第 3 章

變數

3-1 甚麼是變數？

- 變數是用來存放暫時的資料，以便後續的處理

3-1-1 變數的宣告

程式 Variable.java 宣告變數並指派內容給變數

```
01 public class Variable {
02   public static void main(String[] argv) {
03     int i;
04     i = 20;
05     System.out.println(i);
06   }
07 }
```

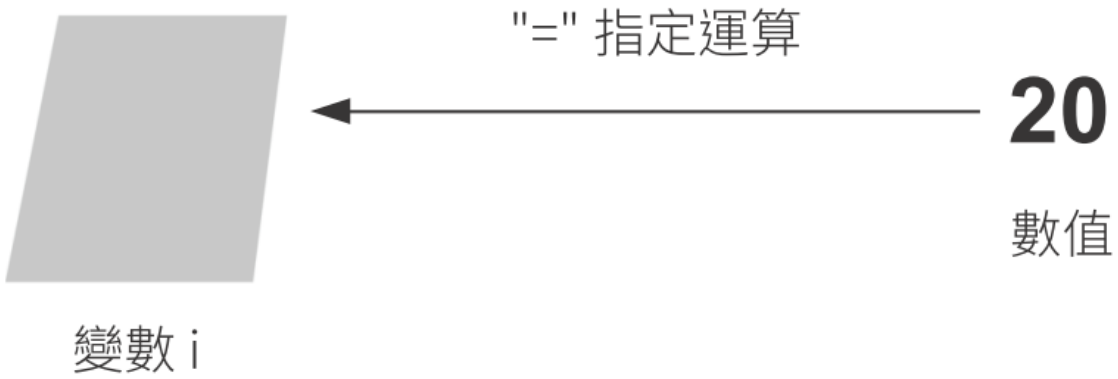
執行結果

```
20
```

- 第3行的意思就是宣告(Declare)一個變數 i
- 這個 i 的變數是要用來存放整數 (Integer) 類型的資料。

3-1-2 設定變數的內容

- 宣告了變數之後，接著要設定變數的初值
- “=”，稱為指定算符 (Assignment Operator)，它的功用就是將資料放到變數中



宣告同時設定初值

程式 Variable2.java

```
01 public class Variable2 {
02   public static void main(String[] argv) {
03     int i=20;
04     System.out.println("變數 i 的內容為：" + i);
05   }
06 }
```

執行結果

```
變數 i 的內容為：20
```

宣告同時設定初值

```
System.out.println("變數 i 的內容為：" + 20);
```

```
System.out.println("變數 i 的內容為：20");
```

3-1-3 變數的名稱

- 本書的變數名稱均採用**駝峯寫法 (Camel Case)** 也就是變數如果是由一個以上的英文字組成，則英文字之間沒有空格，
  第一個英文字由小寫開頭，
  之後的英文字則為大寫開頭。

變數的命名規則

1. 必須以英文字母開頭，大小寫均可
2. 跟著開頭字元之後的，可以是符合前一條規則的字元，或者是阿拉伯數字0~9
3. 長度沒有限制
4. 不能和 Java 程式語言中的保留字 (Reserved Word)重複
5. 字母相同，但大小寫不同時，視為是不同的名稱

Java 執行動作的關鍵字 (Keywords)

abstract	continue	for	new	switch
assert	default	goto	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const	float	native	super	while

變數的命名規則

程式 LegalVariableName.java 合法的變數名稱

```
01 public class LegalVariableName {
02   public static void main(String[] argv) {
03     int age; //合法變數名稱
04     int AGE; //合法變數名稱
05     int Age; //合法變數名稱
06     int No1; //合法變數名稱
07     int No11111111; //合法變數名稱
08     int _Total; //合法變數名稱
09     age = 19; //合法變數名稱
10     System.out.println("你的年齡是：" + age);
11   }
12 }
```

變數的命名規則

程式 InvalidVariableName.java 不合法的變數名稱

```
01 public class InvalidVariableName {
02   public static void main(String[] argv) {
03     int 3age; // 不能以數字開頭
04     int #AGE; // 不能使用 "#"  字元
05     int A#GE; // 不能使用 "#"  字元
06     int while;    // 不能使用關鍵字
07     int true; // 不能使用內建保留的字面常數
08     3age = 19;
09     System.out.println("你的年齡是：" + 3age);
10   }
11 }
```

變數的命名規則

執行結果

```
InvalidVariableName.java:3: not a statement
    int 3age;    // 不能以數字開頭
    ^
InvalidVariableName.java:3: ';' expected
    int 3age;    // 不能以數字開頭
        ^
InvalidVariableName.java:3: not a statement
    int 3age;    // 不能以數字開頭
         ^
InvalidVariableName.java:4: illegal character: \35
    int #AGE;     // 不能使用 "#"  字元
        ^
InvalidVariableName.java:4: not a statement
    int #AGE;     // 不能使用 "#"  字元
    ^
```

變數的命名規則

```
InvalidVariableName.java:4: not a statement
    int #AGE;     // 不能使用 "#"  字元
        ^
InvalidVariableName.java:5: illegal character: \35
    int A#GE;    // 不能使用 "#"  字元
         ^
InvalidVariableName.java:5: not a statement
    int A#GE;    // 不能使用 "#"  字元
          ^
InvalidVariableName.java:6: not a statement
    int while;   // 不能使用關鍵字
    ^
InvalidVariableName.java:6: ';' expected
    int while;   // 不能使用關鍵字
       ^
InvalidVariableName.java:6: '(' expected
    int while;   // 不能使用關鍵字
              ^
```

變數的命名規則

```
InvalidVariableName.java:7: not a statement
    int true;    // 不能使用內建保留的字面常數
    ^
InvalidVariableName.java:7: ';' expected
    int true;    // 不能使用內建保留的字面常數
        ^
InvalidVariableName.java:8: not a statement
    3age = 19;
    ^
InvalidVariableName.java:8: ';' expected
    3age = 19;
     ^
InvalidVariableName.java:9: ')' expected
    System.out.println("你的年齡是：" + 3age);
                                       ^
InvalidVariableName.java:9: ';' expected
    System.out.println("你的年齡是：" + 3age);
                                         ^
17 errors
```

使用標準萬國碼 (Unicode) 字元為變數命名

- 由於 Java 支援使用標準萬國碼 (Unicode)，可以使用包含中文等亞洲國家語言的文字為變數命名

程式 CVariableName.java 使用中文為變數命名

```
01 public class CVariableName {
02   public static void main(String[] argv) {
03     int 年齡 = 19; // 使用中文的變數名稱
04     System.out.println("你的年齡是：" + 年齡);
05   }
06 }
```

執行結果

```
你的年齡是：19
```

3-2 資料型別 (Data Types)

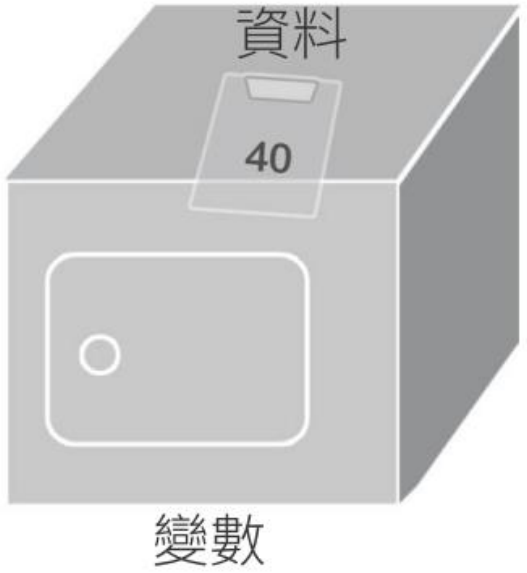
可以先粗略的將 Java 中的資料分成兩種：

- 基本型別 (Primitive Data Types)
- 參照型別 (Reference Data Types)

資料型別 (Data Types)

1. 基本型別：

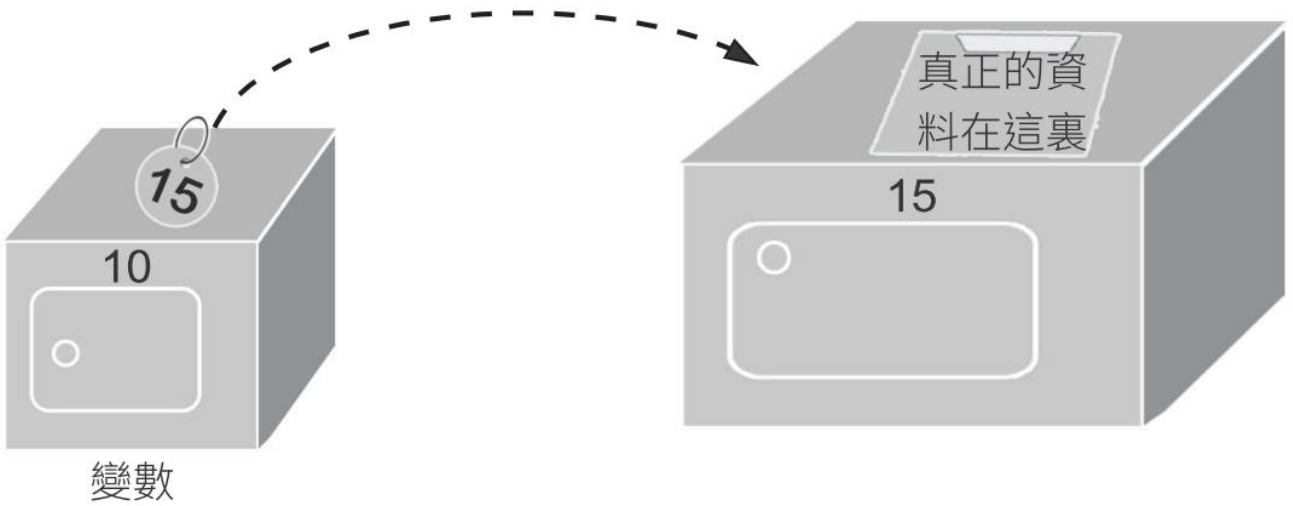
   這種型別的資料是直接放在變數中，像是之前使用過的整數以及浮點數，都屬於這種資料。



資料型別 (Data Types)

2. 參照型別：

這種型別的資料並不是放置在變數中，而是另外配置一塊空間來放置資料



3-3 基本型別 (Primitive Data Types)

Java 的基本型別分為兩大類：

- 布林型別 (Boolean Data Type)
- 數值型別 (Numeric Data Type)

3-3-1 布林型別 (Boolean Data Type)

- 資料只能有兩種可能值，分別是true與false

程式 UsingBoolean.java 使用布林型別的變數

```
01 public class UsingBoolean {
02   public static void main(String[] argv) {
03     boolean test = false;  // false 是 Java 內建的字面常數
04     System.out.println("布林變數 test 的值：" + test );
05
06     test = true;  // true 也是 Java 內建的字面常數
07     System.out.println("布林變數 test 的值：" + test );
08   }
09 }
```

執行結果

```
布林變數 test 的值：false
布林變數 test 的值：true
```

3-3-2 數值型別 (Numeric Data Type)

數值型別又可區分為 2 大類：

- 整數型別 (Integral Data Type)
- 浮點數型別 (Floating Point Data Type)

整數型別 (Integral Data Type)

資料型別	最小值	最大值	佔用空間
byte	-128	127	1 個位元組
short	-32768	32767	2 個位元組
int	-2147483648	2147483647	4 個位元組
long	-9223372036854775808	9223372036854775807	8 個位元組
char	0	65535	2 個位元組

整數型別 (Integral Data Type)

- 10 進位：
  非 0 的數字開頭的數值就是 10 進位的數值
- 2 進位：
  以 0b 或 0B 開頭的數字就是 2 進位數值
- 16 進位：
  以 0x 或是 0X 開頭的數值就是16 進位的數值
- 8 進位：
  以0 開頭的數值就是 8 進位的數值
- 不論是 0、00、或是 0x0、0X0，
  都代表數值 0

整數型別 (Integral Data Type)

程式 IntegerValue.java 使用整數值

```
01 public class IntegerValue {
02   public static void main(String[] argv) {
03     System.out.println("10進位 1357        = " + 1357);
04
05     int i = 0b10011001 ; // int 型別, 2進位
06     System.out.println("2進位  0b10011001 = " + i);
07
08     long l = 0XADEF; // long 型別, 16進位
09     System.out.println("16進位 0XADEF      = " + l);
10
11     short s = 01357; // short 型別, 8進位
12     System.out.println("8進位  01357       = " + s);
13   }
14 }
```

整數型別 (Integral Data Type)

執行結果

```
10進位 1357        = 1357
2進位  0b10011001  = 153
16進位 0XADEF      = 44527
8進位  01357       = 751
```

整數型別 (Integral Data Type)

程式 UsingNumber.java 善用不同數字標示法

```
01 public class UsingNumber {
02   public static void main(String[] argv) {
03     short s = -13666;      // 負數
04     System.out.println("變數 s = " + s);
05
06     long  l = 2_135_482_789L;  // 用 L 標示 long 整數
07     System.out.println("變數 l = " + l);
08
09     int   i = 0b1100_0110_0011_1010;  // 底線字元
10     System.out.println("變數 i = " + i);
11   }
12 }
```

執行結果

```
變數 s = -13666
變數 l = 2135482789
變數 i = 50746
```

整數型別 (Integral Data Type)

程式 LongValueError.java 超過 int 範圍的整數

```
01 public class LongValueError {
02   public static void main(String[] argv) {
03     long l = 2_147_483_649; // 未指定為long數值, 編譯會出現錯誤
04     System.out.println("變數  l = " + l);
05   }
06 }
```

執行結果

```
LongValueError.java:3: error: integer number too large
    long  l = 2_147_483_649; // 未指定為long數值, 編譯會出現錯誤
                ^
1 error
```

整數型別 (Integral Data Type)

- 如果需要直接以字碼的方式設定資料值，可以使用**跳脫序列 (Escape Sequence)** 的方式：**'\uXXXX'**
- XXXX 是字元以 16 進位表示的字碼，**一定要用 4 位數和小寫的 u**。

整數型別中的 char

- 跳脫序列可以表示的特殊字元，裡頭包含了一些無法顯示的字元：

跳脫序列	字碼	字元
\b	\u0008	BS (← 鍵)
\t	\u0009	HT (Tab 鍵)
\n	\u000a	LF (換行)
\f	\u000c	FF (換頁)
\r	\u000d	CR (歸位)
\"	\u0022	" (雙引號)
\'	\u0027	' (單引號)
\\	\u005c	\ (反斜線)

跳脫序列

程式 EscapeValue.java 使用跳脫序列

```
01 public class EscapeValue {
02   public static void main(String[] argv) {
03     char  ch = '\u5b57'; // 16 進位 5b57 是 '字' 的 Unicode 編碼
04     System.out.println("變數 ch 的內容為：" + ch);
05
06     ch = '\\';       // 反斜線 \
07     System.out.println("變數 ch 的內容為：" + ch);
08
09     ch = '\'';       // 單引號 '
10     System.out.println("變數 ch 的內容為：" + ch);
11   }
12 }
```

執行結果

```
變數 ch 的內容為：字
變數 ch 的內容為：\
變數 ch 的內容為：'
```

- **帶小數點的數值：**
  例如 3.4、3.0、0.1234
- **使用科學記號：**
  例如 1.3E2、2.0E-3、0.4E2，
  指數部分也可以用小寫的 e。

資料型別	可表示範圍	佔用空間
float	± 3.40282347E+38 ~ ± 1.40239846E-45	4 個位元組
double	± 1.79769313486231570E+308 ~ ± 4.94065645841246544E-324	8 個位元組

使用浮點數

程式 DoubleValue.java 使用浮點數

```
01 public class DoubleValue {
02   public static void main(String[] argv) {
03     double  d;
04     d = 3.4;
05     System.out.println("變數 d 的內容為：" + d);
06
07     d = 3.0;
08     System.out.println("變數 d 的內容為：" + d);
09
10     d = 0.1234;
11     System.out.println("變數 d 的內容為：" + d);
12
13     d = 3;
14     System.out.println("變數 d 的內容為：" + d);
15
16     d = .1234;
17     System.out.println("變數 d 的內容為：" + d);
```

使用浮點數


```
18
19      d = 2.0E-3;
20      System.out.println("變數 d 的內容為：" + d);
21
22      d = 6.022_140_78E23
23      System.out.println("變數 d 的內容為：" + d);
24    }
25 }
```

執行結果

```
變數 d 的內容為：3.4
變數 d 的內容為：3.0
變數 d 的內容為：0.1234
變數 d 的內容為：3.0
變數 d 的內容為：0.1234
變數 d 的內容為：0.002
變數 d 的內容為：6.02214078E23
```

使用浮點數

- 任何帶有小數點的數值用在 float 型別，必須加上一個 "f" 或是 "F"

程式 Floating.java 使用 f 標示為 float 型別

```
01 public class Floating {
02   public static void main(String[] argv) {
03     float f1 = 0.01f;
04     float f2 = 0.99f;
05
06     f1 = f1 + f2; // 加法運算
07     System.out.println("計算的結果是：" + f1);
08   }
09 }
```

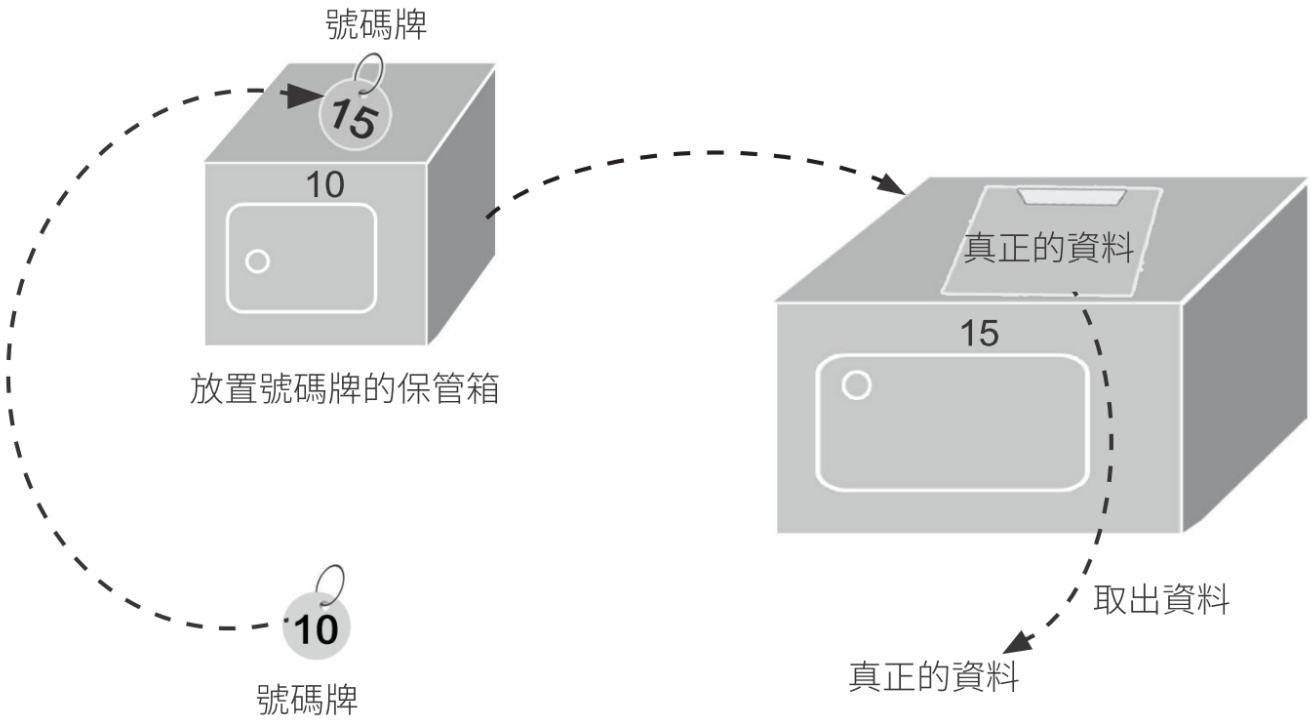
數值必須加 f，否則會被視為 double 而造成編譯錯誤（因 Java 不允許直接將 double 資料存入 float 變數中，以免因數值太大放不下而導致錯誤）。

執行結果

```
計算的結果是：1.0
```

3-4 參照型別 (Reference Data Types)

- 參照型別的變數本身並不放置資料，而是存放資料的位址。



3-4 參照型別 (Reference Data Types)

Java 共有 3 類參照型別：

- 字串 (String)
- 陣列 (Array)
- 物件 (Object)

字串 (String) 型別

- 用來儲存字串的變數，必須使用 **String 型別**

程式 StringVariable.java 使用字串

```
01 public class StringVariable  {
02   public static void main(String[] argv) {
03     String s1= "第一個字串";
04     String s2= "第二個\t字串"; // 字串中可使用跳脫序列
05
06     System.out.println(s1);
07     System.out.println(s2);
08     System.out.println(s1 + '\n' + s2); // 字串也可與字元相加
09   }
10 }
```

字串 (String) 型別

執行結果

```
第一個字串
第二個    字串
第一個字串
第二個  字串
```

'\t' 為定位字元 Tab，所以輸出字串中會有空白

'\n' 代表換行字元，所以輸出字串 s1 的內容後，會換行再輸出字串 s2 的內容

字串 (String) 型別

注意事項：

- 字串型別的資料值必須以雙引號括 " 起來
- 字串可以使用 "+" 來連接

字串 (String) 型別

程式 StringLength.java 使用 length 方法取得字串長度

```
01 public class StringLength  {
02   public static void main(String[] argv) {
03     String s1 = "第一個\t字串";
04     String s2 = "Second 字串";
05
06     System.out.println("變數 s1 的長度：" + s1.length());
07     System.out.println("變數 s2 的長度：" + s2.length());
08   }
09 }
```

執行結果

```
變數 s1 的長度：6
變數 s2 的長度：9
```

3-5 宣告變數的技巧

與變數宣告相關的技巧與注意事項：

- 一次宣告多個變數
- 變數的初值

一次宣告多個變數

- 如果有多個相同型別的變數，可以使用逗號","分隔，在單一敘述中同時宣告

程式 MultipleVariable.java 同時宣告多個同型別的變數

```
01 public class MultipleVariable {
02   public static void main(String[] argv) {
03     int i ,j ,k, sum;
04     i = 10;
05     j = 20;
06     k = 30;
07     sum = i + j + k;
08     System.out.println("總和等於：" + sum);
09   }
10 }
```

執行結果

```
總和等於：60
```

變數的初值

- 在宣告多個變數的同時可以設定變數值，也可以直接用運算式來設定變數的初值

程式 MultiVarInit.java 同時宣告多個變數並設定初值

```
01 public class MultiVarInit {
02   public static void main(String[] argv) {
03     int i = 10,j = 20, k = 30, sum;
04     sum = i + j + k;
05     System.out.println("總和等於：" + sum);
06   }
07 }
```

執行結果

```
總和等於：60
```

3-6 常數

- 常數所儲存的資料則是恆常不變
- 在 Java 中，
  有兩種形式的常數，
  一種稱為字面常數 (Literal)，
  另一種稱為具名常數 (Named Constant)

3-6-1 字面常數 (Literal)

- **字面常數，**
  就是直接以文字表達其數值的意思

程式 MultipleVariableInitAll.java 使用字面常數

```
01 public class MultipleVariableInitAll {
02   public static void main (String [] argv) {
03     int i = 10,j = 20, k = 30, sum = i + j + k ;
04     System.out.println (" 總和等於： " + sum);
05   }
06 }
```

3-6-1 字面常數 (Literal)

注意事項：

- 有些資料型別必須在字面常數的數值之後加上代表該型別的字尾
- Char 型別的字面常數以字元來表達時，必須以單引號括起來
- 要表示一串文字，
  則必須用一對雙引號(“)括起來

3-6-2 具名常數 (Named Constant)

- 使用一個具有名字的常數，
  以代表某個具有特定意義的數值

程式 NamedConstant.java 計算圓面積與圓週

```
01 public class NamedConstant {
02   public static void main(String[] argv) {
03     double r = 3.0;          //半徑
04     final double PI = 3.14; // 圓周率
05     System.out.println("圓周：" + 2 * PI * r);
06     System.out.println("面積：" + PI * r * r);
07   }
08 }
```

執行結果

```
圓周：18.84
面積：28.259999999999998
```

- 在宣告變數時的資料型別之前加上final字符，就會限制該變數在設定初值之後無法再做任何更改，
  我們稱這樣的變數為具名常數。

使用具名常數有以下幾個好處

- 具說明意義：
  具名常數的名稱可以說明其所代表的意義，在閱讀程式時容易理解
- 避免手誤
- 方便修改程式

3-7 良好的命名方式

1. 變數的名稱通常都以小寫字母開頭，
   並且應該能說明變數的用途
2. 組合多個單字來為變數命名時，
   可以採取字首字母大寫的方式
3. 組合的單字過長時，
   可以採用適當的首字母縮寫
4. 可以為不同型別的變數名稱加上一個字頭，
   以彰顯其為某種型別的變數
5. 對於具名常數，
   一般慣例都是採全部大寫字母的命名方式