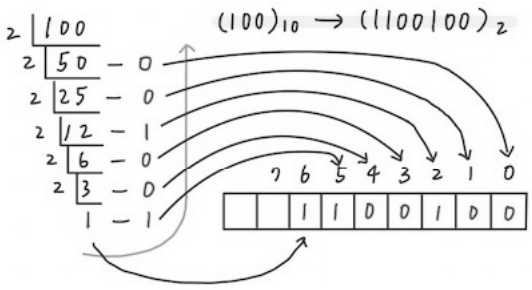

圖 3-5 將十進位 100 轉換爲二進位

最後由下往上寫出其餘數，結果是 $(100)_{10} = (1100100)_2$。這個二進位的資訊告訴我們是以 $64(2^6)$ 加上 $32(2^5)$和 $4(2^2)$的總和。如圖 3-6 所示：

2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	1	0	0	1	0	0

圖 3-6 二進位 1100100 轉換爲十進位

因此，利用上述轉換的方式，將十進位的 30 轉換為二進位，結果是 $(30)_{10} = (11110)_2$，也就是 30 是以 $16(2^4)$ 加上 $8(2^3)$ 加上 $4(2^2)$加上 $2(2^1)$ 的總和。我們可將$(11110)_2$ 看成是五個位元，由右至左編號是從 0 開始，我們將它看成是第一個位元，那麼 30 會出現在第二個、第三個、第四個和第五個的位元上。將這些位元所代表的數字加總就是這個數字 30。

一個人的生日是何日，其實只要將 1~31 共 31 個數字以二進位表示，以五個位元就夠了，因為 2^5 是 32 了。這些數字只要指定某些位元為 1 就可以。如圖 3-7 所示：

6-6 防止更改函式的陣列參數

傳遞陣列只是傳遞陣列在記憶體的起始位址。陣列的元素不會被複製。這可節省記憶體空間。然而使用陣列參數，若在函式中意外的改變陣列的元素，將會導致錯誤的結果。為了防止這項錯誤的發生，您可以在陣列參數前加上關鍵字 const，告訴編譯器此陣列不可以更改。若在函式中試圖要更改陣列的元素，編譯器將會有錯誤訊息發生。

在範例程式 arrayAsParameter.cpp 中，若將下一敘述中

```
void newArray(int arr2[], int n);
```

的參數改為

```
void newArray(const int arr2[], int n);
```

多加了 const 的話，表示 arr2 陣列元素不可更改，而在 newArray() 函式中有更改 arr2 陣列元素的值，因此會產生錯誤的訊息，

6-7 從函式回傳陣列

當函式回傳陣列時，回傳的是陣列的位址。您可以宣告一函式回傳基本型態值或是物件。例如：

```
int total(const int arr[], int size);
```

您能使用相同的語法，從函式回傳一陣列嗎？如嘗試要宣告一函式，用以回傳一新陣列，此新陣列是另一陣列的反轉，如下所示：

```
int[] transpose(const int arr[], int size);
```

這在 C++ 是不允許的。然而您可以在函式中，傳送兩個陣列當作引數克服此問題：

```
void reverse(const int arr[], int reverseArr[], int size);
```

6-8 陣列元素的排序

此小節將以隨機亂數產生大樂透號碼，然後再以氣泡排序法由小至大排序之。

6-8-1 大樂透號碼

以下是利用隨機亂數產生六個介於 1 到 49 之間的數字，這也就是大樂透號碼。程式如下所示：

範例程式：lottoNum-1.cpp

```
01  #include <iostream>
02  #include <iomanip>
03  using namespace std;
04
05  int main()
06  {
07      int lotto[6];
08      srand((unsigned)time(NULL));
09      for (int i=0; i<6; i++) {
10          lotto[i] = rand() % 49 + 1;
11          cout << setw(3) << lotto[i];
12      }
13      cout << endl;
14      return 0;
15  }
```

```
 4 49 36 47 18 33
```

```
 4 18  9 47 29 18
```

因為此程式利用時間來做隨機。在第二個輸出結果，我們發現有重複的數字 18。為了不出現重複的數字，可以先檢視產生的樂透號碼是否先前已產生過，若此號碼已經產生，則不予以採用，然後再重新產生一個號碼，如範例程式 lottoNum-2.cpp 所示：

範例程式：lottoNum-2.cpp

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

int main()
{
    int lotto[6];
    srand((unsigned)time(NULL));
    lotto[0] = rand() % 49 + 1;
    int i=1, flag=0;
    while (i<6) {
        int lottoNum = rand() % 49 + 1;
        for (int j=0; j<i; j++) {
            if (lotto[j] == lottoNum) {
                flag = 1;
                break;
            }
        }
        if (flag == 0) {
            lotto[i] = lottoNum;
            i++;
        }
        else {
            flag = 0;
        }
    }

    for (int k=0; k<6; k++) {
        cout << setw(3) << lotto[k];
    }
    cout << endl;
    return 0;
}
```

```
 4 34 24 37 16 25
```

```
int main()
{
    int arr[] = {11, 23, 15, 47, 39};
    int size = sizeof(arr)/sizeof(int);
    for (int i=0; i<size; i++) {
        cout << setw(4) << arr[i];
    }
    cout << endl;

    int big = greatest(arr, size);
    cout << "最大值是 " << big << endl;
    return 0;
}

int greatest(int arr2[], int n)
{
    int max = arr2[0];
    for (int i=1; i<n; i++) {
        if (arr2[i] > max) {
            max = arr2[i];
        }
    }
    return max;
}
```

6-13 習題

1. 請撰寫一程式，讀取 10 個浮點數，接著以相反的順序顯示其所讀取的數字。
2. 請撰寫一程式，讀取學生分數（score），找到最高分（highest），接著根據以下規則給予成績等級：

   若 scores 大於等於 highest – 10，則等級（Grade）為 A

   若 scores 大於等於 highest – 20，則等級（Grade）為 B

   若 scores 大於等於 highest – 30，則等級（Grade）為 C

   若 scores 大於等於 highest – 40，則等級（Grade）為 D

其他，等級為 F

此程式提示使用者輸入學生人數，接著輸入所有分數，最後顯示等級。以下為程式的執行結果：

```
請輸入學生人數: 6
輸入 6 位學生的分數: 90 89 76 56 57 49
學生 1  分數： 90 ，grade 是 A
學生 2  分數： 89 ，grade 是 A
學生 3  分數： 76 ，grade 是 B
學生 4  分數： 56 ，grade 是 D
學生 5  分數： 57 ，grade 是 D
學生 6  分數： 49 ，grade 是 F
```

3. 試撰寫一程式，自動產生 1~100 二進位的數字，輸出時每五個數字印一列。

4. 請撰寫一程式，讀取未指定個數的成績資料，判斷有多少分數高於或等於平均值，有多少分數低於平均值。以輸入負數值表示輸入資訊的結尾。假設最多可輸入 50 筆分數資料。

5. 請撰寫一程式，讀取十個數字並顯示不同的數字（若一數字出現多次時，則只顯示出一次）。（提示：讀取一數字，若是新的，則將它儲存於陣列中，若數字已經在陣列中，則將它捨掉。輸入完之後，陣列只包含不同的數字。）以下為程式的執行結果：

```
輸入十個整數: 1 3 5 2 3 5 2 7 6 8
不同的數字為: 1 3 5 2 7 6 8
```

6. 請撰寫一程式，產生 100 個介於 0 到 9 的隨機整數，接著顯示各數字的個數。（提示：使用 rand() % 10 來產生 0 到 9 之間的隨機整數。利用有十個整數的陣列名為 counts，來儲存各數字 0、1、…、9 的個數。）

7. 請撰寫一個函式，輸出整數陣列裡的最大元素對與其對應的索引、最小元素與其對應的索引，請使用以下函式標頭：

```
void minMaxIndex(int arr2[], int size);
```

再撰寫一測試程式，提示使用者輸入 10 個數字，接著呼叫此函式印出其結果。