CHAPTER 04

選擇結構

本章學習重點

- 程式流程控制
- 關係運算子與邏輯運算子
- if 敘述
- switch – case 敘述
- APCS 實作題 – 選擇結構

本章學習範例

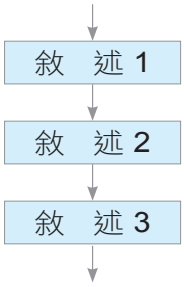
- 範例 4.2.1　關係運算子 (d068)
- 範例 4.2.2　0 與 1 (d063)
- 範例 4.3.1　成績判斷
- 範例 4.3.1-2　偶數個數
- 範例 4.3.2　奇偶數 (d064)
- 範例 4.3.2-2　大寫字母判斷
- 範例 4.3.2-3　打折問題
- 範例 4.3.2-4　三角形面積 (d489)
- 範例 4.3.3　三數最大者 (d065)
- 範例 4.3.3-2　計分程式 (a053)
- 範例 4.3.3-3　閏年判斷 (a004)
- 範例 4.4　二元五則運算
- 範例 4.4-2　等第判斷
- 範例 4.4-3　月份轉季節
- 範例 4.5.1　籃球賽 (201906 APCS 第 1 題)
- 範例 4.5.2　邏輯運算子 (201710 APCS 第 1 題)
- 範例 4.5.3　三角形辨別 (201610 APCS 第 1 題)

4.1 程式流程控制

程式通常會依照敘述的順序，從第一行、第二行、直到最後一行，一步一步循序地執行。但很少程式會如此簡單，程式常需要根據條件判斷，選擇執行不同的程式碼，或重複執行相同的程式碼。這種選擇程式分支和決定敘述執行的順序，就是程式流程控制。

為了使程式容易閱讀、除錯、及維護，最好使用結構化程式設計，也就是只使用下列三種結構設計程式，避免使用跳躍結構 goto。

1. 循序（sequence）結構
   如下圖，按照敘述出現的順序，一步一步循序地執行



2. 選擇（selection）結構
   根據條件判斷的結果，選擇執行不同的程式碼。如 if、if - else、switch 等指令。

3. 重複（repetition）結構
   重複執行某些程式碼，直到滿足特定條件，如 for、while、do while 等指令。

4.2 關係運算子與邏輯運算子

4.2.1 關係運算子（==, !=, >, <, >=, <=）

選擇結構會根據條件判斷的結果，選擇不同的程式碼執行。重複結構也會根據條件判斷的結果，決定是否繼續執行相同的程式碼。條件判斷常會用到關係運算式或邏輯運算式。

關係運算是用來比較兩個運算式的大小關係，運算結果是布林值，不是 1（true），就是 0（false）。關係運算子如下，其運算順序是由左至右。

運算子	數學式	意義	運算式	結果
==	=	等於	(2 == 1)	0
!=	≠	不等於	(2 != 1)	1
>	>	大於	(2 > 1)	1
<	<	小於	(2 < 1)	0
>=	≥	大於等於	(2 >= 1)	1
<=	≤	小於等於	(2 <= 1)	0

1. 關係運算子的優先權低於算術運算子。如下例中，若 a = 2, b = 3, c = 6

```
(a == 5)        ——— a = 2 不等於 5，所以為 false
(a * b >= c)    ——— (2 * 3 >= 6)，(6 >= 6)，所以為 true
(b + 4 > a * c) ——— (3 + 4 > 2 * 6)，(7 > 12)，所以為 false
```

2. 關係運算子中間，不可以留有空白，次序也不可以更換。例如：

```
0 == 0      0 = = 0      5 >= 3      5 => 3
          錯誤，中間不可以留有空白    錯誤，>= 的次序不可更換
```

3. 指定運算子 = 和關係運算子 == 是不同的。

=	指定運算子	將右邊運算式的值指定給左邊的變數
==	關係運算子	比較兩個運算元的值是否相等

範例 4.2.1 關係運算子 (d068)

老師宣布，學期成績 59 分者，一律以 60 分計，不用補考。寫一程式，使用關係運算子，將 59 分自動加 1 分，其餘成績不變。

輸入：學生成績

輸出：處理後的成績

解題方法

若成績變數為 s，成績 59 分的條件是 (s == 59)，條件成立時，關係式會得 1，不成立則會得 0，所以可直接輸出 s + (s == 59)。

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int s;
7      cout << "學生成績 ";
8      cin >> s;
9
10     cout << "處理後的成績 " << s + (s == 59) << endl;
11     return 0;
12 }
```

若 s 等於 59，(s == 59) 會是 1，輸出 59+1。
若 s 不等於 59，(s == 59) 會是 0，輸出 s+0

執行結果

```
學生成績 59
處理後的成績 60

學生成績 65
處理後的成績 65
```

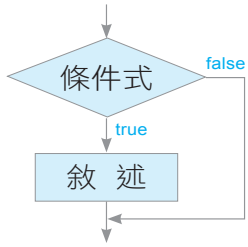
4.3 if 敘述

4.3.1 if 指令

if 的語法與流程圖如下，如果條件式為 true，才執行敘述，否則執行 if 敘述外的下一行敘述。if 屬於單向選擇結構。

```
if （條件式）
    敘述；
```

```
if （條件式）
{
    敘述 1;
    敘述 2;
    ……;
}
```



if 內有多個敘述要執行時，應使用 { } 將敘述括起來。如下例，若 a = 0，會輸出 "1 是正數 "，因為 if 內的敘述只有一個，所以程式應該改寫成下方第二個例子，將兩個敘述使用 { } 括起來。

```
if (a == 1)
    cout << " 輸入 1" ;          ← 屬於 if 內的敘述
    cout << "1 是正數 ";         ← 不屬於 if 內的敘述
```

```
if (a == 1)
{
    cout << " 輸入 1" ;
    cout << "1 是正數 ";
}                                ← 屬於 if 內的敘述
```

非 0 的數值，其布林值是 true，所以以下敘述是相同的。

```
if （運算式）
```

```
if （運算式 != 0）
```

範例 4.3.1 成績判斷

輸入成績，如果及格，輸出「恭喜過關！」。

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int score;
7      cout << " 輸入成績 ";
8      cin >> score;
9
10     if (score >= 60)
11         cout << " 恭喜過關 !" << endl;
12
13     return 0;
14 }
```

若輸入的成績 score 大於等於 60，會輸出 " 恭喜過關 ! "。

執行結果

```
輸入成績 75
恭喜過關 !
```

動動腦

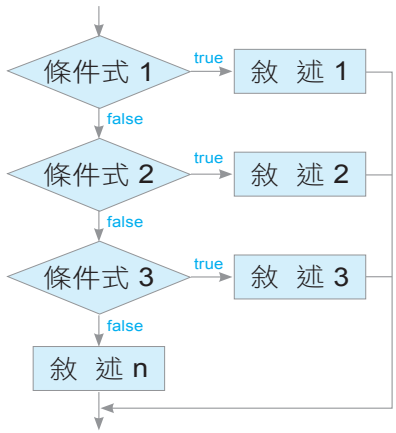
上例中，如果 50 ≤ 成績 <60，輸出「再努力一下」。以下程式碼的空格內應填入甚麼？

```
if (_____________)
   cout << " 再努力一下 " << endl;
```

4.3.3 if - else if 敘述

有多種條件式要判斷時，可使用 if - else if，其語法與流程圖如下。若條件式 1 成立，則執行敘述 1，不再往下繼續判斷其他條件式；若條件式 1 不成立，則繼續判斷條件式 2，依此類推，當所有條件式都不成立時，執行 else 內的敘述 n。此結構只會執行其中一個敘述。最後一個 else 後不需有 if 條件式。

```
if (條件式 1)
    敘述 1;
else if (條件式 2)
    // 不滿足條件式 1，但符合條件式 2
    敘述 2;
else if (條件式 3)
    // 不滿足條件式 1 及 2，但符合條件式 3
    敘述 3;
else
    // 不滿足條件式 1, 2 及 3
    敘述 n;
```

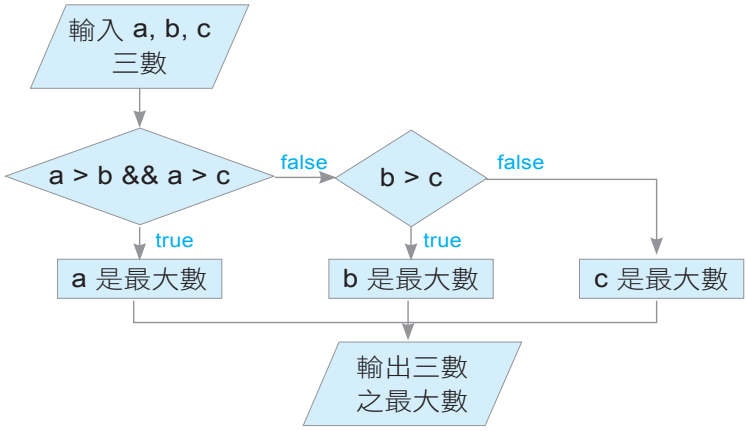


範例 4.3.3 三數最大者 (d065)

輸入三個整數，找出三數中最大者與最小者。

解題方法

1. 若三個整數為 a, b, c，先判斷 a 是不是最大數，若 a > b 且 a > c，則 a 是最大數。
2. 若 a 不是最大數，則判斷 b 是不是最大數，若 b > c，則 b 是最大數。
3. 若 b 也不是最大數，c 就是最大數了。
4. 解題流程圖



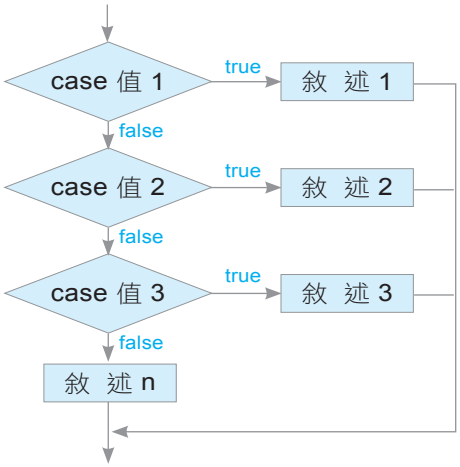

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5     int a, b, c, max, min;
6     cout << " 輸入三個整數 ";
7     cin >> a >> b >> c;
8     if (a > b && a > c)
9        max = a;
10    else if (b > c)
11       max = b;
12    else
13       max = c;
14    if (a < b && a < c)
15       min = a;
16    else if (b < c)
17       min = b;
18    else
19       min = c;
20    cout << a << ", " << b << ", " << c << " 三數之最大數為 "
21         << max << " 最小數為 " << min << endl;;
22    return 0;
23 }
```

若 a > b 且 a > c，則 a 是最大數，否則 a 不是最大數。
否則比較 b 和 c，若 b > c，則 b 是最大數。
否則 a, b 都不是最大數，c 就是最大數。

4.4 switch – case 敘述

switch 語法與流程圖如下

```
switch （運算式） {
    case 值 1 : 敘述 1;
       break;
    case 值 2 : 敘述 2;
       break;
    ......
    case 值 n : 敘述 n;
       break;
    default : 敘述 ;
}
```



需要判斷多個條件時，switch 比 if 簡單明瞭，其使用方式如下：

1. 運算式和 case 的值都必須是整數或字元，不能是浮點數，浮點數必須使用 if 敘述。
2. 運算式的值會依序和每個 case 的值比較，如果相等，則執行此 case 後的敘述，直到碰到 break 指令。
3. break 可讓程式跳離 switch 區塊，但 break 可被省略。若省略 break，程式不會跳離 switch，會繼續往下執行其他 case 子句。
4. 若運算式的值與全部 case 的值都不相等，則會執行 default 後的敘述。default 敘述也是可以被省略的。
5. case 子句若包含多個敘述，不必用大括號括起來。

範例 4.4 二元五則運算

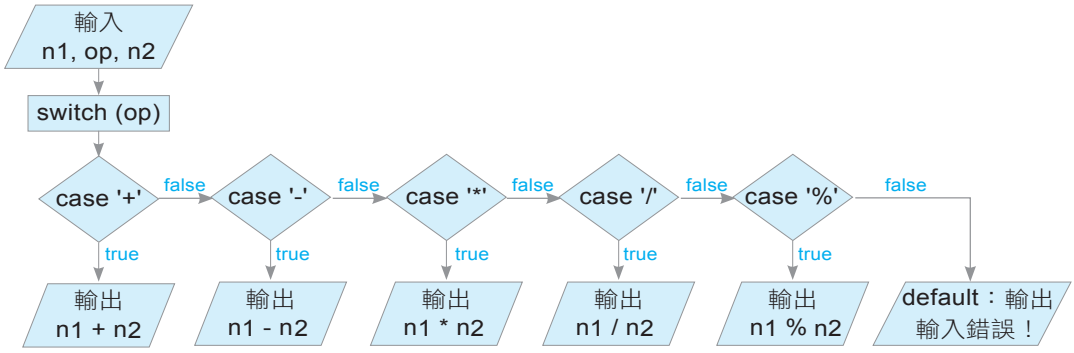
寫一程式，能進行加、減、乘、求商、求餘數的運算。

輸入：依序輸入第 1 個整數、運算子（+ - * / % 其中一個）、第 2 個整數

輸出：運算結果

解題方法

解題流程圖




```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5     int n1, n2;
6     char op;
7     cout << "依序輸入第 1 個數、運算子、第 2 個數 ";
8     cin >> n1 >> op >> n2;
9     switch (op)
10    {
11       case '+': cout << n1 + n2 << endl;
12          break;
13       case '-': cout << n1 - n2 << endl;
14          break;
```

判斷輸入的是那一個運算子，依不同運算子，進行不同的運算

```
15          case '*': cout << n1 * n2 << endl;
16             break;
17          case '/': cout << n1 / n2 << endl;
18             break;
19          case '%': cout << n1 % n2 << endl;
20             break;
21          default: cout << "輸入錯誤！" << endl;
22       }
23       return 0;
24    }
```

若輸入的運算子不在範圍內，會輸出 "輸入錯誤！"

執行結果

```
依序輸入第 1 個數、運算子、第 2 個數 57 % 12
9

依序輸入第 1 個數、運算子、第 2 個數 98 / 16
6
```

比對一個數值範圍，可使用下列方式

```
switch (score) {
   case 90 ... 100 :
      敘述 1; break;
   case 80 ... 89 :
      敘述 2; break;
   .........
}
```

範圍只能用三個點 ... 表示，且數值與 ... 之間一定要有空白，否則編譯器會認為 . 是小數點，編譯時會產生錯誤。

4.5 APCS 實作題 - 選擇結構

範例 4.5.1 籃球賽 (201906 APCS 第 1 題)

APCS 舉辦籃球賽，每場都有主隊與客隊。寫一程式，讀入兩場籃賽的四節分數，自動產生比賽結果。

輸入：共 4 行，每行有 4 個數字。第 1, 2 行分別代表主隊與客隊第一場比賽四節的得分，第 3, 4 行則是第二場比賽四節的得分，所有得分都介於 0 ~ 100。

輸出：共 3 行，第 1, 2 行以「主隊總分 : 客隊總分」的方式，輸出兩場比賽結果。若主隊贏兩場，第 3 行輸出 Win，平手輸出 Tie，客隊贏兩場輸出 Lose。每場一定要分出勝負，不會有同分的形情。

範例一：輸入
22 16 23 22
18 16 25 20
20 18 25 20
20 22 24 20

範例一：正確輸出
83:79
83:86
Tie

範例二：輸入
10 20 20 10
14 13 14 20
21 20 25 12
20 22 23 16

範例二：正確輸出
60:61
78:81
Lose

解題方法

輸入資料共四行，每行有四節得分，所以可以一次輸入四個整數，加總後，再指定給所代表的變數。解題演算法如下

(1) 讀入第 1 行整數，加總，指定給主隊第 1 場得分 h1。
(2) 讀入第 2 行整數，加總，指定給客隊第 1 場得分 a1。
(3) 讀入第 3 行整數，加總，指定給主隊第 2 場得分 h2。
(4) 讀入第 4 行整數，加總，指定給客隊第 2 場得分 a2。
(5) 若 h1 > a1，主隊勝場 w 加 1。

(6) 若 h2 > a2，主隊勝場 w 加 1。

(7) 輸出兩場比賽的比數，也就是 h1 : a1 和 h2 : a2。

(8) 若 w == 2，表示主隊贏兩場，輸出 Win，否則若 w == 1，表示主客平手，輸出 Tie，否則就是客隊贏兩場，輸出 Lose。所以可使用 if - else if 結構來撰寫。

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

int main()
{
    int a, b, c, d;
    int h1, a1, h2, a2, w = 0;

    cin >> a >> b >> c >> d;
    h1 = a + b + c + d;
    cin >> a >> b >> c >> d;
    a1 = a + b + c + d;
    cin >> a >> b >> c >> d;
    h2 = a + b + c + d;
    cin >> a >> b >> c >> d;
    a2 = a + b + c + d;

    if (h1 > a1)
        w++;
    if (h2 > a2)
       w++;

    cout << h1 << ":" << a1 << endl;
    cout << h2 << ":" << a2 << endl;
```

h1, a1 為第 1 場主隊與客隊得分，h2, a2 為第 2 場得分。w 為主隊勝場數，預設為 0

第 1 行 4 個分數加總是主隊第 1 場得分

第 2 行 4 個分數加總是客隊第 1 場得分

第 3 行 4 個分數加總是主隊第 2 場得分

第 4 行 4 個分數加總是客隊第 2 場得分

若第 1 場主隊獲勝，勝場 w 就加 1

輸出第 1, 2 場比分

```
25
26      if (w == 2)
27          cout << "Win";
28      else if (w == 1)
29          cout << "Tie";
30      else
31          cout << "Lose";
32
33      return 0;
34  }
```

執行結果

```
27 27 22 27                 17 20 32 24
22 17 27 24                 19 23 16 22
16 23 35 20                 24 25 24 25
27 24 24 29                 22 24 11 18
103:90                      93:80
94:104                      98:75
Tie                         Win
```

範例 4.5.2 邏輯運算子 (201710 APCS 第 1 題)

a, b 兩數之 AND, OR, XOR 運算結果如下：

a AND b

	b 為 0	b 不為 0
a 為 0	0	0
a 不為 0	0	1

a OR b

	b 為 0	b 不為 0
a 為 0	0	1
a 不為 0	1	1

a XOR b

	b 為 0	b 不為 0
a 為 0	0	1
a 不為 0	1	0

例如：(1) 0 AND 0 結果為 0，0 OR 0 及 0 XOR 0 結果也為 0。

(2) 0 AND 3 結果為 0，0 OR 3 及 0 XOR 3 結果則為 1。

(3) 4 AND 9 結果為 1，4 OR 9 結果為 1，4 XOR 9 結果為 0。

寫一個程式，輸入 a, b 及運算結果，輸出可能的 AND, OR, XOR 運算子。

輸入：三個以空白隔開的非負整數，前兩數為 a, b 之值，第三數為運算結果，只會是 0 或 1。

輸出：輸出可能得到指定結果的運算，若有多個，輸出順序為 AND, OR, XOR，每個可能的運算單獨一行，若不可能得到指定結果，輸出 IMPOSSIBLE。

範例一：輸入	範例二：輸入	範例三：輸入
0 0 0	1 1 1	0 0 1
範例一：正確輸出	**範例二：正確輸出**	**範例三：正確輸出**
AND OR XOR	AND OR	IMPOSSIBLE

解題方法

1. AND 的位元運算子是 &，OR 是 |，XOR 是 ^ (p3-16)。要進行運算，a, b 要先轉換成 0 與 1 的布林值。解題演算法如下

(1) 輸入三數 a, b, r，r 為運算結果 (result)。

(2) 將 a, b 要先轉換成 0 與 1。

(3) 依序比較 a & b, a | b, a ^ b 的運算結果與 r 是否相等，若相等，輸出對應的運算子。

(4) 宣告一個變數 found 作為旗幟，預設為 0，表示未找到符合的運算。步驟 (3) 若找符合的運算，就改變旗幟的值為 1。

(5) 最後再檢查此旗幟是否為 0，若是，表示未找到符合的運算，所以不可能得到指定的結果。

學習挑戰

一、選擇題

1. (　　) 下列何者不是結構化程式設計的指令？

   (A) goto　(B) if　(C) while　(D) for

2. (　　) 若 a = 3, b = 3, c = 6，下列那一個不是正確的關係運算式？

   (A) a * b >= c　(B) a = b　(C) b + 4 < a * c　(D) a <= b

3. (　　) 若 ! (x1 || x2) 為 true，則 x1 與 x2 的值應為何？

   (A) x1 為 false，x2 為 false　(B) x1 為 true，x2 為 true
   (C) x1 為 true，x2 為 false　(D) x1 為 false，x2 為 true

4. (　　) 執行下列敘述，那一個 g 值會輸出 good! ？

   ```
   if (g >= 90) cout << "good ! ";
   ```

   (A) 90　(B) 80　(C) 70　(D) 60

5. (　　) 若 a = 5, b = 5, c = 6，運算式 (a == b) && (b <= c) 的結果為

   (A) 5　(B) 6　(C) 1　(D) 0

6. (　　) 若 a = 5, b = 4，執行下列敘述後，d 值為何？

   ```
   a > b ? d = a : d = b;
   ```

   (A) 5　(B) 4　(C) 1　(D) 0

7. (　　) 若 x = 5, y = 4, z = 3，執行下列敘述後，z 值為何？

   ```
   if (x >= y)
       z = x - y;
   else
       z = y - x;
   ```

   (A) 5　(B) -1　(C) 1　(D) 3

二、應用題

1. 若 i = 1, j = 2, k = 3, m = 4，請寫出下列敘述輸出的結果。

   (1) cout << (j == 2);

   (2) cout << (i >= 1 && j < 4);

   (3) cout << (j >= i || k == m);

   (4) cout << (! (k – m));

   (5) cout << (k + m < j || 3 - j >= k);

2. 若 x,y,z 為布林變數，x = y = true, z = false。下列各運算式的值為何？

   (1) !(y || z) || x

   (2) !y || (z || !x)

   (3) z || (x && (y || z))

   (4) (x || x) && z

3. 請寫出下列程式在各種 n 值時，輸出的結果。

```
switch (n / 4) {
   case 1:
   case 2: n = 2 * n; break;
   case 3: n += 5;
   default: n++;
}
cout << n << endl;
```

   (1) n = 1　　(2) n = 4　　(3) n = 6　　(4) n = 8　　(5) n = 13

4. 請將下列程式使用 if - else 改寫。

```
switch (x) {
  case 10: y = 'a'; break;
  case 20:
  case 30: y = 'b'; break;
  default: y = 'c';
}
```

5. 學校到校時間是 7:50，16:30 才能離校，寫一程式可以輸入時間，判斷現在是不是在校時間。輸入資料為兩個正整數，分別代表小時與分鐘，例如：16:50 輸入 16 50。