範例 2.4.2-2 小寫轉大寫

輸入一個小寫字母，將它以大寫顯示出來。

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      char ch;
7      cout << " 輸入一個小寫字母 ";
8      cin >> ch;
9
10     cout << " 此字母的大寫為 " << ch - 32 << endl;
11
12     return 0;
13 }
```

因為大寫 ch = 小寫 ch – 32

執行結果

```
輸入一個小寫字母 f
此字母的大寫為 F
```

另一類特殊字元是以反斜線 \ 開頭的跳脫字元（escape character），如下表。跳脫字元不會在螢幕上顯示，而是會被轉譯成另外的功能。例如：\n 不會顯示 \n，而是會被轉換成換行。跳脫就是指跳離原字元的意思。

跳脫字元也可使用 ASCII 碼表示，例如：移到定位點的字元 \t 可用 ASCII 碼 9 來表示。若要輸出 ?、\、'、" 等字元，需在字元前再加一個反斜線 \，例如：\?、\\、\'、\"。

執行結果

```
輸入三個整數 8 6 2
8, 6, 2 三數之最大數為 8 最小數為 2
```

範例 4.3.3-2 計分程式 (a053)

某次考試老師依答對題數，訂定給分的規則如下：

1. 1~10 題，每題 6 分。
2. 11~20 題，每題 2 分，前 10 題每題還是 6 分。
3. 21~40 題，每題 1 分。
4. 40 題以上，一律 100 分。

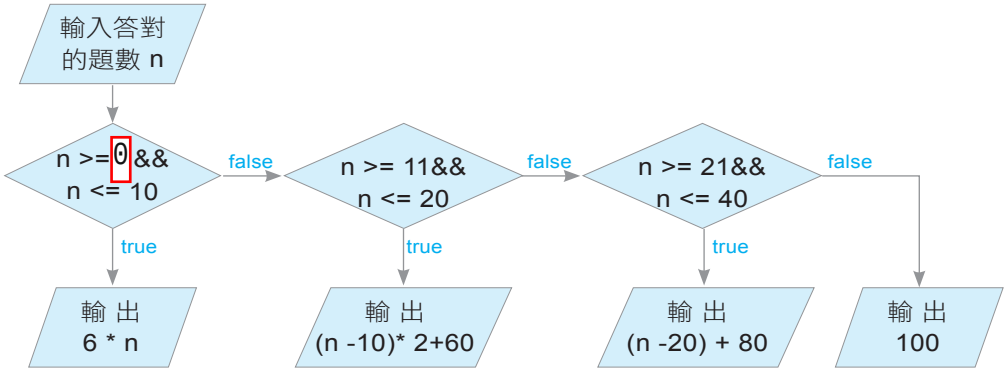
寫一程式，能依此規則，協助老師計分。

輸入：答對題數

輸出：得分

解題方法

1. 此題有多個條件式，可使用 if - else - if 結構。
2. 若答對題數為 n，答對 0~10 題的判斷式為 (n >= 0 && n <= 10)，可得 6 * n 分。
3. 答對 11~20 題 (n >= 11 && n <= 20)，前 10 題得 10 * 6 = 60 分，超過的題數 n - 10，可得 (n - 10) * 2，共得 60 + (n – 10) * 2 分。
4. 答對超過 21~40 題，前 20 題得 10 * 6 + 10 * 2 = 80 分，超過的題數 n - 20，可得 (n - 20) 分，共 80 + (n – 20) 分。
5. 解題流程圖




```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5     int n;
6     cout << " 輸入答對題數 ";
7     cin >> n;
8     cout << " 得分 ";
9     if (n >= 0 && n <= 10)
10       cout << n * 6;                    答對 0~10 題，得 6 * n 分
11    else if (n >= 11 && n <= 20)
12       cout << (n - 10)*2 + 60;          答對 11~20 題，共得 (n-10)*2+60 分
13    else if (n >= 21 && n <= 40)
14       cout << (n - 20) + 80;            答對 21~40 題，共得 (n-20)+80 分
15    else
16       cout << 100;                      答對超過 40 題，得 100 分
17    return 0;
18 }
```

執行結果

```
輸入答對題數 35
得分 95
```

範例 4.4-2 等第判斷

將成績轉換成對應的等第，轉換規則如下：

優：90 分 (含) 至 100 分　　甲：80 分 (含) 以上，未滿 90 分

乙：70 分 (含) 以上，未滿 80 分　　丙：60 分 (含) 以上，未滿 70 分

丁：未滿 60 分

輸入：某項成績

輸出：對應的等第

解題方法

1. 共五種等第，可以使用 switch 解題（或使用 if – else if）。

2. 等第判斷之值為一數值範圍，所以可以使用比對一個數值範圍的方式。

```
#include <iostream>
using namespace std;
int main()
{
    int score;
    cout << "輸入成績 ";
    cin >> score;
    switch (score) {
        case 90 ... 100:
            cout << "優" << endl;
            break;
        case 80 ... 89 :
            cout << "甲" << endl;
            break;
        case 70 ... 79 :
            cout << "乙" << endl;
```

```
18            break;
19         case 60 ... 69 :
.             cout << "丙" << endl;
.             break;
.          default :
              cout << "丁" << endl;
       }
       return 0;
26  }
```

執行結果

```
輸入成績 58
丁
```

每一個 case 的值必須不同。多個 case 的值可以連接在一起的，如下例，字元 letter 的值為 'a' 或 'A' 時，都會執行敘述 1。

```
switch (letter) {
   case 'a' :
   case 'A' :
      敘述 1; break;
   ..........
}
```

解題方法

1. 若某一整數為 n，a, b, c 三數都介於 1~n，要找出所有畢氏三元數，需一一檢查 a, b, c 三數的各種組合 (1, 1, 1), (1, 1, 2).....(1, 1, n), (1, 2, 1) ..... (1, 2, n).....(n, n, n)，是否符合畢氏三元數。

2. 由步驟 1 可以發現，本題可使用三重 for 迴圈解題。

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5     int n, a, b, c, total = 0;
6     cout << "輸入一正整數 ";
7     cin >> n;
8     for (a = 1; a <= n; a++)
9        for (b = 1; b <= n; b++)
10          for (c = 1; c <= n; c++)
11             if (a * a + b * b == c * c)
12                if (a < b && b < c) {
13                   cout << a << ", " << b << ", " << c << endl;
14                   total++;
15                }
16    cout << "小於等於 " << n << " 的畢氏三元數共有以上 "
17         << total << " 組" << endl;
18
19    return 0;
20 }
```

n 是輸入的整數值，a, b, c 是迴圈索引，total 是畢氏三元數的組數

三重迴圈

a, b, c 三邊要滿足畢氏定理

三邊由小而大排列

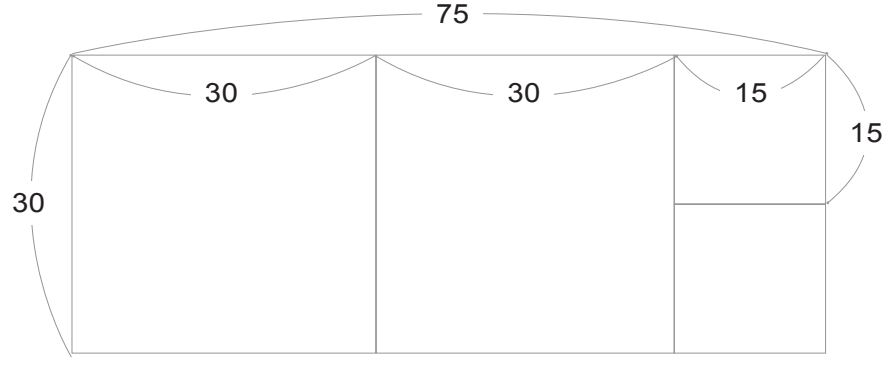
2. 例如：找出 75 與 30 之最大公因數的步驟如下：

(1) 繪出一個 75×30 的矩形。

(2) 以短邊 30 為邊長，切割出兩個 30×30 的正方形，矩形變成 15×30。

(3) 再以短邊 15 為邊長，切割出兩個 15×15 的正方形。

(4) 15×15 是最小的正方形，所以 15 就是 75 與 30 的最大公因數。

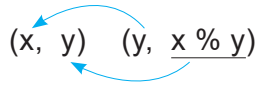


3. 觀察另一個例子，找出 58 和 40 最大公因數的步驟：

x		y		x % y
58	%	40	=	18
40	%	18	=	4
18	%	4	=	2
4	%	2	=	0
2	.....	最大公因數		

(1) 若 x % y == 0，則 y 是最大公因數。

(2) 若 x % y != 0，則將 (x, y) 轉換成 (y, x % y)。



將敘述寫成 x = y; y = x % y; 是錯的，因為先執行 x = y，所以 x 的值已被變為 y，因此 y = x % y = y % y = 0，y 值永遠為 0。

```
1  #include <iostream>
2  #include <iomanip>
3  using namespace std;
4  int main()
5  {
6     float height, total = 0;
7     cout << "輸入初始高度 (m) ";
8     cin >> height;
9     total = height;
10    if (height >= 0.5)
11       do
12       {
13          height /= 2;
14          total += height;
15       } while (height >= 0.5);
16    cout << "最後高度為 " << fixed << setprecision(2)
17         << total << " m" << endl;
18
19    return 0;
20 }
```

使用 setprecision 時，需先引入標頭檔 <iomanip>

height 為每次長高的高度，total 為總高度

將總高度設為初始高度

高度 >= 0.5，才執行迴圈

生長的高度為前一天長高之高度的 1/2

總高度 = 原來的高度 + 長高的高度

重複執行迴圈，直到高度 >= 0.5 不成立

執行結果

```
輸入初始高度 (m) 25.94
最後高度為 51.47 m
```

程式說明

- 第 16 行

  fixed << setprecision(2) 是採四捨五入取至小數點以下第 2 位。

範例 6.1.4-3 陣列記錄問題

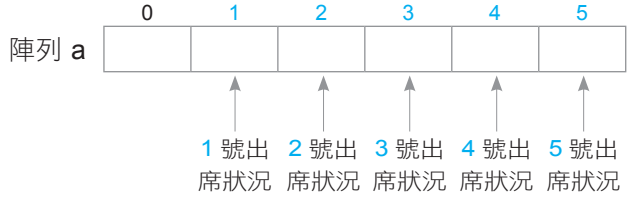
上電腦課時，學生陸續進入電腦教室，小老師記錄每位進教室同學的座號，寫一程式，協助小老師找出缺席同學的座號。

輸入：一串正整數，第一個數字為班級人數，第二個數字為實到人數，後面接續這些實到同學的座號。例如：5 3 4 1 2，表示全班 5 人，實到 3 人，分別是 4 號、1 號、2 號。

輸出：缺席同學的座號，例如：3 5 。

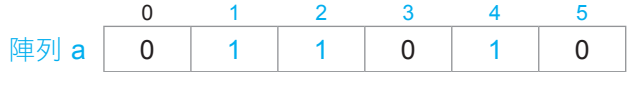
解題方法

1. 宣告一個陣列 a，用來記錄每位同學出席的情形，索引代表學生的座號，若某位學生出席，便以該生的座號為索引，將對應的元素值設為 1，未出席的陣列元素則設為 0。



2. 將陣列 a 每個元素的初始值設為 0，表示皆尚未出席。

3. 將出席學生對應的元素值設為 1，例如：4, 1, 2 號出席，將 a[4], a[1] 及 a[2] 設為 1，依此類推。



4. 要找出缺席同學的座號時，只要一一檢查陣列的每個元素即可，若元素值不等於 1，便輸出缺席同學的座號。

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5     int a[] = {1, 2, 3, 4, 5, 6};
6     int i, n = 6;

7     for (i = 0; i <= n - 2; i++)          // 陣列有 n 個元素，需交換 n - 1 次
8          swap(a[i], a[i + 1]);            // 兩數交換

9     cout << a[0];                         // 輸出陣列 a 的元素值
10    for (i = 1; i < n - 1; i++)
11       cout << ", " << a[i];
12    return 0;
13 }
```

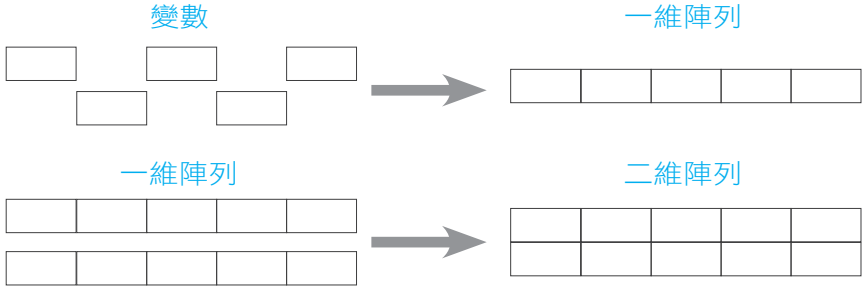
執行結果

```
2, 3, 4, 5, 6, 1
```

6.3 二維陣列與多維陣列

6.3.1 二維陣列的宣告

如下圖，多個相同資料型態的變數，可組成一維陣列；同樣地，多個相同型態與大小的一維陣列，也可組成二維陣列（two-dimensional array）。



例如：下表是某 6 位同學 5 學科的成績，可使用 6 個一維陣列 a~f，每個陣列有 5 個元素，來儲存每位學生的各科成績。例如：陣列 a 存 1 號的成績，陣列 b 存 2 號的成績，依此類推，陣列 f 存 6 號的成績。

	國文	英文	數學	自然	社會
1	86	92	57	81	65
2	96	98	81	67	74
3	91	73	92	78	67
4	68	88	90	89	78
5	87	96	70	62	68
6	84	75	94	80	90

這些資料若使用一維陣列來表示，處理起來會較複雜。上例的表格式資料共有 30（6×5）筆成績，成績都是整數，具有相同資料型態，所以可使用二維陣列來表示。表格式資料使用二維陣列表示，可以使用 for 雙重迴圈處理，非常方便。

前面介紹過的陣列，都是一維陣列，只使用一個索引；二維陣列類似數學的矩陣，使用兩個索引。宣告的格式如下

```
資料型態 陣列名稱 [ 列的大小 ] [ 行的大小 ];
```

4. 因為每列的第 1 個數是 1，所以 p[i][0] = 1。

5. 如下圖，第 2 個以後的數字是上一列同一欄元素與前一個元素的和，所以 p[i][j] = p[i-1][j-1] + p[i-1][j]。

p[i-1][j-1]	p[i-1][j]
	p[i][j]

6. 最後輸出陣列 p 內非 0 的元素，即可輸出巴斯卡三角形。

```
1  #include <iostream>
2  #include <iomanip>
3  using namespace std;
4  int main() {
5          int i, j, n;
6          cin >> n;
7          int p[n][n + 1] ={};
8      p[0][0] = p[0][1] = 1;
9      for (i = 1; i < n; i++){
10         p[i][0] = 1;
11         for (j = 1; j < i + 1; j++)
12             p[i][j] = p[i-1][j-1] + p[i-1][j];
10         p[i][j] = 1;
13     }
14
15     for (i = 0; i < n; i++){
16         for (j = 0; j < n + 1; j++)
17             if (p[i][j] != 0)
18                 cout << setw(4) << p[i][j];
19         cout << endl;
20     }
21     return 0;
22 }
```

使用 setw 函數，需引入此標頭檔

將二維陣列 p 的初始值設為 0

將第 1 列的兩個元素設為 1

每列的第 1 個元素設為 1

每列第 2 個以後的數字 = 上一列同欄的數及其前一個數的和

將字元欄位的寬度設為 4

列印二維陣列的值

二、應用題

1. 寫出產生下列範圍內之整數亂數的語法

   (1) 0 ≦ n ≦ 9

   (2) 100 ≦ n ≦ 999

   (3) -1 ≦ n ≦ 1

2. 寫一程式，能逆向複製一個陣列，例如：將陣列 {1, 2, 3, 4, 5} 複製到陣列 {5, 4, 3, 2, 1}。

3. 寫一程式，能輸入一串正整數，其中第 1 個數代表後面會出現的正整數個數，例如：輸入 5 1 9 6 8 3，第一個數 5 代表後面會出現 5 個正整數。請找出後面正整數的最大數，及其出現的位置，如上例，輸出 9 2。(b002)

4. 寫一程式，能先輸入一串整數，再輸入另一個整數，此程式能找出這一串整數中，比最後輸入之整數大的個數。例如：輸入 100 200 129 134 198，再輸入 150，會輸出 2。(b138)

5. 若某一 n 個整數的序列，其相鄰 2 數之差的絕對值序列為 1 到 n-1，則稱為 jolly jumper，例如：1 2 4 1 5 就是 jolly jumper（n = 5），因為相鄰 2 數差的絕對值為 1, 2, 3, 4。

   寫一程式，可以輸入一串整數，第一個正整數為 n（n < 100），代表此整數序列的長度，判斷此整數序列是否為 jolly jumper。(d097)

6. 費氏數列的第 1, 2 項分別為 1, 1，其後的每一項為前 2 項的合，所以第 3 項以後分別為 2, 3, 5, 8, 13, 21.....，使用陣列，寫一程式，輸入一正整數 n，輸出此數列第 n 項的值。

7. 陣列**轉置**是將一個陣列的第 i 列第 j 行元素換到第 j 列第 i 行。例如：陣列 A 的轉置陣列是 B，則 A[i][j] = B[j][i]。寫一程式，輸入一個 n×n 的二維陣列後，能輸出對應的轉置陣列。

學習挑戰

一、選擇題

1. (　　) 下列何者不是正確的字元或字串常數？

   (A) ""　(B) "hello"　(C) 'aa'　(D) 'm'

2. (　　) ~~若宣告 char str[3];~~，下列何者不是正確的字串表示？

   (A) char s[3] = "c++"　(B) char s[3] = "c"　(C) char s[3] = ""　(D) char s[3] = "/0"

3. (　　) 字串 "Hello\" 共占幾個 bytes ？

   (A) 5　(B) 6　(C) 10　(D) 12

4. (　　) 下列那個字元是字串結束的符號？

   (A) \\　(B) //　(C) \0　(D) /0

5. (　　) 若要一次讀取含空白的一整串文字，可使用那一個函數？

   (A) cin.getline　(B) cin.scanf

   (C) cout.getline　(D) cin.endl

6. (　　) 下列何者的 ASCII 碼為 0 ？

   (A) 0　(B) '0'　(C) '\0'　(D) "/0"

7. (　　) 下列那一個函數能將英文字母大寫轉成小寫？

   (A) toupper　(B) todown　(C) toup　(D) tolower

8. (　　) 若 char str[20] = "Hello world!"; 則 str[12] 值為何？

   (A) 未宣告　(B) \0　(C) !　(D) \n

9. (　　) 執行下列程式片段的結果為何？

```
char word[len] = "C++ program";
word[3] = '\0';
cout << myword << endl;
```

   (A) C　(B) C+　(C) C++　(D) C++ program