Chapter 2

基本輸出入

本章大綱

- 2-1 printf()
- 2-2 scanf()
- 2-3 亂數與發聲
- 2-4 線上程式解題系統
- 2-5 APCS 初體驗

輸出入是程式設計的第一步，C 語言採用 printf() 與 scanf() 作爲基本輸出入函式，請看以下說明。

2-1 printf()

printf() 可輸出結果，其中括號內須放置一對雙引號『""』來將欲輸出的資料當作引數傳給 printf() 函式。例如，

```
printf("a");
printf("aa");
```

的輸出結果是：

```
aaa
```

但若要換行與歸位，則應加上跳脫字元『\n』，例如，

```
printf("a\n");
printf("aa");
```

的輸出結果是：

```
a
aa
```

以下程式，是我們以 DevC++ 開新 C 專案，完成以上操作的完整程式，請自行鍵入練習。

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[]) {
   printf("a\n");
   printf("aa");
   return 0;
}
```

其次，如果是變數的輸出，則必須在輸出敘述中加入一個列印格式，此列印格式以控制字元『%』符號開頭，如下表，。

資料型別	列印格式
整數	%d
長整數	%ld
浮點數 float	%f
浮點數 double	%lf
字元	%c
字串	%s

然後編譯器會依照控制字元的順序，到後面引數區取對應資料來輸出。例如，

```
int a=3;
printf("%d", a);
```

『%』即表示要到後面引數區取變數來取代，『d』是將變數視爲整數，所以輸出結果是

```
3
```

若是，

```
int a=3, b=5;
printf("%d %d", a, b);
```

有兩個控制字元，那就到後面引數區依照順序取變數輸出，所以輸出是

```
3 5
```

字串區有可能有一般字元也有控制字元『%』，那一般字元就直接輸出，控制字元就到後面引數區依序取變數輸出。例如，

```
int a=3, b=5;
printf("a=%d  b=%d", a, b);
```

因爲字串內容『"a="』就是要輸出的字元，『% d』是控制字元，才去引數區按照順序取對應變數來取代，第一個『%d』取到 a，第二個『%d』取到 b，所以輸出結果是：

```
a=3  b=5
```

輸出過程中如須換行與歸位，是以跳脫字元序列『 \n 』來完成。例如：

```
int a=3, b=5;
printf("a=%d\n", a);
printf("b=%d", b);
```

的結果是：

```
a=3
b=5
```

輸出資料時，亦可指定資料長度，那就可以將所有資料對齊。例如：

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[]) {
   int a=3,b=55;
   double c1=3.4,c2=15.678;
   printf("a =%8d,b =%8d\n",a,b);//佔8格，靠右
   printf("c1=%8.2f,c2=%8.2f",c1,c2);//共8格，靠右，小數點取2位
   //請留意遇到『%』就是控制字元，就到後面按順序取對應資料輸出
   //非控制字元，就直接輸出，例如『a=』，就輸出。
   return 0;
}
```

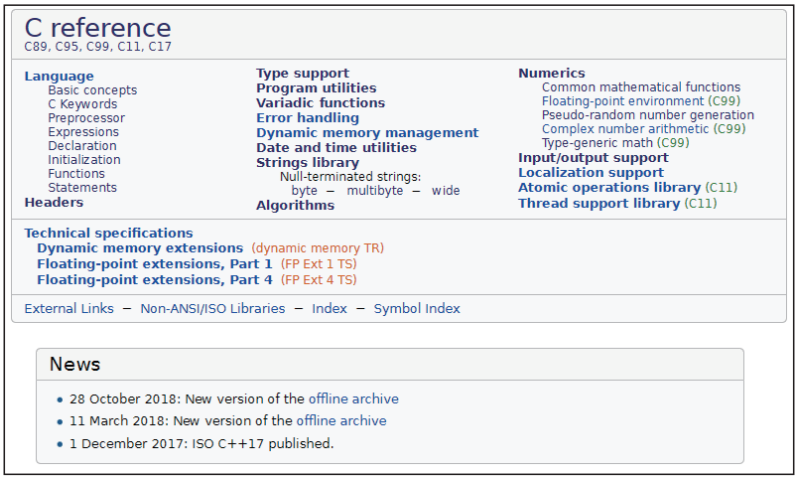
的執行結果如下圖，這樣就可以對齊所有資料。(上面『//』是註解文字，請看 4-2 節，這裡先不要理會。)

```
a =       3,b =      55
c1=    3.40,c2=   15.68
--------------------------------
```

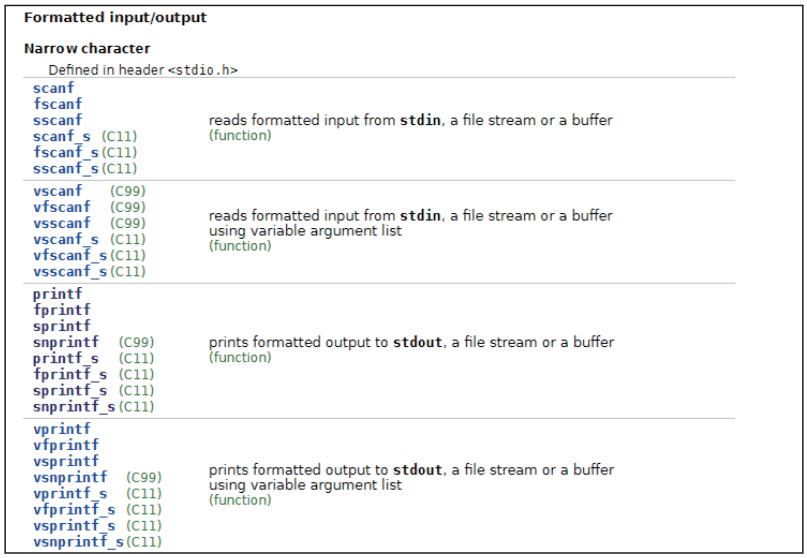
網路C語言使用手冊

網路 C 語言的使用手冊很多，但是，https://en.cppreference.com/w 是我目前找到最齊全的參考手冊，有 C 也有 C++ 如下圖：

C 語言的詳細資料如下（上圖的下面）：上一章曾提到編譯器版本問題，請看下圖就會明瞭，若標示『C99』表示那是 1999 新增的指令；若標示『C11』表示那是 2011 新增的指令，沒標示則是 1990 制訂的標準。



於上圖點選『input/output support』，再往下捲動，即可找到 printf() 的詳細說明，如下圖：



於上圖點選『printf』，畫面如下：這裡有 printf() 的詳細用法：

C File input/output

printf, fprintf, sprintf, snprintf, printf_s, fprintf_s, sprintf_s, snprintf_s

Defined in header <stdio.h>

`int printf( const char *format, ... );` `int printf( const char *restrict format, ... );`	(1)	(until C99) (since C99)
`int fprintf( FILE *stream, const char *format, ... );` `int fprintf( FILE *restrict stream, const char *restrict format, ... );`	(2)	(until C99) (since C99)
`int sprintf( char *buffer, const char *format, ... );` `int sprintf( char *restrict buffer, const char *restrict format, ... );`	(3)	(until C99) (since C99)
`int snprintf( char *restrict buffer, size_t bufsz, const char *restrict format, ... );`	(4)	(since C99)
`int printf_s(const char *restrict format, ...);`	(5)	(since C11)
`int fprintf_s(FILE *restrict stream, const char *restrict format, ...);`	(6)	(since C11)
`int sprintf_s(char *restrict buffer, rsize_t bufsz, const char *restrict format, ...);`	(7)	(since C11)
`int snprintf_s(char *restrict buffer, rsize_t bufsz, const char *restrict format, ...);`	(8)	(since C11)

Loads the data from the given locations, converts them to character string equivalents and writes the results to a variety of sinks.

1) Writes the results to the output stream `stdout`.
2) Writes the results to the output stream `stream`.
3) Writes the results to a character string `buffer`. The behavior is undefined if the string to be written (plus the terminating null character) exceeds the size of the array pointed to by `buffer`.
4) Writes the results to a character string `buffer`. At most `bufsz - 1` characters are written. The resulting character string will be terminated with a null character, unless `bufsz` is zero. If `bufsz` is zero, nothing is written and `buffer` may be a null pointer, however the return value (number of bytes that would be written not including the null terminator) is still calculated and returned.

上圖就有如字典般的詳盡了，所有參數還有詳細解說，如下圖：

Parameters

stream - output file stream to write to

buffer - pointer to a character string to write to

bufsz - up to bufsz - 1 characters may be written, plus the null terminator

format - pointer to a null-terminated multibyte string specifying how to interpret the data.

The format string consists of ordinary multibyte characters (except %), which are copied unchanged into the output stream, and conversion specifications. Each conversion specification has the following format:

- introductory % character
- (optional) one or more flags that modify the behavior of the conversion:
  - **-**: the result of the conversion is left-justified within the field (by default it is right-justified)
  - **+**: the sign of signed conversions is always prepended to the result of the conversion (by default the result is preceded by minus only when it is negative)
  - *space*: if the result of a signed conversion does not start with a sign character, or is empty, space is prepended to the result. It is ignored if + flag is present.
  - **#** : *alternative form* of the conversion is performed. See the table below for exact effects otherwise the behavior is undefined.
  - **0** : for integer and floating point number conversions, leading zeros are used to pad the field instead of *space* characters. For integer numbers it is ignored if the precision is explicitly specified. For other conversions using this flag results in undefined behavior. It is ignored if - flag is present.

還有範例與執行結果，如下圖：

Example

Run this code

```c
#include <stdio.h>

int main(void)
{
    printf("Strings:\n");
    const char* s = "Hello";
    printf("\t.%10s.\n\t.%-10s.\n\t.%*s.\n", s, s, 10, s);

    printf("Characters:\t%c %%\n", 65);

    printf("Integers\n");
    printf("Decimal:\t%i %d %.6i %i %.0i %+i %i\n", 1, 2, 3, 0, 0, 4, -4);
    printf("Hexadecimal:\t%x %x %X %#x\n", 5, 10, 10, 6);
    printf("Octal:\t%o %#o %#o\n", 10, 10, 4);

    printf("Floating point\n");
    printf("Rounding:\t%f %.0f %.32f\n", 1.5, 1.5, 1.3);
    printf("Padding:\t%05.2f %.2f %5.2f\n", 1.5, 1.5, 1.5);
    printf("Scientific:\t%E %e\n", 1.5, 1.5);
    printf("Hexadecimal:\t%a %A\n", 1.5, 1.5);
}
```

Output:

```
Strings:
    .     Hello.
    .Hello     .
    .     Hello.
Characters:     A %
Integers
Decimal:        1 2 000003 0  +4 -4
Hexadecimal:    5 a A 0x6
Octal:          12 012 04
Floating point
Rounding:       1.500000 2 1.30000000000000004440892098500626
Padding:        01.50 1.50  1.50
Scientific:     1.500000E+00 1.500000e+00
Hexadecimal:    0x1.8p+0 0X1.8P+0
```

以上詳盡的資料，對於程式設計者來說，可說是一大福音，因爲程式設計的過程，難免遇到問題，此時就需要查詢這些指令與函式用法，以上資料就非常實用。

2-2　scanf()

scanf( ) 可輸入資料，其使用方式與輸出 printf() 是一樣的方式，二者均須使用到控制字元『%』格式來控制輸出入的資料。因此，輸入字元、整數、浮點數及字串等會有不同的處理方式，請看以下說明。

輸入字元

輸入字元的敘述如下：

```
char a;
scanf("%c", &a);//請留意是&
```

可將輸入的資料放入在某個變數所指定位址中，取址符號『&』用來取得欲放置輸入資料的變數位址。例如，以下程式可輸入一個字元。

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[]) {
   char a;
   printf("Please press any char: ");
   scanf("%c", &a); //請留意是&
   printf("Your char is %c", a);
   return 0;
}
```

請留意輸入資料要有取址符號『&』，漏掉了就是錯誤。

自我練習

1. 請鍵入以上程式，並觀察執行結果。
2. 請鍵入以下程式，並觀察執行結果。

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[]) {
   char a,b;
   printf("Please press any char: ");
   scanf("%c", &a);
   printf("Your char is %c", a);
   printf("Please press any char: ");
   scanf("%c", &b);
   printf("Your char is %c", b);
   return 0;
}
```

以上程式，第二次輸入字元

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

會因鍵盤緩衝問題（只要有先輸入東西，例如整數、實數都有此問題），而直接跳過，這時請將字元前放一個前導空白字元就可以。所以程式修正如下：

```
scanf(" %c", &b);//留意%c的前導空白
```

這在使用手冊有提到 https://en.cppreference.com/w/c，請自行瀏覽。

輸入數值

常用的數值型態有 int、long 與 double 等（關於 int 短整數、long 長整數與 double 實數的差別，下一章有詳細解說），若要輸入這些型態的數值，則應使用不同的列印格式，例如，%d、%ld 與 %lf 等，請輸入以下程式，並觀察執行結果。

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[]) {
   int    a;
   long   b;
   double  c;
   printf("Enter an integer:     ");
   scanf("%d", &a); //請留意是&
   printf("Enter a long integer: ");
   scanf("%ld", &b);//請留意是&
   printf("Enter a float:        ");
   scanf("%lf", &c);//請留意是&
   printf("\n Your enter..... \n");
   printf("integer      a =%d  \n",  a);
   printf("long integer b =%ld \n", b);
   printf("float        c =%f", c);
   return 0;
}
```

請留意輸入資料要有取址符號『&』，漏掉了就是錯誤。以下可輸入兩個數值，輸入資料時，資料中間以逗號分隔。

```
int a,b,c,d;
printf("input a,b:");
d=scanf("%d,%d",&a,&b);//請留意中間是逗號
printf("a+b=%d\n",a+b);
printf("d=%d\n",d);
```

以上執行結果如下：（請自行補程式的頭尾）

```
input a,b:2,3
a+b=5
d=2
```

以下程式可輸入兩筆資料，輸入資料時，資料中間以空白分隔。請留意輸入資料時，兩筆資料中間是空白或逗號『 ,』分隔，是由 scanf("") 內的格式控制。

```
int a,b,c,d;
printf("input a b:");
d=scanf("%d %d",&a,&b);//請留意中間是空白
printf("a+b=%d\n",a+b);
printf("d=%d\n",d);
```

以上程式執行結果如下：（請自行補程式的頭尾）

```
input a b:1 2
a+b=3
d=2
```

其次，scanf() 還傳回正確的資料個數，若輸入資料型態不對，例如，故意輸入 a 3，由於輸入資料型態錯誤；或沒有輸入資料，直接按『Enter』，將傳回『0』，或『EOF』，如下圖：這個觀念在 6_3 節，還有高中解題系統會用到。

```
input a b:a 0
a+b=1
d=0
```

輸入字串

大部分程式語言都有字串 string 型態，但是 C 語言爲了加速處理速度，並沒有字串型態，而是使用字元陣列，因此在輸入字串時，我們必須利用字元陣列來完成輸入。例如，以下程式可輸入一個字串。

```
char c[20];//字元陣列，請看第七章
printf("Please enter a string: ");
scanf("%s", &c);//
printf("Your enter is %s", c);
```

以上輸出結果如下圖：請自行鍵入程式，並觀察執行結果。

```
Please enter a string: Gwosheng
Your enter is Gwosheng
```

關於字串的使用，請看本書第八章。

自我練習

1. 請鍵入以上程式，並觀察執行結果。
2. 請於 https://en.cppreference.com/w 查詢『scanf()』的詳細用法。

2-3　亂數與發聲

亂數

我們寫程式時，有時需要一個亂數，例如樂透開獎，樸克牌發牌等都需要產生一個亂數，所以電腦內部就以記憶體儲存一個亂數表，以供使用者取用，取用亂數的函式是：

```
rand();
```

它將傳回一個 0 到 32767 的任意整數，所以我們宣告一個整數變數暫存，如以下程式：

```
int a;
a=rand();
```

關於 int a，下一章有詳細說明。其次，此函式放在 stdlib.h，所以還要在標頭檔宣告如下：

```
#include <stdlib.h>
```

還有，因爲亂數表是固定的，那要如何取到不同亂數呢？答案是，可取系統時間當依據，因爲每次程式執行時，系統的時間一定不一樣，那就可取到不同的亂數。以下敘述，以當時的系統時間，當作亂數表的起點。

```
srand(time(0));//以當時的系統時間，當作亂數表的起點
```

其次，time() 函式放在 time.h，所以也要載入以下標頭檔。

```
#include <time.h>//本例使用Time()函式
```

範例2-3a

示範以上亂數的取得。

程式列印

```
#include <stdio.h>
#include <stdlib.h>
//步驟1，引入標頭檔
#include <time.h>//本例使用Time()函式
int main(int argc, char *argv[]) {
   int a;
   //步驟2，以當時的系統時間，當作亂數表的起點
   srand(time(0));
   //步驟3，取亂數
   a=rand();
   printf("The Random Number is %d .\n", a);
```

```
    return 0;
}
```

自我練習

1. 請將以上程式的『srand(time(0));』去掉，並觀察執行結果，是否每次都一樣。
2. 請練習於 https://en.cppreference.com/w 尋找『Pseudo-random number generation』，如下圖：

C / Numerics / **Pseudo-random number generation**

Pseudo-random number generation

Defined in header <stdlib.h>

rand	generates a pseudo-random number (function)
srand	seeds pseudo-random number generator (function)
RAND_MAX	maximum possible value generated by rand() (macro constant)

3. 請練習取 0 到 5 的整數亂數。請參考 4-1 節取餘運算子『%』。
4. 請練習取 1 到 6 的整數亂數。請參考 4-1 節取餘運算子『%』。
5. 請練習取 3 到 8 的整數亂數。請參考 4-1 節取餘運算子『%』。
6. 請練習取 -1 到 1 的實數亂數。請參考 4-1 節取餘運算子『%』與除法『/』。
7. 請於 www.tutorialspoint.com 網站探索 C 語言線上說明。
8. 請於 www.programiz.com 網站探索 C 語言線上說明。
9. 請於 www.learn-c.org 網站探索 C 語言線上說明。

發聲

要讓電腦發出聲音，可使用 _beep() 函式，其語法如下：

```
void _beep(unsigned intx,unsignedint y);
```

x 是要發出的頻率，單位是 Hz，y 是發聲的時間，單位是 ms。其次，鋼琴鍵盤頻率對照表如下：

	n	1	2	3	4	5	6	7	8	9	10	11	12
音階	音符	C (Do)	C# (Do#)	D (Re)	D# (Re#)	E (Mi)	F (Fa)	F# (Fa#)	G (So)	G# (So#)	A (La)	A# (La#)	B (Si)
低音	頻率 (Hz)	262	277	294	311	330	349	370	392	415	440	466	494
中音	頻率 (Hz)	523	554	587	622	659	698	740	784	831	880	932	988
高音	頻率 (Hz)	1046	1109	1175	1245	1318	1397	1480	1568	1661	1760	1865	1976

例如，以下敘述可發出『Do』的音，且持續 0.5 秒。

```
_beep(523,500);
```

此函式放在 stdlib.h，所以要載入標頭檔 #include <stdlib.h>

範例2-3b

示範發出 Do、Re、Me、Fa…等八個音。

程式列印

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
   _beep(523,500);
   _beep(587,500);
```

```
    _beep(659,500);
    _beep(698,500);
    _beep(784,500);
    _beep(880,500);
    _beep(988,500);
    _beep(1046,500);
    return 0;
}
```

演奏音樂

要讓電腦演奏音樂，那就要先下載一個簡譜，如下圖。（摘自 http://blog.roodo.com/midiland/archives/1679070.html，特別致謝）

本例一分鐘 80 拍，所以 1 拍的時間是，60000ms/80，以一個四分之一音符爲 1 拍，但實際上每小節是 2 拍，所以一個四分之一音符所佔的時間是 60000ms/160。

範例2_3c

示範演奏音樂。本範例僅作 4 小節，『我有一隻小毛驢我從來也不騎』。

程式列印

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int t=60000/160;
    _beep(523,t);
    _beep(523,t);
    _beep(523,t);
    _beep(659,t);
    _beep(784,t);
    _beep(784,t);
    _beep(784,t);
    _beep(784,t);
    _beep(880,t);
    _beep(880,t);
    _beep(880,t);
    _beep(1064,t);
    _beep(784,4*t);
    return 0;
}
```

自我練習

1. 請繼續完成以上音樂。
2. 請用輸出指令讓輸出音樂的同時，銀幕也輸出歌詞，那就是 KTV 了。
3. 請自己找一首歌，完成輸出歌詞與音階。

補充說明

1. 第五章還會介紹如何製作電子琴。
2. 以上僅輸出一小節，程式就有點冗長，那寫完一首歌不就累垮。請放心，待介紹迴圈與陣列就會明瞭，屆時請讀者重新以迴圈與陣列完成。

警車聲音

警車的警報聲是連續發出頻率 265Hz 與 350Hz 所形成，以下範例示範發出此聲音。

範例2-3d

示範發出警車警報聲。

程式列印

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    while(1){//無窮迴圈，請看第六章
        _beep(265,300);
        _beep(350,700);
        _sleep(50);//空白
    }
    return 0;
}
```

自我練習

1. 市內電話來電大都採用 1000Hz 與 500Hz 交錯形成的樂音，請寫程式完成。
2. 同上題，練習讓鈴聲可越來越急。（此題可待學完迴圈再作）

3. 音感練習。請觀察您家微波爐、洗衣機、汽車安全帶未繫、倒車警告、手機的各種聲響，並寫程式完成。
4. 作曲練習。請嘗試發出不同頻率的聲音，觀察是否悅耳而吸引人，您是否能作曲呢？

2-4 線上程式解題系統

線上程式解題系統目前最熱門的網站有『高中生程式解題系統』與『台中女中程式解題系統』，分別說明如下：

高中生程式解題系統

高中生程式解題系統（https://zerojudge.tw/）畫面如下圖：

高中生程式解題系統 (Online Judge)，它是一個適合所有中學生及初學者的程式設計題庫系統，內有基礎題庫與競賽題庫，每一個題目都有詳盡說明與測試資料，學生還可送出程式，由系統協助檢查是否正確，其次，這個系統不僅能評判是否正確，若有錯誤，還能提示錯誤的地方，真是一個很好的學習資源。下圖是基礎題庫：

基礎題庫　競賽題庫　IOI/APCS　UVa 題庫　原創/不分類題庫

第 1 2 3 4 5 6 7 8 9 10 11 12 頁 |

序號	題目		標籤
1	a001. 哈囉 -- Brian Kernighan	91% / 40805 人	基本輸出輸入
2	a002. 簡易加法	91% / 37590 人	基本輸出輸入
3	a003. 兩光法師占卜術	96% / 30603 人	曆法
4	a004. 文文的求婚	94% / 27915 人	曆法
5	a005. Eva 的回家作業 -- POJ	94% / 23082 人	迴圈
6	a006. 一元二次方程式	94% / 18751 人	數學
7	a009. 解碼器 -- ACM 458	96% / 13609 人	字元處理
8	a010. 因數分解	90% / 9540 人	數論
9	a013. 羅馬數字 -- NPSC 模擬試題	83% / 3358 人	進位制
10	a015. 矩陣的翻轉	96% / 11441 人	陣列

只要學完本書第二章，就可開始從基礎解題。其中第一題參考解答如下，您可以複製此程式，練習此系統的使用。

範例2-4a

請寫一個程式，可以讀入指定的字串，並且輸出此指定的字串。

輸入說明

輸入指定的文字。

輸出說明

輸出指定的文字

輸入範例

```
world
```

輸出範例

```
hello, world
```

參考解答

1. 本例是所有程式語言的第一道任務，要能輸入與輸出，本例要能輸入字串與輸出字串，所以參考程式如下：

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
   char a[1000];
   scanf("%s",&a);//留意&
   printf("hello, %s",a);
   return 0;
}
```

2. 請留意大小寫與空白要全對，才能通過。
3. 先在自己電腦測試，測試無誤，再傳到系統，由系統檢驗，請於下圖點選『測試程式』。

畫面出現如下圖，請點選解題語言『C』，並將程式複製，貼到測試程式碼藍色框框內，如下圖：

測試執行說明：

- “測試執行”讓您得以在送出程式碼之前先測試系統的編譯器。
- 自定測資一律以單測資點進行，測資長度限制在各 1000 字元以內。
- “測試執行”不會計入成績及送出次數。

請注意： 測試執行僅測試您所提供的輸入輸出測資，並非實際題目測資。

解題語言 ：

◉ **C**: gcc -std=c11(gcc 7.3.0)

○ **CPP**: g++ -std=c++14(g++ 7.3.0)

○ **JAVA**: OpenJDK java version 1.8.0

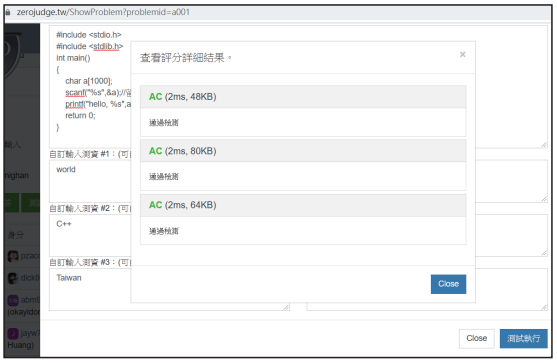
○ **PASCAL**: Free Pascal Compiler version 3.0.4

○ **PYTHON**: Python 3.6.6

測試的 程式碼 ：

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    char a[1000];
    scanf("%s",&a);//留意&
    printf("hello, %s",a);
    return 0;
}
```

於上圖點選『測試執行』，即可測試程式。若完全正確，將會出現下圖：



若有些許錯誤，還有錯誤提示，如下圖，表示遺漏了『;』號。

CE ()

```
/6250647/code_6250647.c: In function ‘main’:
/6250647/code_6250647.c:6:13: warning: format ‘%s’ expects argument of
type ‘char *’, but argument 2 has type ‘char (*)[1000]’ [-Wformat=]
     scanf("%s",&a);//留意&
            ~^   ~~
/6250647/code_6250647.c:8:5: error: expected ‘;’ before ‘return’
```

以上錯誤訊息的意義如下：（以下摘自原網頁）

AC: Accept 即表示通過
NA: Not Accept 表示在多個測資點的情況下，有部分測資點無法通過
WA: Wrong Answer 表示答案錯誤 , 請仔細比對，務必符合題目要求
TLE: Time Limit Exceed 表示執行超過時間限制
MLE: Memory Limit Exceed 表示程序超過記憶體限制
OLE: Output Limit Exceed 表示程序輸出超過限制
RE: Runtime Error 表示執行時錯誤，通常爲記憶體配置錯誤如：使用了超過陣列大小的位置
RF: Restricted Function 表示使用了被禁止使用的函式，並在錯誤訊息中指明使用了什麼不合法的函式。
CE: Compile Error 表示編譯錯誤
SE: System Error 包含 Compile, Runtime 等未定義的錯誤均屬於 System Error
由於本系統以初學者爲導向，因此錯誤訊息儘可能提供最詳細的錯誤資訊，如 WA 訊息會告知您正確答案 (題目測資已公開時)，以及在第幾組資料時發生，讓使用者比較容易判斷錯誤發生在哪哩，而不會像在 ACM/UVa 裡明明做對了，但是就是有莫名的格式錯誤，卻找不出格式錯誤在哪的困境。

4. 若能通過測試，那就可點選『送出解答』，畫面與『測試程式』相近，請自己練習。

5. 以上僅能每次測試一筆資料，但是題目若規定要能通過多筆且不定數量的測試資料，都要加上 while 迴圈，程式如下：

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
   char a[1000];
   while (scanf("%s",&a)!=EOF){
      printf("hello, %s\n", a);
   }
   return 0;
}
```

6. 高中生解題系統比較傾向不定數量的測試資料，而 APCS 則都有指定測試資料的筆數。

台中女中程式解題系統

台中女中程式解題系統 (http://www.tcgs.tc.edu.tw:1218/) 首頁如下圖：

也是一個非常實用熱門的線上解題系統，也是有分基礎題庫與競賽題庫。且還有國中組題庫，下圖是基礎題庫。

基礎題第 1 題是純輸出的題目，系統有提供解答，如下圖：這樣就可以複製此解答，練習系統的操作。

但此系統目前僅開放 C 與 C++，以下我使用 C 寫的參考解答，這樣就可以練習操作此系統。

```
#include <stdio.h>
#include <stdlib.h>
int main() {
   printf("Hello!\n");
      return 0;
}
```

台灣國際資訊奧林匹亞競賽

台灣國際資訊奧林匹亞競賽 toi-reg.csie.ntnu.edu.tw 網站如下圖，裡面有很多新手題目、歷屆試題與解答，也非常值得探索。

2-5 APCS初體驗

若要考 APCS，那 printf() 就要詳細研讀細部語法如下：

```
printf("%[f][w][.p][l]x",arg1[,arg2])
```

f: 可以是 -、+，輸出預設靠右對齊，若要靠左對齊，請加負號(-)。例如，

```
int a=8;
printf("%5d\n",a);//    8
printf("%-5d\n",a);//8
```

正數輸出預設空白，若一定要加上正號，那就加上正號（+）。例如，

```
printf("%-+5d\n",a);//+8
```

請輸入以下程式，並觀察執行結果。

```
int a,b,c;
a=3;b=-4;c=5;
printf("%dx%+dy%+d=0",a,b,c);
```

w: 用來指定資料的總欄位寬度。例如，以下的『5』，表示欄位寬度 5。

```
printf("%5d\n",a);//    8
```

.p: 用來指定小數點的長度。例如，以下的『2』，表示小數佔 2 格。

```
float b=0.034;
printf("%5.2f\n",b);// 0.03 本例共5格，小數點『.』一格，整數僅剩2格
```

l: 用來強調資料是長整數。例如，

```
int a=1234;
long b=123456;
printf("%d\n",a);
printf("%ld\n",b);
```

x: 用來指定資料輸出格式，常用格式有 d,f,c,s,u,e,E,g,G,o,x,X，說明如下表：

符號	說明

d	視為整數輸出。例如，
	`int a=97;`
	`printf("%5d\n",a);//  97`

c	視為字元輸出。例如，
	`int a=97;`
	`printf("%5c\n",a);//  a`

f	視為浮點數輸出。
s	視為字串輸出。

e,E	以科學符號表示。例如，
	`float b=0.034`
	`printf("%E\n",b); //3.400000E-002`

g,G	取浮點數或科學符號較短者輸出。例如，
	`float b=0.034;`
	`float c=0.00000000034;`
	`printf("%g\n",b);// 0.034`
	`printf("%G\n",c);//3.4E-10`
	0.034 會輸出 0.034，但是 0.000000034 則會以科學符號表示。

o	轉為八進位輸出。例如，
	`int d=17;`
	`printf("%o\n",d);//21`

x,X	轉為十六進位輸出。例如，
	`int d=17;`
	`printf("%x\n",d);//11`
	`printf("%X\n",d);//11`

自我練習

題號	題目	結果

1.	`#include <stdio.h>`
	`#include <stdlib.h>`
	`int main() {`
	`  int a=8;`
	`  printf("%5d\n",a);//`
	`  printf("%-5d\n",a);//`
	`  printf("%-+5d\n",a);//`
	`  float b=0.034;`
	`  float c=0.00000000034;`
	`  printf("%5.2f\n",b);//`
	`  printf("%e\n",b);`
	`  printf("%E\n",b);`
	`  printf("%g\n",b);`
	`  printf("%G\n",c);`
	`  int d=17;`
	`  printf("%o\n",d);//`
	`  printf("%x\n",d);//`
	`  printf("%X\n",d);//`
	`  return 0;`
	`}`	

2.	`#include <stdio.h>`
	`#include <stdlib.h>`
	`int main() {`
	`  int a=97;`
	`  printf("%d\n",a);`
	`  printf("%c\n",a);`
	`  return 0;`
	`}`	

範例2-5a

兩點式，兩點可決定一直線，直線方程式爲(y-y1)/(x-x1)=(y2-y1)/(x2-x1))。請寫一程式，可輸入兩個二維座標，並求出此直線方程式。例如，輸入(2,1),(-4,6)，則直線方程式爲 5x+6y-16=0。

設計步驟

1. 假設輸入爲 (x1,y1),(x2,y2)。
2. 由兩點式，直線方程式爲 (y-y1)/(x-x1)=(y2-y1)/(x2-x1))
3. 先計算 d=y2-y1 ,f=x2-x1
4. 代入 (y-y1)/(x-x1)=d/f
5. 乘開 d(x-x1)-f(y-y1)=0
6. 整理上式得方程式爲 dx-fy-dx1+fy1=0

輸出結果

```
5x+6y-16=0
```

程式列印

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[]) {
   int x1,y1,x2,y2;
   int d,f;
   x1=2;
   y1=1;
   x2=4;
   y2=6;
   d=y2-y1;
   f=x2-x1;
   printf("%dx%+dy%+d=0", d,-f,-d*x1+f*y1);
   return 0;
}
```

自我練習

1. 點斜式。輸入一點 p 與斜率 m 可造出一條直線 (y-y1)/(x-x1)=m，請寫程式完成。例如，輸入 p 點是 (1,2),m=3，可得到直線 3x-y-1=0。
2. 截距式。輸入 x 軸與 y 軸截距，也可造出一條直線，由截距式 x/a+y/b=1，可得直線方程式爲 bx+ay-ab=0。例如，x 截距是 4,y 截距是 3 直線方程式是 3x+4y-12=0。