Chapter

3

決策運算

本章大綱

- ☆ 3-1 決策運算子
- ☆ 3-2 if~else ~
- ☆ 3-3 if~elif~
- ☆ 3-4 位元運算子
- ☆ 3-5 實例探討
- ☆ 3-6 APCS 實例探討

人類的生活必須不斷面對決策問題，連我家一個不到三歲的小孩，也常要思考他手裡的十元是要坐電動車還是買棒棒糖。程式語言是協助解決人類問題的工具，當然也有決策指令。本章即是探討 Python 決策所需使用的運算子與指令。請先鍵入以下程式，並觀察執行結果。

```
a=66
if a>=60:
    b="Pass"
else:
    b="Fail"
print(b)
```

請留意冒號『:』與縮排問題。冒號『:』不可遺漏，打完冒號『:』，電腦會自動縮排。鍵入『else:』前，要先前進再打字。其次，請將 a=66 換成 55，並觀察執行結果。以上『>=』稱爲『關係運算子』，於 3-1 節介紹，『if else』稱爲決策指令，於 3-2 節介紹。請再鍵入以下程式，並觀察執行結果。

```
x=3
y=4
if(x>0 and y>0):
    print('第一象限')
```

以上『and』稱爲『邏輯運算子』，它可以讓我們同時連結兩個以上的關係運算子，請看 3-1 節。

3-1 決策運算子

決策運算子包含關係運算子與邏輯運算子，分別說明如下：

關係運算子(Relational Operators)

關係運算子又稱爲比較 (Comparison) 運算子，用於資料之間的大小比較，比較的結果可得到 True 或 False，下表是 Python 中的關係運算子符號，此與 C/C++ 相同。

運算子	定義	優先順序	結合律
<	小於	9	由左至右
>	大於	9	由左至右
<=	小於等於	9	由左至右
>=	大於等於	9	由左至右
==	等於	9	由左至右
!=	不等於	9	由左至右

例如：

```
x=3;y=4
print(x==y) #False
print(x!=y)#True
print(x<y)#True
print(x=y)#錯
```

邏輯運算子(Logical Operators)

當同一個運算式要同時存在兩個以上的關係運算子時，每兩個關係運算子之間必須使用邏輯運算子連結，例如，您要找『男生』且『年齡大於 40』，此一選擇就同時含有兩個關係運算式，此時就要使用邏輯運算子。下表是 Python 的邏輯運算子，C/C++ 用符號『！ ,&&,||』表示，Python 直接用單字表示，如下表，這樣反而比較好記。

運算子	定義	優先順序	結合律
not	邏輯否定運算	10	由右至左
and	邏輯 and 運算	11	由左至右
or	邏輯 or 運算	12	由左至右

not

邏輯否定 not 是將『True』轉爲『False』,『False』轉爲『True』。

and

邏輯 and 是兩件事都 True，結果才是 True，其餘都是 False，其眞值表如下：

事件1	事件2	結果
True	True	True
True	False	False
False	True	False
False	False	False

or

邏輯 or 是兩件事只要有一件爲 True，那就爲 True，只有兩件事全爲 False，才爲 False，其眞值表如下：

事件1	事件2	結果
True	True	True
True	False	True
False	True	True
False	False	False

例如：

```
x=3;y=4
print(not(x==y))#True
print(x>0 and y>0) #True
print(x>0 or x==y) #True
```

範例3-1a

若有一數學式，判斷 x 是否滿足 1<x<=6 , 請轉換爲 Python 語言敘述。

✍說明

1<x<=6 是數學語言，Python 是

x>1 and x<=6

請鍵入以下程式，並觀察執行結果。

```
x=3
print(x>1 and x<=6)
x=8
print(x>1 and x<=6)
```

✍自我練習

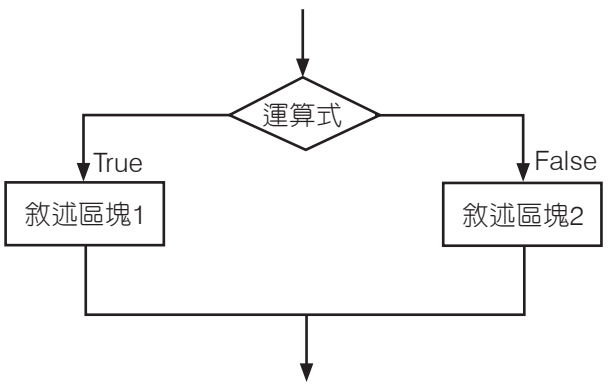
若有一數學式，判斷 x 是否滿足 x>3 or x<-2 , 請轉換爲 Python 語言敘述。

3-2 if~else ~

生活上常會面對『假如～否則～』，Python的指令是『if~else~』，其語法如下：

```
if運算式:
    敘述區塊1；
[else :
    敘述區塊2；]
```

請特別留意冒號『:』與『程式縮排』，因爲冒號『:』是 Python 特有；『縮排』在其他語言是美觀性質，但 Python 是語法也就是規定，因爲縮排才是屬於 if else 要執行的敘述子區塊（若使用 Spyder 編寫程式，Spyder 會幫您補上冒號與縮排），以上程式其流程圖如下：



例如：

```
a=66
if a>=60:
    b="Pass"
else:
    b="Fail"
print(b)
```

以上一定要縮排，因爲，程式縮排在其它語言是美觀問題，但 Python 是語法，所以沒有縮排就不行。例如，以下程式就不行。

```
a=66
if a>=60:
b="Pass"
else:
b="Fail"
print(b)
```

但若是僅執行一個敘述，這樣也行，但是視覺效果很差。

```
a=55
if a>=60:b="Pass"
else:b="Fail"
print(b)
```

請留意，以下兩個程式的執行結果就不同了。

```
a=66
if a>=60:
    b="Pass"
else:
    b="Fail"
print(b)
```

```
a=66
if a>=60:
    b="Pass"
else:
    b="Fail"
    print(b)
```

C 等用大括號『{}』來表示程式區塊的範圍，Python 直接用縮排表示，那就更簡潔了。其次，也可以將否則的部分放在前面當預設值，那程式如下：

```
a=46;b="Fail"
if a>=60:
    b="Pass"
print(b)
```

或

```
a=66;b="Fail"
if a>=60:b="Pass"
print(b)
```

if 內可否放 if，當然可以，此稱爲巢狀 if，請看範例 3-2b。

冒號(:)

冒號 (:) 也是 Python 特有的符號，往後的 for、while、函式、類別定義，都需要這個符號。

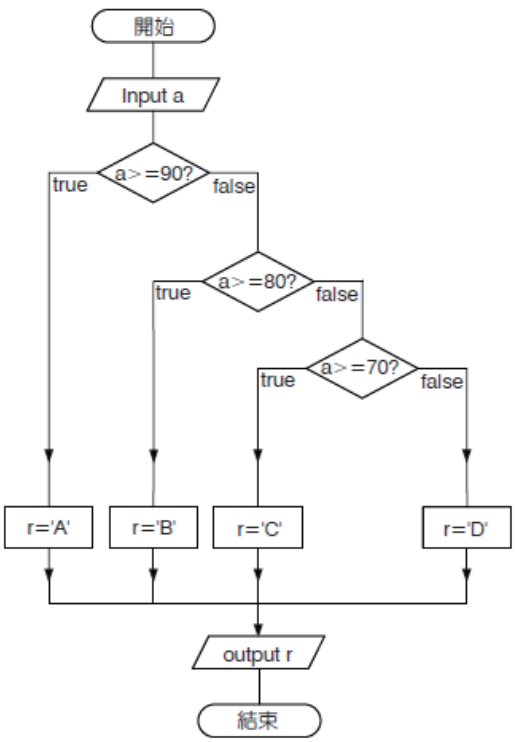
範例 3-2a

請寫一個程式，完成以下要求：

1. 輸入一個 0~100 的分數。
2. 當分數大於 90 分時，輸出 A。
3. 當分數介於 80~89 時，輸出 B。
4. 當分數介於 70~79 時，輸出 C。
5. 當分數介於 0~69 時，輸出 D。

✍流程分析

1. 使用流程圖分析如下：



2. 以上每一個決策流程點，都有兩個分歧點，所以適用 if~else。
3. 每一個 else 後面均需進一步決策流程，所以可在每一 else 後面再放置 if。

✍輸出結果

```
input a grade: 99
the grade is A
```

✍程式列印

```
a=int(input("input a grade: "))#請留意input傳回字串型態
if(a>=90) :        # 高於90分爲A
    r='A'
else :
    if(a>=80):    # 介於 80與90分爲B */
        r='B'
    else :
        if(a>=70):
            r='C'
        else :
            r='D'
print("the grade is %c" % r)
```

✍補充說明

1. 請留意 input 傳回字串型態。
2. 此題有人會寫成 if (90<a<100)，但 Python 並沒有這種運算式，因爲 90<a<100 是數學語言，不是程式語言。
3. 有人會寫成

```
if (a<100 and a>=90) :b='A'
if (a<90  and a>=80) : b='B'
if (a<80  and b>=70) : b='C'
if (a<70  and b>=0) :  b='D'
```

這樣雖然也可以，但是其執行效率非常差，因爲不管分數爲何，僅有一項會成立，如此會讓電腦不斷進行無效的判斷。而且，電腦的判斷時間遠大於執行計算的時間。

4. 也有人會這樣寫，但這是運算思維錯誤，而且結果也不對。

```
a=int(input("input a grade: "))
if(a>=60):
    r='D'
else:
    if(a>=70) :
        r='C'
    else:
        if(a>=80):
           r='B'
        else:
           r='A'
print("the grade is %c" % r)
```

✍自我練習

1. 請寫一程式，判斷所輸入的數是正數、0 或負數。
2. 某一貨品定價 100 元，若購買 10 件 (含) 以上打 9 折，若購買 30 ～ 99 件則打 8 折，若購買 100 件以上則打 7 件，試寫一程式可以輸入購買件數而得總價。測試資料如下：

輸入	輸出
5	500
10	900(100*10*0.9)
30	2400(100*30*0.8)
100	7000(100*100*0.7)

3. 直線。直線標準式爲 ax+by+c=0，請寫一程式，可以指派一直線係數 a,b,c。其次，可再輸入任一點座標，並判斷所輸入點是否在直線上。例如，指派 a,b,c 分別是 1,2,-4，那方程式

就是 x+2y-4=0，其次，輸入點若是 (2,1)，那就是在直線上，若是 (1,2)，那就不在直線上。

4. 請寫一程式，可以輸入一個小寫字元，請判斷其是否爲母音。說明，字元 a,e,i,o,u 稱爲母音，其餘爲子音。
5. 同上題，請寫一程式，可以自動產生一個小寫字元，並判斷是否爲母音。
6. 請寫一個程式，可以產生一個 0 到 25 的亂數，且依以下分數顯示燈號

   21 ～ 25：五個燈。

   16 ～ 20：四個燈。

   11 ～ 15：三個燈。

   6 ～ 10：兩個燈。

   1 ～ 5：一個燈。

   0：零個燈。
7. 表決器。假設有一項評審工作，有 3 位評審，當其中兩人或以上同意，則表示過關，請設計此程式。
8. 請輸入一個整數 x，並判斷此整數是否滿足 1<x<=6。
9. 請輸入一個整數 x，並判斷此整數是否滿足 x>3 or x<-2。
10. 請輸入 3 個線段長度，判斷這 3 個線段，是否可圍一個三角形。圍成三角形的條件是，任兩邊之和要大於第三邊？(提示：任兩邊之和大於第三邊的數學語言是 :a+b > c and a+c > b and b+c > a，請將以上數學語言轉爲 Python。測試資料 3,4,5 可以 1,1,3 不可以)
11. 若有一數學式，同時判斷六個數字是否滿足 $\frac{a_1}{a_2}=\frac{b_1}{b_2}=\frac{c_1}{c_2}$，請轉換爲 Python 敘述。例如，1,2,3 與 2,4,6 就滿足；1,2,3 與 1,2,5 就不滿足。

12. 若有一數學式，同時判斷六個數字是否滿足 $\frac{a_1}{a_2}=\frac{b_1}{b_2}\neq\frac{c_1}{c_2}$ 又如何表示呢？

範例 3-2b

請寫一個程式，可以判斷所輸入座標的所在象限。

✍執行結果

```
input x,y:3,4
I
```

✍程式列印

```
x,y=eval(input("input x,y:")) #數字請用逗號『,』隔開，例如輸入 3,4
if x>0 :
    if y>0:
        b="I"
    else:
        b="IV"
else:
    if y>0:
        b="II"
    else:
        b="III"
print(b)
```

✍補充說明

1. 請留意 eval() 傳回數值型態。
2. 此題目有人會寫成

```
x,y=eval(input("input x,y:")) #數字請用逗號『,』隔開，例如 3,4
if (x>0 and y>0) :
    b="I"
if (x<0 and y>0) :
    b="II";
```

```
if (x<0 and y<0) :
    b="III"
if (x>0 and y<0 ):
    b="IV"
print(b)
```

這樣雖然沒有錯，但是執行效率非常差，因爲電腦要不斷的比較。

✍自我練習

1. 同上範例，但增加先判斷是否在原點或 x、y 軸上。測試資料如下：

輸入 (x,y)	輸出
(0,0)	原點
(0,3)	y 軸
(3,4)	I

2. 同上題，但程式一執行，要求先輸入密碼，密碼若爲『abcd』，那才可執行本程式。
3. 同上題，但程式一執行，要求先輸入密碼，密碼若爲『aa11』、『bb22』、『cc33』或『abcd』，那才可執行本程式。

極大與極小

極大與極小是日常資料處理最常見的問題，以下我們先介紹 2 筆資料的極大值的求法，3 筆或 3 筆以上資料的極大或極小請自行擴充，5 筆以上資料，待學完迴圈與串列也都是相同的道理。

範例 3-2c

請寫一程式，滿足以下條件。

1. 輸入兩個數。
2. 求此兩數極大值。
3. 輸出極大值。

✍演算法則

1. 輸入第一數，本例以變數 a 儲存。
2. 輸入第二數，本例以變數 b 儲存。
3. 設定較大值 (max) 爲第一數。max=a
4. 當第二數 (b) 大於極大值時，極大值即以 b 取代。

```
if (b>max)
    max=b;
```

5. 輸出極大值 (max) 即爲所求。

✍執行結果

```
input a,b:8,2
8
```

✍程式列印

```
a,b=eval(input("input a,b:")) #輸入數字請用逗號『,』隔開，例如 3,4
max=a
if b> max:
    max=b
print(max)
```

✍補充說明

以上是求極大值的演算法，但 Python 已經有 max() 函式，所以本例也可以寫成如下：

```
a=4;b=5;c=1
d=max(a,b)
print(d)
print(max(a,b,c))
```

✍自我練習

1. 請寫一程式，滿足以下條件。
   (1) 輸入三個數。
   (2) 求此三個數的極大值。
   (3) 輸出極大值。

✍演算法則

(1) 輸入第一個數，本例以變數 a 儲存。
(2) 輸入第二個數，本例以變數 b 儲存。
(3) 輸入第三個數，本例以變數 c 儲存。
(4) 設定極大值 (max) 爲第一數 a。
(5) 當第二數 (b) 大於極大值時，極大值即以 b 取代。
(6) 當第三數 (c) 大於極大值時，極大值即以 c 取代。
(7) 輸出極大值 max。

2. 請寫一程式，可以輸入 5 個數值，請去掉最大值與最小值，再求其平均。

排序

排序也是資料處理最常見的問題，排序的演算法很多，例如氣泡、介入、選擇、快速等排序法，本單元先介紹氣泡排序法，且

先以 3 筆資料爲例，然後推廣 4、5 筆，當資料數量變多時，請繼續研讀串列結構與迴圈，那程式就會精簡。

範例 3-2d

請寫一程式，滿足以下條件。

1. 輸入三個數。
2. 將此三個數由小而大輸出。

✍演算法則

1. 分別以 a、b 及 c 表示欲排序的資料，氣泡排序法的演算法如下：
2. 假如 a 大於 b，則 a 與 b 交換，如下圖的 (1)。
3. 假如 b 大於 c，則 b 與 c 交換，如下圖的 (2)。
4. 假如 a 大於 b，則 a 與 b 交換，如下圖的 (3) 排序已完成，共需進行 3 次的比較與交換，如下圖所示。

a	b	c
(1)	(2)	
(3)		

✍執行結果

```
a=2,b=3,c=5
```

✍程式列印

```
a=12
a,b,c=eval(input("input a,b,c:"))
if a>b:
    t=a;a=b;b=t#a,b兩數交換
if b>c:
    t=b;b=c;c=t
```

```
if a>b:
    t=a;a=b;b=t
print ("a=%d,b=%d,c=%d" % (a,b,c))
```

✍補充說明

1. 因爲兩數交換在程式設計領域使用頻率很高，Python 就有同時交換指令，

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

2. 所以，以上程式可簡化如下：

```
a,b,c=eval(input("input a,b,c:"))
if a>b:
    a,b=b,a
if b>c:
    b,c=c,b
if a>b:
    a,b=b,a
print ("a=%d,b=%d,c=%d" % (a,b,c))
```

3. 其次，排序也是程式設計經常使用的步驟，下一章的串列還有 list 方法，不管資料數量爲何？就是一行程式就搞定。
4. 若有 4 筆資料需要排序，則共需進行 6 次比較與交換，如下圖所示。

a	b	c	d
(1)	(2)	(3)	
(4)	(5)		
(6)			

5. 若有 5 筆資料需排序，則共需進行 10 次比較與交換，如下圖所示，此演算法即爲『氣泡排序法』。

a	b	c	d	e
(1)	(2)	(3)	(4)	
(5)	(6)	(7)		
(8)	(9)			
(10)				

6. 以上氣泡沫排序法，處理 3、4 或 5 筆資料的比較與排序，其比較與交換的次數尚可克服。但是，若欲排序的資料超過 5 個，例如 20 筆資料欲排序，那要如何呢？請大家不用擔心，待迴圈與串列 (list) 資料結構介紹以後，程式就有精簡的寫法。
7. 請寫一程式，可以輸入四個人名與其分數，並依照分數由小而大輸出，輸出應含人名與分數。(提示：這題要練習，變數如何取，程式較好寫，其次，當分數交換，人名也要跟著交換，這都是一些程式設計運算思維，請大家好好練習。)
8. 要測試排序是否完成，那測試資料都要給最惡劣的情況，那才能看出排序是否有效。例如，要將資料由大而小排列，那測試資料就要由小而大。

3-3 if～elif～

C 等都有 switch case 解決多個分歧決策問題，Python 卻沒有，但使用 if~elif~ 替代。例如，範例 3-2 的程式，以 if~elif~ 重做如下：

```
a=int(input("input a grade: "))
if(a>=90) :        # 高於90分爲A
    r='A'
elif(a>=80):   # 介於 80與90分爲B */
    r='B'
```

```
elif(a>=70):    # 介於 70與80分爲C */
    r='C'
else :
    r='D'    # 不符合上述情況則爲D */
print("the grade is %c" % r);
```

✍自我練習

成以

1. 請寫一程式，將所輸入的 0、1、2…6，轉爲 ' 星期日 '、' 星期一 '…' 星期六 ' 等字串。
2. 同範例 3_2a 的自我練習第 2 題，請用 if~elif 重作。
3. 請寫一程式，可以自動產生一個 0 到 127 的亂數，以字元方式輸出此亂數，並判斷是否爲大寫字元、小寫字元、或數字。
4. 叫號器。請設計一個程式，可以完成以下功能。
   按『A』鍵，計數值加 10。
   按『S』鍵，計數值加 1。
   按『D』鍵，計數值減 1。
   按『F』鍵，計數值減 10。

3-4 位元運算子

前面有談到電腦如何將時間數位化，電腦並不需要大費周章宣告年月日時分秒等多個變數，只要用一個整數，就可處理與記錄 100 年的日期與時間。目前又有一個問題，請您思考一下複選題答案應該如何數位化。例如，答案有可能 ABC，也有可能 BE，那如何將這些資料數位化，並用適當的變數型態處理呢？是不是用一個字串表示呢？例如，以

```
a="ABD"
```

表示。這樣當然也可以，但每個答案就要占 1 ～ 5 byte(1 byte=8bit)，但 "ABD" 若是以 11010 表示呢，"A" 就以 10000 表示，這樣就可以用 26($11010=2^4+2^3+2^1=26$) 與 16($10000=2^4=16$) 表示答案，這樣竟然只要 1byte 就可以記錄所有答案。好的，以上已經完成非常精簡的資料數位化方式，那電腦如何評分呢，答案就是電腦也備有位元運算子，可以錙銖必較的處理每個位元 (bit)。

位元（Bitwise）運算子

位元運算子是將一個 8 位元的整數逐位元進行運算，此類運算子還可以分爲兩類：位移（Shift）運算子與布林運算子。位移運算子可以用來將各個位元向左或是向右移；布林運算子則可以逐位元進行布林運算。下面是 Python 語言位元操作運算子的列表：

運算子	定義	優先順序	結合律
~	位元 not 運算	5	由右至左
<<	逐位元向左位移	8	由左至右
>>	逐位元向右位移	8	由左至右
&	位元 and 運算	9	由左至右
^	位元 xor 運算	10	由左至右
|	位元 or 運算	11	由左至右

~

『~』是位元 not 運算，not 是將 0 變 1，1 變 0，運算結果如下表：

c=~a	a
0	1
1	0

例如：

```
a=1
print(~a)#-2
```

爲什麼答案是 -2 呢？這有點難懂，筆者簡單說明如下，因爲電腦是以 2 補數表示負數，a=1，分解爲 2 進位是 00000001

~a= not 00000001=11111110

首位元是 1 表示負數，那到底負多少，就取 2 補數。2 補數的步驟是先取 1 補數再加 1。其次，所謂 1 補數是『逐位元將 1 變 0，0 變 1』，所以上面 11111110 的 1 補數是

00000001

將 1 補數再加 1，就是 2 補數，上面 00000001 加 1 就是 00000010，其大小是 2。上面首位元符號既然是『1』，那就表示其爲負數，所以 11111110 代表此數是 -2，所以輸出 -2，此即爲二補數的觀念。

✍補充說明

1. 正數的數位化。正數就直接以 2 進位編碼，本例以 8 位元爲例，如下表：

正數	二進位編碼
0	00000000
1	00000001
2	00000010
3	00000011
4	00000100
5	00000101
6	00000110
7	00000111
254	11111110
255	11111111

2. 正負數的數位化：正負數的數位化，直覺的作法，就將最高位元放 0 代表正數，放 1 代表負數，那直覺的編法如下：

二進位編碼	數字
0000000	0
0000001	1
01111111	127
10000000	-0
10000001	-1
10000010	-2
11111110	-126
11111111	-127

由上表發現，『00000000』與『10000000』都對應到 0，這樣不是一對一函式，這樣無法求得反函式，請好好想念數學老師一再強調，一對一才能求反函式，所以科學家就不斷討論與修正，發現負數的轉換需要兩個步驟，第一，求其 1 補數（將 0 變 1，1 變 0），第二，求 2 補數（1 補數加 1 稱爲 2 補數。）

3. 將負數部分取 2 補數的過程如下：

數字	取正數	1補數	2補數
-0	00000000	11111111	00000000
-1	00000001	11111110	11111111
-2	00000010	11111101	11111110
-3	00000011	11111100	11111101
-126	01111110	10000001	10000010
-127	01111111	10000000	10000001
-128	10000000	01111111	10000000

4. 將上表合併正數，如下表：（請留意正數直接編碼，負數取 2 補數編碼）

數字	取正數	1補數	編碼
127			01111111
126			01111110
2			00000010
1			00000001
0			00000000
-0	00000000	11111111	00000000
-1	00000001	11111110	11111111
-2	00000010	11111101	11111110
-3	00000011	11111100	11111101
-126	01111110	10000001	10000010
-127	01111111	10000000	10000001
-128	10000000	01111111	10000000

5. 由上表可知，若負數先取 2 補數在編碼，竟然就解決正負 0 的問題。這樣就 1 對 1 了，這樣函式，反函式通通有解，這就是爲什麼負數要取 2 補數了。而且這種轉換竟然一石二鳥，也解決減法問題，因爲

   5-2

   可以看成

   5+(-2)

6. 由上表可知，剛剛

```
a=1#a=00000001
print(~a)#~a=11111110   11111110 is  -2
```

爲什麼是 -2 了，若您還是沒有轉過來，麻煩您去買一個 Arduino 實驗板，就可以看到記憶體的內容是『11111110』，請看酌著『中小學自造與程式設計，泉勝』

✍自我練習

請輸入以下程式，並觀察執行結果，且自行使用以上原理推敲，看答案是否一致。

```
a=-3
print(~a)
a=-127
print(~a)
```

<<

『<<』是左移運算子，例如，

```
a=1
print(a<<1)#2    1向左移爲1位元，結果是2
a=1;
print(a<<2)#4    1向左移爲2位元，結果是4
```

『>>』是右移運算子，例如，

```
a=4;
a=a>>1
print(a)#2    4向右移爲1位元，結果是2
a=7
a=a>>2
print(a)#1
```

&

『&』是位元 and 運算，and 運算眞值表如下，當 a 與 b 同時爲 1，c 才得到 1：

c= a & b	a	b
0	0	0
0	0	1
0	1	0
1	1	1

例如：

```
a=1;b=5
print(a&b)#00000001 & 00000101=00000001=1
```

^

『^』是位元 xor 運算，xor 運算眞值表如下，當 a 與 b 不同時，c 才得到 1。

c=a^b	a	b
0	0	0
1	0	1
1	1	0
0	1	1

例如：

```
a=1;b=5
print(a^b)#00000001 ^00000101=00000100=4
```

|

『|』是位元 or 運算，or 運算眞值表如下，當 a 與 b 有一爲 1，c 就得到 1。

c=a | b	a	b
0	0	0
1	0	1
1	1	0
1	1	1

例如：

```
a=1;b=5
print(a|b)#00000001 | 00000101 =00000101=5
```

本書目前已經介紹很多運算子，茲將使用手冊的運算子優先順序摘錄如下：有些還沒介紹，那就先瀏覽。

Operator	Description	優先順序（1最優先）
`lambda`	Lambda expression	17
`if – else`	Conditional expression	16
`or`	Boolean OR	15
`and`	Boolean AND	14
`not x`	Boolean NOT	13
`in, not in, is, is not, <, <=, >, >=, !=, ==`	Comparisons, including membership tests and identity tests	12
`|`	Bitwise OR	11
`^`	Bitwise XOR	10
`&`	Bitwise AND	9
`<<, >>`	Shifts	8

Operator	Description	優先順序（1最優先）
`+, -`	Addition and subtraction	7
`*, @, /, //, %`	Multiplication, matrix multiplication, division, floor division, remainder	6
`+x, -x, ~x`	Positive, negative, bitwise NOT	5
`**`	Exponentiation	4
`await x`	Await expression	3
`x[index], x[index:index], x(arguments...), x.attribute`	Subscription, slicing, call, attribute reference	2
`(expressions...), [expressions...], {key: value...}, {expressions...}`	Binding or tuple display, list display, dictionary display, set display	1

範例3-4a 複選題評分。

大學學測考試的複選題共五個選項，五個選項全答對得 5 分，錯一個選項得 3 分，錯兩個選項得 1 分，錯超過兩個選項得 0 分，未做答也是 0 分，且答錯不倒扣，請寫一程式，可以協助評分，請問您的資料結構爲何？

資料結構

1. 答案要如何表示呢？ 若答案是 ABE，那用 ABE 表示嗎？ 這樣雖然可以，但需要 5 個 bytes，非常浪費記憶體，若將其看爲二進位，那只要以 $(11001)_2=(25)_{10}$ 表示，也就是 1 個 byte 就足夠。

✍執行結果

```
8
-9
4
3
```

✍程式列印

以下的測試資料，請自行用 # 號調整。

```
a=25#標準答案ABE
b=25#答對3個5分
b=17#AE答對2個，錯1個3分
#b=1#E答對1個，錯2個1分
#b=2#D 0分
#b=0#未作答
c=0
d=0
e=0#得分
if (b==0) :
    e=0
else :
    c=a^b#逐位元xor，位元相同是0，不同是1
    d=0;
    print(c);
    c=~c#取補數，1變0，0變1
    print(c);
    d=d+(c & 1)#計算1的個數
    #d=d+c &1 // 錯，因爲+的優先順序大於&
    c=c>>1#右移
    d=d+(c & 1)
    c=c>>1
    d=d+(c & 1)
    c=c>>1
    d=d+(c & 1)
    c=c>>1
    d=d+(c & 1)
    if (d==5):
        e=5;
    elif(d==4):
```

```
        e=3
    elif (d==3):
        e=1
    else:
        e=0
print(d)
print(e)
```

3-5 實例探討

猜拳遊戲

範例3-5a 猜拳遊戲。

請寫一個程式，可以由人和電腦猜拳，並評定勝負。

✍執行結果

```
input 1,2,3:1
您出 剪刀,電腦出 石頭,結果是 computer win
```

✍演算法則

1. 這一任務就是寫人工智慧程式的入門了，寫程式前先想一下您和一個 3 歲小孩猜拳，那您們如何評判勝負？首先，我們先規定有三種拳，分別是『剪刀、石頭與布』，且『剪刀贏布，石頭贏剪刀、布贏石頭，兩者相同則平手等』。『人工智慧』就是要把這些規定以程式語言表示，並由以上規定來評判勝負。

2. 本例因爲電腦有三種拳法，人也有三種拳法，所以共有 9 種情況，您要讓電腦也能思考，其實就是將以上所有可能發生的情況，以程式語言先規定好。

3. 資料的數位化。我們人類猜拳是直接用手勢表示『剪刀、石頭與布』，但是如何讓電腦瞭解您的拳法呢，裝一個攝影機當然可以，但是也還要影像處理，這不是我們這一單元任務，我們電腦只有鍵盤，所以可以先將以上『剪刀、石頭與布』以『1,2,3』表示，此即爲資料的數位化。若使用攝影機，也是要影像處理，再將您的拳法以簡單的數字表示，也是要用『1,2,3』表示，因爲用『1,2,3』，表示只要 1byte 就可以，若您用『剪刀、石頭與布』表示，當然也可以，但一個中文字就佔用 3byte，且往後的處理也是比較複雜。

✍程式列印

```
import random
a=int(input("input 1,2,3:"))#記得要轉型
b=random.randint(1,3)
astr=''
bstr=''
r=''
if a==1:#people
    astr='剪刀'
    if b==1:  #computer
        bstr='剪刀'
        r='平手'
    elif b==2:
        bstr='石頭'
        r='computer win'
    elif b==3:
        bstr='布'
        r='people win'
elif a==2:
    astr='石頭'
    if b==1:
        bstr='剪刀'
        r='people win'
    elif b==2:
        bstr='石頭'
```

```
        r='平手'
    elif b==3:
        bstr='布'
        r='computer win'
    astr='布'
    if b==1:
        bstr='剪刀'
        r='computer win'
    elif b==2:
        bstr='石頭'
        r='people win'
    elif b==3:
        bstr='布'
        r='平手'
print("您出 %s,電腦出 %s,結果是 %s" % (astr,bstr,r))
```

✍補充說明

這一題輸入資料時，若忘了將資料轉爲數值，如以下程式，那眞的哭不完，因爲沒有錯誤，但就是沒結果。

```
a=input("input 1,2,3:")#請留意a是字串型態
```

✍自我練習

那三個人同時猜拳呢，由電腦產生兩個亂數，由您和電腦猜，且評判輸贏。（提示：兩個人共 9 種情況，用兩層決策，三個人共 27 種情況，可用 3 層決策）

✍路口紅綠燈

路口的紅綠燈如何設計燈號的變化，這一問題要讓電腦處理，也是要先想好如何將資料數位化，這種問題通常將燈號的變化和時間建立函數關係，就稱爲時序圖。

範例3-5b　紅綠燈程式。

假設有單向紅綠燈時序如下，請設計程式完成其功能。

時序	燈號
0	綠燈
1	綠燈
2	綠燈
3	綠燈
4	綠燈
5	黃燈
6	黃燈
7	紅燈
8	紅燈
9	紅燈
10	紅燈

✍執行結果

```
t=0,綠燈
t=1,綠燈
t=2,綠燈
```

✍運算思維

1. 您要讓程式不間斷執行，那就要一個無窮迴圈如下：（下一章會介紹）

```
while 1:
```

2. 要讓程式依照時序無限循環，且不溢位，就要聯想取餘運算『%』。

```
while 1:
    t=(t+1)% 11
```

3. 全部程式如下：

```
import time
a='綠燈'
b='黃燈'
c='紅燈'
d=''
t=0
while 1:
    if t<=4:
        d=a
    elif (t==5 or t==6):
        d=b
    else:
        d=c
    print('t=%d,%s'%(t,d) )
    time.sleep(1)
    t=(t+1)% 11
```

4. 這一題，也可以藉助串列（第六章介紹），先將所有情況以串列儲存，然後每一個時間點，就輸出其對應值。

```
import time
a=['綠燈','綠燈','綠燈','綠燈','綠燈','黃燈','黃燈','紅燈','紅燈',
'紅燈','紅燈']
t=0
while 1:
    print('%d,%s' %(t,a[t]))
    time.sleep(1)
    t=(t+1)% 11
```

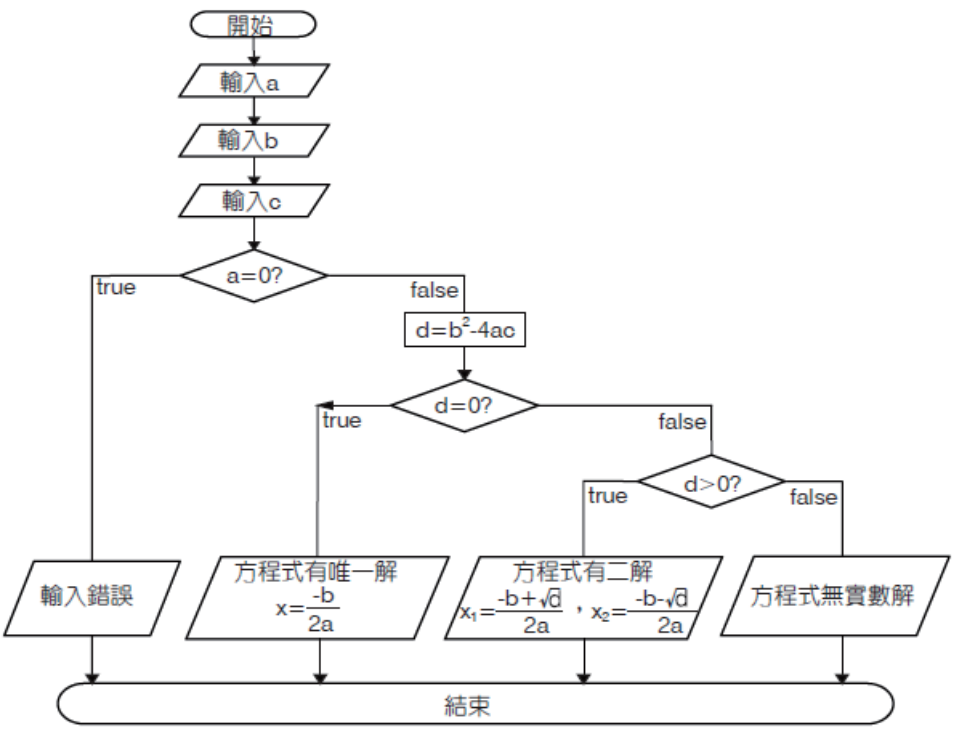
✍自我練習

觀察十字路口雙向紅綠燈，規劃其時序，寫出其程式。

一元二次方程式

前面我們直接假設所輸入的一元二次方程式有解，因爲不是隨便給三個係數，一個方程式就通通有解，本例則要加上判斷了，解一元二次方程式的演算法如下：

1. 設有一元二次方程式如下：
   $ax^2 + bx + c = 0$
2. 若 a=0 則輸出“輸入錯誤”。
3. 令 $d=b^2 - 4ac$。
4. 若 d=0，則方程式有唯一解 $x = \frac{-b}{2a}$；否則，若 d>0，則方程式有二解 $x_1 = \frac{-b+\sqrt{d}}{2a}$，$x_2 = \frac{-b-\sqrt{d}}{2a}$；否則，無實數解。
   以上演算分析，以流程圖說明如下：



範例 3-5c

請設計一個程式，可以解一元二次方程式 $ax^2 + bx + c = 0$。

執行結果

```
input a,b,c:1,3,-10
two answer,x1= 2.000000,x2=-5.000000
```

程式列印

```
a1,b1,c1=eval(input("input a,b,c:"))
a=int(a1)
b=int(b1)
c=int(c1)
if(a==0):
    print("input error")# 若a=0，則列印錯誤訊息
else:
    d=b**2-4*a*c# 計算d值 */
    if(d==0):
        print("only one answer,x1,x2= %d" % (-b/(2*a)))
    elif d>0 :
        d=d**(1/2)
        x1=(-b+d)/(2*a)
        x2=(-b-d)/(2*a)
        print("two answer,x1= %f,x2=%f"%(x1,x2))
    else:
        print("no real answer")
```

自我練習

1. 解二元一次方程式。解二元一次方程式的演算法如下：

   (1) 設二元一次方程式如下：

$$a_1x + b_1y = c_1$$
$$a_2x + b_2y = c_2$$

(2) 令 $d = \begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix}$（表示 $d = a_1b_2 - a_2b_1$）

(3) 假如 $\frac{a_1}{a_2} = \frac{b_1}{b_2} = \frac{c_1}{c_2}$，則方程式無限多解，且程式結束。

（代表兩重疊直線）

(4) 假如 $\frac{a_1}{a_2} = \frac{b_1}{b_2} \neq \frac{c_1}{c_2}$，則方程式無解，且程式結束。

（代表兩平行直線）

(5) $x = \frac{\begin{vmatrix} c_1 & b_1 \\ c_2 & b_2 \end{vmatrix}}{d} = (c_1b_2 - c_2b_1)/d$

(6) $y = \frac{\begin{vmatrix} a_1 & c_1 \\ a_2 & c_2 \end{vmatrix}}{d} = (a_1c_2 - a_2c_1)/d$

2. 請設計一個程式，可以解二元一次方程式。

閏年的判斷

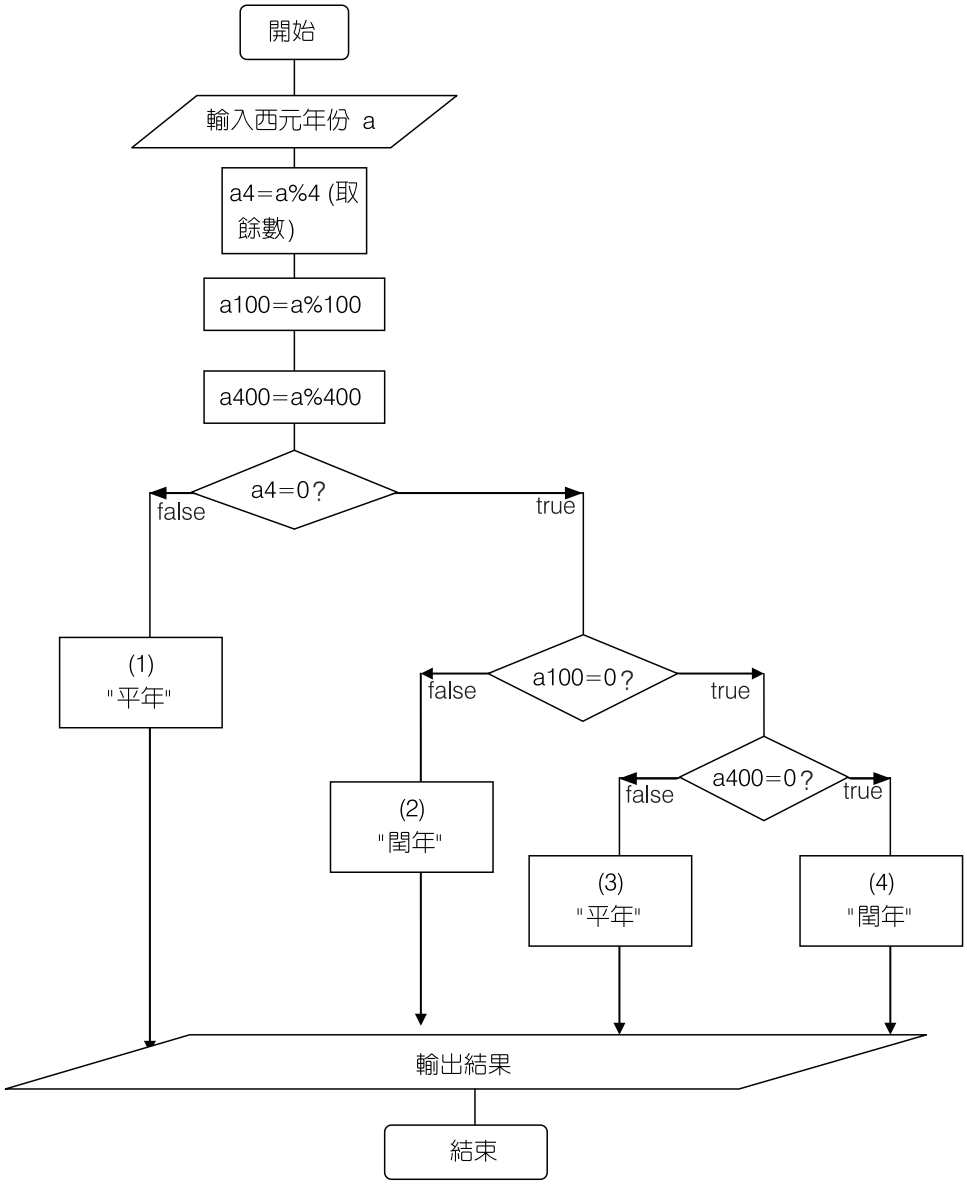
西元的閏年爲每 400 年必須有 97 次閏年，其規劃方式如下：

(1) 4 的倍數。依此條件共有 100 次。

(2) 於 (1) 的條件，扣掉 100 的倍數。依此條件，共有 96 次。

(3) 於 (2) 的條件，再加回 400 的倍數。所以共有 97 次。

以上演算分析，流程圖分析如下：


開始
輸入西元年份 a
a4=a%4 (取餘數)
a100=a%100
a400=a%400
a4=0?
false
true
(1)
"平年"
a100=0?
false
true
(2)
"閏年"
a400=0?
false
true
(3)
"平年"
(4)
"閏年"
輸出結果
結束

以下提供一些測試資料，以便對照流程路線。

西元年份	性質	流程路線
3	平年	(1)
4	閏年	(2)
100	平年	(3)
200	平年	(3)
300	平年	(3)
400	閏年	(4)
600	平年	(3)
1200	閏年	(4)
2000	閏年	(4)

範例5-3d

請寫一個程式，可以將使用者所輸入的西元年份，判斷其爲平年或閏年。

程式列印

```
a=2000
a4=a%4
a100=a%100
a400=a%400
if a4==0:
    if a100==0:
        if a400==0:
            d='閏年4'
        else :
            d='平年3'
    else:
        d='平年2'
else :
    d='平年1'
print(d)
```

補充說明

以上程式，亦可簡化如下：

```
a=2001
a4=a%4
a100=a%100
a400=a%400
if (a % 4 == 0 and not(a % 100 == 0) or (a % 400 == 0 )):
    d="leap year"
else:
    d="common year"
print(d)
```

3-6 APCS實例探討

一、三角形判斷（APCS105第二梯次試題）

若已知三角形三邊長，判斷是否構成三角形、評定三角形種類與計算三角形面積的演算法如下：

1. 輸入三角形的三邊長 a、b、c。
2. 任兩邊之和要大於第三邊。
3. 若 c 是最長邊，假如 $a^2 + b^2 > c^2$ 則爲銳角三角形；否則，假如 $a^2 + b^2 = c^2$，則爲直角三角形；否則此三角形爲鈍角三角形。
4. 令 $d = \frac{1}{2}(a+b+c)$
5. 三角形面積 $= \sqrt{d(d-a)(d-b)(d-c)}$

請輸入三角形三邊長，首先判斷是否構成三角形、其次判別三角形的種類，最後計算其面積。

範例一：輸入	範例二：輸入	範例三：輸入
3 4 5	101 100 99	10 100 10
範例一：正確輸出	**範例二：正確輸出**	**範例三：正確輸出**
3 4 5 Right	99 100 101 Acute	10 10 100 No
（說明） a×a＋b×b＝c×c 成立時為直角三角形。	（說明） 邊長排序由小到大輸出， a×a＋b×b＞c×c 成立時為銳角三角形。	（說明） 由於無法構成三角形， 因此第二行須印出 「No」。

✍執行結果

```
input a,b,c:3,4,5
直角三角形
6.0
```

✍程式列印

任兩邊之和要大於第三邊，本例就將最大邊找出來，這樣就一石二鳥，也可以方便判斷是否形成三角形，也能判斷三角形種類。

```
a1,b1,c1=eval(input("input a,b,c:"))
a=int(a1)
b=int(b1)
c=int(c1)
if a>b:
    a,b=b,a
if b>c:
    b,c=c,b
if a>b:
    a,b=b,a
if (a+b>c)  :
    c2=c**2
    b2=b**2
    a2=a**2
    if ((a2+b2)>c2):
        t="銳角三角形"
```

```
    elif ((a2+b2)==c2):
        t="直角三角形"
    else:
        t="鈍角三角形"
    d=(a+b+c)/2
    area=(d*(d-a)*(d-b)*(d-c))**(1/2)#留意括號的對稱
    print(t)
    print(area)
else:
    print("無法構成三角形")
```

✍補充說明

1. 『任兩邊之和要大於第三邊』如何以程式表示？
   這句話同義於，a+b>c and b+c>a and c+a>b 或將三邊長由小而大重新排列，最小的放入 a，其次放入 b，最大的放入 c。最小的兩邊之和若小於等於第三邊，則此三邊未能構成三角形，程式提早離開。本例因爲還要判斷三角形的種類，所以採用後者。
2. 本題測試資料是資料與資料之間給空白，所以資料輸入程式如下，且用 split() 分割，sort() 排序。

```
d=input('a b c:')
e=d.split(' ')#分割資料
print (e) #e的型態是字串
d=[int(e1) for e1 in e] #串列生成式，請看第六章，本例是將每一元素轉爲數值
d.sort()  #排序
print(d)
a=d[0];b=d[1];c=d[2]
if  (a+b)<= c:
    print('No')
elif (a*a+b*b< c*c) :
    print('Obtuse triangle' )
elif(a*a+b*b==c*c):
    print('Right triangle')
else :
    print('Acute triangle')
```

二、城市移動（108/第一梯次）

假設有一個遊戲，有 3 個城市，玩家依亂數 L 在 3 個城市移動，移到城市 1 得 L 分，如果 L 是偶數則下一回合移到城市 2，否則留在城市 1；移到城市 2 得到 2L 分，如果 L 是 3 的倍數，下一回合移到城市 3，否則前往城市 1；移到城市 3 得分是 L 除以 10 的商，如果 L 是 5 的倍數，下一回合移到城市 3，否則移到城市 1。測試資料如下：

```
4 1 //玩4回合，起始城市是1
1 2 3 12 //四個亂數
```

備註：此題目爲憑個人記憶，題目很好，還是呼籲主辦單位公佈試題，讓學生練習。

演算法

因爲每一城市有其不同移動規則與得分方式，所以可用 if 分成 3 部分，且因每一城市都有其移動規則，所以每一城市都要自己用 1 個 if else，這樣就可以計算每一次轉移到哪一城市與每一次的得分。所以這題是屬於雙層決策問題，拿到分數沒有問題。參考解答如下：

```
p=input('input n a:')
q=p.split(' ')
n=int(q[0])#次數 ;串列，請看第六章
a=int(q[1])#起始城市
bt=0#總得分
for i in range(n):
    l=int(input('input l:'))
    if (a==1):
        b=1#得分
        if(l %2==0):
            a=2
        else:
            a=1
```

```
    elif(a==2):
        b=2*1#得分
        if(l %3==0):
            a=3
        else:
            a=1
    else:
        b=l//10 #得分
        if(l %5==0):
            a=3
        else:
            a=1
    bt=bt+b
print(bt)
```

```
input n a:4 1

input l:1

input l:2

input l:3

input l:12
5
```

三、主客場籃球賽制。（108/第二梯次）

1. 連打兩場，主場球隊兩場皆贏輸出 Win。
2. 兩場皆輸，輸出 Loss。
3. 其餘輸出 Tie。
4. 輸入格式：每場 4 節得分，主場先輸出。以下分別是主隊、客隊兩場比賽的四小節得分

```
15 20 25 30
15 16 17 18
5 10 15 20
5 6 7 8
```

5. 參考解答如下：

```
def inputdata():
    p=input('input a b c d:')
    q=p.split(' ')
    print(q) #字串
    r=[int(q1) for q1 in q] #將字串中的每一元素轉爲數值
    print(r)
    return sum(r)
a1=inputdata()
b1=inputdata()
a2=inputdata()
b2=inputdata()
num=0
if a1>b1 :
    num=num+1
if a2> b2:
    num=num+1
if num==2:
    print('Win')
else:
    if num==0:
        print('Loss')
    else:
        print('Tie')
```

以下僅是第一天主場球隊的輸出結果，其餘略。

```
input a b c d:15 20 25 30
['15', '20', '25', '30']
[15, 20, 25, 30]
```

習 題

1. 假設所得稅稅率累進法則如下：

   (1) 淨所得 30 萬以下繳納 6%。

   (2) 淨所得 30 ～ 80 萬之間，則前面 30 萬繳納 6%，超過 30 萬的部分繳納 13%。

   (3) 淨所得在 80 ～ 200 萬之間繳納 21%。（前面 30 萬繳 6%，30 ～ 80 萬之間繳 13%）

   (4) 淨所得超過 200 萬，超過的部分繳納 30%。

   試寫一程式可以輸入淨所得，並計算應繳納稅額。測資如下：

淨所得	納稅額
20 萬	20 萬 *6% =12000
40 萬	30 萬 ×6% +10 萬 ×13% = 31000
100 萬	30 萬 ×6% +50 萬 ×13% +20 萬 *21% =125000
220 萬	30 萬 ×6% +50 萬 ×13% +120 萬 *21% +20 萬 * 30% =395000

2. 假設自來水費率如下：

   A. 100 度以下，每度 3 元。

   B. 100 ～ 300 度，超過 100 度的部分，每度 5 元。

   C. 300 度以上，超過 300 度的部分，每度 6 元。

   根據以上條件，寫一程式，可輸入用水度數，得到水費。例如，測資如下表：

度數	水費
50	50 * 3 = 150
200	100 * 3 + 100 * 5 = 800
400	100*3+200*5+100*6=1900

3. 假設計程車的計費方式如下：

   1000 公尺以內 60 元。 超過 1000 公尺時，

   (1) 日間以每 500 公尺加收 6 元，不足 500 公尺時，以 500 公尺計算。

   (2) 夜間以每 300 公尺加收 6 元，不足 300 公尺，以 300 公尺計算。

   請根據以上條件，寫一程式完成車資的計算。本例，日間輸入 1，夜間輸入 0，再輸入一個里程數，並計算車資。測資如下表：

模式	里程	車資	模式	里程	車資
1	500	60	0	500	60
1	1800	60+6*2=72	0	1800	60+8*3=84

4. 寫一程式輸入 x 值，並印出其所對應的值，其函數對應如下：

$$y = f(x) = \begin{cases} x+3 & x > 3 \\ x^2 & 1 <= x <= 3 \\ \sqrt{x} & 0 < x < 1 \\ 0 & x <= 0 \end{cases}$$

5. 請寫一程式，可以輸入三點座標，輸出其面積。

   提示：三角形面積公式如下：若面積為零，則表示共線。

$$\frac{1}{2}\begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_1 & y_1 \end{vmatrix} = \frac{1}{2}(x_1y_2+x_2y_3+x_3y_1-x_2y_1-x_3y_2-x_1y_3)$$

6. 同上題，任意輸入 3 點，檢查是否共線，若未共線，輸出其面積。

   提示：任 3 點面積若為 0，則共線。

7. 判斷任意點 D 是否在三角形 ABC 內或外。已知 ABC 三點座標，請寫一程式，可以輸入 D 點座標，並判斷任一點 D 是否在三角形 ABC 內或外。
   （提示：D 若在三角形內，則三角形 ABC 面積 = DAB + DBC + DAC 的面積）

8. 凸或凹四邊形。已知 ABCD 四點座標，請寫一程式，可以判斷四邊形 ABCD 是否為凸或凹四邊形。
   （提示：若為凸四邊形，則任一點，均不得在任三點所圍成的三角形內）

9. 同上題，若為凸四邊形，並計算其面積。 提示：任意凸多邊行，都可以將第 5 題公式推廣，並且代入。

10. 於二維平面中，請寫一程式，可以判斷兩直線平行、或相交或重合，若平行計算其距離，相交計算其交點。

11. 寫一程式，可以任意輸入凸四邊形的四個端點座標，並可將其按順時針方向排列此四個點。（提示：請聯想高中數學的極座標）

12. 停車場計費。假設停車場每小時 30 元，不足 1 小時以 1 小時計，每日最高 180 元，請寫一程式，可以計算其停車費。測試資料如下：

編號	進場時間	離場時間	停車時間	金額
1	09:40	11:30	110	60
2	09:50	20:20	630	180