

## 第四章 運算子、運算式與敘述

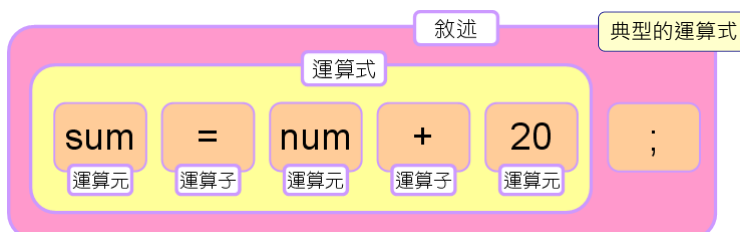
- 4.1 認識運算式與運算子
- 4.2 學習各種常用的運算子
- 4.3 認識運算子的優先順序
- 4.4 學習如何進行運算式之資料型態的轉換

1

### 4.1 運算式與運算子

## 認識運算式

- 運算式由**運算元** ( operand ) 與**運算子** ( operator ) 組成
- **運算元**可以是變數或是常數
- **運算子**就是數學上的運算符號
  - 如「+」、「-」、「\*」、「/」等



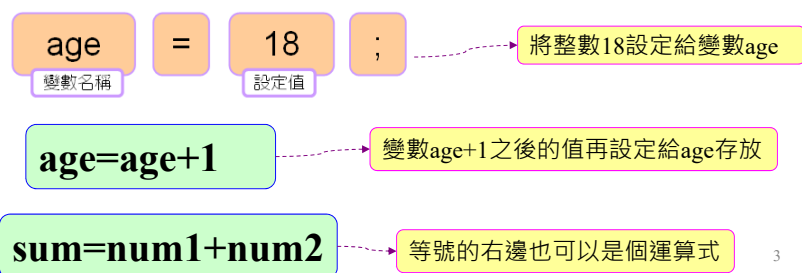
2

## 設定運算子(1/2)

- 為各種不同資料型態的變數設值，可使用**設定運算子**

設定運算子	意義
=	設定

- 等號 (=) 並不是「等於」，而是「設定」



3

## 設定運算子 (2/2)

- 下面的範例示範設定運算子的使用：

```

01 // app4_1,設定運算子「=」
02 public class app4_1
03 {
04     public static void main(String args[])
05     {
06         int age=18; // 宣告整數變數age,並設值為18
07
08         System.out.println("before compute, age="+age);// 印出age的值
09         age=age+1; // 將age加1後再設定給age存放
10         System.out.println("after compute, age="+age);// 印出計算後age的值
11     }
12 }

```

```

/* app4_1 OUTPUT-----
before compute, age=18
after compute, age=19
-----*/

```

4

## 一元運算子 (1/2)

- 下面的敘述是由一元運算子與單一個運算元組成

```
+6;           // 表示正6
~a;           // 表示取a的1補數
x = -y;       // 表示負y的值設定給變數x存放
!a;           // a的NOT運算，若a為0，則!a為1，若a不為0，則!a為0
```

- 一元運算子的成員

一元運算子	意義
+	正號
-	負號
!	NOT · 否
~	取1的補數

5

## 一元運算子 (2/2)

- 「~」與「!」運算子的範例：

```
01 // app4_2,一元運算子「~」與「!」
02 public class app4_2
03 {
04     public static void main(String args[])
05     {
06         byte a=Byte.MIN_VALUE;    // 宣告變數a,並設為該型態之最小值
07         boolean b=true;           // 宣告boolean變數b,並設為true
08
09         System.out.println("a="+a+",~a="+(~a)); // 印出a與~a的值
10         System.out.println("b="+b+",!b="+(!b)); // 印出b與!b的值
11     }
12 }
```

```
/* app4_2 OUTPUT--
--
a=-128,~a=127
b=true,!b=false
-----*/
```

6

原則：首位1表示負數，0表示正數。

題目1：求4的補數

$$\begin{array}{r}
 \begin{array}{cccccccc}
 2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\
 a=4 & \text{---} & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
 \hline
 \sim a=-5 & \text{---} & 1 & 1 & 1 & 1 & 0 & 1 & 1
 \end{array} \\
 = -128 + 123 = -5
 \end{array}$$

題目2：求-128的補數

$$\begin{array}{r}
 \begin{array}{cccccccc}
 2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\
 a=-128 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
 \hline
 \sim a=127 & 0 & 1 & 1 & 1 & 1 & 1 & 1
 \end{array} \\
 = -2^7 + 0 = -128 \\
 = 2^7 = 127
 \end{array}$$

題目3：求7的補數

#### 4.1 運算式與運算子

## 算術運算子 (1/3)

- 算術運算子在數學上經常會使用到
- 算術運算子的成員：

算數運算子	意義
+	加法
-	減法
*	乘法
/	除法
%	取餘數

- 加法運算子「+」可將前後兩個運算元做加法運算

6+2;            // 計算6+2

b=a+15        // 將a的值加15之後，再設定給變數b存放

sum=a+b+c    // 將a,b與c的值相加後，設定給變數sum存放

8

## 4.1 運算式與運算子

## 算術運算子 (2/3)

- 減法運算子「-」可將前後兩個運算元做減法運算

```
age=age-1;      // 計算age-1之後，再將其結果設定給age存放
c=a-b;          // 計算a-b之後，再設定給c存放
54-12;          // 計算54-12的值
```

- 乘法運算子「\*」可將前後兩個運算元相乘

```
b=c*3;          // 計算c*3之後，再將其結果設定給b存放
a=a*a;          // 計算a*a之後，再設定給a存放
17*5;           // 計算17*5的值
```

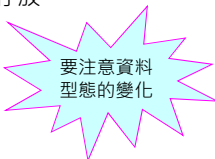
9

## 4.1 運算式與運算子

## 算數運算子 (3/3)

- 除法運算子「/」可將前面的運算元後除以後面的運算元

```
b=a/6;          // 計算a/6之後，再將其結果設定給b存放
d=c/d;          // 計算c/d之後，再設定給d存放
3/8;            // 計算3/8的值
```



要注意資料  
型態的變化

- 餘數運算子「%」：取出二數相除後的餘數

```
age=age%5;       // 計算age/5的餘數，再把計算結果給age存放
c=a%b;           // 計算a/b的餘數，然後把計算結果給c存放
48%7;            // 運算48%7的值
```

10

## 課堂練習

- ex4\_1\_1.java
- 請撰寫程式, 計算出 277 除以 13 的商數以及餘數。(請以鍵盤讀取所需輸入的資料)

```
package ex_9700837;
import java.util.Scanner;
public class e {
    public static void main(String[] argv) {
        int i;
        int j;

        Scanner scn=new Scanner(System.in);
        System.out.print("請輸入分子 : ");
        i=scn.nextInt();
        System.out.print("請輸入分母 : ");
        j=scn.nextInt();
    }
}
```

請輸入分子 : 277  
 請輸入分母 : 13  
 277 / 13 的商為 21  
 277 / 13 的餘數為 4

### 4.1 運算式與運算子

## 關係運算子與 if 敘述 (1/2)

- if 敘述的格式如下：

if 敘述的格式

```
if(條件判斷)
    敘述;
```

- 如下面的程式片段：

```
if (x<0)
    System.out.println("x的值小於0");
```

關係運算子的成員

關係運算子	意義
>	大於
<	小於
>=	大於等於
<=	小於等於
==	等於
!=	不等於

## 4.1 運算式與運算子

## 關係運算子與 if 敘述 (2/2)

```

01 // app4_3,關係運算子
02 public class app4_3
03 {
04     public static void main(String args[])
05     {
06         if (9>4)        // 判斷9>4是否成立
07             System.out.println("9>4成立");        // 印出傳回值
08
09         if (true)       // 判斷true是否成立
10             System.out.println("此行會被執行--true"); // 印出字串
11
12         if (false)      // 判斷false是否成立
13             System.out.println("此行會被執行--false"); // 印出字串
14
15         if (5==7) // 判斷5是否等於7
16             System.out.println("5==7成立");        // 印出字串
17     }
18 }

```

```

/* app4_3 OUTPUT---
9>4成立
此行會被執行--true
-----*/

```

13

## 4.1 運算式與運算子

## 遞增與遞減運算子

- 遞增與遞減運算子的成員：

遞增與遞減運算子	意義
++	遞增・變數值加1
--	遞減・變數值減1

- 想讓變數a加上1，其敘述如下

a=a+1;

a加1後再設定給a  
存放・簡潔的寫  
法為a++;

```

01 // app4_4,遞增運算子「++」
02 public class app4_4
03 {
04     public static void main(String args[])
05     {
06         int a=3,b=3;
07
08         System.out.print("a="+a);    // 印出a
09         System.out.println(",a++="+a+++",a="+a); // 印出a++
10         System.out.print("b="+b);    // 印出b
11         System.out.println(",++b="+(++b)+"",b="+b); // 印出++b
12     }
13 }

```

本程式比較a++  
與++b的不同

```

/* app4_4 OUTPUT--
a=3,a++=3,a=4
b=3,++b=4,b=4
-----*/

```

14

## 4.1 課堂練習

Q. What gets printed when the following program is compiled and run. Select the one correct answer.

```
class ex4_1_2 {
    public static void main(String args[]) {
        int i,j,k,l=0;
        k = l++;
        j = ++k;
        i = j++;
        System.out.println(i);
    }
}
```

A. 0    B. 1    C. 2    D. 3

15

### 4.1 運算式與運算子

## 邏輯運算子 (1/2)

### • 邏輯運算子與真值表：

邏輯運算子的成員

邏輯運算子	意義
&&	AND · 且
	OR · 或

AND與OR真值表

AND	T	F
T	T	F
F	F	F

OR	T	F
T	T	T
F	F	F

### • 邏輯運算子的使用範例：

- (1) `a>0 && b>0`      // 兩個運算元皆為真，運算結果才為真
- (2) `a>0 || b>0`      // 兩個運算元只要一個為真，運算結果就為真

16



## 4.1 運算式與運算子

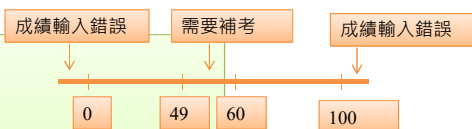
## 邏輯運算子(2/2)

- 邏輯運算子應用在 if 敘述中：

```

01 // app4_5,邏輯運算子
02 public class app4_5
03 {
04     public static void main(String args[])
05     {
06         int a=53;
07
08         if ((a<0) || (a>100))
09             System.out.println("Input error!!"); // 成績輸入錯誤
10         if ((a<60) && (a>49))
11             System.out.println("Make up exam!!"); // 需要補考
12     }
13 }

```



```

/* app4_5 OUTPUT----
Make up exam!!
-----*/

```

17

## 4.1 運算式與運算子

## 括號運算子

- 括號運算子() 用來處理運算式的優先順序：

括號運算子	意義
()	提高括號中運算式的優先順序

- 以一個加減乘除式子為例：

 $12-2*6/4+1$ 

// 未加括號的運算式

運算結果為10

- 想分別計算 $12-2*6$ 及 $4+1$ 之後再將兩數相除則成為

 $(12-2*6)/(4+1)$ 

// 加上括號的運算式

運算結果為0

18

## 4.1 課堂練習

- 請寫出下列程式的輸出結果。

```
public class ex4_1_3
{
    public static void main(String args[])
    {
        int a=15;
        System.out.println("a="+(--a));
        System.out.println("a="+(++a));
        System.out.println("a="+a);
    }
}
```

輸出結果：a=14  
a=14  
a=15

19

## 4.1 課堂練習

```
package myJava.exercise.ch03;
import java.lang.*;
public class ex4_1_4
{
    public static void main(String args[])
    {
        int x=1,y=1,z=1;
        z = ++x-y++;
        z = z + x++ + ++y;

        System.out.println("x=" + x);
        System.out.println("y=" + y);
        System.out.println("z=" + z);
    }
}
```

x=3  
y=3  
z=6

## 4.2 運算子的優先順序

## 運算子的優先順序列表

優先順序	運算子	類別	結合性
1	()	括號運算子	由左至右
1	[]	方括號運算子	由左至右
2	!、+ (正號)、- (負號)	一元運算子	由右至左
2	~	位元邏輯運算子	由右至左
2	++、--	遞增與遞減運算子	由右至左
3	*、/、%	算數運算子	由左至右
4	+、-	算數運算子	由左至右
5	<<、>>	位元左移、右移運算子	由左至右
6	>、>=、<、<=	關係運算子	由左至右
7	==、!=	關係運算子	由左至右
8	& (位元運算的AND)	位元邏輯運算子	由左至右
9	^ (位元運算的XOR)	位元邏輯運算子	由左至右
10	(位元運算的OR)	位元邏輯運算子	由左至右
11	&&	邏輯運算子	由左至右
12		邏輯運算子	由左至右
13	?:	條件運算子	由右至左
14	=	設定運算子	由右至左

數字愈小  
表示優先  
順序愈高

21

## 補充資料

## 邏輯運算子(Logical Operators)

運算子	定義	優先順序	結合律
!	一元邏輯補數運算(NOT)	2	由右而左
&	完全評估的AND運算	9	由左而右
^	XOR	10	由左而右
	完全評估的OR運算	11	由左而右
&&	快捷的AND運算	12	由左而右
	快捷的OR運算	13	由左而右

22

## 完全(&)VS快捷(&&)

- (&)完全評估系列的運算子一定會把兩個運算元拿來運算
- (&&)快捷(Short-circuit)系列的運算子會先算出第一個運算元的值，如果這個值已經可以決定出整個運算式的結果，快捷系統的運算子就不會對第二個運算元進行運算。
- 例如
  - `boolean1 = (3>7)&(5>2)`
  - `boolean2 = (3>7)&&(5>2)`
- 敘述以`boolean2`的速度較快，因為在快捷系列的`boolean2`中`3>7`已得到一個`false`，所以`5>2`不予執行。

23

## 「&」、「&&」的差異

- codeA:
 

```
int a = 10, b = 5;
if (a++ > 10 & b-- < 5)
{
}
System.out.println("a = " + a + ", b = " + b);
```

codeA結果: a = 11, b = 4
- codeB:
 

```
int a = 10, b = 5;
if (a++ > 10 && b-- < 5) //快捷
{
}
System.out.println("a = " + a + ", b = " + b);
```

codeB結果: a = 11, b = 5
- codeA與codeB差別在於  
 & 無論左邊結果為何都會執行右邊的程式  
 && 則是左邊為true才會繼續執行右邊的判斷(也就是說當左邊為false, 那麼不管右邊是否為true或false, 整個邏輯算式的結果一定為false)

24

## 4.2 課堂練習

試判別下列的各敘述的執行結果

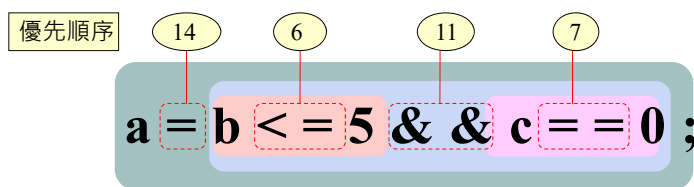
- |                                    |         |
|------------------------------------|---------|
| ① $2+7<15-6$                       | → false |
| ② $5-3*6+2$                        | → -11   |
| ③ $(12-3)*8+25$                    | → 97    |
| ④ $16>2 \ \&\& \ 8<9 \ \&\& \ 2<7$ | → true  |
| ⑤ $28>10 \    \ 7<2$               | → true  |
| ⑥ $6<=6$                           | → true  |
| ⑦ $5+17>16$                        | → true  |
| ⑧ $21+10*6>53$                     | → true  |
| ⑨ $14+6>8 \    \ 32-5>6$           | → true  |
| ⑩ $36>=10$                         | → true  |

25

### 4.2 運算子的優先順序

## 運算子的優先順序

- 運算子優先順序的範例：



1. 先計算 `b<=5` (`<=`的優先順序為6)
2. 再計算 `c==0` (`==`的優先順序為7)
3. 然後進行`&&`運算 (`&&`的優先順序為11)
4. 最後再把運算結果設給變數 `a` 存放 (`=` 的優先順序為14)

26

## 4.2 運算子的優先順序

## 結合性

- **結合性**是指相同優先順序之運算子的執行順序
- 算術運算子的結合性為「由左至右」

`a=b+d/3*6;` // 結合性可以決定運算子的處理順序

d會先除以3再乘以6  
得到的結果加上b後，  
將整個值給a存放

27

## 4.3 運算式

## 運算式的組成

- 運算式是由常數、變數或是其它運算元與運算子所組合而成
- 下面的例子，均是屬於運算式

<code>-18</code>	// 由一元運算子「-」與常數18組成
<code>sum+6</code>	// 由變數sum、算術運算子與常數6組成
<code>a+b-c/(d*3-9)</code>	// 由變數、常數與算術運算子組成的運算式

28

## 4.3 運算式

## 簡潔的運算式 (1/2)

- 下表為簡潔的運算式與使用說明：

運算子	範例用法	說明	意義
+=	a+=b	a+b的值存放到a中	a=a+b
-=	a-=b	a-b的值存放到a中	a=a-b
*=	a*=b	a*b的值存放到a中	a=a*b
/=	a/=b	a/b的值存放到a中	a=a/b
%=	a%=b	a%b的值存放到a中	a=a%b

- 簡潔寫法的運算式範例：

```
i++          // 相當於 i=i+1
a-=3         // 相當於 a=a-3
b%=c         // 相當於 b=b%c
a/=b--       // 相當於計算 a=a/b 之後，再計算 b--
```

29

## 4.3 運算式

## 簡潔的運算式 (2/2)

- 使用簡潔運算式的範例：

```
01 // app4_6,簡潔運算式
02 public class app4_6
03 {
04     public static void main(String args[])
05     {
06         int a=5,b=8;
07
08         System.out.println("before compute, a="+a+",b="+b);
09         a+=b;                // 計算a+=b的值，此式相同於a=a+b
10         System.out.println("after compute, a="+a+",b="+b);
11     }
12 }
```

```
/* app4_6 OUTPUT-----
before compute, a=5, b=8
after compute, a=13, b=8
-----*/
```

30

## 4.3 課堂練習

- 設下列各運算式中，a的初值皆為10，b的初值皆為20，試寫出下列各式中，經運算過後的num、a與b之值：

**每一題的初始值：num=0, a=10, b=20**

- (a) num=(a++)-b
- (b) num=(-b)\*a
- (c) num=(a++)+(++b)
- (d) num=(--a)+(b--)
- (e) a+=a\*(b++)

**Ans:**

- (a) a=11,b=20,num=-10
- (b) a=10,b=20,num=-200
- (c) a=11,b=21,num=31
- (d) a=9,b=19,num=29
- (e) a=210,b=21,num=0

31

### 4.4 運算式的型態轉換

## 運算式的資料型態轉換 (1/4)

- Java處理型態轉換的規則：

byte→short→int→long

- 佔用位元組較少的轉換成位元組較多的型態。
- 字元型態會轉換成short型態（字元會取其unicode碼）。
- short型態遇上int型態，會轉換成int型態。
- float遇上int型態會轉換成float型態。
- 運算式中的某個運算元的型態為double，則另一個運算元也會轉換成double型態。
- 布林型態不能轉換至其它的型態。

32



## 4.4 運算式的型態轉換

## 運算式的資料型態轉換 (2/4)

- 型態轉換的範例：

```

01 // app4_7,運算式的型態轉換
02 public class app4_7
03 {
04     public static void main(String args[])
05     {
06         char ch='a'; //a=97
07         short a=-2;
08         int b=3;
09         float f=5.3f;
10         double d=6.28;
11
12         System.out.print("(ch/a)-(d/f)-(a+b)="); // 印出結果
13         System.out.println((ch/a)-(d/f)-(a+b));
14     }
15 }

```



```

/* app4_7 OUTPUT-----
(ch/a)-(d/f)-(a+b)=-50.18490561773532
-----*/

```

33

## 4.4 運算式的型態轉換

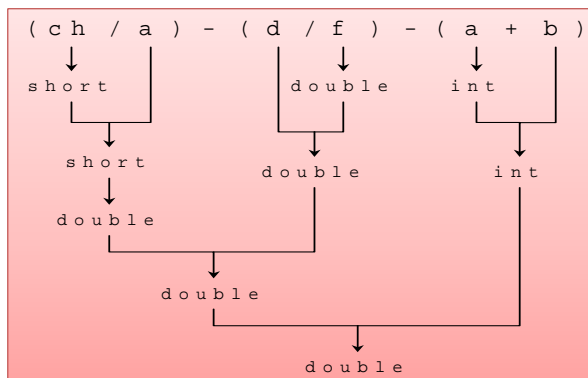
## 運算式的資料型態轉換 (3/4)

- 下圖為app4\_7內變數的資料型態轉換過程解說：

```

06 char ch='a';
07 short a=-2;
08 int b=3;
09 float f=5.3f;
10 double d=6.28;

```



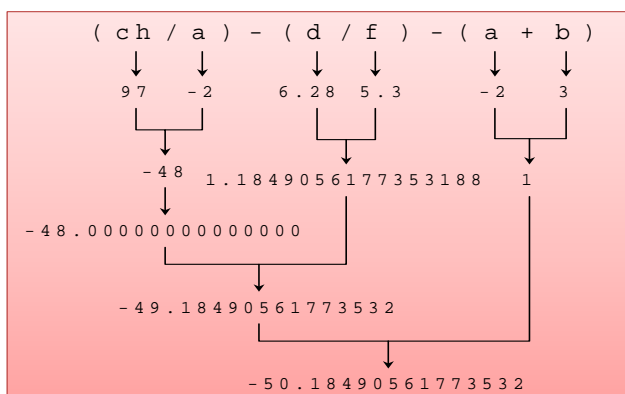
34

## 4.4 運算式的型態轉換

# 運算式的資料型態轉換 (4/4)

- 下圖為app4\_7運算式的運算過程：

```
06 char ch='a';
07 short a=-2;
08 int b=3;
09 float f=5.3f;
10 double
d=6.28;
```

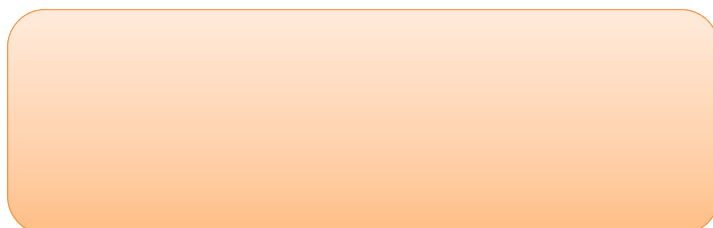


35

## 4.4 課堂練習

- 設有一程式碼，其變數的初值宣告如下：
 

```
char ch='h';
short s=61;
float f=5.3f;
int i=7;
double d=21.83;
```
- 在下面的運算式中，試仿照圖4.4.1與圖4.4.2的畫法，繪出資料型態的轉換過程與資料的運算過程：
 
$$d/(s+i)+(ch*f)=??$$



36

## 回家作業

- hw4\_1.java
- 請撰寫程式, 計算出變數  $i$  與變數  $j$  的和的平方。(請以鍵盤讀取所需輸入的資料)

請輸入  $i$  : 5  
請輸入  $j$  : 15  
 $i$  與  $j$  和的平方為 400

## 回家作業

- hw4\_2.java
- 請撰寫程式, 計算  $4 + 5 + 6 + \dots + 53 + 54 + 55$  的結果。(請以鍵盤讀取所需輸入的資料)

請輸入連加數目的起始數字 : 4  
請輸入連加數目的終點數字 : 55  
總和為 1534

## 回家作業

- hw4\_3.java
- 假設某個籠子裡有雞、兔若干隻, 共有 26 隻腳、8 個頭, 請撰寫程式分別算出雞與兔各有幾隻? ( 請以鍵盤讀取所需輸入的資料 )

請輸入有幾個頭:8  
請輸入有幾隻腳:26  
雞有 3 隻  
兔有 5 隻

## 回家作業

- hw4\_4.java
- 假設火車站的自動售票機只能接受 10 元、5 元、以及 1 元的硬幣, 請撰寫一個程式, 算出購買票價 137 元的車票時, 所需投入各種幣值硬幣最少的數量? ( 請以鍵盤讀取所需輸入的資料 )

請輸入票價: 137  
共需 13 枚 10 元硬幣  
1 枚 5 元硬幣  
2 枚 1 元硬幣