Lecture 3

關係、布林、複合指定運算子
運算優先順序
基本輸入輸出

回顧

- 變數型態
  - 字元: **char**
    - 修飾字: const / unsigned
  - 整數: **short**, **int**, **long**
    - 修飾字: const / unsigned
  - 浮點數: **float**, **double**
    - 修飾字: const
  - 布林數: **bool**
    - 修飾字: const

```
char a = 65;
char b = 'A';
const char newline=13;

int c = 65;
unsigned short d = 65535;
const int three=3;

float pi = 3.14;
double E=3.0E8;
const double v=0.3;

bool old = true;
const bool male = false;
```


回顧

- 算術運算
  - 二元運算子: +, -, *, /, =, %
  - 一元運算子: --, ++
    - 前置、後置

```
int a = 3;
cout << a++;
cout << ++a;
int b = a++;
cout << a;
```


今日內容

- 運算
  - 關係運算子
  - 布林運算子
  - 複合指定運算子
  - 運算的優先順序
- 基本輸入輸出

關係運算子

Relational Operator

關係運算子 / Relational Operator

- 目的：對兩個資料 (運算元) 作比較以判斷他們的關係。
- 運算結果：為一布林值，代表所判斷關係真假。
- 六種關係運算子

運算元	我會這樣唸	動作
==	等於等於	判斷左右是否相等
!=	不等於	判斷左右是否不等
>	大於	判斷左邊是否大於右邊
<	小於	判斷左邊是否小於右邊
>=	大於等於	判斷左邊是否大於或等於右邊
<=	小於等於	判斷左邊是否小於或等於右邊

7

範例 (3-1.cpp)

```
int a = 10, b = 4;
bool c = (a == b);
bool d = (a != b);
bool e = (a >  b);
bool f = (a <  b);
bool g = (a >= b);
bool h = (a <= b);
```

請判斷各變數值！

答案：a=10; b=4; c=false; d=true; e=true;f=false;g=true;h=false;

8

範例 (3-2.cpp)

```
int a = -5, b = 5;
int c = (a == b);
int d = (a != b);
int e = (a >  b);
int f = (a <  b);
int g = (a >= b);
int h = (a <= b);
```

請判斷各變數值！

答案：a=-5; b=5; c=0; d=1; e=0;f=1;g=0;h=1;

9

布林運算子

Boolean Operator

10

布林運算子 / Boolean Operator

- 目的：對布林值作運算時，為用來「複合」關係運算子的結果。
- 運算結果：布林值。
- 又稱為邏輯運算子 (logical operator)
- 三種布林運算子

運算元	英文名稱	中文名稱	動作
&&	AND	且	所運算之兩運算元皆為真時，運算結果為真
||	OR	或	所運算之兩運算元任一為真時，運算結果即為真
!	NOT	否定	否定所運算之運算元 (真 → 假、假→真)

以上何者為二元運算子，何者為一元運算子？

11

範例 (3-3.cpp)

```
bool a = false;
bool b = true;
bool c = (a && b);
bool d = (a || b);
bool e = ! a;
```

請判斷各變數值！

答案：a=false; b=true; c=false; d=true;e=true

12

範例 (3-4.cpp)

```
bool a = true;
bool b = true;
bool c = (a && b);
bool d = (a || b);
bool e = ! a;
```

請判斷各變數值！

答案：a=true; b=true; c=true; d=true; e=false

13

真值表 (truth table)

- 真值表列出了各種邏輯運算的結果
- 在C/C++語言中，非0為真(true)，0為偽(false)

&&	0	1
0	0	0
1	0	1

||	0	1
0	0	1
1	1	1

14

小結

- **關係運算子**與**布林運算子**常放在條件敍述句中用來控制流程/選擇性地執行程式 (下週)
- 關係運算子：==、!=、>、<、>=、<=
- 布林運算子：&&、||、!

15

複合指定運算子

Compound Assignment Operator

16

算術運算的簡寫

- 在 C/C++ 裡，有一些簡寫可以讓程式撰寫人員減少打字 ...

```
A = A + 3;  → A += 3;
B = B - 4;  → B -= 4;
C = C * 5;  → C *= 5;
D = D / 2;  → D /= 2;
E = E % 6;  → E %= 6;
```

- 以上這些運算子 (+=, -=, *=, /=, %=) 稱為複合指定運算子

17

運算子小結

- 指定運算子: =
- 算術運算子: +、-、*、/、%、++、--
- 關係運算子: ==、!=、>、<、>=、<=
- 布林運算子: &&、||、!
- 複合指定運算子: +=、-=、*=、/=、%=

18

運算子優先順序

Precedence of operators

19

運算子優先順序

- 當今天一行程式敍述內含多個運算子時，何者先算、何者後算乃根據運算子的優先順序 (下頁中的 level) ，當順序相同時則根據表中的 grouping 決定運算方向。
- 除了先乘除後加減這個大家從小學都知道的規則外，其它在寫運算敍述時，儘量以括弧將想要先運算的部份括起來。
- a = 3 + 4 – 5;
  - 查表可知 + - level 為 7、= level 為17
  - + - 同 level, 根據 grouping 可知運算方向由左至右
  - 3 + 4 → 7
  - 7 – 5 → 2
  - a = 2;

20

範例 (3-5.cpp)

b = c = 8; (運算方向由右至左 → 8 存入 c, c 存入 b)
c *= 3 + 3 * 2 > b --;

- *=: level 17
- *: level 6
- +: level 7
- >: level 9
- -- (postfix): level 2

c *= 3 + 3 * 2 > (**b --**) → c *= 3 + (**3*2**) > 8
→ c *= (**3 + 6**) > 8 → c *= (**9 > 8**) → c *= 1;

結果: b = 7, c = 8;

21

運算子優先順序 (level) 與運算方向 (grouping)

Level	Operator	Description	Grouping
2	() ++ --	postfix	Left-to-right
3	++ -- !	unary (prefix)	Right-to-left
4	(type)	type casting	Right-to-left
6	* / %	multiplicative	Left-to-right
7	+ -	additive	Left-to-right
8	<< >>	shift	Left-to-right
9	< > <= >=	relational	Left-to-right
10	== !=	equality	Left-to-right
14	&&	logical AND	Left-to-right
15	||	logical OR	Left-to-right
17	= *= /= %= += -=	assignment	Right-to-left

http://www.cplusplus.com/doc/tutorial/operators/

22

運算子優先順序 (由高而低)

- 括弧
- 一元運算子 (後置)
- 一元運算子 (前置)
- 型別轉換
- 算術運算子 (先乘除後加減)
- << 、 >>
- 關係運算子 (先大小關係、後相等關係)
- 布林運算子／邏輯運算子 (&& > ||)
- 指定

23

基本輸入輸出

Basic Input Output

24

基本輸入輸出

- cout (console output) 用來進行輸出至螢幕上。
  ```
  int a=3;
  cout << a;
  ```
- cin (console input) 用來從鍵盤輸入資料，經過適當地轉換成為目的變數的型態後，存入目的變數內。
  ```
  int a;
  cin >> a;
  ```
- 在此， <<、>> 稱為串流運算子 (stream operator)。
- cin, cout 都可以連結多個串流運算子進行輸入輸出的操作。

25

3-6.cpp

```
#include <iostream>
using namespace std;
int main() {
  int a, b;

  cout << "請輸入 a: ";
  cin >> a;
  cout << "請輸入 b: ";
  cin >> b;
  cout << "\na+b=" << a+b;
  cout << "\na-b=" << a-b;
  cout << "\na*b=" << a*b;
  cout << "\na/b=" << a/b;
  cout << "\na%b=" << a%b << endl;
  return 0;
}
```

若輸入的 a = 3, b=5, 輸出的結果?

注意到 \n，其中 \ 稱為 escape sequence，主要用來輸入一些鍵盤無法直接輸入的字元。

26

一些常見的跳脫字元

\n	new line (換行)
\b	back space (擦去前一個字)
\r	carriage return (回到行開頭)
\a	alert (警告音)
\t	horizontal tab (水平定位)
\”	雙引號
\’	單引號
\\	backslash (反斜線)
\o	null character
\xhh	以 16 進位方式指定字碼 (e.g. \x41 → ‘A’)

27

3-7.cpp

```
#include <iostream>
using namespace std;
int main() {
  int a, b;

  cout << "請輸入 a 與 b: ";
  cin >> a >> b;

  cout << "\na+b=" << a+b
       << "\na-b=" << a-b
       << "\na*b=" << a*b
       << "\na/b=" << a/b
       << "\na%b=" << a%b << endl;
  return 0;
}
```

若輸入 3 5, 輸出的結果?

28

輸出格式的設定

Formatted Output

29

輸出格式設定 iomanip

- 使用 cout 輸出資料時，內定格式依資料型別與數字大小決定。
- 若要設定數字的格式，可使用一些定義於 iomanip 標頭檔內的功能：
  - setprecision(n): 設定接下來所有輸出資料的精確度為 n
  - setw(n): 設定下一個輸出資料的最小寬度為 n
  - setfill(ch): 設定接下來所有輸出有寬度時，空白地方所填入的字元

30

3-8.cpp

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

int main() {
  double a = (1.0/3.0);
  float b = (1.0f/3.0f);

  cout << setprecision(20) << a << "," << b << endl;

  return 0;
}
```

在程式沒有指精度的數字,
整數內定為 int 型別
浮點數內定為 double 型別

請試著用 f 拿掉看看結果如何

```
0.33333333333333331,0.33333334326744080
```

31

3-9.cpp

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

int main() {
  cout << setw(5) << "*" << endl;
  cout << setw(4) << "*" << endl;
  cout << setw(3) << "*" << endl;
  cout << setw(2) << "*" << endl;
  cout << setw(1) << "*" << endl;

  return 0;
}
```

32

3-10.cpp

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

int main() {
  cout << setfill('*');
  cout << setw(1) << "-" << endl;
  cout << setw(2) << "-" << endl;
  cout << setw(3) << "-" << endl;
  cout << setw(4) << "-" << endl;
  cout << setw(5) << "-" << endl;

  return 0;
}
```

33

記得它們是作什麼的嗎?

- cin
- cout

- setw()
- setprecision()
- setfill()