

物件導向程式設計基礎 類別與物件

人工智慧與無線感應設備開發專班

湜憶電腦知訊顧問股份有限公司

馬傳義

物件導向的概念

- 「物件導向（Object Oriented Programming，OOP）」程式設計是人類發展程式語言累積出來的成果，這個成果極具影響性。
- 是應用在軟體設計的發展模式（Software Development Model），並著重在物件的分解與相互作用。
 - ◆ 是將一些常用的程式碼分類整合起來，組成可再用的程式零件。
 - ◆ 設計開發應用程式時，將程式零件定義為「類別」，再將類別實體化建立了「物件」。

物件導向的概念

◆ 利用程式碼來記錄此物件的「屬性」、「方法」與「事件」。

- 屬性（Attribute）：

- ◆ 是物件的靜態外觀描述。
- ◆ 例如：一輛車子的顏色、大小等。
- ◆ 如同於Java程式中的類別成員資料（Member data）。

- 方法（Method）：

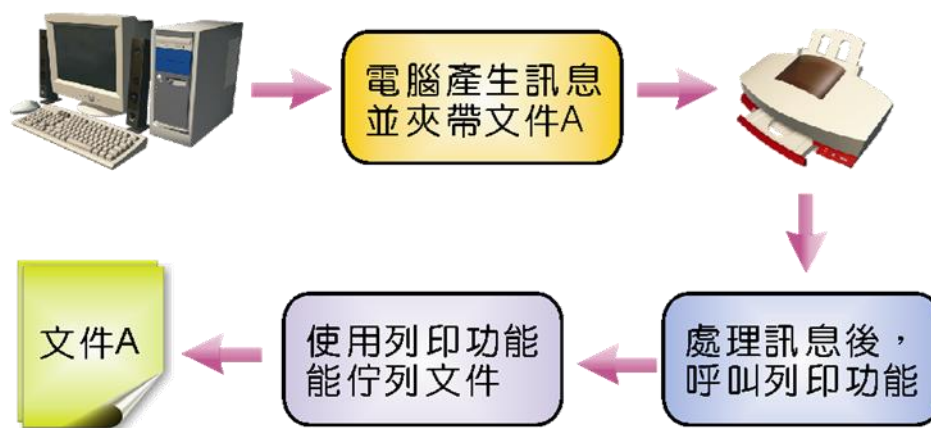
- ◆ 是一種行為模式，是用來代表一個物件的功能。
- ◆ 就是Java中的類別成員方法（Member Method）。

- 事件（Event）：

- ◆ 物件可以針對外部事件做出各種反應。
- ◆ 例如：車子沒油時，引擎就會停止。
- ◆ 物件也可以主動地發出事件訊息（如下圖）。

物件導向的概念

- ◆ 當電腦為了列印「文件A」時，就必須連同訊息一起將「文件A」傳遞給印表機。
- ◆ 這些一起傳遞給印表機的資訊，是用來提供印表機於執行動作前，必須導入的「引數（Arguments）」或稱「參數（Parameters）」。



- ◆ 一個訊息的「組成」，可以分為下列三個部分：
 - 接收訊息的物件：
如前例中，接收訊息的物件就是連接該電腦的印表機。

物件導向的概念

➤所呼叫的方法：

如前例中，電腦呼叫印表機的列印功能（即 列印方法）。

➤方法所需要的引數：

如前例中，印表機的列印功能必須要有資料「文件 A」傳入，才能做正確地列印工作。

◆物件的屬性資料可藉由「封裝」加以保護，物件的方法（功能）可藉由「繼承」得以重複使用。

- 「封裝（Encapsulation）」是隱藏（保護）類別中的成員，防止外部類別不當的存取。
- 「繼承」可以衍生新的物件方法與資料，進而提高軟體的再使用率。

◆開發出來的類別除了自己用以外，還可授權給別人用。

物件導向的概念

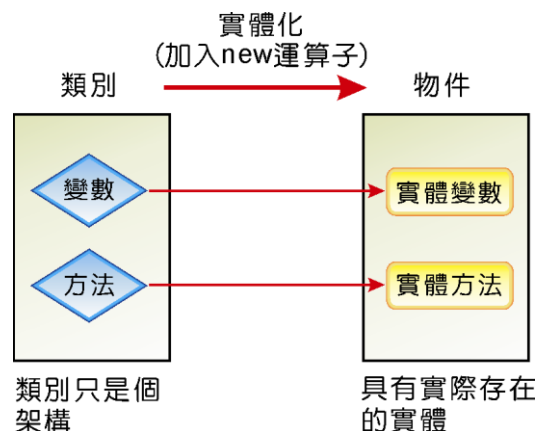
- ◆ 重點是強調軟體的可讀性（Readability）、重覆使用性（Reusability）與延伸性（Extension）。
- ◆ 能夠讓程式設計師在設計程式時，能以一種更生活化、可讀性更高的設計觀念來進程式開發。
- ◆ 目前而言，OOP已是所有程式語言所必須具備的設計功能。

類別

Java為物件導向程式設計語言，所有程式都屬於某一個類別（Class）。

「類別」與「物件」之間的關係：

- 「類別」是「物件」的模型、模組。
- 「物件」是「類別」實際製作後的成品或實體。
 - 實體化後，原本類別中的變數或方法又稱為「實體變數」或「實體方法」。



類別

- 類別中有「資料成員」（欄位、屬性、變數）和「方法成員」（函式、事件），類別中至少含有其中一種成員。
- 以「封裝（encapsulate）」的概念來說，類別是由資料成員與方法成員封裝而成的一種特殊結構。
 - ◆ 以矩形為例：
 - 它具有high（高度）、wide（寬度）兩個屬性與CalArea()（求矩形面積）及GetPeri()（求矩形周長）兩個方法。
 - 如果把高度、寬度兩個屬性與計算矩形的面積或周長的方法封裝在一起，便形成一個矩形類別（例如：CRectangle）。

類別

- 其中high與wide兩個屬性就是這矩形類別的「資料成員」，而CalArea()與GetPeri()兩個方法，就是這矩形類別的「方法成員」。

類別的命名規則：

◆ 類別和介面：

- 第一個字母為大寫，而當名稱由兩個以上的單字組成時，每個單字的第一個字母為大寫。

例如：

Student、StudentName。

◆ 成員變數和成員方法：

- 以小寫為主，如果為複合字時第一個英文單字小寫，其它單字的第一個字母要大寫其餘小寫。

例如：

setColor。

類別

◆ 套件：

- 全部小寫。

例如：

java.io、java.lang.math。


◆ 常數：

- 全部大寫，如為複合字在每個單字間以底線（_）連接。

例如：

PI、MAX_VALUE。

類別

 宣告：

[類別存取層級修飾子] class 類別名稱

{

敘述區段;

⋮

}

類別

◆ 說明：

- 類別存取層級修飾子：
 - ◆ 有 public與預設階層（不宣告）兩種。
 - ◆ 宣告屬於public的類別可在不同套件（package）中使用，存取沒有限制。
 - ◆ 若類別為預設階層，則該類別只能在所定義的相同套件下使用。
- class：
 - ◆ 類別的關鍵字。
- 類別名稱
 - ◆ 自訂，但須符合命名規則。

類別

- 敘述區段：

- ◆ 包括「資料成員（屬性、欄位、...、等）」和「方法成員（行為、函式、...、等）」。

- ◆ 定義成員要用修飾子宣告。

成員的修飾子有public（公有成員）、private（私有成員）、protected（保護成員）及預設階層（default，即不宣告）。

注意：

- 一個*.java程式檔可定義多個類別，但一個程式檔案只能宣告一個public的類別，且public的類別名稱必須和程式檔名相同。
- 一個類別至少含有一種成員。

類別

例如：

```
public class Triangle
{
    double base;
    double height;

    void area()
    {
        System.out.println("三角形面積是：");
        double ans=(base*height)/2;
        System.out.println(ans);
    }
}
```

資料成員

宣告：

[成員存取層級] [修飾字] 資料型態 成員變數名稱[=初始值];

◆ 說明：

- 存取層級：

- ◆ 各存取層級的說明如下：

- ◆ public：

- 代表所有的類別都可以使用。

- ◆ protected：

- 代表只有該類別的衍生類別，或是在相同套件（package）裡的類別才能使用。

- ◆ private：

- 代表只有此類別本身才能使用。

資料成員

- ◆ 未設定：
代表只有相同套件（package）裡的類別，才可以使用。
- 修飾字：
 - ◆ static：
 - ◆ 將成員變數宣告為類別變數（Class Variable），如此一來，此類別中建立的物件都可使用此變數。
 - ◆ final：
 - ◆ 將成員變數宣告成常數的狀態，也就是不能更改此成員變數的值。
- 資料型態：
 - ◆ 成員變數的資料型態，有基本的資料型態如整數、浮點數、布林及字元，和參考型態的字串和陣列等。

資料成員

- 初始值：
 - ◆ 依照資料型態給予成員變數一個初始的設定值。

例如：

```
public class student
{
    public String name;
    public float[ ] score;
}
```

方法成員

宣告：

[成員存取層級][修飾字] 傳回值資料型別 成員方法名稱([參數])

```
{  
    敘述區段;  
    ⋮  
    return 傳回值;  
}
```

◆ 說明：

- 成員存取層級：
 - ◆ 成員方法的存取層級和成員變數相同。

方法成員

- 修飾字：
 - ◆ 成員變數的修飾字一樣可以使用在成員方法上，它的用法上有些不同，說明如下：
 - ◆ static：
將成員方法宣告為類別方法（class method），如此一來，類別可以直接使用成員方法。
 - ◆ final：
利用final宣告的成員方法，只能在該類別中使用，並不能被其衍生類別重新定義，詳細的方法會在繼承中介紹。
- 傳回值資料型別：
 - ◆ 代表的是傳回值的資料型態。
 - ◆ 「傳回值」的資料型態必須符合此處所設定的「傳回值資料型別」。
 - ◆ 如果不需要傳回值，須設為void（即 無傳回值）。

方法成員

- 成員方法名稱：
 - ◆ 自訂，但須符合命名規則。
- 參數：
 - ◆ 包含了參數的資料型態和參數的名稱。
 - ◆ 參數名稱自訂，但須符合命名規則。
 - ◆ 如果需要多個參數，可以用「,」來區隔。
- 敘述區段：
 - 方法內部運作主體。
- return：
 - ◆ 傳回值的關鍵字。
 - ◆ 如果不需要傳回值，則可省略，但傳回值資料型別須設為 void。

方法成員

- 傳回值：
 - ◆ 可以是運算式或變數。
 - ◆ 「傳回值」必須符合「傳回值資料型別」的設定。
 - ◆ 「傳回值」的類型須是合法的類型，如int、char、double...等，也可以是自行建立的類別型態。
 - ◆ 類別方法可以不需要傳回值，但傳回值資料型別須設為void。

例如：

```
void area()  
{  
    double ans=(base*height)/2;  
    System.out.println("三角形的面積是："+ans);  
}
```

方法成員

- 類別方法加入「參數」的目的是讓類別方法在使用上更有彈性、功能性加強。

例如：

```
double area()  
{  
    return (10*25.6)/2;  
}
```

更改為：

```
double area(double i, double j)  
{  
    return (i*j)/2;  
}
```

會更有彈性。

方法成員

注意：

關於參數（引數）的個數，不限制只有一個，可以多個。

建立物件

- 在main()方法或不同類別方法成員的程式碼中，可利用既有的類別來產生不同的物件，而且一個類別可產生出數個物件。
 - ◆ 以三角形類別Triangle為例，它可建立底為20cm、高為10cm的三角形物件tri1，它也可以建立底為15cm、高為12cm的三角形物件tri2。
 - ◆ 這兩個三角形物件雖來自同一類別，但卻是不一樣的物件。
 - ◆ 把這種經由類別宣告而建立出來的物件，稱為實體（Instance）。

建立物件

■ 在Java中，物件是一種類別型別的變數，在產生物件之前，要先宣告此物件所屬類別，進而建立物件。

■ 語法：

類別名稱 物件名稱;

物件名稱 = new 類別名稱();

注意：

可將上述兩行，合併為一行：

類別名稱 物件名稱=new 類別名稱();

建立物件

◆ 例如：

```
Triangle tri1;
```

```
tri1 = new Triangle();
```

或

```
Triangle tri1 = new Triangle();
```

注意：

物件的「建立」與物件的「指派」不同，例如：

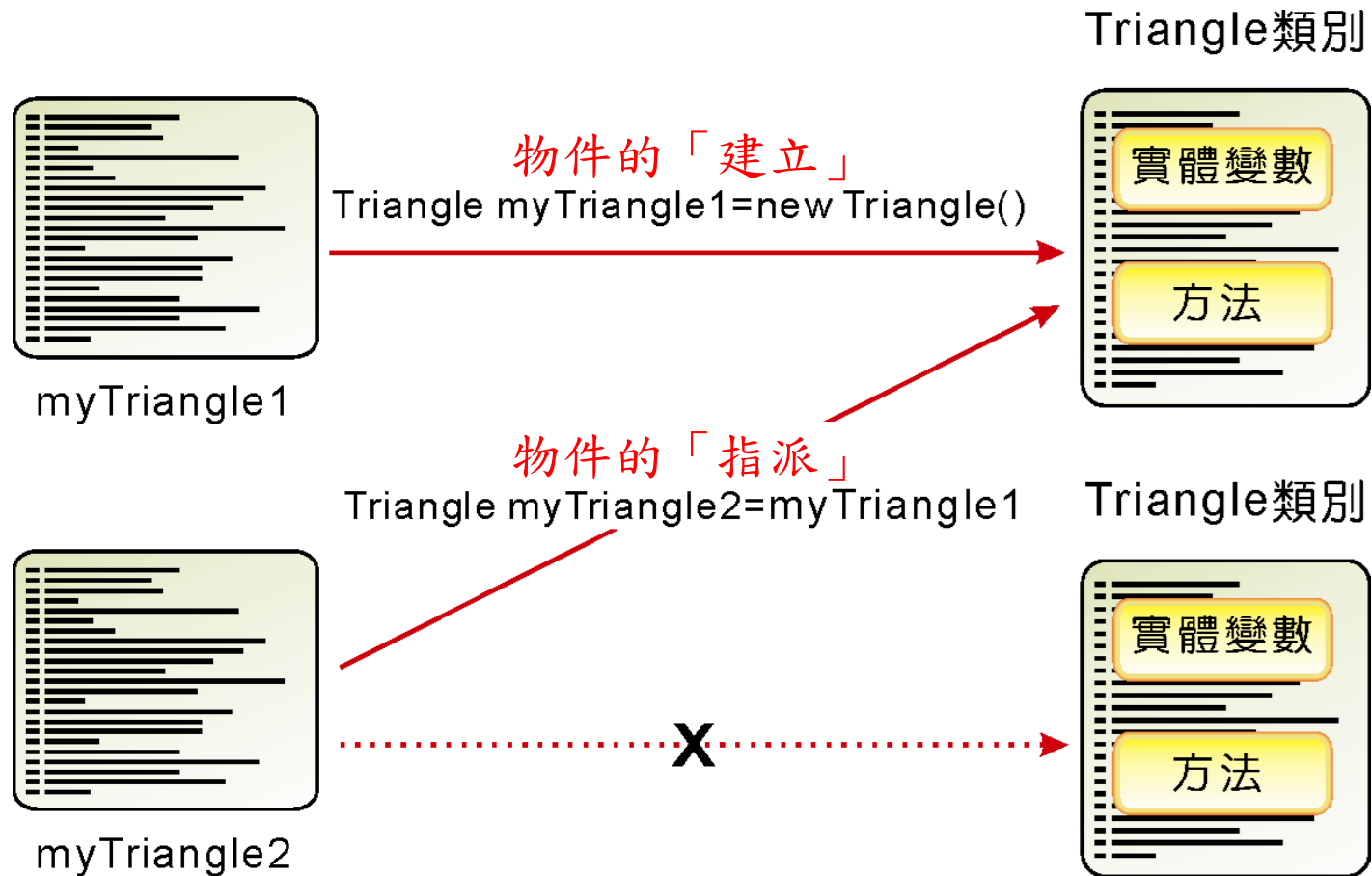
```
Triangle myTriangle1 = new Triangle();
```

與

```
Triangle myTriangle2 = myTriangle1;
```

是不一樣的。（如下圖）

建立物件



建立物件

■ 物件在使用「資料成員」和「方法成員」時，是利用「.» 運算子。

■ 語法：

物件.資料成員;

與

物件.方法成員([參數]);

建立物件

◆ 程式：

```
package CH06_01;
```

```
class Triangle
```

```
{
```

```
    int base;
```

```
    int height;
```

```
    double ans;
```

```
    void Area()
```

```
    {
```

```
        ans = (base * height) / 2;
```

```
        System.out.println("在Area中，底=" + base + "，高=" + height + "，  
                            三角形面積=" + ans + '\n');
```

```
    }
```

```
    double Area_2(int i)
```

```
    {
```

```
        System.out.print("在Area_2中，底=" + i + "，高=" + height + "，");
```


建立物件

```
        return ans = (i * height) / 2;
    }

    double Area_3(int i, int j)
    {
        System.out.print("在Area_3中，底=" + i + "，高=" + j + "，");
        return ans = (i * j) / 2;
    }
}

public class CH06_01
{
    public static void main(String[] args)
    {
        Triangle triangle = new Triangle();

        triangle.base = 2;
        triangle.height = 8;
```

建立物件

```
System.out.println("在main中，底=" + triangle.base + "，高=" +  
triangle.height + '\n');
```

```
System.out.println("不具回傳值的方法，沒有引數：");  
triangle.Area();
```

```
System.out.println("從main中，傳入單一引數base，且具有回傳值：");  
System.out.println("三角形面積=" + triangle.Area_2(4) + '\n');
```

```
System.out.println("從main中，傳入2個引數base及height，且具有回傳  
值：");
```

```
System.out.println("三角形面積=" + triangle.Area_3(4, 10));
```

```
}  
}
```

建立物件

```
1 package CH06_01;
2
3 class Triangle
4 {
5     int base;
6     int height;
7     double ans;
8
9     void Area()
10    {
11        ans = (base * height) / 2;
12        System.out.println("在Area中，底=" + base + "，高=" + height + "，三角形面積=" + ans + "\n");
13    }
14
15    double Area_2(int i)
16    {
17        System.out.print("在Area_2中，底=" + i + "，高=" + height + "，");
18        return ans = (i * height) / 2;
19    }
20
21    double Area_3(int i, int j)
22    {
23        System.out.print("在Area_3中，底=" + i + "，高=" + j + "，");
24        return ans = (i * j) / 2;
25    }
26 }
```

建立物件

```
27
28 public class CH06_01
29 {
30     public static void main(String[] args)
31     {
32         Triangle triangle = new Triangle();
33
34         triangle.base = 2;
35         triangle.height = 8;
36
37         System.out.println("在main中，底=" + triangle.base + "，高=" + triangle.height + "\n");
38
39         System.out.println("不具回傳值的方法，沒有引數：");
40         triangle.Area();
41
42         System.out.println("從main中，傳入單一引數base，且具有回傳值：");
43         System.out.println("三角形面積=" + triangle.Area_2(4) + "\n");
44
45         System.out.println("從main中，傳入2個引數base及height，且具有回傳值：");
46         System.out.println("三角形面積=" + triangle.Area_3(4, 10));
47     }
48 }
```

建立物件

◆ 執行結果：

在main中，底=2，高=8

不具回傳值的方法，沒有引數：

在Area中，底=2，高=8，三角形面積=8.0

從main中，傳入單一引數base，且具有回傳值：

在Area_2中，底=4，高=8，三角形面積=16.0

從main中，傳入2個引數base及height，且具有回傳值：

在Area_3中，底=4，高=10，三角形面積=20.0

◆ 說明：

- 行01：

- ◆ 定義「套件（package）」。

建立物件

- 行03~行26：
 - ◆ 建立「Triangle」類別。
 - ◆ 行05~行07：
 - ◆ 宣告類別的「資料成員」，base（底）、height（高）、ans（面積傳回值）。
 - ◆ 行09~行13：
 - ◆ 建立「不具傳回值」的類別的「方法成員」「Area()」，因此在方法前面需加上關鍵字「void」，表示不具傳回值。
 - ◆ 行15~行19：
 - ◆ 建立「有傳入單一引數，且具有傳回值」的類別「方法成員」「Area_2(int i)」，因此必須給予傳回值的資料類型；傳入的參數為i。

建立物件

- ◆ 行21~行25：
 - ◆ 建立「有傳入2個引數，且具有傳回值」的類別「方法成員」「Area_3(int i, int j)」，因此必須給予傳回值的資料類型；傳入的參數為i和j。
- 行32：
 - ◆ 建立triangle物件（即將類別實體化）。
- 行33~行34：
 - ◆ 指定triangle物件base及height的值。
- 行40：
 - ◆ 因為類別方法「Area()」是不傳回值，並且不具參數，呼叫時括號內不需要指定引數值。
- 行43：
 - ◆ 類別方法「Area_2(int i)」需要傳入1個引數，且有傳回值，因此呼叫時必須指定1個引數值。

建立物件

- 行46：
 - ◆ 類別方法「Area_3(int i, int j)」需要傳入2個引數，且有傳回值，因此呼叫時必須指定2個引數值。

資料的封裝

■ 資料「封裝（Encapsulation）」的目的是為了隱藏（保護）類別中的成員，防止外部類別不當的存取。

- ◆ 前面的三角形類別Triangle中，有base、height、ans三個資料成員及Area()、Area_2(int i)、Area_3(int i, int j)三個方法成員。
- ◆ 當Triangle類別建立了triangle物件以後，這六個類別成員的內容皆可以在Triangle類別的外部被指派或讀取。
- ◆ 這樣的機制是不嚴謹的，因有些資料成員的內容只是用來存放類別內部其它資料成員的運算處理結果。

資料的封裝

例如：

ans是用來存放類別內部 $(base * height) / 2$ 的結果，不希望ans的內容在Triangle類別的外部可再被設定更改。

- ◆ 若要防止類別成員遭受類別外部的敘述存取，在宣告時，將「成員存取層級」，設定適當的關鍵字，即可達到隱藏的目的。

■ 用Java來開發應用程式，類別的建立與使用會相當頻繁，因此每一類別的成員應該只允許在類別自身內部進行存取，編譯或執行時才不會產生不可預期的錯誤。

資料的封裝

■ Java提供了三種的「成員存取層級」，方便程式設計者應用。

◆ public（公開）：

- 代表所宣告的方法或屬性，可以被所有的類別成員所使用。

◆ private（私有）：

- 代表所宣告的方法或屬性，只能被此類別的成員使用。

◆ protected（保護）：

- 代表所宣告的方法或屬性，可以在同類別、同套件範圍內，或其衍生類別的成員所使用。

資料的封裝

◆ 例如：

➤ 公開類別

```
public class checkPassword  
{  
    敘述區段  
}
```

➤ 私有資料成員

```
private int usePassword;
```

➤ 保護方法成員

```
protected getPassword();
```

資料的封裝

- 運用封裝的概念，對資料或方法成員進行資料隱藏後，外部類別將無法對被保護的資料或方法成員進行修改。
- 可新增「公開」的方法成員，間接的對「私有」或「保護」的資料或方法成員進行修改。

資料的封裝

◆ 程式：

```
package CH06_02;
```

```
class Triangle
```

```
{
```

```
    private int base;
```

```
    private int height;
```

```
    private double ans;
```

```
    public void setData(int b, int h)
```

```
    {
```

```
        base = b;
```

```
        height = h;
```

```
        Area();
```

```
    }
```

```
    public double outArea()
```

```
    {
```

```
        return ans;
```

資料的封裝

```
}

private void Area()
{
    ans = (base * height) / 2;
    System.out.println("經由public setData()方法，間接設定，底=" + base + "，  
                        高=" + height + '\n');
}
}

public class CH06_02
{
    public static void main(String[] args)
    {
        Triangle triangle = new Triangle();

        int base = 2, height = 8;
        double ans;
```


資料的封裝

```
triangle.setData(base, height);  
ans = triangle.outArea();
```

```
System.out.println("經由public outArea()方法，間接傳回三角型面積為" +  
                    ans + '\n');
```

```
}
```

```
}
```

資料的封裝

```
1 package CH06_02;
2
3 class Triangle
4 {
5     private int base;
6     private int height;
7     private double ans;
8
9     public void setData(int b, int h)
10    {
11        base = b;
12        height = h;
13        Area();
14    }
15
16    public double outArea()
17    {
18        return ans;
19    }
20
21    private void Area()
22    {
23        ans = (base * height) / 2;
24        System.out.println("經由public setData()方法，間接設定，底=" + base + "，高=" + height + "\n");
25    }
26 }
```

資料的封裝

```
27
28 public class CH06_02
29 {
30     public static void main(String[] args)
31     {
32         Triangle triangle = new Triangle();
33
34         int base = 2, height = 8;
35         double ans;
36
37         triangle.setData(base, height);
38         ans = triangle.outArea();
39
40         System.out.println("經由public outArea()方法，間接傳回三角型面積為" + ans + "\n");
41     }
42 }
```

◆ 執行結果：

經由public setData()方法，間接設定，底=2，高=8

經由public outArea()方法，間接傳回三角型面積為8.0

資料的封裝

◆ 說明：

- 行01：
 - ◆ 定義「套件（package）」。
- 行03~行26：
 - ◆ 建立「Triangle」類別。
 - ◆ 行05~行07：
 - ◆ 宣告「私有」的類別「資料成員」，base（底）、height（高）、ans（面積傳回值）。
 - ◆ 行09~行14：
 - ◆ 建立「公開」且「不具傳回值」的類別「方法成員」
「setData」，因此在方法名稱的前面需加上關鍵字
「public void」，表示「公開」且「不具傳回值」。

資料的封裝

- ◆ 行11~行12：

透過此公開的方法，間接將main中設定的base及height兩個整數變數傳入給「Triangle」類別中的「私有」的「資料成員」「base」及「height」。
- ◆ 行13：

執行「私有」的方法成員「Area()」。
- ◆ 行16~行19：
 - ◆ 建立「公開」且「傳回值」的類別的「方法成員」「setData()」，因此必須給予傳回值的資料類型及傳回的值。
 - ◆ 行18：

傳回ans給main。
- ◆ 行21~行25：
 - ◆ 建立「私有」的計算三角型面積的「方法成員」「Area()」。

資料的封裝

- 行32：
 - ◆ 建立triangle物件（即將類別實體化）。
- 行34：
 - ◆ 宣告整數變數base及height，並指定初值。
- 行34：
 - ◆ 宣告倍精度浮點數變數ans。
- 行37：
 - ◆ 將整數變數base及height的值，傳給「Triangle」類別中，「公開」的「方法成員」`setData()`。
- 行38：
 - ◆ 將「Triangle」類別中，「公開」的「方法成員」`outArea()`傳回的值，指定給倍精度浮點數變數ans。

方法成員的多載

- 不具有OOP功能的程式語言，當方法（或稱函式）中的引數個數不同、引數型別不同時，或方法內的敘述不同時，就要用不同的方法名稱來撰寫不同方法內容。
- 程式設計者常為了方法的命名而困擾，尤其在大型軟體開發中，常遇到功能類似的方法，而要如何命名以避免使用上的困擾，常令程式設計者傷透腦筋。
- Java提供方法多載（Method Overload）的功能，允許程式設計者使用相同的方法名稱。

方法成員的多載

■ 只要引數個數不同或引數型別不同，Java會自動判斷並執行所需求的方法。

- ◆ 以一組數字相加為例，可以設計一個方法，來計算數字相加，並傳回相加的結果。
- ◆ 但相加的數字可能是2個、3個、或一個陣列，若只有兩個數字相加時，也有可能是兩個整數相加或兩個浮點數相加...
- ◆ 這些不同數字相加的情況，可以設計好幾個方法來計算並傳回相加結果，但在這些方法中，我們只需命名同一個方法名稱。

例如：

Total()。

方法成員的多載

◆ 程式：

```
package CH06_03;
```

```
class Triangle
```

```
{
```

```
    public int ibase = 10;
```

```
    public int iheight = 20;
```

```
    public double dbase=10.0;
```

```
    public double dheight=20.0;
```

```
    public int Area()
```

```
{
```

```
        return (ibase * iheight) / 2;
```

```
}
```

```
    public int Area(int b)
```

```
{
```

```
        ibase = b;
```

方法成員的多載

```
    return (ibase * iheight) / 2;  
}
```

```
public int Area(int b, int h)  
{  
    ibase = b;  
    iheight = h;  
    return (ibase * iheight) / 2;  
}
```

```
public double Area(double b, double h)  
{  
    dbase=b;  
    dheight=h;  
    return (dbase * dheight) / 2;  
}  
}
```

```
public class CH06_03
```

方法成員的多載

```
{  
    public static void main(String[] args)  
    {  
        Triangle triangle = new Triangle();  
  
        int ibase = 2, iheight = 8, ians;  
        double dbase = 2.5, dheight = 8.5, dans;  
  
        ians = triangle.Area();  
        System.out.println("執行沒有傳入值的Area()方法,傳回三角型面積為 " +  
                             ians + '\n');  
  
        ians = triangle.Area(ibase);  
        System.out.println("執行有傳入一個整數值的Area()方法,傳回三角型面  
                             積為 " + ians + '\n');  
  
        ians = triangle.Area(ibase, iheight);  
        System.out.println("執行有傳入二個整數值的Area()方法,傳回三角型面  
                             積為 " + ians + '\n');
```

方法成員的多載

```
    dans = triangle.Area(dbase, dheight);  
    System.out.println("執行有傳入二個倍精度浮點數值的Area()方法,傳回  
                        三角型面積為 " + dans + '\n');  
}  
}
```

方法成員的多載

```
1 package CH06_03;
2
3 class Triangle
4 {
5     public int ibase = 10;
6     public int iheight = 20;
7
8     public double dbase=10.0;
9     public double dheight=20.0;
10
11     public int Area()
12     {
13         return (ibase * iheight) / 2;
14     }
15
16     public int Area(int b)
17     {
18         ibase = b;
19         return (ibase * iheight) / 2;
20     }
21
22     public int Area(int b, int h)
23     {
24         ibase = b;
25         iheight = h;
26         return (ibase * iheight) / 2;
27     }
```

方法成員的多載

```
28
29 public double Area(double b, double h)
30 {
31     dbase=b;
32     dheight=h;
33     return (dbase * dheight) / 2;
34 }
35 }
36
37 public class CH06_03
38 {
39     public static void main(String[] args)
40     {
41         Triangle triangle = new Triangle();
42
43         int ibase = 2, iheight = 8, ians;
44         double dbase = 2.5, dheight = 8.5, dans;
45
46         ians = triangle.Area();
47         System.out.println("執行沒有傳入值的Area()方法,傳回三角型面積為 " + ians + "\n");
48
49         ians = triangle.Area(ibase);
50         System.out.println("執行有傳入一個整數值的Area()方法,傳回三角型面積為 " + ians + "\n");
51
52         ians = triangle.Area(ibase, iheight);
53         System.out.println("執行有傳入二個整數值的Area()方法,傳回三角型面積為 " + ians + "\n");
54     }
}
```

方法成員的多載

```
55     dans = triangle.Area(dbase, dheight);  
56     System.out.println("執行有傳入二個倍精度浮點數值的Area0方法,傳回三角型面積為 " + dans + "\n");  
57 }  
58 }
```

◆ 執行結果：

執行沒有傳入值的Area0方法,傳回三角型面積為 100

執行有傳入一個整數值的Area0方法,傳回三角型面積為 20

執行有傳入二個整數值的Area0方法,傳回三角型面積為 8

執行有傳入二個倍精度浮點數值的Area0方法,傳回三角型面積為 10.625

◆ 說明：

• 行01：

- ◆ 定義「套件（package）」。

方法成員的多載

- 行03~行35：
 - ◆ 建立「Triangle」類別。
 - ◆ 行05~行06：
 - ◆ 宣告「公開」的類別「資料成員」，並指定初值。
 - ◆ 行08~行09：
 - ◆ 宣告「公開」的類別「資料成員」，並指定初值。
 - ◆ 行11~行14：
 - ◆ 宣告「公開」的類別「方法成員」，沒有傳入值，傳回計算面積後的結果。
 - ◆ 行16~行20：
 - ◆ 宣告「公開」的類別「方法成員」，有傳入一個整數的值，傳回計算面積後的結果。

方法成員的多載

- ◆ 行22~行26：
 - ◆ 宣告「公開」的類別「方法成員」，有傳入二個整數的值，傳回計算面積後的結果。
- ◆ 行29~行34：
 - ◆ 宣告「公開」的類別「方法成員」，有傳入兩個倍精度浮點數的值，傳回計算面積後的結果。
- 行41：
 - ◆ 建立triangle物件（即將類別實體化）。
- 行43：
 - ◆ 宣告整數變數，其中ibase、iheight有指定初值。
- 行44：
 - ◆ 宣告倍精度浮點數變數，其中dbase、dheight有指定初值。

方法成員的多載

- 行46：
 - ◆ 沒有傳入值給「Triangle」類別中，「公開」的整數「方法成員」`Area()`，並接收傳回的值。
- 行49：
 - ◆ 將整數變數`ibase`的值，傳給「Triangle」類別中，「公開」的整數「方法成員」`Area(int b)`，並接收傳回的值。
- 行52：
 - ◆ 將整數變數`ibase`、`iheight`的值，傳給「Triangle」類別中，「公開」的整數「方法成員」`Area(int b,int h)`，並接收傳回的值。
- 行55：
 - ◆ 將倍精度浮點數變數`dbase`、`dheight`的值，傳給「Triangle」類別中，「公開」的倍精度浮點數變數「方法成員」`Total(double b,double h)`，並接收傳回的值。

建構子

- 宣告變數之同時可以指派變數值；同樣地，類別宣告建立物件之同時，也是可以指派物件資料成員的初始值。
- 這個動作就是要在類別內定義「建構子（Constructor）」。
- 類別在建立物件之同時，便自動執行類別的「建構子」，而要物件成員初始化的敘述，就可寫在「建構子」內。
- 「建構子」也是一種在類別內部的「方法成員」，但是在使用時需注意下面事項：

建構子

- ◆ 「建構子」的定義方式與「方法」相似。
- ◆ 「建構子」的名稱必須與所屬的「類別名稱」相同；類別用new建立物件之同時，便自動執行此物件的「建構子」。
- ◆ 類別中只有一個「建構子」時，該「建構子」一定要用public宣告（或省略）。
 - 若是只有一個「建構子」時，用private宣告，則在建立類別物件時，會出現錯誤訊息。
- ◆ 「建構子」前面不能使用「傳回值型別」，也沒有「傳回值」（即不能用void與return）。
- ◆ 若省略定義類別的「建構子」，系統會自動產生「預設建構子」，內容為空敘述。

建構子

◆ 建構子也可以多載。

- 當建構子多載設計時，有的建構子前面可加private，一般使用於精簡程式用。

建構子

◆ 程式：

```
package CH06_04;
```

```
class Triangle
```

```
{
```

```
    public double dbase, dheight;
```

```
    public Triangle()
```

```
    {
```

```
        dbase = 10.0;
```

```
        dheight = 20.0;
```

```
        System.out.println("建構子Triangle()中定義，底 = " + dbase + "，高 = " + dheight);
```

```
    }
```

```
    public Triangle(double b, double h)
```

```
    {
```

```
        dbase = b;
```

```
        dheight = h;
```

建構子

```
System.out.println("建構子接收Triangle(double b,double h)傳入的值，底  
                    = " + dbase + "，高 = " + dheight);  
}  
  
public double Area()  
{  
    return (dbase * dheight) / 2;  
}  
  
public double Area(double b, double h)  
{  
    dbase = b;  
    dheight = h;  
    System.out.println("從dtriangle.Area(dbase, dheight)接收傳入的二個倍精  
                        度浮點數的值，底 = " + dbase + "，高 = " + dheight);  
    return (dbase * dheight) / 2;  
}  
}
```

建構子

```
public class CH06_04
{
    public static void main(String[] args)
    {
        double dbase = 2.5, dheight = 8.5, dans;

        Triangle triangle = new Triangle();
        dans = triangle.Area();
        System.out.println("執行沒有傳入值的Area()方法，傳回三角型面積為 "
                           + dans + '\n');

        Triangle dtriangle = new Triangle(dbase, dheight);
        dbase = 5.5;
        dheight = 10.5;
        dans = dtriangle.Area(dbase, dheight);
        System.out.println("執行有傳入二個倍精度浮點數值的Area(dbase,
                           dheight)方法，傳回三角型面積為 " + dans + '\n');
    }
}
```


建構子

```
1 package CH06_04;
2
3 class Triangle
4 {
5     public double dbase, dheight;
6
7     public Triangle()
8     {
9         dbase = 10.0;
10        dheight = 20.0;
11        System.out.println("建構子Triangle()中定義，底 = " + dbase + "，高 = " + dheight);
12    }
13
14    public Triangle(double b, double h)
15    {
16        dbase = b;
17        dheight = h;
18        System.out.println("建構子接收Triangle(double b,double h)傳入的值，底 = " + dbase + "，高 = " + dheight);
19    }
20
21    public double Area()
22    {
23        return (dbase * dheight) / 2;
24    }
25
26    public double Area(double b, double h)
27    {
28        dbase = b;
29        dheight = h;
30        System.out.println("從triangle.Area(dbase, dheight)接收傳入的二個倍精度浮點數的值，底 = " + dbase + "，高 = " + dheight);
31        return (dbase * dheight) / 2;
32    }
33 }
34
```

建構子

```
35 public class CH06_04
36 {
37     public static void main(String[] args)
38     {
39         double dbase = 2.5, dheight = 8.5, dans;
40
41         Triangle triangle = new Triangle();
42         dans = triangle.Area();
43         System.out.println("執行沒有傳入值的Area()方法，傳回三角型面積為 " + dans + "\n");
44
45         Triangle dtriangle = new Triangle(dbase, dheight);
46         dbase = 5.5;
47         dheight = 10.5;
48         dans = dtriangle.Area(dbase, dheight);
49         System.out.println("執行有傳入二個倍精度浮點數值的Area(dbase, dheight)方法，傳回三角型面積為 " + dans + "\n");
50     }
51 }
```

◆ 執行結果：

建構子Triangle()中定義，底 = 10.0，高 = 20.0
執行沒有傳入值的Area()方法，傳回三角型面積為 100.0

建構子接收Triangle(double b,double h)傳入的值，底 = 2.5，高 = 8.5
從dtriangle.Area(dbase, dheight)接收傳入的二個倍精度浮點數的值，底 = 5.5，高 = 10.5
執行有傳入二個倍精度浮點數值的Area(dbase, dheight)方法，傳回三角型面積為 28.875

建構子

◆ 說明：

- 行01：
 - ◆ 定義「套件（package）」。
- 行03~行33：
 - ◆ 建立「Triangle」類別。
 - ◆ 行05：
 - ◆ 宣告「公開」的類別「資料成員」。
 - ◆ 行07~行12：
 - ◆ 宣告「公開」的類別「建構子」「Triangle()」，沒有傳入值。
 - ◆ 行14~行19：
 - ◆ 宣告「公開」的類別「建構子」「Triangle(double b, double h)」，有傳入兩個倍精度浮點數的值。

建構子

- ◆ 行21~行24：
 - ◆ 宣告「公開」的類別「方法成員」，沒有傳入值，利用「Triangle()」「建構子」中定義的dbase及dheight來計算面積，並傳回計算結果。
- ◆ 行26~行32：
 - ◆ 宣告「公開」的類別「方法成員」，有傳入兩個倍精度浮點數的值，傳回計算面積後的結果。
- 行39：
 - ◆ 宣告倍精度浮點數變數，其中dbase、dheight有指定初值。
- 行41：
 - ◆ 利用「Triangle()」的「建構子」，建立triangle物件。
- 行42：
 - ◆ 沒有傳入值給「Triangle」類別中，「公開」的類別「方法成員」「Area()」，並接收傳回的值。

建構子

- 行45：
 - ◆ 利用「Triangle(double b, double h)」的「建構子」，建立dtriangle物件。
- 行46~行47：
 - ◆ 指定值給倍精度浮點數變數dbase、dheight。
- 行48：
 - ◆ 傳入兩個倍精度浮點數的值給「Triangle」類別中，「公開」的類別「方法成員」Area(double b, double h)」，並接收傳回的值。

類別變數

- 在「類別」中用「成員存取層級」所定義的變數稱為「成員變數（member variable）」，而「類別」建立了物件後，該成員變數就成為物件的「實體變數（instance variable）」。
- 「實體變數」是由「類別」建立的多個物件所獨佔，各物件實體變數名稱雖然相同，但是彼此之間並無關聯。
- 若「成員變數」要能給同「類別」所建立之各物件共享，該變數需被宣告為「類別變數（class variable）」。

類別變數

- 宣告變數時，在「成員存取層級」後面再加上「static」，則此變數便成為「類別變數」（或稱「靜態變數」）。
- 「類別變數」被存取的方式為：

類別名稱.類別變數名稱

- 「靜態變數」還是有「公有靜態變數」（用public宣告）、「私有靜態變數」（用private宣告）之分別。
- 若要在類別外部直接取用靜態變數，該變數還是要用public宣告為「公有成員」才行。

類別變數

◆ 程式：

```
package CH06_05;
```

```
class Cstudent
```

```
{
```

```
    public static int count = 0;
```

```
    public int num = 0;
```

```
    private int tot;
```

```
    Cstudent()
```

```
    {
```

```
        count++;
```

```
        num++;
```

```
    }
```

```
    public int GetTot(int chi, int eng)
```

```
    {
```

```
        tot = chi + eng;
```

```
        return tot;
```


類別變數

```
}  
}
```

```
public class CH06_05
```

```
{
```

```
    public static void main(String[] args)
```

```
    {
```

```
        Cstudent s1 = new Cstudent();
```

```
        System.out.println("s1.num = " + s1.num);
```

```
        System.out.println("第" + Cstudent.count + "位");
```

```
        System.out.println("總分 = " + s1.GetTot(80, 90));
```

```
        Cstudent s2 = new Cstudent();
```

```
        System.out.println("s2.num = " + s2.num);
```

```
        System.out.println("第" + Cstudent.count + "位");
```

```
        System.out.println("總分 = " + s2.GetTot(100, 100));
```

```
    }
```

```
}
```

類別變數

```
1 package CH06_05;
2
3 class Cstudent
4 {
5     public static int count = 0;
6     public int num = 0;
7     private int tot;
8
9     Cstudent()
10    {
11        count++;
12        num++;
13    }
14
15    public int GetTot(int chi, int eng)
16    {
17        tot = chi + eng;
18        return tot;
19    }
20 }
21
22 public class CH06_05
23 {
24     public static void main(String[] args)
25     {
```

類別變數

```
26      Cstudent s1 = new Cstudent();
27      System.out.println("s1.num = " + s1.num);
28      System.out.println("第" + Cstudent.count + "位");
29      System.out.println("總分 = " + s1.GetTot(80, 90));
30
31      Cstudent s2 = new Cstudent();
32      System.out.println("s2.num = " + s2.num);
33      System.out.println("第" + Cstudent.count + "位");
34      System.out.println("總分 = " + s2.GetTot(100, 100));
35  }
36 }
```

◆ 執行結果：

```
s1.num = 1
第1位
總分 = 170
s2.num = 1
第2位
總分 = 200
```

◆ 說明：

- 行01：

- ◆ 定義「套件（package）」。

類別變數

- 行03~行20：
 - ◆ 建立「Cstudent」類別。
 - ◆ 行05：
 - ◆ 宣告「公開」、「靜態」的「類別變數（即「靜態變數」）」，並指定初值。
 - ◆ 行06：
 - ◆ 宣告「公開」的類別「資料成員」，並指定初值。
 - ◆ 行07：
 - ◆ 宣告「私有」的類別「資料成員」。
 - ◆ 行09~行13：
 - ◆ 宣告類別「建構子」`Cstudent()`，沒有傳入值。此類別「建構子」為「公開」的（省略public）。
 - ◆ 行15~行19：
 - ◆ 宣告「公開」的類別「方法成員」，有傳入兩個整數數的值，傳回加總後的結果。

類別變數

- 行26：
 - ◆ 利用「Cstudent()」的「建構子」，建立s1物件。
- 行27：
 - ◆ 顯示s1物件的「實體變數」 「num」內的值。
- 行28：
 - ◆ 顯示「類別變數」 「count」內的值。
- 行29：
 - ◆ 傳入兩個整數值給s1物件的「方法成員」 「GetTot(int chi, int eng)」，並顯示加總後的結果。
- 行31：
 - ◆ 利用「Cstudent()」的「建構子」，建立s2物件。
- 行32：
 - ◆ 顯示s2物件的「實體變數」 「num」內的值。

類別變數

- 行33：
 - ◆ 顯示「類別變數」 「count」 內的值。
- 行34：
 - ◆ 傳入兩個整數值給s2物件的「方法成員」 「GetTot(int chi, int eng)」，並顯示加總後的結果。

類別方法

- 在類別中的「一般方法（方法成員）」被使用的方式是，先建立該「類別」的物件，再使用「物件名稱.方法名稱」來呼叫。
- 「類別」的「方法成員」在宣告時，若在方法名稱前面加上「static」，便成為「類別方法」（或稱「靜態方法」）。
- 「類別方法」可以不用先「new」來建立物件，就可以直接給類別外部的敘述呼叫使用。
- 「類別方法」被使用的方式：

類別名稱.類別方法名稱()

類別方法

- 要注意的是，在「類別方法」中所使用到的變數皆須為「靜態變數（類別變數）」，即要用「static」宣告的變數才能在類別方法中出現。

類別方法

◆ 程式：

```
package CH06_06;
```

```
class CHeight
```

```
{
```

```
    public static int count = 0;
```

```
    public static double hei_tot = 0;
```

```
    private double hei;
```

```
    public void SetTot(double h)
```

```
    {
```

```
        hei = h;
```

```
        hei_tot += hei;
```

```
        count++;
```

```
    }
```

```
    public static double GetHeiAve()
```

```
    {
```

```
        return hei_tot / count;
```

類別方法

```
    }  
}  
  
public class CH06_06  
{  
    public static void main(String[] args)  
    {  
        CHeight p1 = new CHeight();  
        p1.SetTot(180);  
  
        CHeight p2 = new CHeight();  
        p2.SetTot(140);  
  
        CHeight p3 = new CHeight();  
        p3.SetTot(160);  
  
        System.out.println("總人數 = " + CHeight.count);  
        System.out.println("身高平均 = " + CHeight.GetHeiAve());  
    }  
}
```

類別方法

```
1 package CH06_06;
2
3 class CHeight
4 {
5     public static int count = 0;
6     public static double hei_tot = 0;
7     private double hei;
8
9     public void SetTot(double h)
10    {
11        hei = h;
12        hei_tot += hei;
13        count++;
14    }
15
16    public static double GetHeiAve()
17    {
18        return hei_tot / count;
19    }
20 }
21
22 public class CH06_06
23 {
24     public static void main(String[] args)
25     {
```

類別方法

```
26     CHeight p1 = new CHeight();
27     p1.SetTot(180);
28
29     CHeight p2 = new CHeight();
30     p2.SetTot(140);
31
32     CHeight p3 = new CHeight();
33     p3.SetTot(160);
34
35     System.out.println("總人數 = " + CHeight.count);
36     System.out.println("身高平均 = " + CHeight.GetHeiAve());
37 }
38 }
```

◆ 執行結果：

```
總人數 = 3
身高平均 = 160.0
```

◆ 說明：

- 行01：

- ◆ 定義「套件（package）」。

類別方法

- 行03~行20：
 - ◆ 建立「Cstudent」類別。
 - ◆ 行05：
 - ◆ 宣告「公開」、「靜態」整數的「類別變數（即「靜態變數」）」，並指定初值。
 - ◆ 行06：
 - ◆ 宣告「公開」、「靜態」倍精度浮點數的「類別變數（即「靜態變數」）」，並指定初值。
 - ◆ 行07：
 - ◆ 宣告「私有」的類別「資料成員」。
 - ◆ 行09~行14：
 - ◆ 宣告「公開」且不具傳回值的「方法成員」「SetTot(double h)」，有傳入一個倍精度浮點數的值。

類別方法

- ◆ 行16~行19：
 - ◆ 宣告「公開」的「類別方法（即「靜態方法」）」「`double GetHeiAve()`」，沒有傳入值，傳回身高平均後的結果。
- 行26：
 - ◆ 利用「`CHeight()`」的「建構子」，建立p1物件。
- 行27：
 - ◆ 傳入一個倍精度浮點數的值給p1物件的「方法成員」「`SetTot(double h)`」，進行身高的加總。
- 行29：
 - ◆ 利用「`CHeight()`」的「建構子」，建立p2物件。
- 行30：
 - ◆ 傳入一個倍精度浮點數的值給p2物件的「方法成員」「`SetTot(double h)`」，進行身高的加總。

類別方法

- 行32：
 - ◆ 利用「CHHeight()」的「建構子」，建立p3物件。
- 行33：
 - ◆ 傳入一個倍精度浮點數的值給p3物件的「方法成員」「SetTot(double h)」，進行身高的加總。
- 行35：
 - ◆ 顯示「類別變數」「count」內的值。
- 行36：
 - ◆ 顯示「類別方法」「GetHeiAve()」傳回的值。

- this是表示物件實體本身（即 this = 此類別）。
- 在物件建立後為「這個物件」的參考名稱。
- this只能在類別中使用，它儲存的是物件實體本身的參考。
- 在類別中呼叫this()，表示要執行此類別的建構子（Constructor）。
- 使用this的時機在於要引用物件實體本身的方法、成員或者類別本身實體的建構子。
- Java程式編譯時會在成員加上this關鍵字。

this主要的五種用法：

- 一. 代表使用中的類別（即自身的類別）。
- 二. 當「方法成員」的「參數」與「資料成員」名稱相同時，表示使用「類別」的「資料成員」，而非「方法成員」的「參數」。
- 三. this不能用在static方法（即類別方法）中。
- 四. 在「建構子」中引用，是為了滿足指定「參數」類型的「建構子」（其實也就是建構子）。
- 五. 若在「建構子」中使用，要注意的是「只能引用一個「建構子」，且必須位於程式碼的第一行」。

■ 在「建構子」中，呼叫另一個「建構子」

◆ 語法：

this(參數)

注意：

- 只能在類別的建構子中使用。
- 只能引用一個建構子。
- 程式碼須放在建構子的第一行。
- this跟super不能同時使用。
 - ◆ 透過super可呼叫父類別資源。

■ 「方法成員」的「參數」與「資料成員」名稱相同時：

◆ 語法：

this.資料成員

注意：

- 在沒有同名的情況下，可以直接用「方法成員」的「參數」，也可以用「this.資料成員」。

◆ 程式：

```
package CH06_07;
```

```
class ThisTest
```

```
{
```

```
    private int i = 0;
```

```
    ThisTest(int i)
```

```
    {
```

```
        System.out.println("傳入整數建構子的參數 i = " + i + "\n類別的資料成員 i = " + this.i);
```

```
        this.i = i + 1;
```

```
        System.out.println("類別的資料成員執行 this.i=i+1 的結果：" + this.i);
```

```
        System.out.println("i-1 = " + (i - 1) + "\nthis.i+1 = " + (this.i + 1));
```

```
    }
```

```
    ThisTest(String s)
```

```
    {
```

```
        System.out.println("傳入字串建構子的 s：" + s);
```

this

```
}
```

```
ThisTest(int i, String s)
```

```
{
```

```
    this(s);
```

```
    // this(i);
```

```
    System.out.println("傳入整數建構子的 i = " + i + "\n傳入字串建構子的  
                        s : " + s);
```

```
    this.i = i++;
```

```
    System.out.println("this.i = " + this.i + "\ni++ = " + i + "\nthis.i=i++ : " +  
                      (this.i = i++));
```

```
}
```

```
public ThisTest increment()
```

```
{
```

```
    this.i++;
```

```
    System.out.println("類別的資料成員this.i++ : " + this.i);
```

```
    System.out.println("返回this : " + this);
```

```
    return this;
```

this

```
    }  
}  
  
public class CH06_07  
{  
    public static void main(String[] args)  
    {  
        ThisTest tt0 = new ThisTest(10);  
        ThisTest tt1 = new ThisTest("ok");  
        ThisTest tt2 = new ThisTest(20, "ok again!");  
        System.out.println(tt0.increment());  
    }  
}
```

this

```
1 package CH06_07;
2
3 class ThisTest
4 {
5     private int i = 0;
6
7     ThisTest(int i)
8     {
9         System.out.println("傳入整數建構子的參數 i = " + i + "\n類別的資料成員 i = " + this.i);
10        this.i = i + 1;
11        System.out.println("類別的資料成員執行 this.i=i+1 的結果： " + this.i);
12        System.out.println("i-1 = " + (i - 1) + "\nthis.i+1 = " + (this.i + 1));
13    }
14
15    ThisTest(String s)
16    {
17        System.out.println("傳入字串建構子的 s： " + s);
18    }
19
20    ThisTest(int i, String s)
21    {
22        this(s);
23        // this(i);
24        System.out.println("傳入整數建構子的 i = " + i + "\n傳入字串建構子的 s： " + s);
25        this.i = i++;
26        System.out.println("this.i = " + this.i + "\ni++ = " + i + "\nthis.i=i++： " + (this.i = i++));
27    }
28 }
```

this

```
28
29 public ThisTest increment()
30 {
31     this.i++;
32     System.out.println("類別的資料成員this.i++ : " + this.i);
33     System.out.println("返回this : " + this);
34     return this;
35 }
36 }
37
38 public class CH06_07
39 {
40     public static void main(String[] args)
41     {
42         ThisTest tt0 = new ThisTest(10);
43         ThisTest tt1 = new ThisTest("ok");
44         ThisTest tt2 = new ThisTest(20, "ok again!");
45         System.out.println(tt0.increment());
46     }
47 }
```


◆ 執行結果：

```
傳入整數建構子的參數 i = 10  
類別的資料成員 i = 0  
類別的資料成員執行 this.i=i+1 的結果：11  
i-1 = 9  
this.i+1 = 12  
傳入字串建構子的 s：ok  
傳入字串建構子的 s：ok again!  
傳入整數建構子的 i = 20  
傳入字串建構子的 s：ok again!  
this.i = 20  
i++ = 21  
this.i=i++：21  
類別的資料成員 this.i++：12  
返回 this：CH06_07.ThisTest@15db9742  
CH06_07.ThisTest@15db9742
```

◆ 說明：

- 行01：
 - ◆ 定義「套件（package）」。

- 行03~行36：

- ◆ 建立「ThisTest」類別。

- ◆ 行07~行13：

- ◆ 宣告類別「建構子」`ThisTest (int i)`，有傳入一個整數的值「i」。

- 此類別「建構子」為「公開」的（省略public）。

- 行10：

- 「`This.i`」是指類別的「資料成員」`i`。

- 「i」則是指傳入此建構子的參數「i」。

- ◆ 行15~行18：

- ◆ 宣告類別「建構子」`ThisTest (String s)`，有傳入一個字串的值「s」。

- 此類別「建構子」為「公開」的（省略public）。

◆ 行20~行27：

- ◆ 宣告類別「建構子」`ThisTest (int i,String s)`，有傳入一個整數的值「i」及一個字串的值「s」。
此類別「建構子」為「公開」的（省略public）。

行22：

呼叫類別「建構子」`ThisTest (String s)`。

行23：

呼叫類別「建構子」`ThisTest (int i)`。

注意：

因為『只能引用一個「建構子」，且必須位於程式碼的第一行』，故若加入此行會發生錯誤，所以以註解方式呈現。

行25：

「`This.i`」是指類別的「資料成員」`i`。
「i」則是指傳入此建構子的參數「i」。

- ◆ 行29~行35：
 - ◆ 宣告「公開」、傳回型態為「ThisTest」的「類別方法」`increment()`，沒有傳入值，傳回「自己」。
- 行31：

「`This.i`」是指類別的「資料成員」`i`。
- 行34：

傳回值為「自己」。
- 行42：
 - ◆ 利用「`ThisTest(int i)`」的「建構子」，建立「`tt0`」物件。
- 行43：
 - ◆ 利用「`ThisTest(String s)`」的「建構子」，建立「`tt1`」物件。
- 行44：
 - ◆ 利用「`ThisTest(int i, String s)`」的「建構子」，建立「`tt2`」物件。

- 行45：
 - ◆ 顯示「tt0」物件的方法「increment()」結果。

變數的存取範圍

■ 在同一類別中，變數的存取範圍，大致上可分為四個等級。

◆ 類別等級：

- 宣告為「static」的變數，也就是「類別變數」（或稱為「靜態變數」）。
- 可以被同「類別」中的「類別方法」（或稱「靜態方法」；有宣告「static」的方法）或「方法成員」（沒有宣告「static」的方法）來存取。
- 這個等級的存取範圍最大。

◆ 物件實體等級：

- 沒有宣告「static」的變數，也就是「資料成員」。
- 只能給同類別中的「方法成員」存取。

變數的存取範圍

◆ 方法等級：

- 只能在所宣告的方法中使用。
- 在相同「類別」中的不同方法，可以使用相同的變數名稱，而不會相互影響。

◆ 區塊等級：

- 只能在所宣告出來的區塊中使用。
 - ◆ 例如：
「選擇結構」或「重複結構」中，被大括號（{ }）括起來的範圍。
- 這個等級的存取範圍最小。

物件的生命週期

■ 在Java語言中，會主動將物件視為需要被清除，主要有以下兩種情況：

- 一. 當物件變數超出其有效範圍，也就是其生命週期結束時。
- 二. 將物件變數的值設定成null或是指向其他的物件實體，使得沒有任何物件變數指向該物件實體。

例如：

```
MyObject obj1 = new MyObject;  
    ⋮  
obj1 = null;
```


綜合練習 (1)

題目：計算『面積』

要求：

- 一. 利用方法成員多載的功能，定義一個類別可以用同一個名稱Cal_area來分別計算圓形面積（計算到小數5位，第六位四捨五入）、矩形面積（整數）、立方體表面積（整數）。
- 二. 在程式碼中分別給半徑或邊長（自訂），呼叫類別物件的Cal_area方法，顯示不同面積的計算結果。

提示：

圓面積 $= \pi r^2$

矩形面積 = 長 * 寬

立方體表面積 $= ((長 * 寬) + (寬 * 高) + (長 * 高)) * 2$

綜合練習 (2)

題目：製作『猜密碼』遊戲

要求：

一. 建立一個類別：

- 在「建構子」中，
 - ⇒ 要產生 1 ~ 100 間的『亂數』。
 - ⇒ 要顯示遊戲開始時電腦的西元年、月、日、星期。
- 建立數個「資料成員」，用來存放所產生的『亂數』及『總共猜了幾次數字』，或其它相關的資料，且不得被所建立的物件直接修改。
- 建立數個「方法成員」，用來判斷『密碼』是否猜中『密碼』及計算『總共猜了幾次數字』或其它相關的處理。

二. 在程式碼中，

- 由使用者來猜『密碼』並呼叫「方法成員」完成相關判斷及計算『總共猜了幾次數字』...等。
- 須顯示總共花了多少遊戲時間。

資料來源

- 蔡文龍、何嘉益、張志成、張力元，JAVA SE 10基礎必修課，台北市，基峰資訊股份有限公司，2018年7月，出版。
- 吳燦銘、胡昭民，圖解資料結構-使用Java(第三版)，新北市，博碩文化股份有限公司，2018年5月，出版。
- Ivor Horton，Java 8 教學手冊，台北市，基峰資訊股份有限公司，2016年9月，出版。
- 李春雄，程式邏輯訓練入門與運用---使用JAVA SE 8，台北市，上奇科技股份有限公司，2016年6月，初版。
- 位元文化，Java 8視窗程式設計，台北市，松崗資產管理股份有限公司，2015年12月，出版。
- Benjamin J Evans、David Flanagan，Java 技術手冊 第六版，台北市，基峰資訊股份有限公司，2015年7月，出版。
- 蔡文龍、張志成，JAVA SE 8 基礎必修課，台北市，基峰資訊股份有限公司，2014年11月，出版。
- 陳德來，Java SE 8程式設計實例，台北市，上奇科技股份有限公司，2014年11月，初版。
- 林信良，Java SE 8 技術手冊，台北市，基峰資訊股份有限公司，2014年6月，出版。
- 何嘉益、黃世陽、李篤易、張世杰、黃鳳梅，徐政棠譯，JAVA2 程式設計從零開始--適用JDK7，台北市，上奇資訊股份有限公司，2012年5月，出版。