

# 計算機程式設計

## C語言 Basic

郭忠義

[jykuo@ntut.edu.tw](mailto:jykuo@ntut.edu.tw)

臺北科技大學資訊工程系

# 基本C程式

- ❑ 每一個C程式一定有一個main function(函式)
  - C程式執行的進入點
  - C程式可由許多function組成，但只能有一個main function
- ❑ function 語法

```
回傳值資料型別 函式名稱(參數資料型別 參數名稱, ...) {  
    return 回傳值;  
}
```

- 回傳值資料型別: int(整數), void(無), double(浮點小數), ...
- 參數資料型別: int(整數), void(無), double(浮點小數), ...
- 函式名稱: 除了main為預設之外，其他自訂
- 回傳值: 回傳一個計算後得到的數值，給呼叫此函式的程式
- return是保留字，後續會進一步介紹函式

# 基本C程式

## □ main function

```
#include <stdio.h>
int main(void) {
    int i;          // 宣告變數 i，這一行是註解
    /* 註解有兩種，這是多行註解
    */
    i = 5;
    printf("%d\n", i);
    return 0;
}
```

- {} 大括號裡面稱為 程式區塊 (Block)
- 程式的指令/敘述，稱為 instruction, statement
- 每一行指令，後面一定要以分號(;)結束
- 程式不同層次，最好能 **空四個space**以增加可讀性

# 基本C程式

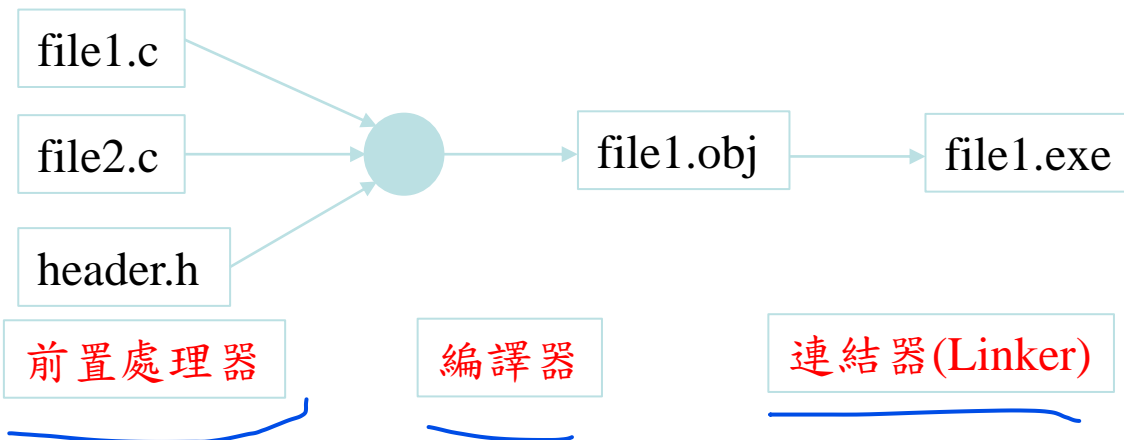
## □ main function

- `stdio.h` 是標準輸出輸入(`standard input out`)函式庫，`.h` 是附屬檔名，`h` 是標頭檔(header file)
- `int i;` 是宣告變數 (variable) `i` 的資料型別是整數，`i` 是程式師命名
- `i = 5;` 是將 5 這個值指定(= assignment)給 `i`。
- `printf` 是 `stdio.h` 函式庫提供的一個函式功能，輸出值到螢幕(標準輸出)
- `printf()`，括號內是參數，有兩個，一個是格式化旗標，第二個是要輸出變數資料
- `main` 後面括號，DOS 傳遞參數，`void` 表示不傳參數
- `return 0` 是將 0 傳回給 DOS。

# 基本C程式

## ❑ 前置處理器(preprocessor)

- # 後程式由前置處理器處理，之後再由編譯器(compiler)處理
- #define，定義一個符號的值，程式之後使用這個符號代表此值
  - #define start 0，定義start代表0，前置處理器先將程式中start全換成0
- #include<內建函式庫檔名>表示載入一個內建函式庫 (library)
- #include"自訂函式庫檔名"表示載入一個自訂函式庫 (library)
- #後面不需要分號;結束



# 變數名稱的使用

- ❑ 變數(variable)，在程式中值是可以改變的，可以指定各種不同的值。
  - 常數(Constant)，在程式中一經指定，值是不可以改變的。
    - `const int i = 9;`      `#define i 9`
- ❑ C語言變數(variable)名稱使用，須以下列三種字元做開頭：
  - 大寫字母。
  - 小寫字母。
  - 底線(\_)
- ❑ 變數名稱由下列四種字元所構成：
  - 大寫字母
  - 小寫字母
  - 底線(\_)
  - 阿拉伯數字0~9

# 變數名稱的使用

- ❑ 大寫和小寫字母代表不同變數，下列代表三個不同變數。
  - sum
  - Sum
  - SUM
- ❑ 系統保留字(又稱關鍵字)，在C編譯程式中代表特別意義，不可使用這些字為變數名稱。下列是ANSI C語言保留字。
  - auto    break    case    char    continue    default
  - do       double    else    enum    extern    float
  - for    goto    if    int    long    register
  - return    short    sizeof    static    struct    switch
  - typedef    union    unsigned    void    while

# Exercise

□ 以下那些變數是合法，哪些是非法

○ SUM

○ Hung

○ sum\_1

○ Fg

○ X5

○ y61

~~○ sum,1~~

~~○ 3y~~

~~○ x\$2~~



# 變數名稱的使用

## ❑ 合法的變數名稱：

- SUM, Hung, sum\_1, Fg, X5, y61

## ❑ 不合法的變數名稱：

- sum,1 ← 變數名稱不可有"，"符號
- 3y ← 變數名稱不可由阿拉伯數字開頭
- x\$2 ← 變數名稱不可含有"\$"符號

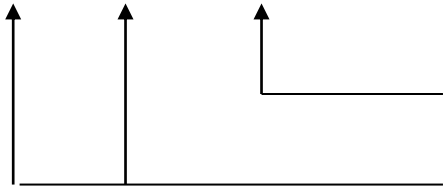
## ❑ 大寫和小寫字母代表不同變數，下列代表三個不同變數。

- sum
- Sum
- SUM

# 變數宣告

- ❑ 將i，j，k三個數宣告為整數，下列宣告均是合法的。

int i, j, k;



整行宣告以";"為結束

各變數間用", "隔開

```
int i;  
int j;  
int k;
```

- 由於i和j間用", "隔開，所以上述是合法宣告方式。
- ❑ 宣告完由";"結束，也可在宣告變數的同時，設定變數的值。
  - 將i宣告成整數，並將其設定成7。
  - `int i = 7;`
- ❑ 把所有的變數集中在程式的開頭宣告，有利於程式的維護。

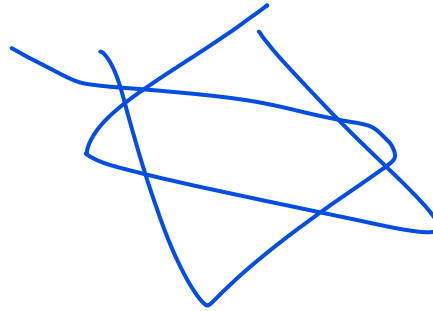
# Exercise

- 撰寫一個main function
  - 設定矩形的高與寬兩個整數變數值分別為15, 12 ,
  - 計算面積和周長並輸出

# 基本資料型別

□ C語言的基本資料型別有：

- void：無型別
- int：整數
- char：字元
- float：單精度浮點數
- double：雙倍精度浮點數



# 基本資料型別

## ❑ void：無型別

- 非屬任何型別，可以轉成任何型別
- 放在函式宣告參數中，表示函數不需要傳入任何參數
- 放在函式宣告回傳資料型別中，表示函數不需要回傳任何值

```
回傳值資料型別 函式名稱(參數資料型別 參數名稱, ...) {  
    return 回傳值;  
}
```

```
void 函式名稱(void) {  
    return;    //可省略此行  
}
```

# 資料型別

關鍵字	bit	scope	printf chars
char	8	-128..127 (或0..255與體系相關)	%c
unsigned char	8	0..255	
signed char	8	-128..127	
int	32(win32/unix)	-2147483648..2147483647	%i, %d
unsigned int/unsigned	32(win32/unix)	0..4294967295	%u
signed int	32(win32/unix)	-2147483648..2147483647	%i, %d
short int	16	-32768..32767	%hi
unsigned short	16	0..65535	%hu
signed short	16	-32768..32767	
long int	32	-2147483648..2147483647	%li, %ld
unsigned long	32	0..4294967295	%lu
signed long	32	-2147483648..2147483647	
long long	64	-9223372036854775808.. 9223372036854775807	%lli
unsigned long long	64	0..18446744073709551615	%llu
float	32	$3.4 \times 10^{-38}$ .. $3.4 \times 10^{+38}$ (7 sf)	%f, %e, %g
double	64	$1.7 \times 10^{-308}$ .. $1.7 \times 10^{+308}$ (15 sf)	%f, %e, %g
long double	64 或以上	編譯器相關	%Lf, %Le, %Lg

# 整數常數

- ❑ 以0(零)為開頭的整數視為8進位數字。
- ❑ 試說明013和026的10進位值。
  - 013 等於 11
  - 026 等於 22
- ❑ 以0x開頭的整數，視為16進位整數。
- ❑ 試說明0x28和0x3A的10進位值。
  - 0x28等於40
  - 0x3A等於58
- ❑ 在數字後加l或L，表示長整數。

# 常數 Constant

- ❑ 常數 (Constant) 是 "不會在程式執行過程中改變的數"。
- ❑ 定義常數的好處
  - 使常數的意義一目瞭然。
  - 使用常數易於維修程式。

```
#include <stdio.h>

int main(void) {
    double radiusA, areaA, radiusB, areaB;
    radiusA = 3.0;
    radiusB = 5.0;
    areaA = 3.14 * radiusA * radiusA;
    areaB = 3.14 * radiusB * radiusB;
    printf("%f, %f\n", areaA, areaB);
    return 0;
}
```

```
#include <stdio.h>
#define PI 3.14159

int main(void) {
    double radiusA, areaA, radiusB, areaB;
    radiusA = 3.0;
    radiusB = 5.0;
    areaA = PI * radiusA * radiusA;
    areaB = PI * radiusB * radiusB;
    printf("%f, %f\n", areaA, areaB);
    return 0;
}
```



# 字元char

- ❑ 一個char宣告的變數，佔記憶體空間是8位元。 $2^8 = 256$ 
  - 代表256個不同的值。
  - ASCII碼值，含大小寫字母、數字、標點符號及其它特殊符號。
- ❑ 宣告字元變數single\_char
  - char single\_char;
- ❑ 宣告字元變數single\_char，將其值設定為'a'。
  - char single\_char = 'a';
  - 單引號間的字元，如：'a'，';'，'3'。
  - '\0'值是0， '0'值是48
- ❑ 字元和一般整數混合進行加法和減法運算。
  - ch = 'a';
  - ch = 'a' + 1;
  - 因'a' 值是97，執行加法後ch值是98，所以最後ch值是'b'。

# 字元char

- ❑ 無法列印字元，如：'\0'，雖在單引號中有\和0，但合併只算一個字元，這些字元為逸出(escape)字元。

整數值	字元表示方式	字元名稱
0	'\0'	空格(null space)
7	'\a'	響鈴(bell ring)
8	'\b'	退格(backspace)
9	'\t'	標識(tab)
10	'\n'	新列(newline)
12	'\f'	送表(form feed)
13	'\r'	回轉(carriage return)
34	'\"'	雙引號(double quote)
39	'\''	單引號(single quote)
92	'\\'	倒斜線(back slash)

# 字元常數

- ❑ 單引號間的字元，稱字元常數。例如：'a'，';'，'3'。
  - '\0'的ASCII值值是0
  - '0'的ASCII值值是48
- ❑ 有時也將字元常數和一般整數混合進行加法和減法運算。
  - 假設有一字元變數`ch = 'a'`；
  - 假設有一指令是`ch = 'a' + 1`；
  - 因 'a' 值是97，執行加法後ch值是98，最後ch值是 'b'。

# 字串常數

▣ 雙引號間任意個數的字元符號。例如：

○ "This is a good book" ← 一個字串

○ "" ← 一個空字串

▣ 字串

○ 是一個陣列，

○ 每個元素存一個字元，編譯程式自動把'\0'放入陣列最後，代表字串結束。

T	h	i	s		i	s		a		g	o	o	d		b	o	o	k	\0
---	---	---	---	--	---	---	--	---	--	---	---	---	---	--	---	---	---	---	----

# 浮點數

- ❑ 宣告一個浮點變數average，則其宣告如下：
  - float average；
- ❑ double雙倍精度浮點數，容量是浮點數的一倍。而long double則稱長雙倍浮點數，長度是80位元。
- ❑ 宣告tmp為雙倍精度浮點數，tmp1為長雙倍精度浮點數。
  - double tmp；
  - long double tmp1；
- ❑ 下表是三種浮點數/實數有關資料表

浮點數宣告型別	長度	值的範圍
float	32	$3.4 \times 10^{-38} \sim 3.4 \times 10^{38}$
double	64	$1.7 \times 10^{-308} \sim 1.7 \times 10^{308}$
long double	80	$3.4 \times 10^{-4932} \sim 1.1 \times 10^{4932}$

# 字串資料型別

- ❑ 字串指在兩個雙引號中任意字元。例如：
  - " ", "hello, How are you?"
- ❑ 若雙引號中沒有字元，稱空字串。
- ❑ 字串存入記憶體會將'\0'(空字元)加在字串最後，表示字串結束。
- ❑ 假設有一字串是"hello!"，實際記憶體儲存此字串如下：



- 雙引號非字串一部份，若字串是He say, "Hello !"，則此字串表示法： "He say, \"Hello !\""

# 基本算術運算

- ❑  $\text{sum} = a + b;$
- ❑  $s = a - b;$
- ❑  $s = a * b;$
- ❑  $s = a / b;$
- ❑  $s = a \% b;$
- ❑  $s = -a - b;$
- ❑  $s = a * (b + c);$

# Homework I

- 寫一程式計算一元二次方程式的兩個實根



# 型別轉換

□ 不同型別間運算，如整數轉換成浮點來運算。

○ 會轉換成最長空間後再運算

○  $s = a + b$ ;

➤ 整數 $a = 3$ ，浮點數 $b = 2.5$ ，運算後 $s = 5.5$

○  $s = a + b$ ;

➤  $a = 3$ ， $b$ 是浮點數 $b = 2.5$ ， $s = ?$

○  $a = 3$ ， $b = 2$ ， $s = a/b$ ， $s = ?$

➤  $s = (\text{float}) a / (\text{float}) b$ ;  $s = ?$

○  $a$ 和 $b$ 是浮點數， $a = 4.6$ ， $b = 2.1$ ， $s = ?$

➤  $i = 'a' - 'A'$ ;

➤  $i$ 是整數，先將 $'a'$ 轉成ASCII碼97，再將 $'A'$ 轉成ASCII碼65，運算完後 $i$ 的值是32。

# 遞增和遞減運算式

□ "++"將某個運算元加1，"--"會主動將某個運算元減1。

○  $i++$ ；執行前  $i = 2$ ，則執行後  $i = 3$

○  $i--$ ；執行前  $i = 2$ ，則執行後  $i = 1$

□  $i++$ ，後置運算

○ 先取出  $i$  的值， $i$  再加。

□  $++i$ ，稱前置運算。

○ 先加1，再取出  $i$  的值

○  $a = (++i) + 3$ ；

➤  $a = 3$ ， $i = 5$ ，

➤ 先做  $i$  加1， $i$  變為6，再進行加算， $a$  值是9。

○  $a = (3 + i)++$ ；

➤  $a = 3$ ， $i = 5$ ，

➤ 先執行  $3 + i$ ， $a$  值是8， $i$  本身再加1，值是6。

# 遞增和遞減運算式

## □ ++ 和 -- 前置運算子和後置運算子應用

```
#include <stdio.h>
int main() {
    int x,y,z;
    x = y = z = 0;
    /* testing the increment by 1 */
    x = (++y) + (++z);           2 11
    printf("\1: line 14 result ... %d %d %d\n",x,y,z);
    x = y++ + z++;              1 + 1           2 2 2
    printf("\2: line 16 result ... %d %d %d\n",x,y,z);
    /* testing the decrement by 1 */
    x = y = z = 0;
    x = (--y) + (--z);           -1 -1 -1
    printf("\1: line 20 result ... %d %d %d\n",x,y,z);
    x = y-- + z--;              -2 -1 -2
    printf("\2: line 22 result ... %d %d %d\n",x,y,z);
}
```

# 輸入和輸出函數

## □ 加、減、乘、除及求餘數(+=, -=, \*=, /=, %=)等特殊運算子的程式應用

○  $a *= c$  ;

➤  $a = 3$  ,  $c = 2$  , 則執行後  $c = 2$  ,  $a = 6$

○  $a += c * d$  ;

➤  $a = a + (c * d)$  ;

○  $a *= c + d$  ;

➤  $a = 2$  ,  $c = 3$  ,  $d = 4$  ,

○  $a = a * (c + d)$

➤  $c + d$  等於 7 ,  $7 * 2 = 14$  ,

➤ 可得  $a = 14$

```
#include <stdio.h>
int main() {
    int a,b,c,d,e;
    a = b = c = d = e = 0;
    a += 2;
    printf("a is %d\n",a); 2
    b -= 2;
    printf("b is %d\n",b); -2
    c *= c = 2;
    printf("c is %d\n",c); 4
    d %= d = 3;
    printf("d is %d\n",d); 0
    e /= e = 4;
    printf("e is %d\n",e); 1
}
```

# 位移運算子(Shift Operator)

- 位移運算，把位元(bit)向左移(<<)或向右移(>>)幾個位置。
  - 向左移n個位元，相當於乘2的n次方；
  - 向右移n個位元，相當於除以2的n次方。

```
#include<stdio.h>
int main(){
    int a = 5, b = 13;
    a = a << 2;
    b = b >> 1;
    printf(" 5 << 2 = %i \n", a);
    printf("13 >> 1 = %i \n", b);
    return 0;
}
```

5x4  
13÷2

5 << 2 = 20 13 >> 1 = 6
----------------------------

# 位元運算子(Bit Operator)

## □ 位元邏輯運算

011  
101

```
#include<stdio.h>
int main(){
    int A = 3, B = 5;
    printf("A & B = %i \n", A & B);
    printf("A | B = %i \n", A | B);
    printf("A ^ B = %i \n", A ^ B);
    printf(" ~A  = %i \n", ~A );
    return 0;
}
```

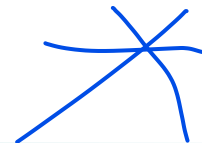
1  
7  
6

運算子	使用方式	功能敘述
&	i & j	i AND j
	i   j	i OR j
^	i ^ j	i XOR j
~	~ i	NOT i

A & B = 1  
A | B = 7  
A ^ B = 6  
~A = -4

→ 100

# Homework



## □ 集合元素 0~7

○ 集合編碼S: {3, 2, 1, 0} 為二進位00001111，十進位15

7	6	5	4	3	2	1	0
0	0	0	0	1	1	1	1

$$2^3 + 2^2 + 2^1 + 2^0 = 15$$

○ 集合編碼S: {3, , , 0} 為二進位00001001，十進位9

7	6	5	4	3	2	1	0
0	0	0	0	1	0	0	1

$$2^3 + 0^2 + 0^1 + 2^0 = 9$$

○ 空集合編碼S: {} 為二進位00000000，十進位0

○ S, 是集合，e是集合元素，均以十進位表示

➤ 集合S，加入元素e， $(S | (1 \ll e))$ ;

7	6	5	4	3	2	1	0
0	0	0	0	1	0	0	1
0	0	0	0	0	1	0	0
0	0	0	0	1	1	0	1

$$S = [3, , , 0] = 9$$

$$e = 2, 1 \ll 2 = 4$$

$$9 | 4 = 13 = [3, 2, , 0] \quad 2^3 + 2^2 + 0^1 + 2^0 = 13$$

# Homework

- 集合S，判斷元素e是否屬於集合S， $(S \& (1 \ll e)) == (1 \ll e)$

7	6	5	4	3	2	1	0
0	0	0	0	1	1	0	1
0	0	0	0	0	1	0	0
0	0	0	0	0	1	0	0

$$S = [3, 2, 0] = 13$$

$$e = 2, 1 \ll 2 = 4$$

$$13 \& 4 = 4 = 2^2, 0^3 + 2^2 + 0^1 + 0^0 = 4$$

- 集合S，移除元素e， $(S \& \sim(1 \ll e))$

7	6	5	4	3	2	1	0
0	0	0	0	1	1	0	1
1	1	1	1	1	0	1	1
0	0	0	0	1	0	0	1

$$S = [3, 2, 0] = 13$$

$$e = 2, \sim(1 \ll 2) = 251$$

$$13 \& 251 = 9, 1^3 + 0^2 + 0^1 + 1^0 = 4$$

- 集合S1，集合S2，計算S1是否為S2子集合，  
➤ 集合S1，集合S2，計算S1與S2交集， $(S1 \& S2)$   
➤ 集合S1，集合S2，計算S1與S2聯集， $(S1 | S2)$   
➤ 印出集合元素

```
for (int i=0; i<n; i++) {  
    if (((S>>i)&1)==1)  
        printf("%d, ", i);  
}
```



# Homework

```
#include <stdio.h>
#define N 8
int isEmpty(int set) {
    [redacted]
}
int insertElement(int set, int e) {
    [redacted]
}
✓ int removeElement(int set, int e) {
    [redacted]
}
int intersection(int set1, int set2) {
    [redacted]
}
int unionSet(int set1, int set2) {
    [redacted]
}
void print(int set) {
    printf("[");
    for (int i=0; i<N; i++) {
        if (((set>>i)&1)==1)
            printf("%d, ", i);
    }
    printf("]\n");
}
```

```
void test() {
    int set1=0, set2=0, set3=0;
    printf("%d\n", isEmpty(set1));
    set1 = insertElement(set1, 6);
    set2 = insertElement(set2, 6);
    printf("set1:"); print(set1=insertElement(set1, 1));
    printf("set2:"); print(set2=insertElement(set2, 3));
    printf("set3:"); print(set3 = unionSet(set1, set2));
    printf("set3:"); print(set3 = removeElement(set3, 1));
    printf("----:"); print(intersection(set1, set2));
    printf("set3:"); print(set3=insertElement(set3, 1));
}
int main() {
    test();
    return 0;
}
```

```
1
set1:[1, 6, ]
set2:[3, 6, ]
set3:[1, 3, 6, ]
set3:[3, 6, ]
----:[6, ]
set3:[1, 3, 6, ]
```

# sizeof

- ❑ 算出型別資料佔用記憶體位置數量(以byte為單位)
  - sizeof(某個資料型別)
  - $n = \text{sizeof}(\text{char})$  ;
  - char字元是一個位元組(byte)，執行完後n值是1。

```
#include <stdio.h>
int main() {
    printf("The size of char  is %d\n",sizeof(char));
    printf("The size of int   is %d\n",sizeof(int));
    printf("The size of float is %d\n",sizeof(float));
    printf("The size of double is %d\n",sizeof(double));
}
```

```
The size of char  is 1
The size of int   is 4
The size of float is 4
The size of double is 8
```

# 基本輸入輸出

## □ 標準輸入與輸出函數基本定義：

- #include <stdio.h>

- printf()：這是一個最常用的輸出函數

- scanf()：這是一個最常用的輸入函數

- putchar(c)：列印字元的輸出函數

- getche()：讀取字元的輸入函數 → 入、出

- getchar()：讀取字元的輸入函數

- getch()：讀取字元的輸入函數 ~~出~~

//依不同格式列印字串，

//Introduction to C language字串兩次，依不同格列印。

```
#include <stdio.h>
```

```
int main() {
```

```
    printf("Introduction\n to C language.\n");
```

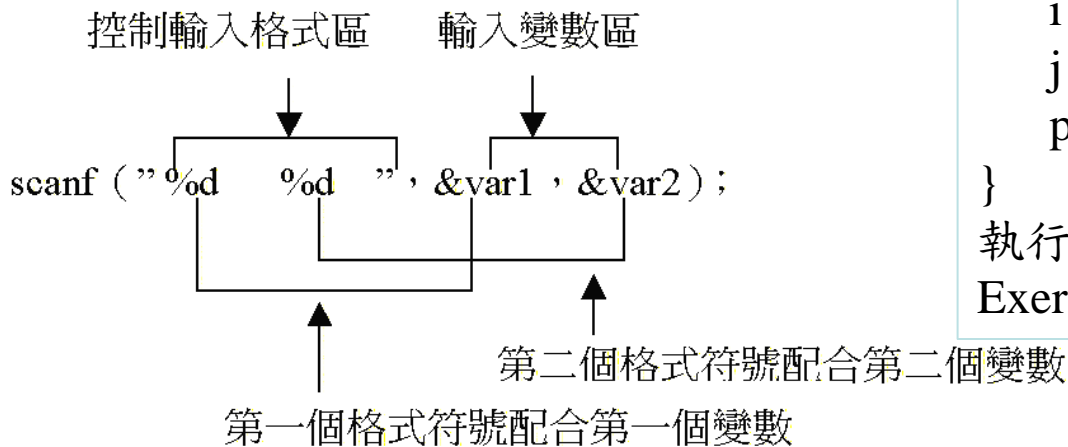
```
    printf("Introduction to\n C language.\n");
```

```
}
```

執行結果  
Introduction  
To C language.  
C language.

# %d 十進位整數的列印

## □ 格式化控制輸出的結果



```
#include <stdio.h>
int main() {
    int i,j;
    i = 3;
    j = 4;
    printf("exercice ch%d_%d.c \n",i,j);
}
```

執行結果

Exercie ch3-4.c

- 第一個格式符號配合第一個欲列印的變數。
- 控制輸入格式區的格式符號間，可有許多空格，或沒有空格。
- 輸入變數區，各變數間要有逗點隔開。
- 控制輸入格式區需用雙引號" "包夾起來。
- 控制輸入格式區和輸入變數區間需用逗號隔開。

# 格式化控制輸出

## □ %5d

```
#include <stdio.h>
int main() {
    int i;
    i = 356;
    printf("/%d\n",i);
    printf("/%2d\n",i);
    printf("/%5d\n",i);
    printf("/%-5d\n",i);
}
```

//控制字元 '\n'

執行結果

/356/

/356/

/ 356/ →

/356 / ←

## %f 浮點數的列印

```
#include <stdio.h>
int main() {
    float i;
    i = 123.56;
    printf("/%f\n",i);
    printf("/%3.2f\n",i);
    printf("/%8.2f\n",i);
    printf("/%-8.2f\n",i);
}
```

執行結果

/123.559998/

/123.56/

/ 123.56/

/123.56 /

IEEE 754

# % c 字元的列印

- 格式化出某一字元變數。

```
#include <stdio.h>
int main() {
    char i;
    i = 'a';
    printf("/%c\n",i);
    printf("/%3c\n",i);
    printf("/%-3c\n",i);
}
```

執行結果

/a/

/ a/ →

/a / ←

# 其他格式化資料列印

## □ printf()提供格式化列印方式：

○ %e：以e記號(科學符號)表示法，輸出浮點數。

○ %u：不帶符號的10進位整數輸出。

○ %x：16進位整數輸出。

○ %o：8進位整數輸出。

○ %s：列印字串。

○ %lld：long long

○ %llu：unsigned long long

```
format octal value output
/12/
/12  /
format hexadecimal value output
/a/
/  a/
format unsigned value output
/10/
/  10/
format scientific symbol value output
/1.235600e+002/
/1.236e+002/
```

```
#include <stdio.h>
int main() {
    int i = 10;
    float j = 123.56;
    printf("format octal value output\n");
    printf("/%o\n",i);
    printf("/%-8o\n",i);
    printf("format hexadecimal value output\n");
    printf("/%x\n",i);
    printf("/%8x\n",i);
    printf("format unsigned value output\n");
    printf("/%u\n",i);
    printf("/%8u\n",i);
    printf("format scientific symbol value output\n");
    printf("/%e\n",j);
    printf("/%8.3e\n",j);
}
```

# scanf()

## ❑ 使用兩次scanf()的問題

- 執行結果錯誤，輸入一個值(例如 1)，按下 enter 就結束。
- scanf() 未處理換行符號，造成第二個 scanf("%c",&b) 吃到換行符號 enter。

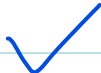
```
int a;  
char b;  
scanf( "%c", &a );  
scanf( "%c", &b );  
printf( "a = %c, b = %c\n", a, b );
```

```
1  
a = 1, b =
```

## ❑ 解決方法：

- 在第二個 scanf() 的 %c 之前加入空白字元，scanf( " %c", &b );
- 在第一個 scanf() 的 %d 之後使用 %\*c 跳脫一個字元，scanf( "%d%\*c", &a );
- 在兩個 scanf() 之間加入 getchar() 吸收換行

```
getchar();  
scanf( "%c", &b );
```





# scanf()

❑ scanf 可自定欲接收的字元，包括空白等字元：

- `scanf("%[^\\n]",str);` // 接收除了 `\\n` 以外的所有字元

- `printf("%s",str);` // 輸出完整的「hello world」

❑ 如果輸入不是integer時，程式會跑個不停。

- 因scanf 透過stdin 讀入資料，存放在memory一個queue中。

- 只要queue裡有資料，程式就須scanf 將裡面資料消化掉。

- 「%」符號是match，「%d」表示queue中資料須是integer才可。

- 若輸入char，會使queue中資料沒有辦法消化，程式將在回圈中不繼尋找一個可以消化的scanf。

```
int a=0;
while(a!=1){
    scanf("%d",&a);
}
```

# scanf()

## ❑ 解決方法：清掉queue裡的資料

- 用 fflush(stdin)，把 input queue 清除，但此函式偶而有誤。可以使用 getche 把 queue 裡的值讀完。

```
while(a!=1){  
    scanf("%d",&a);  
    while(getche() != '\n');  
}
```

- 若輸入資料不符合scanf時，getche可把queue裡的值讀完，
- 若輸入資料符合scanf，不會因多getche那一行程式就停在那邊不動，因enter剛好可留給它消化，如此程式就不會出問題。

# scanf()

- 輸入兩字元，顛倒順序輸出，輸入三個整數求總和，將它列印出，輸入兩個浮點數，將平均浮點數值印出

```
#include <stdio.h>
int main() {
    int i,j,k,sum;
    char ch1,ch2;
    float x1,x2,ave;
    printf("Enter 2 characters ==>");
    scanf("%c%c",&ch1,&ch2);
    printf("Reverse of 2 char ==>");
    printf("%c%c\n",ch2,ch1);
    printf("Enter 3 integer ==>");
    scanf("%d %d %d",&i,&j,&k);
    sum = i + j + k;
    printf("The sum of input ==> %d\n",sum);
    printf("Enter 2 floating ==>");
    scanf("%f %f",&x1,&x2);
    ave = ( x1 + x2 ) / 2.0;
    printf("Average of input ==> %6.2f\n",ave);
}
```

Enter 2 characters ==>ab  
Reverse of 2 char ==>ba  
Enter 3 integer ==>3 1 5  
The sum of input ==> 9  
Enter 2 floating ==>1.6 3.8  
Average of input ==> 2.70

**注意：輸入整數或浮點數時，可用空格區別所輸入的資料。但輸入字元時，字元間不可有空格。**

# 字元輸入和輸出函數


## □ Function

- getche()讀取一個字元，putchar(ch)每次輸出一個字元。
  - getche()函數式不包含任何參數，使用：ch = getche()；
  - putchar(ch)須包含一個字元變數：
  - getche()函數注意事項
    - 以scanf()讀取字元值，按enter鍵後，才正式讀取字元資料，以getche()讀取字元值，只要一有輸入，輸入值立刻讀入所設定的字元變數內。
  - getchar()，使用格式：ch = getchar()；
    - 讀取的字元會被存至變數ch內。
- {
- 以getche()讀取字元時，不必按enter鍵，程式自動讀該字元。
  - 以getchar()讀取字元時，輸入完字元後，須按enter鍵。

# 字元輸入和輸出函數

- 以getch()讀取字元時，所輸入的字元不顯示在螢幕上。

```
#include <stdio.h>
int main() {
    char ch1, ch2;
    printf("Please enter 2 characters \n==>");
    ch1 = getch();
    ch2 = getch();
    printf("\n");
    printf("The first character is \n==>");
    putchar(ch1);
    printf("\n");
    printf("The second character is \n==>");
    putchar(ch2);
}
```

Please enter 2 characters  
==>   
The first character is  
==>p  
The second character is  
==>q

# 輸入和輸出函數

- 輸入英哩和碼數，轉換成公里。1英哩=1.609公里，1英哩=1760碼

```
#include <stdio.h>
int main() {
    int mile, yard;
    float km;
    printf("Convert the mile and yard to kilometer\n");
    printf("Please enter mile here \n==> ");
    scanf("%d",&mile);
    printf("Please enter yard here \n==> ");
    scanf("%d",&yard);
    km = 1.609 * ( mile + (float) yard / 1760 );
    printf("The result is %8.3f \n",km);
}
```

# Exercise

- 輸入公里、公尺，轉換英哩和碼。1英哩=1.609公里=1760碼

```
#include <stdio.h>
double getMile(int km, int m) {
    double mile = (km + m/1000.0)/1.609;
    return mile;
}
void compute(int km, int m) {
    printf("mile = %d, yard = %d\n", mile_i, yard);
}
int main() {
    compute(5, 500);
    return 0;
}
```

*Handwritten notes:* A blue box highlights the `int` parameter in the `compute` function call. The text `double` is written above `int` in the box. There are blue scribbles under the `mile_i` and `yard` variables in the `printf` statement.

# 輸入和輸出函數

## □ 不同型別資料運算與強制運算元的基本應用

```
#include <stdio.h>
int main() {
    float x = 5.3;
    int y = 9;
    int z = 4;
    x = y / z;
    printf("The result is %6.2f\n",x);
    x = (float) y / (float) z;
    printf("The result is %6.2f\n",x);
}
```

The result is 2.00

The result is 2.25



# 輸入和輸出函數

- 輸入華氏溫度，轉換成攝氏溫度輸出，溫度轉換公式：
  - 攝氏溫度 =  $(5.0 / 9.0) * (\text{華氏溫度} - 32.0)$

```
#include <stdio.h>
int main() {
    float f,c;
    printf("Input fahrenheit degree \n==>");
    scanf("%f",&f);
    c = ( 5.0 / 9.0 ) * ( f - 32.0 );
    printf("The celsius is %6.2f \n",c);
}
```

```
Input fahrenheit degree
==>103
The celsius is  39.44
```

# Exercise

- BMI值計算公式 (正常範圍 BMI=18.5~24)
  - BMI = 體重(公斤) / 身高<sup>2</sup>(公尺<sup>2</sup>)
  - 例如：一個52公斤的人，身高是155公分，則BMI為:  $52(\text{公斤}) / 1.55^2 (\text{公尺}^2) = 21.6$

```
#include <stdio.h>
int main() {
    double BMI =0, weight=0, heigh=0;

    return 0;
}
```


# Exercise

- ❑ 工資計算，假設工作一小時12元，而薪資所得的12%是稅金，請輸入工作時數，列出扣完稅之後實際所得。

```
#include <stdio.h>
int main() {
    int hourpay = 12;
    int hour, totalpay;
    float taxrate = 0.12;
    float netpay;
    printf("\1: Please input the working hours. ==> ");
    scanf("%d",&hour);
    totalpay = hour * hourpay;
    netpay = totalpay - (totalpay * taxrate);
    printf("\2: The netpay is %8.2f \n",netpay);
}
```

# Homework II

- ❑ 工資計算，假設工作一小時 $x$ 元(整數)，而全部薪資所得的8%取到小數第一位是稅金，每個月勞保費為最低工資 $y$ 元(整數)的5%取到小數第一位。
- ❑ 請輸入每月工作時數(整數)，輸入二個月， $x, y$ 。
- ❑ 輸出扣完稅與勞保後實際所得、勞保費、稅金。

```
double f(int hour, int x, int y ) {  
    double fee;    //勞保費  
    double salary; //薪資  
    double tax;    //稅金  
  
  
  
    salary = salary - fee - tax;  
    printf("salary = %.1f, fee=%.1f, tax=%.1f", salary,  
fee, tax);  
}
```

# 程式執行時間

## □ 計算程式執行時間

```
#include <stdio.h>
#include <time.h>
long g(int n) {
    if (n<1) return 1;
    else return (g(n-1)+g(n-2)+g(n-3));
}
int main() {
    long begin, end;
    begin = clock();
    g(32); g(32);
    end = clock();
    printf("%d ms\n", (end-begin)*100/CLK_TCK); //毫秒
    return 0;
}
```

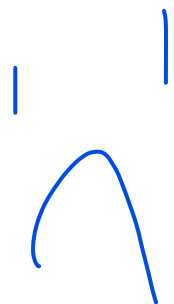
# fork() 程式執行時間

## ❑ fork

- 計算程式執行時間
- clock() 會被歸0

```
struct timespec {  
    time_t  tv_sec;      /* seconds */  
    long    tv_nsec;     /* nanoseconds */  
};
```

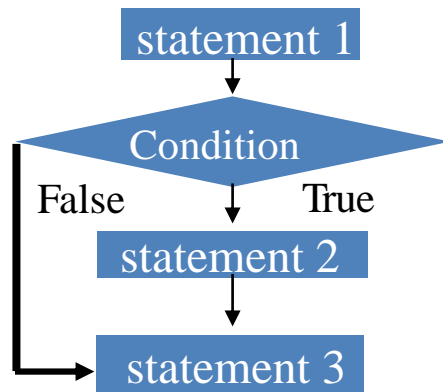
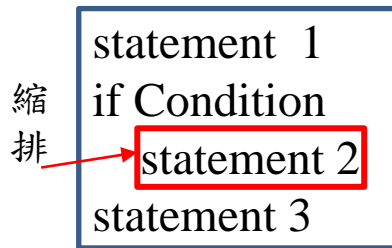
```
#include <stdio.h>  
#include <time.h>  
long g(int n) {  
    if (n<1) return 1;  
    else return (g(n-1)+g(n-2)+g(n-3));  
}  
int main() {  
    struct timespec st = {0, 0};  
    struct timespec et = {0, 0};  
    clock_gettime(CLOCK_REALTIME, &st);  
    g(32); g(32);  
    clock_gettime(CLOCK_REALTIME, &et);  
    printf("%ld ms\n", (et.tv_sec-st.tv_sec)*1000+(et.tv_nsec-st.tv_nsec)/1000000); //毫秒  
    return 0;  
}
```



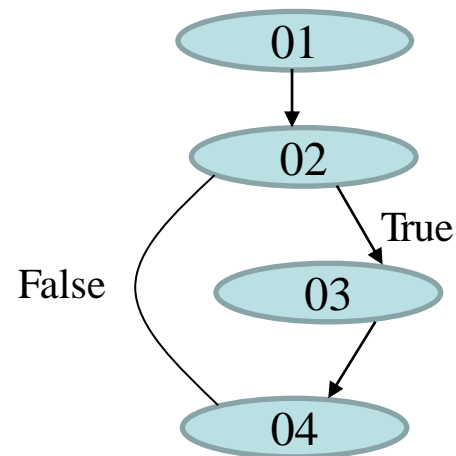
# 選擇 - if條件判斷

## □ 基本語法

- if (condition), if 後跟隨"條件判斷"(Condition)
- statement 2, if本體區塊 (body block), 縮排一層, 若"條件判斷"True, 執行此區指令



```
01 x = 2
02 if (x*10 == 20)
03     printf('2*10=20');
04 x = 3;
```



# Exercise

## □ 105APCS Q4

- 輸入棒球隊球員打擊結果，計算出隊得分。假設球員打擊情況：
  - 安打：以1, 2, 3 和4代表一、二、三和全(四)壘打。
  - 出局：以0表示。
- 簡化版的規則如下：
  - 球場上有四個壘包，稱為本壘、一、二和三壘。
  - 本壘握球棒打的稱「擊球員」，在另外三個壘包的稱為「跑員」。
  - 當擊球員打擊「安打」時，擊球員與跑壘員可移動；「出局」時，跑壘員不動，擊球員離場換下一位。
  - 比賽開始由第1位打擊，接著2, 3位球員。
  - 打出 K 壘打時，場上球員(擊球員和跑壘員)會前進 K 個壘包。本壘到一壘，接著二、三壘，最後回到本壘。回到本壘可得1分。



# Exercise

## □ 105APCS Q4

### ○ 輸出格式

- 輸入3位打者打擊資料，輸出最後一、二、三壘狀況，有人為1，沒人為0。

輸入範例一

1

0

1

正確輸出

1 1 0

輸入範例二

1

4

0

正確輸出

0 0 0

輸入範例三

1

0

0

正確輸出

1 0 0

輸入範例四

1

2

0

正確輸出

0 1 1

# Exercise

```
#include <stdio.h>
int main() {
    int state=0, input=0, score=0;
    scanf("%d", &input);
    state = (state <<input) | (1<<(input-1));
    state = state&7;
    scanf("%d", &input);
    state = (state <<input) | (1<<(input-1));
    state = state&7;
    scanf("%d", &input);
    state = (state <<input) | (1<<(input-1));
    state = state&7;
    printf("%d %d %d\n", state&1, (state>>1)&1, (state>>2)&1);
    return 0;
}
```

```
input = 2
s =          0 0 0 0 0 0 1 1
s<<2 =       0 0 0 0 1 1 0 0
1<<(input-1) = 0 0 0 0 0 0 1 0
-----
s =          0 0 0 0 1 1 1 0
7 =          0 0 0 0 0 1 1 1
-----
s&7=         0 0 0 0 0 1 1 0
```

# Homework III

## □ 105APCS Q4

- 輸入棒球隊球員打擊結果，計算出隊得分。假設球員打擊情況：
  - 安打：以 1B, 2B, 3B 和 HB 代表一、二、三和全(四)壘打。
  - 出局：以 O1, O2, O3 表示 1, 2, 3 人出局 (OUT)。
- 簡化版的規則如下：
  - 球場上有四個壘包，稱為本壘、一、二和三壘。
  - 本壘握球棒打的稱「擊球員」，在另外三個壘包的稱為「跑員」。
  - 當擊球員打擊「安打」時，擊球員與跑壘員可移動；「出局」時，跑壘員不動，擊球員離場換下一位。
  - 比賽開始由第 1 位打擊，接著 2, 3, ..., 9 位球員。
  - 打出 K 壘打時，場上球員(擊球員和跑壘員)會前進 K 個壘包。本壘到一壘，接著二、三壘，最後回到本壘。回到本壘可得 1 分。
  - 每達到三個出局數時，壘包清空(跑壘員都得離開)，重新開始。

# Exercise

## ○輸入格式

- 1. 每組測試資料固定有十行。
- 2. 第一到九行，第一到九行，依照球員順序，每一行代表一位球員打擊資訊。每一行開始有一個正整數  $a$  ( $1 \leq a \leq 5$ )，代表球員總共打  $a$  次。接下來有  $a$  個字串(均為兩字元)，依序代表每次打擊結果。資料間均以一個空白隔開代表每次打擊結果。球員打擊資訊不會有錯誤也不會缺漏。
- 3. 第十行有一個正整數  $b$  ( $1 \leq b \leq 27$ )，表示要計算當總出局數累計到  $b$  時，該球隊的得分。輸入的打擊資訊中至少包含  $b$  個出局。

## ○輸出: 計算在第 $b$ 個出局數發生時的總得分，並將此得分輸出。

```
5 1B 1B FO GO 1B
5 1B 2B FO FO SO
4 SO HR SO 1B
4 FO FO FO HR
4 1B 1B 1B 1B
4 GO GO 3B GO
4 1B GO GO SO
4 SO GO 2B 2B
4 3B GO FO FO
3
```

輸出  
0

```
5 1B 1B FO GO 1B
5 1B 2B FO FO SO
4 SO HR SO 1B
4 FO FO FO HR
4 1B 1B 1B 1B
4 GO GO 3B GO
4 1B GO GO SO
4 SO GO 2B 2B
4 3B GO FO FO
6
```

輸出  
5

輸入範例二

```
1
4
O
3
1
正確輸出
0 0 0
```