



## Chapter [5]

使用類別  
和物件

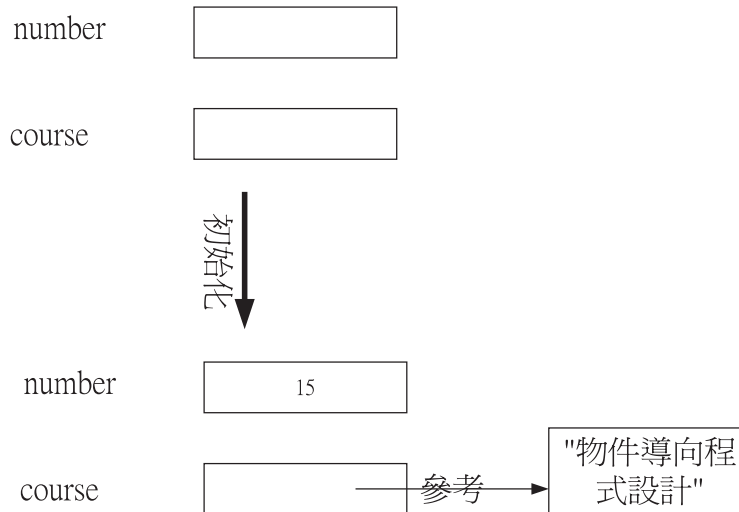
## 5-1 建立物件

我們宣告了兩個變數，一個是原始資料型態的整數變數 `number`，一個是字串參考型態的變數 `course`。整數資料型態 `int` 是屬於原始資料型態，但字串變數是屬於參考型態，字串變數會去參考字串物件。一個物件變數它儲存的是物件的位置，而不是物件本身。

```
int number ;  
String course
```

一開始宣告 `number` 變數和 `course` 變數，這時後在記憶體配置空間給它們。然後我們初始化這些變數，並且將整數 15 給 `number` 變數；我們將參考 ( 字串在記憶體的位置 ) 給 `course` 變數。`number` 變數所儲存的是 15 的值，而 `course` 變數所儲存的是字串“物件導向程式設計”的參考位置。`course` 參考到物件字串。

```
Number=15;  
course=new String( "物件導向程式設計" );
```



因為 Java 物件導向的參考就像是 C 語言的指標一樣，它們在記憶體的空间都是放置物件的位置。當物件被初始化時，我們使用物件接逗點再接變數或方法來存取該物件的成員。我們使用逗點運算元來呼叫 `println()` 函數來輸出字串，例如我們使用 `System.out.println()`。

```
System.out.println();
```

我們建立參考和新增物件的方法，我們使用 `new` 這個關鍵字。

```
course=new String(“物件導向程式設計”);
```

因為物件變數放置的是參考位置，因此我們要小心的使用和管理物件。

這是兩個整數變數，我們首先宣告整數變數 `number1` 和 `number2`，並且初始化將 15 的數值給 `number1`，將 25 的數值給 `number2`。這是原始資料型態的初始化。

```
int number1,number2  
number1=15;  
number2=25;
```

number1

15

number2

25

這時我們將 `number2` 變數的數值 25 給 `number1`，這時 `number1` 和 `number2` 的數值就變成 25 了，但是存放 25 數值的記憶體所在位置都是不相同的。

```
number1=number2 ;
```

number1

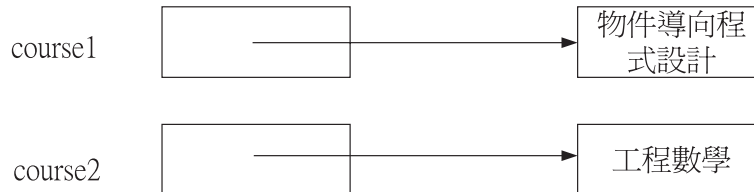
25

number2

25

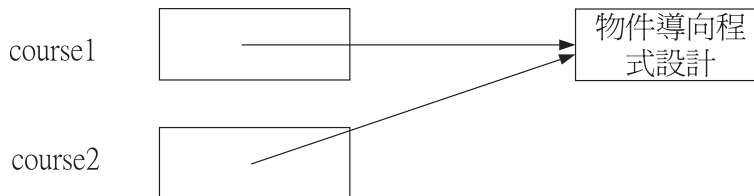
我們建立字串物件 `course1` 和 `course2`，並且 `course1` 變數參考到”物件導向程式設計”，而 `course2` 變數參考到”工程數學”。這是物件參考型態的初始化。

```
String course1=" 物件導向程式設計" ;  
String course2=" 工程數學" ;
```



現在我們將 `course1` 變數的資料 ( 指向”物件導向程式設計”的參考 ) 給 `course2` 變數，這就像整數型態變數的拷背一樣，但是物件變數放置的是物件的參考位置。在 `course1` 的資料 ( 參考 ) 分配給 `course2` 時，`course2` 和 `course1` 就會參考到相同的物件 ( “物件導向程式設計” )。

```
course2=course1
```



`course1` 和 `course2` 稱為 *aliases*( 不同的變數名稱卻參考相同物件 )。當我們使用其中一個參考來將物件的資料改變時，另外一個參考的物件也會同時改變，因為它們都參考了相同物件。Java 使用參考變數來使用物件。當所有指向該物件的參考都被刪除時，這時這個物件就沒有在程式裏有任何作用，程式將無法使用該物件的變數或方法，這個物件被稱為 *garbage*( 消失 )，也就是當作垃圾被移除了。

## 5-2 字串類別

`String` 類別有許多有用的函數，這些都是在 `Java.lang.String` 類別函式庫裏面。

<code>string(String str)</code>	為建構子，使用輸入的 <code>str</code> 來建立一個新的字串物件。
<code>char charAt(int index)</code>	回傳指定索引的字元。
<code>int compareTo(String str)</code>	假如比較的字串先於這個字串的字母則回傳正整數；如果相等則回傳負數；如果比較的字串後於這個字串的字母則回傳負整數。
<code>String concat(String str)</code>	回傳由這兩個字串組合而成的新字串。
<code>boolean equals(String str)</code>	假如這兩個字串的字元都相同(包含字元大小寫)則回傳 <code>true</code> ，否則回傳 <code>false</code> 。
<code>boolean equalsIgnoreCase(String str)</code>	假如這兩個字串的字元都相同(忽略字元大小寫)則回傳 <code>true</code> ，否則回傳 <code>false</code> 。
<code>int length()</code>	回傳字串中字元的個數。
<code>String replace(char 舊字串,char 新字串)</code>	回傳由”新字串”取代”舊字串”的字串。
<code>String substring(int offset,int endIndex)</code>	回傳刪去部份字串後的字串， <code>offset</code> 是起始，到 <code>endIndex-1</code> 的字串。
<code>String toLowerCase()</code>	回傳轉成小寫的字串。
<code>String toUpperCase()</code>	回傳轉成大寫的字串。



### 範例 String\_method.java

第三行宣告字串物件變數`sentence`，並且將字串” I am a good student” 給物件變數`sentence`。

第四行宣告字串物件變數`chinese_words`，並且將中文字串” 中文字” 給物件變數`chinese_words`。

第六行列印出`sentence`物件的字串。

第七行使用`sentence.length()`來列印出`sentence`字串物件的長度。

第八行使用 `chinese_words.length()` 來列印出 `chinese_words` 字串物件的長度。

第九行將字串物件 `sentence` 的字串轉成小寫。

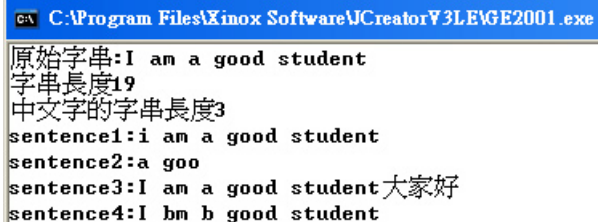
第十行 `sentence.substring(5,10)` 來將 `sentence` 字串物件截取第五到第十字元。

第十一行使用 `sentence.concat(“大家好”)`，來將字串 “I am a good student” 和中文字串 “大家好” 連接。

第十二行將 `sentence` 字串物件中的 `a` 字元取代成 `b` 字元。

```
1 public class String_method{
2     public static void main(String[] args){
3         String sentence="I am a good student";
4         String chinese_words=" 中文字";
5         String sentence1,sentence2,sentence3,sentence4;
6         System.out.println("原始字串:"+sentence);
7         System.out.println("字串長度"+sentence.length());
8         System.out.println("中文字的字串長度"+chinese_words.length());
9         sentence1=sentence.toLowerCase();
10        sentence2=sentence.substring(5,10);
11        sentence3=sentence.concat("大家好");
12        sentence4=sentence.replace('a','b');
13        System.out.println("sentence1:"+sentence1);
14        System.out.println("sentence2:"+sentence2);
15        System.out.println("sentence3:"+sentence3);
16        System.out.println("sentence4:"+sentence4);
17    }
18 }
```

這是執行的結果。因為 “中文字” 字串中有三個中文字，所以顯示字串長度為3。



```
C:\Program Files\Xinox Software\JCreator\Y3LE\GE2001.exe
原始字串:I am a good student
字串長度19
中文字的字串長度3
sentence1:i am a good student
sentence2:a goo
sentence3:I am a good student大家好
sentence4:I bm b good student
```

## 5-3 套件

套件就是群組和命名一群的類別，我們可以在程式中來使用它，而不用將它們放在程式同一個目錄中。

一個套件就是將一群的類別，群組放到同一個目錄中，然後再給這個目錄一個套件名稱。在套件中的類別是放置在不同的檔案中，而檔案的名稱和類別的名稱相同。每一個檔案在起始的地方都有下列的敘述。

**Package** 套件名稱；

它一定要放在檔案的最前面。套件名稱通常都是小寫的字母，然後通常接標點符號，當作連接目錄。例如 `example.ch01` 就是套件名稱，然後每一個在這個套件中的檔案起始的地方都有下列的敘述。

**Package** `example.ch01`；

每一個程式或類別都可以使用套件裏的所有類別，它們只要加入 `import` 敘述在檔案的最前面就可以了。就算程式和套件裏面的類別不在同一個目錄也可以使用。假如我們希望使用在 `example.ch01` 套件裏面的類別，我們可以使用下列的敘述。

**import** `example.ch01.*`；

星號 `*` 代表我們輸入 `example.ch01` 套件中的所有類別。

### 5-3-1 套件名稱和目錄

套件名稱不是識別子。套件名稱告訴編譯器哪裏可以找到這個套件中的類別。套件的名稱告訴編譯器套件中類別的路徑。為了找到套件中的目錄，Java 需要作兩件事，套件的名稱和目錄的列表在我們類別路徑變數的值。類別路徑變數的值告訴 Java 從哪裏開始搜尋套件。類別路徑不是 Java 的變數，它是我們作業系統中目錄的路徑名稱列表。當 Java 搜尋套件時，它開始在類別路徑基礎目錄中搜尋。

套件的名稱指定了目錄的相對路徑，而這目錄包含這套件類別。它是相對路徑，因為它假設我們在類別路徑基礎目錄開始，然後順從這套件名稱的子目錄。例如假如下面是類別路徑基礎 CLASSPATH 目錄。

**\example\ch01**

假設我們的套件類別在 \example\ch01\client\run 目錄，這個套件一定要被命名為

**client.run**

我們可以指定類別路徑基礎目錄，我們使用設定作業系統的類別變數。類別路徑就是 CLASSPATH。

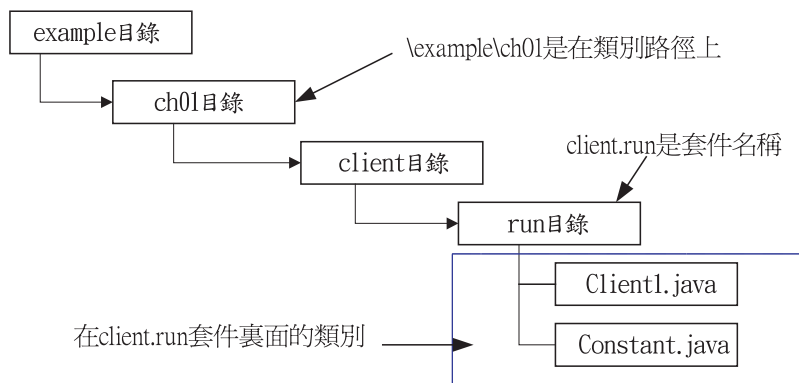
假如我們使用 Unix 作業系統，我們使用 set 來設定 CLASSPATH 類別路徑，我們會使用 export 來輸出路徑。

```
set CLASSPATH=/example/ch01;  
export CLASSPATH
```

我們在 Windows 上的命令模式下也可以暫時的指定類別路徑。

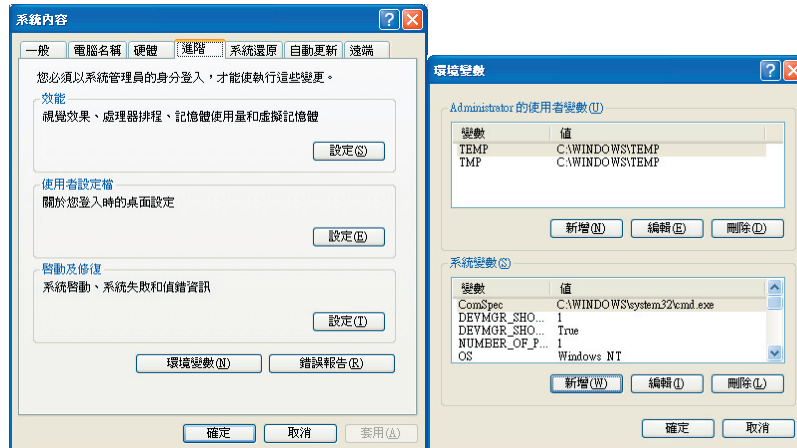
```
Set CLASSPATH=%CLASSPATH%;C:\example\ch01
```

假如 \example\ch01 是在類別路徑上。client.run 就是套件的名稱。

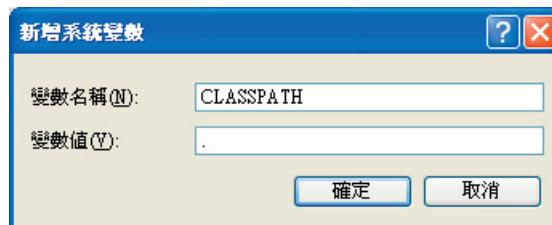




假如我們使用 Windows 作業系統，我們可以設定類別路徑變數，使用設定控制台→效能及維護→系統→進階→環境變數，來新增 CLASSPATH 的環境變數。這將會編譯 MyClass.java，並且改寫 CLASSPATH 的類別路徑設定。我們選取環境變數。然後我們選取新增系統變數。



我們增加一個 CLASSPATH 變數，並設定變數值為'.'。逗點代表目前目錄。



-classpath 後面的逗點指的是目前的目錄，這樣我們的類別就可以存取目前目錄的類別。每一組路徑都使用分號來格開。我們也可以編譯在 C:\example\ch01 目錄下的檔案。我們這樣子就可以編譯 MyClass.java 的檔案了。當我們編譯類別時，我們可以指定類別路徑。我們使用 -classpath 的參數。

```
javac -classpath .;C:\example\ch01 MyClass.java
```

當我們執行已經編譯過的程式，我們也使用 -classpath 類別路徑。

```
java -classpath .; C:\example\ch01 MyClass
```

CLASSPATH 類別路徑變數預設是設定在本目錄'.'，預設是可以省略環境變數。Java 把 class 分成三類，第一類是 Bootstrap 類別，第二類是 Extension 延伸類別，第三類是 User 使用者自訂類別。第一類的 Bootstrap 類別是放置在 \Java\jdk1.5.0\jre\lib 目錄下的 rt.jar 中，它們是 Java2 平台所內建的類別函式庫。Extension 延伸類別也是屬於 Java2 平台所內建的類別函式庫，它們是放在 \Java\jdk1.5.0\jre\lib\ext 目錄下。Bootstrap 和 Extension 類別不用設定路徑，Java 都能找到，但，我們使用者自訂的類別就要設定 CLASSPATH 類別路徑，否則，Java 會找不到。

### 5-3-2 套件和import

當我們在程式裏使用 Java 標準資料庫的類別時，我們要指定程式讀取這個類別的路徑，包含了該類別的套件名稱。import java.util.\* 指的是輸入 java.util 套件中所有的類別，\* 星號指的是全部類別。如果我們輸入 java.util 套件中兩個類別時，也是使用星號 \*，如果只輸入一個類別，則指定套件和該類別名稱的路徑。

```
import java.util.*
```

### 5-3-3 實作套件

#### 範例 Class0.java

假如我們有兩個檔案，一個是Class0.java一個是Class3.java。Class3.java是Class0.java的子類別，但是，Class0是在First目錄下，而Class3是在Second目錄下。

我們將已經編譯好的Class0.class放到First目錄下，因此第一行要宣告First套件。

```
1 package First;
2 public class Class0{
3     public int x=1;
4     protected int y=2;
5     int z=3;
6     private int u=4;
7     protected void fun(){
8         System.out.println("Class0的函數");
9     }
10 }
```

### 範例 Class3.java

我們將已經編譯好的Class3.class放到Second目錄下，因此第一行要宣告Second套件。

第二行輸入First.Class0類別。因為Class3.java會用到Class0.class。

```
1 package Second;
2 import First.Class0;
3 public class Class3 extends Class0{
4     public static void main(String[] args){
5         Class3 c=new Class3();
6         System.out.println(c.x);
7         System.out.println(c.y);
8         //System.out.println(c.z);
9         //System.out.println(c.u);
10        c.fun();
11    }
12 }
```

在 F:\java2\6\example\1-3-2 目錄下放置兩個目錄，分別是 First 目錄和 Second 目錄。然後我們將 Class0.java 編譯好的檔案放到 First 目錄中。我們先編譯 Class0.java 的檔案，我們輸入完整路徑。編譯好的 Class0.class 是放置到 First 目錄中。

```
C:\>javac F:\java2\6\example\1-3-2\Class0.java
```

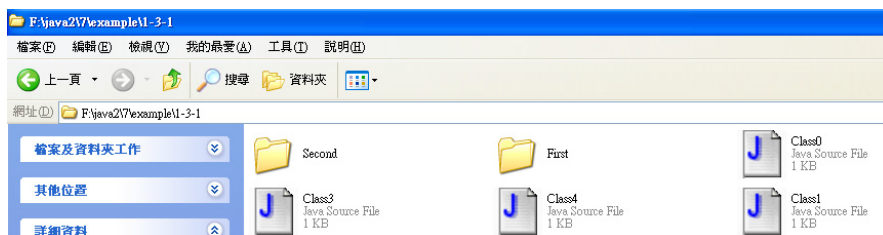
在這裏 cp 指的是 classpath 類別路徑參數。我們設定我們所在的目錄 F:\java2\6\example\1-3-2 為我們的類別路徑，這樣在編譯時，Java 就能找到 Class0.class 的檔案，然後作輸入的動作。Class0.class 是在 F:\java2\6\example\1-3-2 目錄下的 First 目錄套件中。

```
C:\>javac -cp F:\java2\6\example\1-3-2; F:\java2\6\example\1-3-2\Class3.java
```

我們也可以先使用設定 CLASSPATH 類別路徑，這樣在編譯時，Java 會自動找到要輸入的類別所在的目錄。

```
C:\>set CLASSPATH=F:\java2\6\example\1-3-2;
```

```
C:\>javac F:\java2\6\example\1-3-2\Class3.java
```



我們使用 java 來執行。Cp 是 classpath 指定類別路徑，在這裏是 F:\java2\6\example\1-3-2 目錄。這樣 Java 就可以根據類別路徑找到 First.Class0.class 的類別和要執行的 Second.Class3 的類別。Second.Class3 為套件名稱加上逗點再加上類別名稱，這樣完整指出類別所在的地方。

```
C:\>java -cp F:\java2\6\example\1-3-2; Second.Class3
```

這是執行的情況。

```
C:\>java -cp F:\java2\6\example\1-3-2; Second.Class3
1
2
Class0的函數
```

如果已經設定好 CLASSPATH 環境變數指到 F:\java2\6\example\1-3-2 目錄，那麼我們就不用設定 cp 了。

```
C:\>set CLASSPATH=F:\java2\6\example\1-3-2;
C:\>java Second.Class3
1
2
Class0的函數
```

### 5-3-3 javadoc

javadoc 程式將抽取我們類別的名稱和所有 public 函數的名稱、公眾實體變數、公眾靜態類變數和特定的註解。

我們可以執行 javadoc 然後接上要顯示資訊的類別，在這邊是 Class1.java 類別，它會產生許多資訊的網頁。

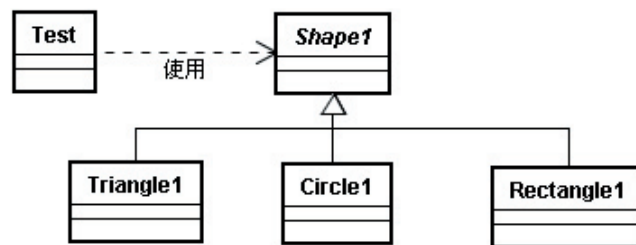
```
javadoc Class1.java
```

version 為顯示版本選項。

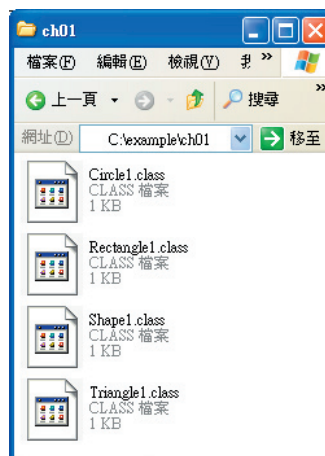
```
javadoc -version Class1.java
```

### 5-3-4 實作套件2

我們有四個檔案，分別是 Shape1.java、Triangle1.java、Rectangle1.java 和 Square1.java 這四個檔案，他們這四個檔案都是放置到 example.ch01 套件中，也就是在 / example/ch01 目錄下。因此在每一個類別前我們都要加上 package example.ch01。Shape1 為形狀抽象類別，Triangle1 三角形、Circle1 圓形和 Rectangle1 矩形類別繼承自 Shape1 抽象類別。Test 使用 Shape1 形狀抽象類別。



Circle1.class、Rectangle1.class、Shape1.class 和 Triangle1.class 類別都是放在 example.ch01 套件與目錄中。



## 範例

## Test.java

我們在Test.java中要加上下列的敘述，來輸入套件內的類別。

第一行輸入import example.ch01.Shape1;，因為Shape1類別在example.ch01的套件中。

第二行輸入import example.ch01.Circle1;，因為Circle1類別在example.ch01的套件中。

第三行輸入import example.ch01.Rectangle1;，因為Rectangle1類別在example.ch01的套件中。

第四行輸入import example.ch01.Triangle1;，因為Triangle1類別在example.ch01的套件中。

第九行是宣告型態為Shape1形狀的變數s。

第十行是宣告並建立Circle1圓形的物件c。

第十一行是宣告並建立Rectangle1矩形的物件r。

第十二行是宣告並建立Triangle1三角形的物件t。

第十三行到第十九行是多型，向上型別轉換。將圓形物件c轉換成型態為s，並且第十四行使用s.area()來顯示圓的面積。

第十六行將矩形物件r轉換成型態為s，並且第十七行使用s.area()來顯示矩形的面積。

第十九行將三角形物件t轉換成型態s，並且第二十行使用s.area()來顯示三角形的面積。

```

1 import example.ch01.Shape1;
2 import example.ch01.Circle1;
3 import example.ch01.Rectangle1;
4 import example.ch01.Triangle1;
5 public class Test
6 {
7     public static void main(String[] args)
8     {
9         Shape1 s;
10        Circle1 c = new Circle1(10);
11        Rectangle1 r=new Rectangle1(5,2);
12        Triangle1 t=new Triangle1(2,2);
13        s = c;
14        System.out.print(s.area());
15        System.out.println();
16        s = r;
17        System.out.print(s.area());
18        System.out.println();
19        s= t;
20        System.out.print(s.area());
21        System.out.println();
22    }
23 }

```

這是執行的情況。

```

C:\C:\PROGRAMS\1\XINOS\1\WCREAT~1\GE2001.exe
314.1592653589793
10.0
2.0

```

### 範例

### Shape1.java

我們在第一行就要套件敘述。前面只可以加上註解。

第一行是說Shape1抽象類別是在example.ch01套件中。

第二行宣告Shape1為抽象abstract類別。

第三行到第四行為成員屬性。

第五行為抽象方法area()。我們只有宣告抽象方法，但沒有實作。

```

1 package example.ch01;
2 public abstract class Shape1{
3     public double x;
4     public double y;
5     public abstract double area();
6 }

```

**範例** Circle1.java

我們在第一行就要套件敘述。前面只可以加上註解。

第一行是說Circle1類別是在example.ch01套件中。

第二行宣告Circle1類別，它繼承了抽象類別Shape1。

第四行到第六行為Circle1建構子。

第七行到第九行為實作抽象類別Shape1的抽象方法area()。

```
1 package example.ch01;
2 public class Circle1 extends Shape1{
3     public double r;
4     public Circle1(double r){
5         this.r=r;
6     }
7     public double area(){
8         return r*r*Math.PI;
9     }
10 }
```

**範例** Rectangle1.java

我們在第一行就要套件敘述。前面只可以加上註解。

第一行是說Rectangle1類別是在example.ch01套件中。

第二行宣告Rectangle1類別，它繼承了抽象類別Shape1。

第五行到第九行為Rectangle1建構子。

第十行到第十二行為實作抽象類別Shape1的抽象方法area()。每一個子類別實作抽象類別Shape1函數area()的內容都不同。

```
1 package example.ch01;
2 public class Rectangle1 extends Shape1{
3     public double width;
4     public double height;
5     public Rectangle1(double width,double height)
6     {
7         this.width = width;
8         this.height = height;
9     }
10    public double area(){
11        return width*height;
12    }
13 }
```



### 範例 Triangle1.java

我們在第一行就要套件敘述。前面只可以加上註解。

第一行是說Triangle1類別是在example.ch01套件中。

第二行宣告Triangle1類別，它繼承了抽象類別Shape1。

第六行到第十行為Triangle1建構子。

第十一行到第十四行為實作抽象類別Shape1的抽象方法area()。

```
1 package example.ch01;
2 public class Triangle1 extends Shape1
3 {
4     private double height;
5     private double bottom;
6     public Triangle1(double height, double bottom)
7     {
8         this.height = height;
9         this.bottom = bottom;
10    }
11    public double area()
12    {
13        return(height*bottom/2);
14    }
15 }
```

### 5-3-5 Java的標準函式庫

在 Java 語言中，有內件的標準類別函式庫。Java 標準類別資料庫為 Java 程式開發的內建類別資料庫。Java 內建類別資料庫通常都內建在 Java 的編譯器上或 Java 的開發環境上。我們程式設計師經常會用到 Java 標準類別資料庫，在程式設計時，我們只要呼叫它，就可以完成特定的功能，而不用再去撰寫新的類別或方法。字串類別就是 Java 標準類別資料庫，我們直接使用它，只要用物件，後面再接方法就可以用它了。

Java 類別資料庫是由幾個相關的套件 (類別群) 所組成，我們稱為應用程式介面或者是 Java API(Application programming interfaces)。例如 Java Swing API 是定義特別的圖形使用者界面的圖形元件。Java 標準類別函式庫通常都會組成套件。每一個類別都屬於某一個特別套件。例如字串類別 String 和 System 類別是屬於 java.lang 套件。

這是 Java 所提供的標準類別函式庫，Java 類別資料庫是由幾個相關的套件（類別群）所組成。

套件名稱	提供功能
java.applet	建立applet網路程式。
java.awt	繪製圖形和建立使用者圖形介面：AWT提供抽象視窗工具。
java.beans	定義能被簡單組合應用的軟體元件。
java.io	提供廣範的輸入輸出功能。
java.lang	一般支援，預設是自動import到java程式中。
java.math	執行任何精密的數學計算。
java.net	網路溝通。
java.rmi	呼叫遠端的方法(remote Method Invocation)。
java.security	加強安全。
java.sql	存取SQL資料庫。
java.text	文字格式輸出。
java.util	一般的工具。
javax.swing	延伸AWT建立使用者圖形介面元件。
javax.xml.parsers	處理XML元件。

java.lang 套件的類別是自動的在所有程式中被輸入，因為它們是 Java 語言的基礎和延伸。因此任何在 java.lang 套件中的類別都可以任意的使用，而不用事先 import 輸入進來程式中。例如我們時常直接就使用 String 物件和 System 物件，而沒有先 import java.lang.\* 套件，因為預設是已經 import 進來了。

```
import java.lang.*;
```

## 5-4 數學Math類別

Math 類別裏面的所有函數都是靜態 static 函數，因此呼叫這些數學函數時，只要前面加上 Math 類別名稱即可。

數學函數	說 明
Static int abs(int num)	回傳參數num的絕對值。
Static double acos(double num)	回傳arc cosine的值。
Static double asin(double num)	回傳arc sine的值。
Static double atan(double num)	回傳arc tangent的值。
Static double cos(double 角度)	回傳cosine的角度，以弧度表示。
Static double ceil(double num)	回傳num無條件進位的值。
Static double exp(double power)	回傳e的power次方的值。
Static double floor(double num)	回傳num無條件捨去的值。
Static double pow(double num,double power)	回傳num的power次方的值。
Static double random()	回傳從0到1.0(不包含1)的值。
Static double sqrt(double num)	回傳平方根的值。



### math\_quadratic.java

這是二次方程式，我們要求解x的值。

$$a x^2 + b x + c$$

這是解二次方程式x的解。我們只要得到二次方程式a、b和c的值，就可以得到x的解。

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

第一行輸入了java.util套件的Scanner類別，這樣就可以在程式中使用該類別。

第六行我們新增了Scanner類別的物件input，這樣我們就可以將資料輸入到Java程式中。System.in為我們在鍵盤上輸入之串流資料。

第八行得到第一個a的係數，使用input.nextInt()函數來得到輸入的值。

第十行得到第二個b的係數。

第十三行得到b平方減去4\*a\*c得值，我們使用Math.pow(b,2)。pow()函數為Math類別的靜態函數，我們在Math類別後面接逗點再接pow()函數。

第十四行得到我們第一個root1根的值。

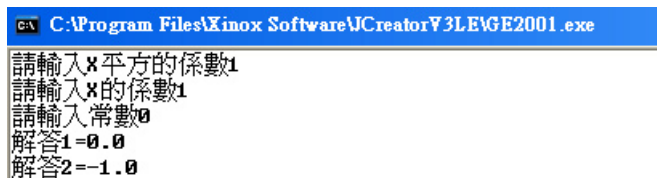
第十五行得到第二個root2根的值。

```

1 import java.util.Scanner;
2 public class math_quadratic{
3     public static void main(String[] args){
4         int a,b,c;
5         double value,root1,root2;
6         Scanner input=new Scanner (System.in);
7         System.out.print("請輸入X平方的係數");
8         a=input.nextInt();
9         System.out.print("請輸入X的係數");
10        b=input.nextInt();
11        System.out.print("請輸入常數");
12        c=input.nextInt();
13        value=Math.pow(b,2)-(4*a*c);
14        root1=(-1*b+Math.sqrt(value))/(2*a);
15        root2=(-1*b-Math.sqrt(value))/(2*a);
16        System.out.println("解答1="+root1);
17        System.out.println("解答2="+root2);
18    }
19 }

```

我們將a=1、b=1和c=0帶入，就可以得到x=0和-1的值。



```

C:\Program Files\Xinox Software\UCreator\3LEIGE2001.exe
請輸入x平方的係數1
請輸入x的係數1
請輸入常數0
解答1=0.0
解答2=-1.0

```

## 5-5 列舉型態

Java 語言提供列舉型態，列舉型態就是當宣告時就可以使用的變數型態。列舉型態建立每一個變數可能的數值。這些數值都是識別名稱，可以是任何的資料。例如我們可以定義季節的列舉型態是春、夏、秋和冬。

```
enum Season {spring,summer,fall,winter}
```

當我們定義好列舉型態時，我們可以宣告變數的型態。

```
Season x;
```

這個變數 x 的數值，就限定在下列的列舉型態 spring、summer、fall 和 winter，如果使用其它的數值，java 在編譯時就會發生錯誤。我們可以使用下列的方式來存取型態名稱的資料。

```
X=Season.winter;
```

任何一個被初始化的變數擁有 Grade 列舉型態，則它就保證有 A,B,C,D,E 其中之一的成績。任何一個列舉型態的數值都是排序編號的數值。第一個排序編號的數值為 0，第二個排序編號的數值為 1，這樣依此類推。列舉型態為一種特別的類別，列舉型態的變數為物件變數。

```
enum Grade{A,B,C,D,E}
```



### enum\_grade.java

任何一個被初始化的變數擁有 Grade 列舉型態，則它就保證有 A,B,C,D,E 其中之一的成績。任何一個列舉型態的數值都是排序編號的數值。第一個排序編號的數值為 0，第二個排序編號的數值為 1，這樣依此類推。ordinal() 函數為列舉型態的函數，它會回傳該變數的排序編號。

第二行我們宣告了列舉型態成績 Grade，它有五種列舉型態的數值，分別是 A、B、C、D 和 E。

第四行我們宣告列舉型態成績的物件變數student1、student2和student3。

第五行我們將Grade.A列舉型態的數值給student1物件變數。

第七行我們將student1變數參考到student3變數，student1和student3變數會參考到相同物件資料。

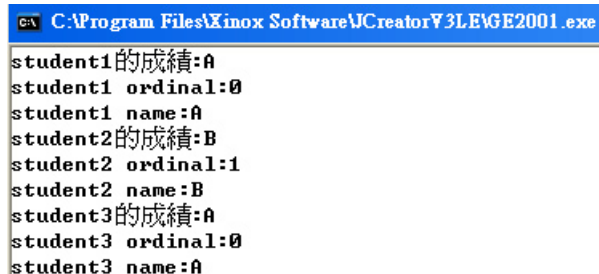
第九行student1.ordinal()會顯示student1的排序編號，因為它是第一位，所以顯示0。

```

1 public class enum_grade{
2     enum Grade {A, B, C, D, E}
3     public static void main(String[] args){
4         Grade student1, student2, student3;
5         student1=Grade.A;
6         student2=Grade.B;
7         student3=student1;
8         System.out.println("student1的成績:"+student1);
9         System.out.println("student1 ordinal:"+student1.ordinal());
10        System.out.println("student1 name:"+student1.name());
11        System.out.println("student2的成績:"+student2);
12        System.out.println("student2 ordinal:"+student2.ordinal());
13        System.out.println("student2 name:"+student2.name());
14        System.out.println("student3的成績:"+student3);
15        System.out.println("student3 ordinal:"+student3.ordinal());
16        System.out.println("student3 name:"+student3.name());
17    }
18 }

```

這是顯示的情況。



```

C:\Program Files\Xinox Software\WCreator\3LEVGE2001.exe
student1的成績:A
student1 ordinal:0
student1 name:A
student2的成績:B
student2 ordinal:1
student2 name:B
student3的成績:A
student3 ordinal:0
student3 name:A

```

## 5-6 元件和容器

容器是特殊的使用者圖形化介面，它可以組織其它的元件。一個框 `frame` 可以當作分離的視窗 `window`，但是面版 `panel` 只能是其它元件的一部份。一個 GUI (圖形使用者介面) 元件是一個物件，它包含了用來顯示資訊的螢幕元件，螢幕 `screen` 元件也允許我們和程式交談。Java 相關的圖形化介面類別和 Java 元件主要是定義在 `java.awt` 套件和 `javax.swing` 套件中，而抽象視窗管理工具 (AWT) 是原始的 Java 圖形使用者介面。容器是一個特殊的元件型態，它是用來組織和握有其它的元件。`frame` 框和 `panel` 面版是 Java 的容器 `container`，Swing 則是新增的視窗管理工具。

一個 `frame` 框是一種容器，它用來顯示以圖形化為介面的 Java 應用程式。一個 `frame` 包含了在角落的開關按鈕和放大縮小按鈕。`frame` 框是由 `JFrame` 類別所定義。一個 `panel` 面版也是一個容器，但是 `panel` 面版卻不能顯示自己。一個 `panel` 面版必需增加到其它的容器才能被顯示。`panel` 面版是用來在圖形化介面中來組織其它的元件。`panel` 面版是定義 `Jpanel` 類別。

容器可分為重量級容器與輕量級容器。重量級容器是由作業系統來控制程式的運作，輕量級容器是由 Java 程式自行控制。`frame` 框是屬於重量級容器，而 `panel` 面版是屬於輕量級容器。一般來說，重量級元件比輕量級元件還要複雜。一般，一個框 `frame` 有許多的面版。所有 `java` 界面的可顯示元件都是在框 `frame` 面版的 `panel` 上顯示。

一般我們先建立 `frame` 框，來顯示 `java` 圖形化介面的程式。`Label` 標籤元件是用來顯示圖形化界面的文字。一個標籤可以用來顯示圖形、文字或主題，而且標籤在我們的圖形化界面中廣範的被使用。

一個框 `frame` 可以當作一個分離的視窗來被顯示，而面版 `panel` 只能當作其它容器的部份來被顯示。

Layout manager 物件管理每一個容器，Layout manager 管理每一個容器如何佈置。

**範例** Java\_image.java

我們在這裏用到了frame框、面版panel和標籤label。它會顯示新的視窗，並且顯示一段文字。在這裏，panel面版組織了標籤label，而面版panel則在框上作顯示。

第一行import輸入原始圖形化介面發展的awt抽象視窗工具(Abstract Windowing Toolkit)套件。

第二行import輸入swing套件，這是用來作使用者圖形化界面發展的套件。

第五行新增了JFrame框，並且將字串”第一個視窗”當參數帶入，當frame框的建構子，文字會顯示在frame框的標題上。

第六行frame框的函數setDefaultCloseOperation()會設定當按下框frame的關閉鈕時，會發生什麼事，EXIT\_ON\_CLOSE常數是會關閉這個框frame。

第七行新增panel面版。

第九行新增標籤label，並且建立標籤的建構子，將字串”暢銷書籍帶入”。

第十行設定面版的顏色為紅色red，使用panel框的函數setBackground()。

第十一行設定frame框的大小，使用setPreferredSize()函數，setPreferredSize()函數使用Dimension物件當作參數，用來指定框frame的寬和高。

第十二行panel面版容器使用add()方法將label標籤元件加入到panel面版容器中。

第十三行frame框的內容方格使用getContentPane()方法將面版panel加入。

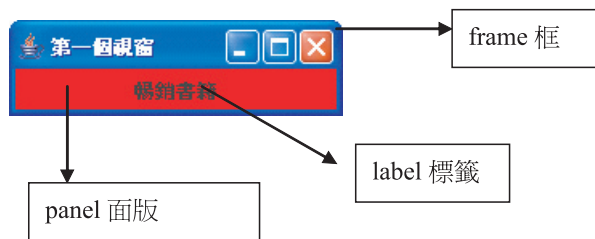
第十四行frame框的方法pack()設定frame框的大小來容納面版panel。

第十五行frame框的方法setVisible(true)來顯示frame框。



```
1 import java.awt.*;
2 import javax.swing.*;
3 public class Java_image{
4     public static void main(String[] args){
5         JFrame frame=new JFrame("第一個視窗");
6         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
7         JPanel panel=new JPanel();
8         JLabel label1;
9         label1=new JLabel("暢銷書籍");
10        panel.setBackground(Color.red);
11        panel.setPreferredSize(new Dimension(200, 25));
12        panel.add(label1);
13        frame.getContentPane().add(panel);
14        frame.pack();
15        frame.setVisible(true);
16    }
17 }
```

這是執行的情況。





## 5-7 包裹類別wrapper

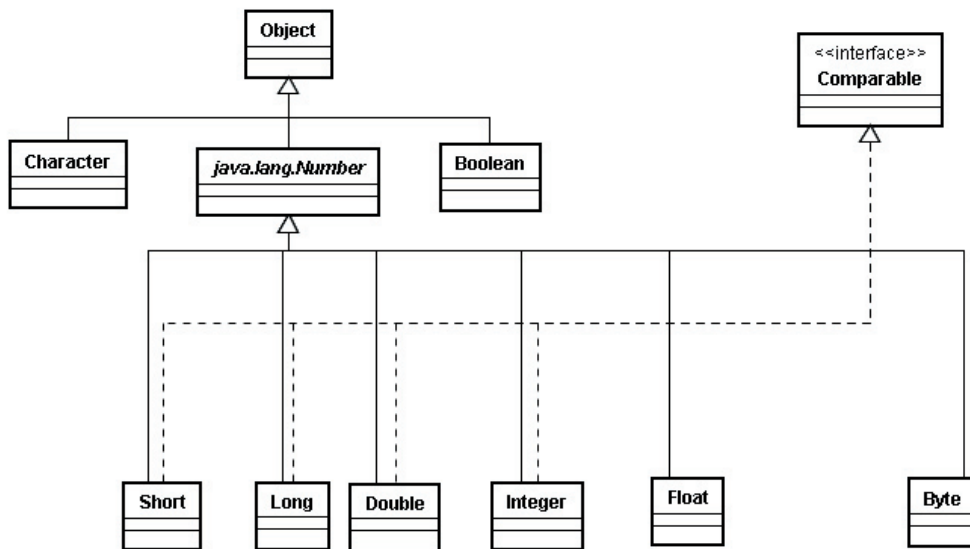
Java 的資料有分為原始資料型態和參考型態資料。我們可以建立一個物件來當作容器來包住各種型態的物件。我們使用物件包裹來包住其它的原始資料。一個包裹類別 wrapper 代表了一種特別的原始型態。例如 Integer 整數類別代表了簡單的整數。從整數類別建立的物件，包含了單一的 int 原始資料型態數值。我們可以宣告一個雙精度浮點數的物件 `do1`，它包含了雙精度浮點數 10.1 的數值。

```
Double do1=new Double(10.1);
```

所有的 wrapper 包裹類別都定義在 `java.lang` 套件中。這是原始資料型態和它相對應的包裹類別。一個包裹類別允許原始資料型態資料像物件一樣被操作。

原始型態	包裹類別
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
char	Character
boolean	Boolean
void	Void

每一個包裹物件都繼承自 `Object` 物件。`Short`、`Long`、`Double`、`Integer` 包裹物件實作 `Comparable` 介面，因此它們有 `compareTo()` 比較大小的方法。`Short`、`Long`、`Double`、`Integer`、`Float` 和 `Byte` 包裹物件繼承自 `java.lang.Number` 數值物件。



## 範例

## WrapperSort.java

第三行和第四行新增整數包裹物件，並且將這些物件裝到整數物件陣列 `intArray` 中。

第五行和第六行新增雙精度浮點數包裹物件，並且將這些物件裝到雙精度浮點數物件陣列 `doubleArray` 中。

第七行和第八行新增字元包裹物件，並且將這些物件裝到字元物件陣列 `doubleArray` 中。

第九到第十一行分別呼叫 `sort()` 函數，並且將陣列物件帶入，來排序各個陣列物件。

第十九行到第三十六行為 `sort()` 函數的定義，它會將輸入的陣列物件作排序。

第二十行我們宣告 `currentMax` 為物件 `Object`。

第二十六行因為 `Integer` 整數物件、`Double` 雙精度浮點數物件和 `Character` 字元物件都是實作 `Comparable` 介面，所以可以將 `currentMax` 轉成 `Comparable`，然後再使用 `compareTo()` 方法來作比較大小。

第三十七行到第四十一行的 `printList()` 函數會列印出排序後的陣列。

```

1 public class WrapperSort {
2     public static void main(String[] args) {
3         Integer[] intArray = {new Integer(1), new Integer(3),
4             new Integer(2), new Integer(5), new Integer(4)};
5         Double[] doubleArray = {new Double(5.1), new Double(2.1), new Double(1.1),
6             new Double(3.1), new Double(4.1)};
7         Character[] charArray = {new Character('a'), new Character('e'),
8             new Character('d'), new Character('b'), new Character('c')};
9         sort(intArray);
10        sort(doubleArray);
11        sort(charArray);
12        System.out.print(" 排序整數物件: ");
13        printList(intArray);
14        System.out.print(" 排序雙精度浮點數物件: ");
15        printList(doubleArray);
16        System.out.print(" 排序字元物件: ");
17        printList(charArray);
18    }
19    public static void sort(Object[] list) {
20        Object currentMax;
21        int currentMaxIndex;
22        for (int i = list.length - 1; i >= 1; i--) {
23            currentMax = list[i];
24            currentMaxIndex = i;
25            for (int j = i - 1; j >= 0; j--) {
26                if (((Comparable)currentMax).compareTo(list[j]) < 0) {
27                    currentMax = list[j];
28                    currentMaxIndex = j;
29                }
30            }
31            if (currentMaxIndex != i) {
32                list[currentMaxIndex] = list[i];
33                list[i] = currentMax;
34            }
35        }
36    }
37    public static void printList(Object[] list) {
38        for (int i = 0; i < list.length; i++)
39            System.out.print(list[i] + " ");
40        System.out.println();
41    }
42 }

```

這是執行排序陣列物件的情況。

```

C:\Program Files\Xinox Software\UCreatorY3LEVGE2001.exe
排序整數物件: 1 2 3 4 5
排序雙精度浮點數物件: 1.1 2.1 3.1 4.1 5.1
排序字元物件: a b c d e

```

## 5-8 Autoboxing

Autoboxing 是自動的在原始資料型態和相對應的包裹物件作轉換。我們宣告整數物件 `grade1`，我們宣告原始資料型態整數 `n1` 變數並且將數值 88 給它。我們將 `n1` 數值給 `grade1` 整數物件，這時就自動建立了整數物件。

```
Integer grade1 ;  
int n1=88 ;  
grade1=n1;
```

我們宣告整數物件 `grade1`，我們宣告並建立整數物件 `grade2` 並且將數值 88 給它。我們宣告整數原始資料型態 `n2` 變數。我們將 `grade2` 整數物件轉換成原始整數資料型態，並將它分配給 `n2` 變數。

```
Integer grade2=new Integer(88);  
int n2 ;  
n2=grade2 ;
```



### 範例 Autoboxing.java

第三行我們宣告整數物件`grade1`。

第四行我們宣告原始資料型態整數`n1`變數並且將數值88給它。

第五行我們將`n1`數值給`grade1`整數物件，這時就自動建立了整數物件。

第七行我們宣告整數物件`grade1`，我們宣告並建立整數物件`grade2`並且將數值88給它。

第八行我們宣告整數原始資料型態`n2`變數。

第九行我們將`grade2`整數物件自動轉換成原始整數資料型態，並將它分配給`n2`變數。

```
1 public class Autoboxing{
2     public static void main(String[] arg){
3         Integer grade1;
4         int n1=88;
5         grade1=n1;
6         System.out.println(grade1);
7         Integer grade2=new Integer(88);
8         int n2;
9         n2=grade2;
10        System.out.println(n2);
11    }
12 }
```

這是自動轉換型別執行的情況，都一樣。



88

88

## 5-9 記憶體回收Garbage Collection

Java 有自動回收記憶體的機制。當我們新增物件時，Java 就會分配記憶體空間給該物件。當該物件結束時，Java 會自動的將該記憶體給收回。

### 範例

#### Garbage.java

第四行到第七行為大括號，當執行完裏面內容時，Java會自動的回收記憶體。

第五行我們新增了str1字串物件。

第六行執行顯示該物件。

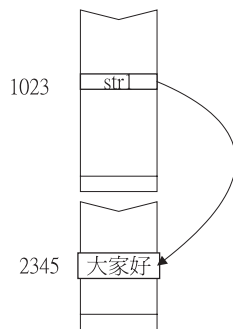
第七行大括號結束時，也就是整個敘述結束，這時JAVA自動回收分配出去的記憶體，因此字串物件str1就消失了。因此第八行在編譯時會發生找不到str1參考的物件，而發生錯誤。

```
1 public class Garbage{
2     public static void main(String[] args){
3     {
4         if(true){
5             String str1=new String("大家好");
6             System.out.println(str1);
7         }
8         System.out.println(str1);
9     }
10 }
11 }
```

因為str1物件參考在第七行大括號結束時消滅，因此第八行會找不到str1符號。

```
F:\java2\6\example>javac Garbage.java
Garbage.java:8: cannot find symbol
symbol : variable str1
location: class Garbage
        System.out.println(str1);
                           ^
1 error
```

我們在記憶體位置為1023的地方新增字串物件”大家好”的參考，這時，str1就會指向物件”大家好”所在的位置。str1變數放的是物件的參考位置。當大括號執行完並消失時，整個程式自動結束，Java會將這些記憶體給回收。



## 習題

1. 請簡述物件的參考？
2. 請簡述Java.lang.String類別的函數？
3. 何謂套件？

### 【答案】

套件就是群組和命名一群的類別，我們可以在程式中來使用它，而不用將它們放在程式同一個目錄中。



一個套件就是將一群的類別，群組放到同一個目錄中，然後再給這個目錄一個套件名稱。在套件中的類別是放在不同的檔案中，而檔案的名稱和類別的名稱相同。每一個檔案在起始的地方都有下列的敘述。

**Package** 套件名稱；

它一定要放在檔案的最前面。套件名稱通常都是小寫的字母，然後通常接標點符號，當作連接目錄。例如example.ch01就是套件名稱，然後每一個在這個套件中的檔案起始的地方都有下列的敘述。

**Package** example.ch01；

4. 請簡述如何輸入import套件？

**【答案】**

每一個程式或類別都可以使用套件裏的所有類別，它們只要加入import敘述在檔案的最前面就可以了。就算程式和套件裏面的類別不在同一個目錄也可以使用。假如我們希望使用在example.ch01套件裏面的類別，我們可以使用下列的敘述。

**import** example.ch01.\*；

星號\*代表我們輸入example.ch01套件中的所有類別。

5. 請簡述在Windows系統上設定類別環境的過程？

6. 請簡單說明Java所提供的套件及其功能？



7. 請簡單說明一些常用的數學Math類別函數？
8. 請簡述列舉型態？
9. 請簡述Java的容器和元件？
10. 請簡述包裹類別wrapper？
11. 請簡述Autoboxing？
12. 請簡述Java的記憶體回收機制？

**【答案】**

Java有自動回收記憶體的機制。當我們新增物件時，Java就會分配記憶體空間給該物件。當該物件結束時，Java會自動的將該記憶體給收回。