

# 計算機程式設計

Computer Programming

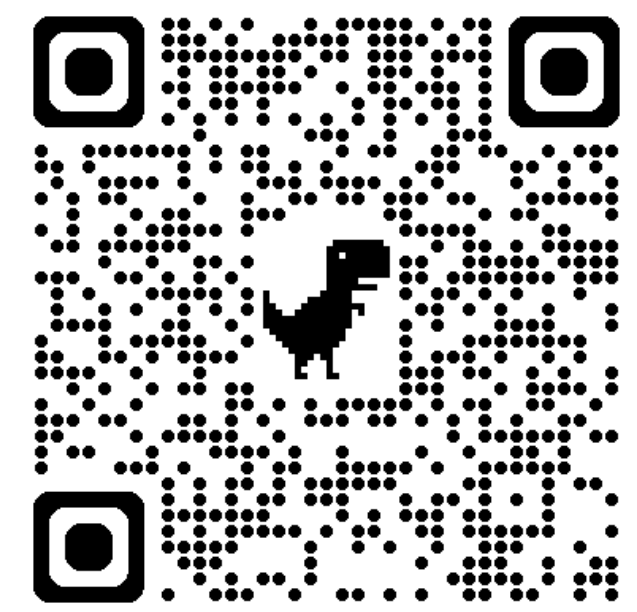
## Data Types and Operators

Instructor: 林英嘉

2024/09/16



[W2 Slido: #6981334](#)



[GitHub repo](#)

# Outline

---

- 算數運算子
- C語言常見資料型態
- 關係運算子
- 邏輯運算子以及 if

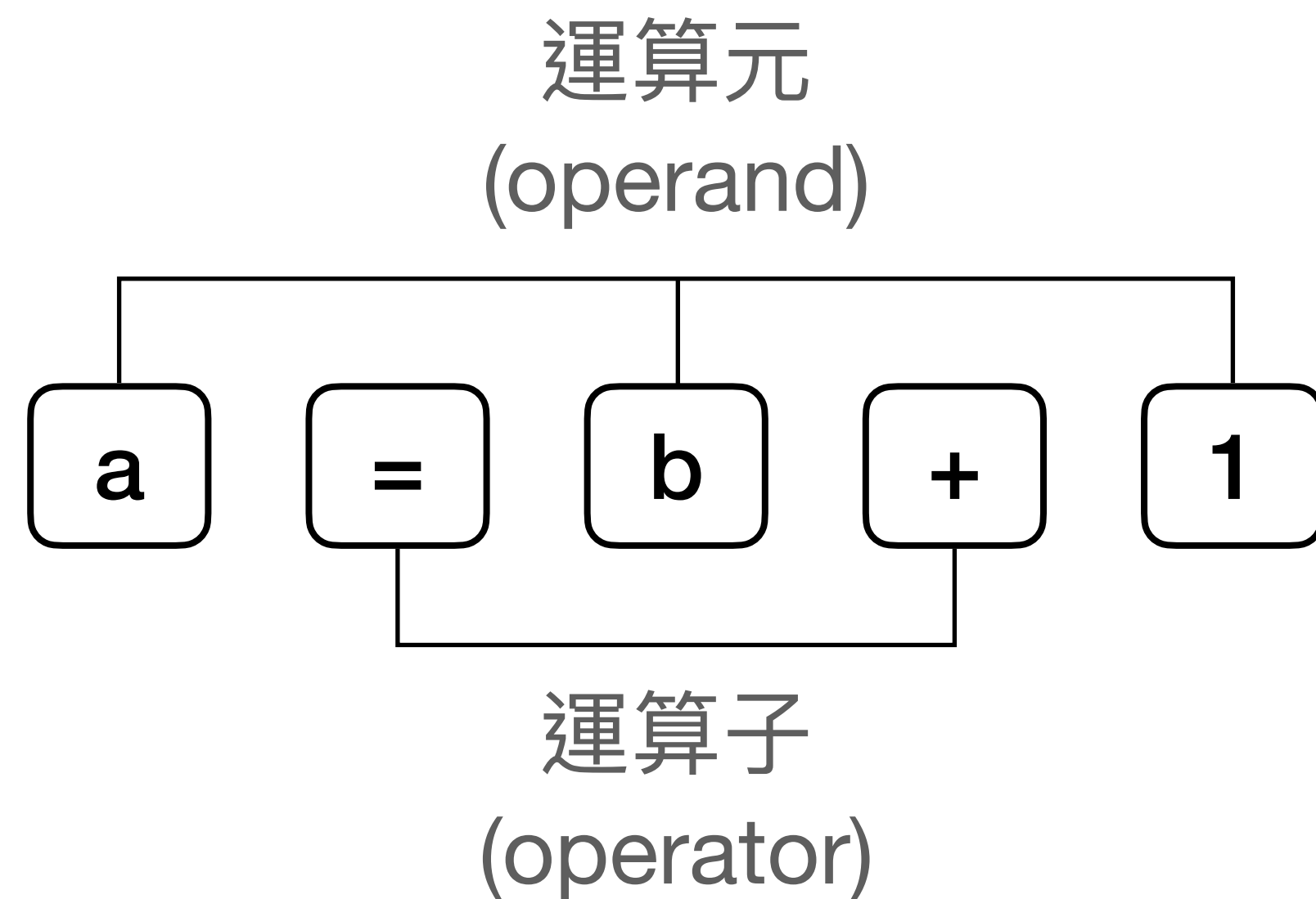
# 算數運算子 (Arithmetic Operators)

---

- 又稱為「運算符」
- 可以對於兩個數值進行運算

Operator	Meaning	Example in C
+	加	$a + b$
-	減	$a - b$
*	乘	$a * b$
/	除	$a / b$
%	取餘數	$a \% b$
-	取負數	$-a$

# 運算元與運算子



- 一元運算子 (Unary Arithmetic Operators)
  - 只需要一個運算元的運算子
  - 取正 / 負
- 二元運算子 (Binary Arithmetic Operators)
  - 需要兩個運算元的運算子
  - 加 / 減 / 乘 / 除 / 取餘數

# 進行四則運算

```
#include <stdio.h>
int main(){
    int a, b;
    scanf("%d %d", &a, &b);
    printf("a + b 等於 %d\n", a + b);
    printf("a - b 等於 %d\n", a - b);
    printf("a * b 等於 %d\n", a * b);
    printf("a \\/ b 等於 %d\n", a / b);
    printf("a %% b 等於 %d\n", a % b);
    printf("把 a 取負數 : %d", -a);
}
```

\表示跳脫符號

%%表示會印出百分比

# 常見使用於printf的跳脫序列

表示方式	\	%%	\\	\“	\‘
實際印出	/	%	\	“	‘
功能	印出斜線	印出百分比	印出反斜線	印出雙引號	印出單引號

# 精簡寫法：增量賦值 / 複合賦值

```
#include <stdio.h>
int main(){
    int a = 2, b = 1
    a += b;
    printf("%d\n", a);
    a -= b;
    printf("%d\n", a);
    a *= b;
    printf("%d\n", a);
    a /= b;
    printf("%d\n", a);
    a %= b;
    printf("%d\n", a);
}
```

# 精簡寫法：遞增運算子, 遞減運算子

```
#include <stdio.h>
int main(){
    int a = 2, b = 1;
    a++; // a = a + 1;
    printf("%d\n", a);
    a/=b--; // 先用b的值進行運算，再將b減1
    printf("a值為%d\n", a);
    printf("b值為%d\n", b);
}
```

++ 遞增運算子

-- 遞減運算子



# 計算優先順序

---

```
1 #include <stdio.h>
2 int main(){
3     int i = 1, j = 2, k = 3;
4     printf("%d\n", i + j * k); // 答案為7
5     printf("%d\n", (i + j) * k); // 答案為9
6 }
```

# C語言的資料型態

C Data Types

# C語言中常見的資料型態

---

int	char	float	double
整數	字元	浮點數	倍準浮點數

# Format Specifiers

---

Type	int	char	float	double
Meaning	整數	字元	浮點數	倍準浮點數
大小 (Byte)*	4	1	4	8
Specifier	%d	%c	%f	%lf

\*以64-bit系統為例

# 變數初始化 (以float為例)

---

```
#include <stdio.h>
int main() {
    float a = 0.05; 宣告變數 (declaration)
    printf("%f", a);
}
```

# 比較float與double的精度

---

```
#include <stdio.h>
int main(){
    float pi = 3.14159265358979323846;
    double double_pi = 3.14159265358979323846;
    printf("pi: %.10f\n", pi);
    printf("double pi: %.10f\n", double_pi);
}
```

- float: 精度約 6 位十進制之小數
- double: 精度約 15-16 位十進制之小數
  - 在printf中，%f和%lf會產生相等的結果

# 使用scanf讀取double值時一定要用%lf

---

```
#include <stdio.h>
int main() {
    double d;
    scanf("%lf", &d); // this will cause an error.
    printf("d = %f", d);
}
```

# 資料類型比較

	大小 (Byte)*	Specifier	數值範圍
int	4	%d	-2,147,483,648 到 2,147,483,647 (範圍2的32次方)
char	1	%c	-128 到 127 或 0 到 255 (取決於是否有符號)
float	4	%f	約 1.2E-38 到 3.4E+38，精度約 6 位十進制之小數
double	8	%lf	約 2.2E-308 到 1.7E+308，精度約 15-16 位十進制之小數

\*以64-bit系統為例



# sizeof

---

```
#include <stdio.h>
int main(){
    int int_a;
    printf("Size of int: %d\n", sizeof(int_a));
    float float_a;
    printf("Size of float: %d\n", sizeof(float_a));
    char char_a;
    printf("Size of char: %d\n", sizeof(char_a));
    double double_a;
    printf("Size of double: %d\n", sizeof(double_a));
}
```

# 溢位

```
1 #include <stdio.h>
2 int main(){
3     int int_a;
4     scanf("%d", &int_a);
5     printf("%d\n", int_a);
6     int_a ++;
7     printf("%d\n", int_a);
8 }
```

輸入	2147483647
輸出	2147483647 -2147483648

# 溢位定義

---

- 算術溢位 (arithmetic overflow)
- 「溢位」是當一個數值超出其能夠表示的範圍時發生的
- 實作時需要注意變數的數值與資料型態

# 自動轉換資料型態

---

```
#include <stdio.h>
int main(){
    int a = 100;
    float b = 0.5;
    printf("自動轉型浮點數 a + b = %f\n", a + b);
}
```

- 型別提升 (Type Promotion): int -> float -> double
- 因為float的表示範圍比int大，double的表示範圍又比float大

# 手動轉換資料型態 (說明)

---

手動資料型態轉換用法 (Type Casting)

(type) 表達式

# 手動轉換資料型態

---

```
#include <stdio.h>
int main() {
    int i = 10;
    float f = 3.14;
    int result;

    result = i + (int)f;
    printf("Result: %d\n", result);
    printf("Result float: %f\n", (float)result);
}
```

# 手動轉換資料型態以避免不符合預期的結果

```
#include <stdio.h>
int main() {
    int i = 5;

    // 不進行強制轉型
    float result1 = i / 2;
    // 進行強制轉型
    float result2 = (float)i / 2;

    printf("不進行強制轉型的結果: %f\n", result1);
    printf("進行強制轉型的結果: %f\n", result2);
}
```

# 手動轉換資料型態 (範例3)

---

```
#include <stdio.h>
int main() {
    int score1 = 95, score2 = 96;
    float average;
    int count = 2;
    printf("平均: %.2f\n", (float)(score1 + score2) / count);
}
```


將和轉型




# 浮點數？

---

- 浮點數 (floating point number) 是基於科學記號來進行表示的數值

$$123456.789 \longrightarrow 1.23456789 \times 10^5$$


- 小數點可以根據指數的位置「浮動」，因此稱為浮點數，方便進行運算或近似

$$1.2345 \times 10^5 + 1.23 \times 10^3 = 123.45 \times 10^3 + 1.23 \times 10^3$$


$$1.23456789 \times 10^5 \approx 1.234568 \times 10^5$$

# 控制printf輸出小數點位數

---

- 控制printf輸出小數點位數的格式化字串：`%.nf`

```
printf("%.nf", 浮點數變數);
```

`n`代表代表要保留的小數位數

# 控制printf輸出小數點位數

- 控制printf輸出小數點位數的格式化字串：%.nf

```
# include <stdio.h>
int main(){
    float pi = 3.1415926;
    printf("(保留2位小數) pi = %.2f\n", pi);
    printf("(保留3位小數) pi = %.3f\n", pi);
    printf("(保留4位小數) pi = %.4f\n", pi);
}
```

# 利用printf輸出小數點位數

---

- printf 會根據要求保留的小數位數，對該位置後的數字進行四捨五入操作
- 若沒有指定欲保留的小數位數，則printf預設輸出6位小數點位數

# C語言邏輯控制

# 關係運算子

---

Operator	Meaning	C Example
>	大於	a > b
<	小於	a < b
>=	大於等於	a >= b
<=	小於等於	a <= b
==	等於	a == b
!=	不等於	a != b

- C語言邏輯控制
- 判斷變數與變數之間的關係
  - 可以得到布林值 (Boolean) ，也就是 True (真) 或 False (假)

# 變數關係判斷

---

```
#include <stdio.h>
int main(){
    int a, b, c;
    scanf("%d %d", &a, &b);
    printf("a是否等於b: %d\n", a == b);
    c = a != b;
    printf("c的值為%d\n", c);
}
```

- 1代表 True (真)
- 0 代表 False (假)

# 三個變數相等嗎？

---

```
#include <stdio.h>
int main(){
    int a = 5, b = 5, c = 1;
    printf("%d\n", a == b == c);
    printf("%d\n", a == b);
    printf("%d", a == b && b == c);
}
```



# 邏輯運算子

---

Operator	Meaning	C Example
&&	且	a && b
	或	a    b
!	非	!a

# 變數邏輯判斷 (1)

---

```
#include <stdio.h>
int main(){
    int a = 1, b = 0, c = 0;
    printf("a && b 等於 %d\n", a && b);
    printf("a || b 等於 %d\n", a || b);
    printf("b && c 等於 %d\n", b && c);
    printf("b || c 等於 %d\n", b || c);
    printf("非 b 等於 %d\n", !b);
}
```

## 變數邏輯判斷 (2)

---

```
#include <stdio.h>
int main(){
    int a = 1, b = -1;
    printf("a && b 等於 %d\n", a && b);
    printf("a || b 等於 %d\n", a || b);
}
```

- 在 C 語言中，邏輯運算符 && 和 || 是基於布林邏輯運行的。即使數值是負的，只要不是零，邏輯運算中都會視為 "真" (True)

# 變數關係+邏輯判斷

---

```
#include <stdio.h>
int main(){
    int a = 1, b = 2, c = 3, d;
    d = (a > b) && (b <= c);
    printf("d的值為%d\n", d);
    d = (a > b) || (b < c);
    printf("d的值為%d\n", d);
}
```

# 運算子優先順序說明

優先順序	Operator	Meaning	連在一起用？
1	()	大於	由左至右
2	[]	小於	由左至右
3	! + -	非、取正負	由右至左
4	++ --	遞增、遞減	由右至左
5	* / %	算數運算子	由左至右
6	+ -	算數運算子	由左至右
7	> >= < <=	關係運算子	由左至右
8	= = !=	關係運算子	由左至右
9	&&	邏輯運算子	由左至右
10		邏輯運算子	由左至右
11	=	設定運算子	由右至左

# 運算子優先順序

---

```
#include <stdio.h>
int main() {
    int a = 5, b = 10, c = 15;
    // 優先順序將決定運算順序
    int result = a < b || b > c && a == 5;
    printf("Result: %d\n", result);
    return 0;
}
```

# if 敘述

---

- 程式語言的關係和邏輯運算經常會用到 if 敘述

```
if (判斷條件)  
    判斷主體；
```

代表「如果 ... 就 ...」

# if 敘述範例

---

```
#include <stdio.h>
int main(){
    int a = 5;
    int b = 10;
    if (a != b)
        printf("a is not equal to b\n");
        printf("b is not equal to a\n");
}
```



# if 敘述搭配兩種以上的條件

```
#include <stdio.h>
int main(){
    int a = 5;
    int b = 10;
    if ((a < b) && (a >= 0))
        printf("a < b\n");
        printf("a >= 0\n");
    if ((a < b) || (b > 10))
        printf("Or 成立\n");
}
```

注意! 如果在if中使用兩種以上的條件，需要再多一層小括號