

# Java

## 數的運算



# 運算的五則符號

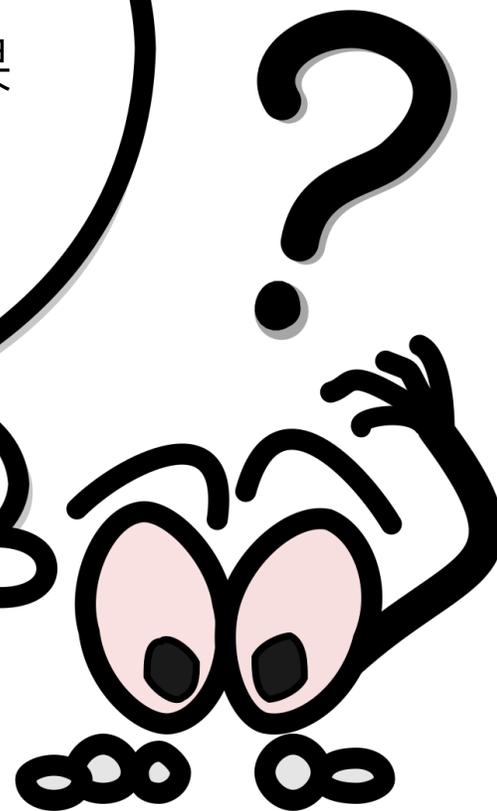
符號	功能
+	加
-	減
*	乘
/	除
%	取餘數

# 五則運算的用法

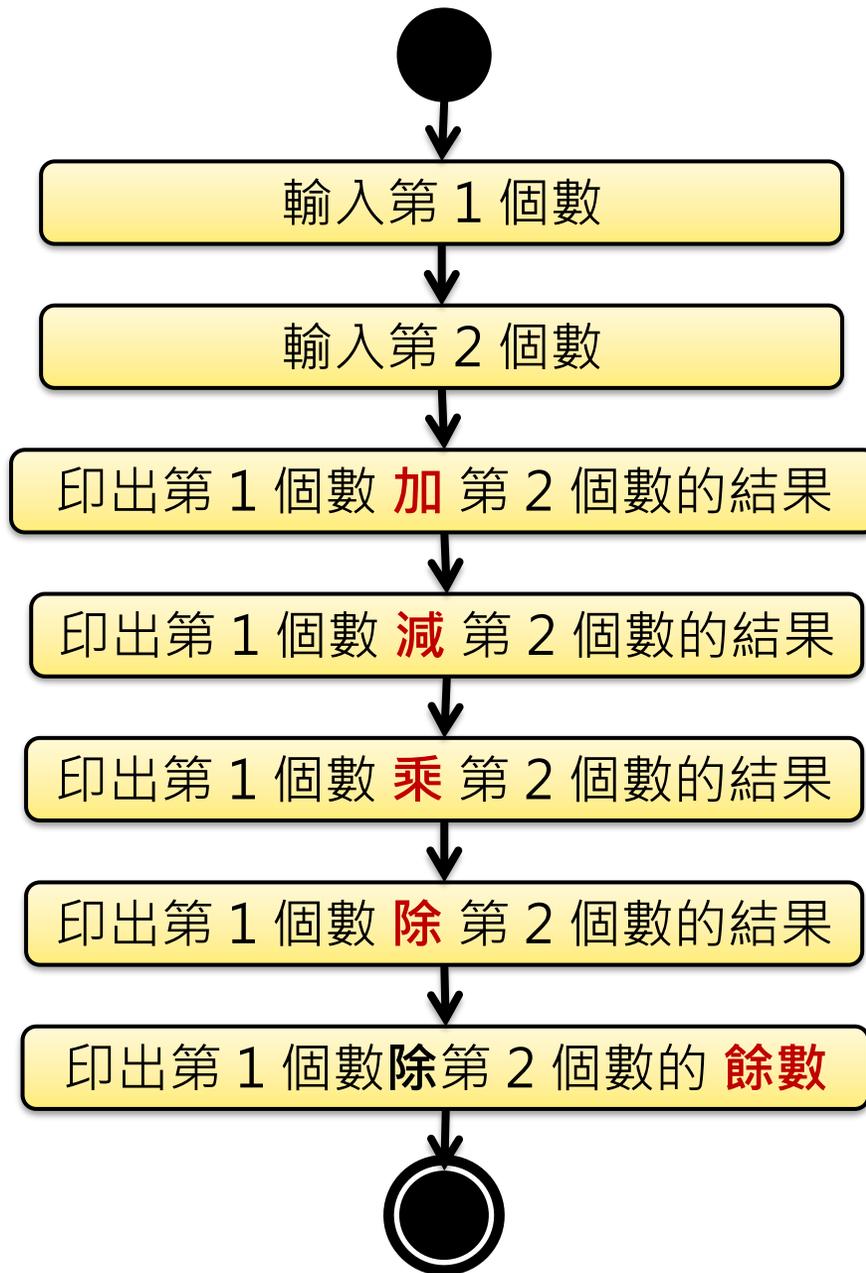
符號	功能	語法	實例
+	加	$x + y$	$10 + 3 = 13$
-	減	$x - y$	$10 - 3 = 7$
*	乘	$x * y$	$10 * 3 = 30$
/	除	$x / y$	$10 / 3 = 3.333$
%	取餘數	$x \% y$	$10 \% 3 = 1$

寫一個程式，  
輸入2個整數，  
印出兩數加、減、乘、除與餘數的結果

問題



# 流程



# Java 的五則運算

```
import java.util.Scanner;
class web{
    public static void main(String[] args) {
        int a , b;
        Scanner sc = new Scanner(System.in);
        a = sc.nextInt();
        b = sc.nextInt();

        System.out.printf("%d + %d = %d\n",a, b, a+b);
        System.out.printf("%d - %d = %d\n",a, b, a-b);
        System.out.printf("%d * %d = %d\n",a, b, a*b);
        System.out.printf("%d / %d = %d\n",a, b, a/b);
        System.out.printf("%d %% %d = %d\n",a, b, a%b);
    }
}
```



執行結果

輸入第1  
個整數

10  
2

輸入第2  
個整數

10	+	2	=	12
10	-	2	=	8
10	*	2	=	20
10	/	2	=	5
10	%	2	=	0

# JAVA

## 延伸的概念

# 概念1：算術運算子

符號
+
-
*
/
%

這 5 種運算符號稱為

**算數運算子**  
**(Arithmetic Operators)**

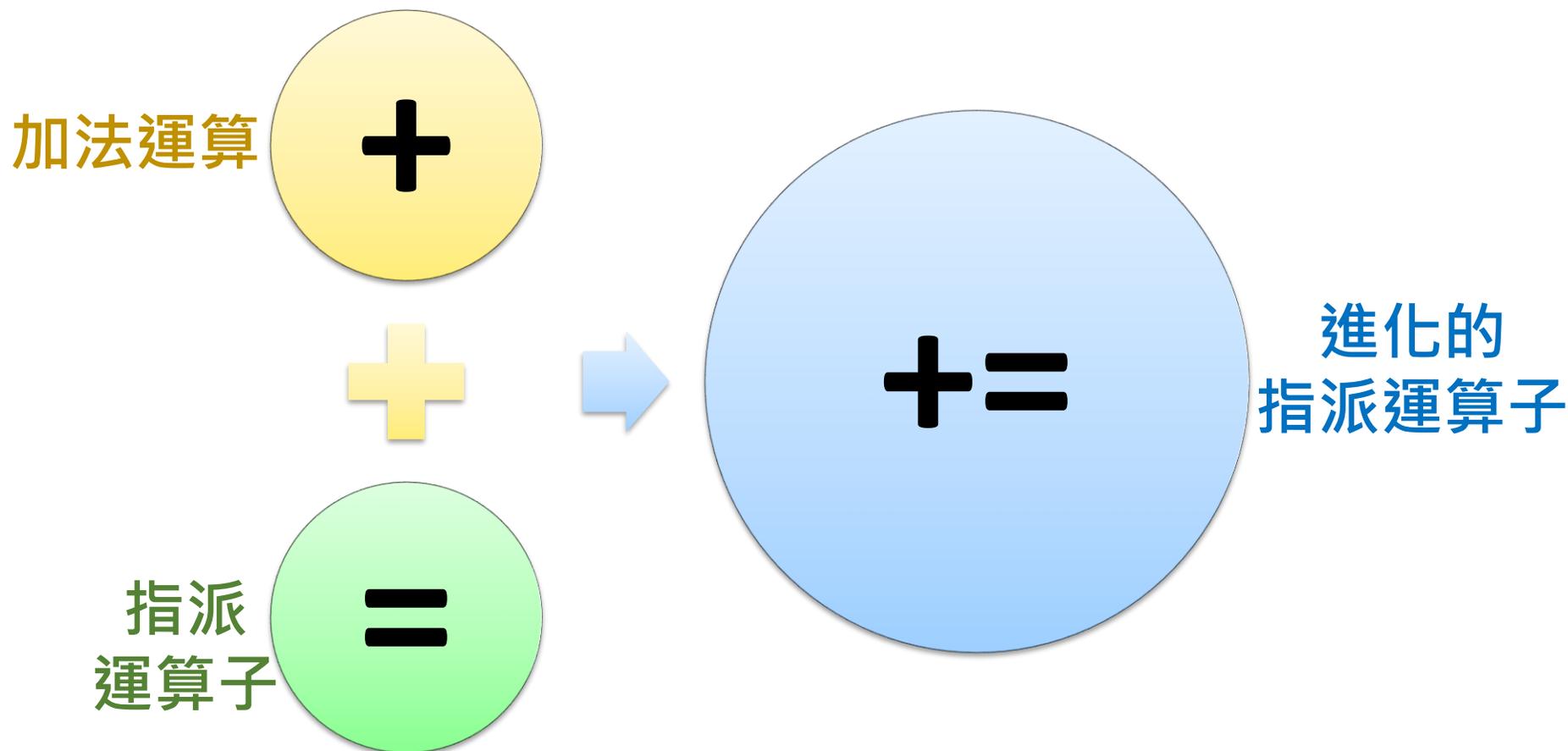


## 概念2：運算的優先順序，與數學運算是一樣的

- 跨號優先
- 乘、除(整數除法/符點除法)與取餘數優先
- 同一運算式含有數個加與減，由左至右逐一運算

# 概念3：更多的指派運算子(1)

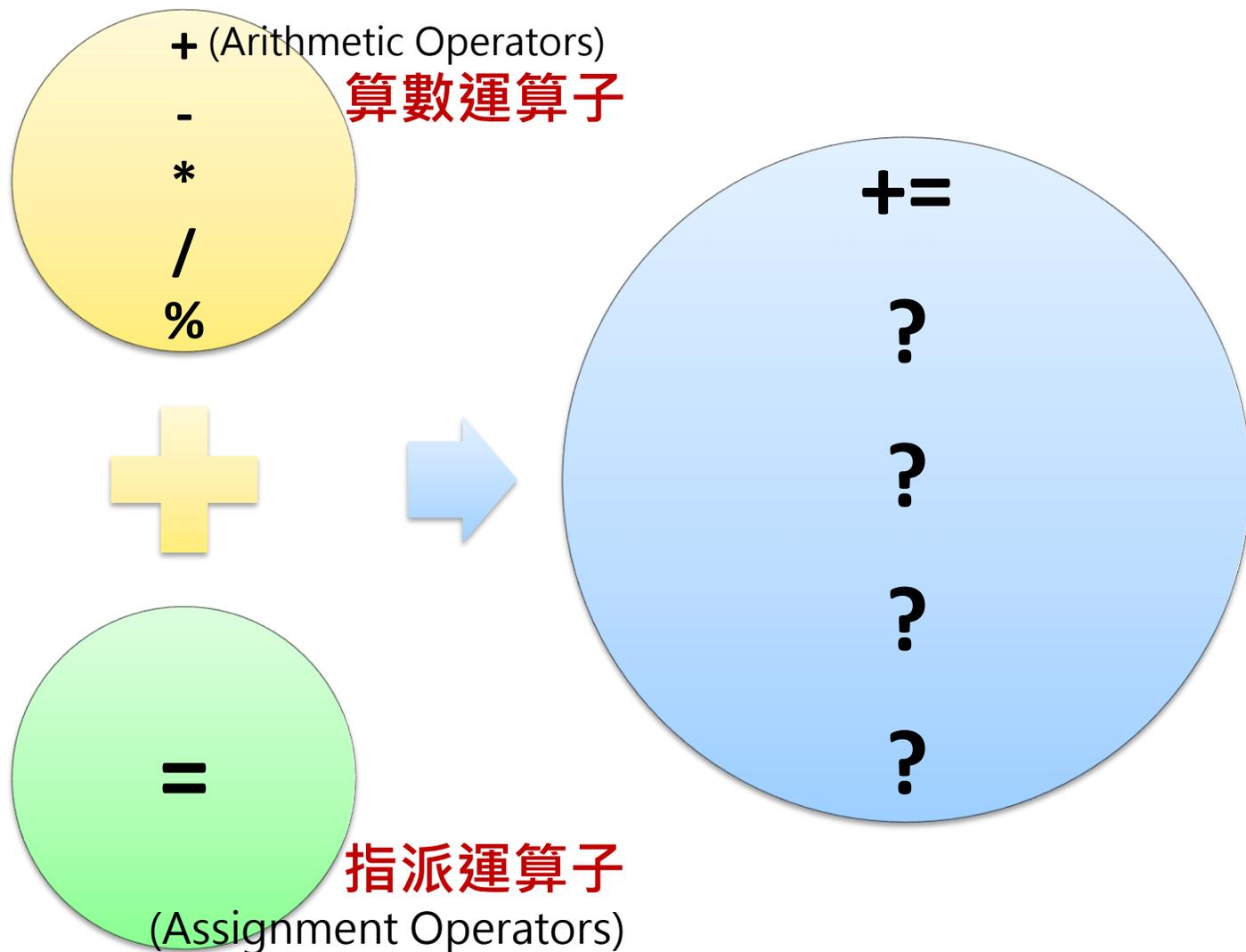
- 還記得**加法運算(+)**遇上**指派運算子(=)**變成什麼嗎？



# 概念3： 更多的指派運算子(2)

當其它的  
算數運算子  
(Arithmetic Operators)  
碰上  
指派運算子  
(Assignment Operators)  
又會變成什麼呢？

- 還記得**加法運算(+)**遇上**指派運算子(=)**變成什麼嗎？



# 概念3：更多的指派運算子(3)

算數運算子	用法
+	$x = x + y$
-	$x = x - y$
*	$x = x * y$
/	$x = x / y$
%	$x = x \% y$



指派運算子	用法
+=	$x += y$
-=	$x -= y$
*=	$x *= y$
/=	$x /= y$
%=	$x \% = y$

# 概念4：除法要小心(1)

兩個整數相除，  
不論是否會整除，  
預設都會得到整數值。

若相除的兩個數，  
至少含有一個浮點數，  
相除後的值自動就會  
是浮點數的形式。

```
class web{  
    public static void main(String[] args) {  
        System.out.println(10 / 2);  
        System.out.println(10 / 3);  
        System.out.println(10 / 3.0);  
        System.out.println(10.0 / 3);  
        System.out.println(10.0 / 3.0);  
        System.out.println((float)10 / 3);  
    }  
}
```



兩個整數相除，若想得到浮點數的結果，可以在運算式前面加上 **(float)**，就可以將運算結果轉成浮點數格式。

執行結果

```
5  
3
```

```
3.33333333333333333335  
3.33333333333333333335  
3.33333333333333333335
```

```
3.3333333
```

# 概念4：除法要小心(2)

整數與浮點數在系統中儲存的空間大小是不同的，因此使用 (float) 強迫從整數轉浮點數時，會有**精準度遺漏**的情況，因此使用 float 轉換後得到的小數位數與至少有一個浮點數相除後得到的小數位數是不相同的！



被除數	除數	執行動作	結果	例子
整數	整數	/	整數	$10 / 3 = 3$
整數	浮點數	/	浮點數	$10 / 3.0 = 3.3333333333333335$
浮點數	整數	/	浮點數	$10.0 / 3 = 3.3333333333333335$
浮點數	浮點數	/	浮點數	$10.0 / 3.0 = 3.3333333333333335$
整數	整數	相除並加上 (float)	浮點數	(float) $10 / 3 = 3.3333333$

# 概念5：印出 %

```
import java.util.Scanner;
class web{
    public static void main(String[] args) {
        int a , b;
        Scanner sc = new Scanner(System.in);
        a = sc.nextInt();
        b = sc.nextInt();

        System.out.printf("%d + %d = %d\n",a, b, a+b);
        System.out.printf("%d %% %d = %d\n",a, b, a%b);
    }
}
```

printf() 內若遇到 % 開頭時，通常代表是要輸出某變數，如 %d 是整數、%f 是浮點數，因此若只是要輸出 % 時，要在 printf 內寫 %% 才可正確輸出！



**%d %% %d = %d**

