



C++

算術運算子



算術運算子(arithmetic operator)分為兩大類

- 一元運算子(unary operator) (只需要一個運算元)

例如: 遞增/遞減運算子

$K = K++;$

- 二元運算子(binary operator) (需要兩個運算元)

$K = \text{num1} + \text{num2};$

$K = \text{num1} - \text{num2};$

$K = \text{num1} * \text{num2};$

$K = \text{num1} / \text{num2};$

$K = \text{num1} \% \text{num2};$

運算元(operand):常數(1 ,2, 3,.....)或是變數(比如: num1,num2,.....)。

算術運算子 - 二元運算子(binary operator)

運算子	說明	範例運算式
+	提供兩個運算元的加法運算	$x+y$
-	提供兩個運算元的減法運算	$x-y$
*	提供兩個運算元的乘法運算	$x*y$
/	提供兩個運算元的除法運算	x/y
%	提供兩個運算元的餘數運算	$x\%y$

運算元(operand)：常數(1, 2, 3,.....)或是變數(比如: num1,num2,.....)。

運算式(expression)：運算子(operator由)與運算元(operand)組成。

練習

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。

輸入範例：

5
2

輸出範例：

$$5+2=7$$

$$5-2=3$$

$$5*2=10$$

$$5/2=2\text{餘}1$$



參考程式碼-1

```
1 #include<iostream>
2 #include<iomanip>
3 using namespace std;
4 int main()
5 {
6     int a,b; //宣告兩個變數
7     cin>>a>>b; //輸入兩個整數變數
8     cout<<a<<"+"<<b<<"="<<a+b<<endl; //加
9     cout<<a<<"-"<<b<<"="<<a-b<<endl; //減
10    cout<<a<<"*"<<b<<"="<<a*b<<endl; //乘
11    cout<<a<<"/"<<b<<"="<<a/b<<endl; //除
12    return 0;
13 }
```

指定運算子(assignment operator)

(有很多種，先介紹其中一種)

將右邊的值指定給左邊

=

指定運算子 =

```
int a,b,result;  
cin>>a>>b;  
result = a+b;
```

result 會得到a+b

例子1：

```
int x = 50;      #x會得到50  
x = x+100;      #x會得到150
```


例子2：

將 j 的值指定給 i (如以下所示)。

$i = j;$

運算順序

與代數運算相同

- 括號優先
- 乘、除與取餘數優先
- 由左至右逐一運算



C++

延伸的概念

Extended concept



資料型態強迫轉換

```
#include <iostream>
#include <iomanip>
using namespace std;

int main() {
    int s,a,b;
    float f;
    cin>>f>>endl;
    f= (float)9/5*s+32;
    cout<<"華氏溫度為"<<fixed<<setprecision(1)<<f<<endl;
    return 0;
}
```

9.0/5.0=1.8

為什麼資料型態要強迫轉換?

如果沒有將int(整數型態)轉換成float(單精度浮點數型態)，會怎樣?

例子：假設a為浮點數，且 $a=9/5$ ，a的值為多少(四捨五入到小數點後第六位)？

```
#include <iostream>
#include <iomanip>
using namespace std;

int main(){
    float a;
    a=9/5;
    cout<<fixed<<setprecision(6)<<a<<endl;
    return 0;
}
```

結果：**1.000000**
(因為9和5為整數，所以商取到9/5的值的**整數位數**(1.8的1)。)

若有資料轉型，9/5的值(四捨五入到小數點後第六位)應為**1.800000**。

複合指定運算子(不只這些)

(compound assignment operator)

```
int c = 12, d = 12, e = 12, f = 12, g = 12;
```

複合指定運算子	範例運算式	展開式	指定值
+=	c += 7	c = c+7	c 得到 19
-=	d -= 4	d = d-4	d 得到 8
*=	e *= 5	e = e*5	e 得到 60
/=	f /= 3	f = f/3	f 得到 4
%=	g %= 9	g = g%9	g 得到 3

運算式(expression): 由運算子(operator)與運算元(operand)組成。

複合指定運算子+=

例子:

```
c = c+3;
```

利用設定運算子 += ,

縮寫成 **c += 3;**

參考程式碼

```
#include <iostream>
using namespace std;

int main(){
    int c=12,d=12,e=12,f=12,g=12;
    c+=7;
    d-=4;
    e*=5;
    f/=3;
    g%=9;
    cout<<c<<endl; /* 19 */
    cout<<d<<endl; /* 8 */
    cout<<e<<endl; /* 60 */
    cout<<f<<endl; /* 4 */
    cout<<g<<endl; /* 3 */
    return 0;
}
```


遞增運算子/ 遞減運算子

(increment operator/ decrement operator)

- 算術運算子(arithmetic operator)的其中一種
- 為一元運算子(unary operator)

遞增運算子	範例運算式	說明
++	++a	先將a遞增1，再以a的新值進行運算
++	a++	以a目前的值進行運算，再將a遞增1
遞減運算子	範例運算式	說明
--	--b	先將b遞減1，再以b的新值進行運算
--	b--	以b目前的值進行運算，再將b遞減1

運算式(expression): 由運算子(operator)與運算元(operand)組成。

遞增運算子範例

```
#include <iostream>
using namespace std;

int main(){
    int c;
    c = 5;
    cout<<c<<endl; /* 5 */
    cout<<c++<<endl; /* 5 */
    cout<<c<<endl; /* 6 */
    c = 5;
    cout<<c<<endl; /* 5 */
    cout<<++c<<endl; /* 6 */
    cout<<c<<endl; /* 6 */
}
```

遞減運算子範例

```
#include <iostream>
using namespace std;

int main(){
    int c;
    c = 5;
    cout<<c<<endl; /* 5 */
    cout<<c--<<endl; /* 5 */
    cout<<c<<endl; /* 4 */
    c = 5;
    cout<<c<<endl; /* 5 */
    cout<<--c<<endl; /* 4 */
    cout<<c<<endl; /* 4 */
}
```