python 程式設計

第 二 講

邏 輯 與 迴 圈

布林數

- 布林數：真或假

假：False , None , 0 , 0.0 , 0j , "" , [] , () , {}

真：True , 非假

例如：

```
>>> a = True                    # a 為真
>>> b = False                   # b 為假
>>> c = ( 3 > 4 )               # c 為假
>>> d = 0                       # d 為假
>>> e = not ( 3 > 4 )           # e 為真
```

❖ 數字如果不為零都代表為 True

比較運算子

- 六個比較運算子：< <= > >= == !=

符　號	名　稱
<	小於
<=	小於或等於
==	等於
>	大於
>=	大於或等於
!=	不等於

例如：

```
>>> x = 7
>>> a = ( x < 5 )          # a 為假
>>> b = ( x >= 3 )         # b 為真
>>> c = ( 2**2 == x )      # c 為假
>>> d = ( 2 < x <= 10 )    # d 為真
```

布林運算子： not and or

- not A ： 用來逆轉 A 的真假值
- A and B ： 當 A 與 B 兩個皆為真才為真
- A or B ： 為 A 與 B 其中一個為真即為真

```
>>> x = 30
>>> a = not ( x > 20 )                # a 為假
>>> b = ( x < 50 and 3*x > 70 )       # b 為真
>>> c = ( x == 20 or x > 40 )         # c 為假
```

- 遇到複雜邏輯運算，可適時使用小括號藉以簡化式子

```
d = ( ( 0 < x < 30 and x%2 == 0 ) or
      ( x > 50 and x%5 != 0 ) )
```

流程控制 (一)

- 流程控制：控制程式執行路徑
  - **A if C else B**: 如果 C 為真則執行 A 否則執行 B

```
s = int( input("> ") )
print( "P" if s >= 60 else "F" )    # s >= 60 印出 P 否則印出 F


#  A 與 B 可以為運算式
s = int( input("> ") )                  # A = 3*5+1   B = 2+3*4
x = 3 * 5 + 1 if s else 2 + 3 * 4   # s 非零 x=A , s 為零 x=B
```

此種條件式經常併入運算式中，例如：

```
s1 = int( input("> ") )
s2 = s1 + ( 60-s1 if 55<s1<60 else 0 ) # 若去除小括號，則式子等同
                                       # s2 = 60 if 55<s1<60 else 0
```

流程控制 (二)

- **if A : B :** 如果 **A** 為真則執行 **B**

```
if x > 3 : print(x)

# 如果 B 不只一個式子，則 B 的每一行都要使用縮排
  if 55 < x < 60 :
       x = 60
       print(x)
```

- **if A1 : B1 elif A2 : B2 … else :**

```
if score >= 90 :
    print( "A" )
elif 80 <= score < 90 :
    print( "B" )
elif 70 <= score < 80 :
    print( "C" )
else :
    print( "F" )
```

❖ 根據問題需要，elif 或 else 可加以省略，elif 的數量不受限制。

條件式為數字

- 條件式為數字：當數字為 0 或者 0.0 為假，其它數字皆為真

```
for i in range(-3,4) :
    print( ”a” if i else ”b” , end=”” )    # 印出 aaabaaa

以上等同:
for i in range(-3,4) :
    if i != 0 :
        print( "a" , end="" )
    else :
        print( "b" , end="" )
```

迴圈與條件式 (一)

- 迴圈與條件式經常混合交織在一起

➢ **X 圖形**

```
n = int( input("> ") )

for i in range(n):
    for j in range(n):
        if i == j or i+j == n-1 :
            print( 'x' , end="" )
        else :
            print( ' ' , end="" )
    print()
```

```
x     x
 x   x
  x x
   x
  x x
 x   x
x     x

 n = 7
```

迴圈與條件式 (二)

- 漏斗圖案

```
n = int( input(”> ” ) )

for i in range(n) :
    print( " "*i + "*"*(2*(n-i)-1) )

for i in range(n-2,-1,-1 ) :
    print( " "*i + "*"*(2*(n-i)-1) )
```

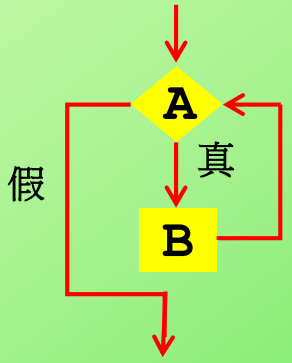
```
*******
 *****
  ***
   *
  ***
 *****
*******
 n = 4
```

while 迴圈 (一)

- while A : B : 當 A 為真，重複執行 B 直到 A 為假

```
while A : B
```

```
while A :
    B
```



- while 迴圈經常可改用 for 迴圈達到相同效果

```
i , s = 0 , 0
while i < 10 :
    s += i
    i += 1
```

```
s = 0
for i in range(10) :
    s += i
```

while 迴圈 (二)

- 列印 1! 到 10!

```
# 讓 p , i , n 三個變數分別 1 , 1 , 10
p , i , n = 1 , 1 , 10
while i <= n :
    p *= i
    print( i , "! = " , p , sep="" )
    i += 1
```

```
1! = 1
2! = 2
3! = 6
4! = 24
5! = 120
6! = 720
7! = 5040
8! = 40320
9! = 362880
10! = 3628800
```

無窮迴圈：無止盡重複執行

```
while True :
    n = int(input("> ") )
    for i in range(n) :
        print( " "*(n-1-i) + "*"*(2*i+1) )
    for i in range(n-2,-1,-1) :
        print( " "*(n-1-i) + "*"*(2*i+1) )
```

```
> 5
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
```

```
> 4
   *
  ***
 *****
*******
 *****
  ***
   *
```

break (一)： 提前跳出迴圈

- 使用 **break** 可跳出一層迴圈，提前結束迴圈

```
for a in A :
    if B : break
C
```

```
while A :
    for b in B :
        if C : break
    D
    if d : break
E
```

break (二)

- **while True 常與 break 合用 :**

```
while True :

    n = int( input(”> ” ) )

    # n <= 0 則跳離迴圈
    if n <= 0 : break

    for i in range(n) :
        print( ” ”*(n-1-i) + str(i+1)*(2*i+1) )
```

```
> 4
   1
  222
 33333
4444444
```

```
> 5
    1
   222
  33333
 4444444
555555555
```

break (三)

- 列印不在九九乘法乘積的兩位數

```
i = 0
for n in range(10,100) :
    found = True                          # 先設定 found 「找到」為真
    for x in range(2,10) :                # 除數範圍 x 為 [2,9]
        if n%x == 0 and n//x < 10 :       # 當 x 能整除 n 且商為個位數
            found = False                 # 排除數字 n, 設定 found 為假
            break                         # 並提早跳離迴圈

    if found :                            # 當 found 「找到」仍為真
        i += 1
        print( n , end=" " )
        if i%20 == 0 : print()            # 每 20 個數換行
print()
```

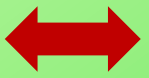
輸出:

```
11 13 17 19 22 23 26 29 31 33 34 37 38 39 41 43 44 46 47 50
51 52 53 55 57 58 59 60 61 62 65 66 67 68 69 70 71 73 74 75
76 77 78 79 80 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96
97 98 99
```

continue (一)：提前進入下個佚代

- **continue** 可提前進入下個佚代步驟，但並非如**break**跳出迴圈

```
while True :
    if A : continue
    B
```



```
while True :
    if not A :
        B
```

continue (二)

- 列印不是 3 或 5 或 7 倍數的兩位數

```
i = 1
for  n  in range(10,100) :
    if n%3 == 0 or n%5 == 0 or n%7 == 0 : continue
    print( i , n )
    i += 1
```

以上等同:

```
i = 1
for  n  in range(10,100) :
    if not ( n%3 == 0 or n%5 == 0 or n%7 == 0 ) :
        print( i , n )
        i += 1
```

❖ 當「not」條件式不容易理解時，就是使用 continue 的時機

簡單格式輸出 (一)：format

- **format** 設定資料的輸出格式，以字串表示
- 可設定資料輸出寬度 **(w)**，向靠右 **(>)** 或向左 **(<)** 對齊，填補字元 **(f)**
- **"{:f>w}".format(a)：資料 a 用 w 格輸出，靠右對齊，如果有剩餘格數，填補上 f 字元，若沒有設定 f，則以空格替代**

語法	產生字串
"{:#<5}".format(17)	'17###'
"{:>2}-{:>2}".format(3,28)	' 3-28'
"20{:<2}/{:0>2}/{:0>2}".format(18,4,5)	'2018/04/05'
"{:<3}={:#>5}".format("pi",3.14)	'pi =#3.14'

簡單格式輸出 (二)

```
for i in range(1,6) :
    for j in range(1,6) :
        print( "{:>1}x{:>1}={:>2}".format(i,j,i*j) , end=" ")
    print()
```

輸出:

```
1x1= 1 1x2= 2 1x3= 3 1x4= 4 1x5= 5
2x1= 2 2x2= 4 2x3= 6 2x4= 8 2x5=10
3x1= 3 3x2= 6 3x3= 9 3x4=12 3x5=15
4x1= 4 4x2= 8 4x3=12 4x4=16 4x5=20
5x1= 5 5x2=10 5x3=15 5x4=20 5x5=25
```

❖ 更詳細的 format 使用方式請參考第 ?? 頁

pass：空式子

- pass 不執行任何動作，為空式子

```
for s in range(1,100) :
    if s < 60 :
        pass                    # 尚未決定處理方式，先以 pass 暫代
    else :
        print(s)
```

❖ 當程式仍在開發階段時，未完成區塊可先使用 pass 替代藉以保持語法的正確性

exit()： 提前離開程式 (一)

- 提前離開程式

```
while True :
    n = int( input("> ") )
    if n < 0 : exit()              # 當 n 小於 0 時隨即離開程式
    ...
```

- **exit(str)** ： 印出字串 **str** 後離開程式
- **exit(n)** ： 離開程式時將整數 **n** 回傳給作業系統

❖ 以 exit 中斷程式，若不刻意回傳整數，作業系統也會收到整數 1

exit()： 提前離開程式 (二)

- **sys.exit(): exit 函式多用於互動式操作上。**
  若為一般程式，最好使用定義於 sys 套件的 sys.exit 函式，用法與 exit 函式相同。使用前需用 import sys 將 sys 套件加入程式內

```
import sys                                  # 將 sys 套件加入程式中使用

while True :
    n = int( input("> ") )
    if n < 0 : sys.exit("n < 0") # 印出 n < 0 後隨即離開程式
    ...
```

常用的預設函式 (一)

- 函式語法:

函式	作用
`abs(x)`	回傳 `x` 的絕對值
`pow(x,y)`	回傳指數函數 x^y 的數值
`min(x,y,...)`	回傳輸入參數中的最小值，參數數量不限
`max(x,y,...)`	回傳輸入參數中的最大值，參數數量不限
`round(x,n)`	回傳最靠近浮點數 `x` 的數字，`n` 設定取位的小數位數預設為 `0` 。若接近 `x` 的數有兩個，則選擇偶數

常用的預設函式 (二)

- 範例:

用法	結果	說明
`abs(-3)`	`3`	
`pow(2,3)` `pow(2,-2)`	`8` `0.25`	
`min(2,9,3)`	`2`	
`max(2,9)`	`9`	

`round(3.56)`
`round(2.5)`
`round(2.35,1)`
`round(2.345,2)`
	`4`
	`2`
	`2.4`
	`2.35`
		
		取偶數
		
		截去誤差影響取位結果

❖ `round(x,n)` 常會因數字 `x` 的截去誤差影響到計算結果

練習題 (一)

1.找出三位數的數字和為 10 且數字都不同的所有三位數，例如：325、910，驗證共有 40 個數。

練習題 (二)

2.撰寫程式印出由 2 到 99 所有數的質因數乘積。

```
2 = 2
3 = 3
4 = 2 x 2
5 = 5
6 = 2 x 3
7 = 7
8 = 2 x 2 x 2
9 = 3 x 3
10 = 2 x 5
11 = 11
12 = 2 x 2 x 3
```

```
13 = 13
14 = 2 x 7
15 = 3 x 5
16 = 2 x 2 x 2 x 2
17 = 17
18 = 2 x 3 x 3
19 = 19
20 = 2 x 2 x 5
21 = 3 x 7
22 = 2 x 11
```

```
23 = 23
24 = 2 x 2 x 2 x 3
25 = 5 x 5
26 = 2 x 13
27 = 3 x 3 x 3
28 = 2 x 2 x 7
29 = 29
30 = 2 x 3 x 5
...
```

練習題 (三)

3.輸入兩整數，印出兩數的直式乘積運算式:

```
> 238 , 11208
     238
x  11208
---------
    1904
   476
  238
 238
---------
 2667504
```

```
> 98123 , 3084
    98123
x    3084
----------
   392492
  784984
 294369
----------
 302611332
```

練習題 (四)

4.以下運算式的每一個中文代表不同的數字，撰寫程式印出滿足的數學式子。

```
    學 習 再 學 習
×           學
-------------------
  優 優 優 優 優 優
```

練習題 (五)

5.撰寫程式印出直式九九乘法表:

```
  1   1   1   1   1   1   1   1   1
x 1 x 2 x 3 x 4 x 5 x 6 x 7 x 8 x 9
--- --- --- --- --- --- --- --- ---
  1   2   3   4   5   6   7   8   9
  2   2   2   2   2   2   2   2   2
x 1 x 2 x 3 x 4 x 5 x 6 x 7 x 8 x 9
--- --- --- --- --- --- --- --- ---
  2   4   6   8  10  12  14  16  18
...
  9   9   9   9   9   9   9   9   9
x 1 x 2 x 3 x 4 x 5 x 6 x 7 x 8 x 9
--- --- --- --- --- --- --- --- ---
  9  18  27  36  45  54  63  72  81
```

練習題 (六)

6.輸入列數產生以下 W 圖形:

```
> 4
*       *       *
 *     * *     *
  *   *   *   *
   * *     * *
    *       *
```

```
> 5
*         *         *
 *       * *       *
  *     *   *     *
   *   *     *   *
    * *       * *
     *         *
```

練習題 (七)

7.輸入列數產生以下 W 數字遞增圖形:

```
> 4
1       7       3
 2    6  8    2
  3  5    9  1
    4       0
```

```
> 5
1         9         7
 2      8  0      6
  3    7     1    5
    4  6      2  4
      5        3
```

練習題 (八)

8.輸入數字，使用一層迴圈產生以下圖案:

```
> 2
* * - - - -
* * - - - -
+ + - - - -
+ + - - - -
```

```
> 3
* * * - - - - - -
* * * - - - - - -
* * * - - - - - -
+ + + - - - - - -
+ + + - - - - - -
+ + + - - - - - -
```

練習題 (九)

9.輸入方格內最大數字，印出以下方形數字分佈圖案:

```
> 3
1 1 1 1 1
1 2 2 2 1
1 2 3 2 1
1 2 2 2 1
1 1 1 1 1
```

```
> 4
1 1 1 1 1 1 1
1 2 2 2 2 2 1
1 2 3 3 3 2 1
1 2 3 4 3 2 1
1 2 3 3 3 2 1
1 2 2 2 2 2 1
1 1 1 1 1 1 1
```

練習題 (十)

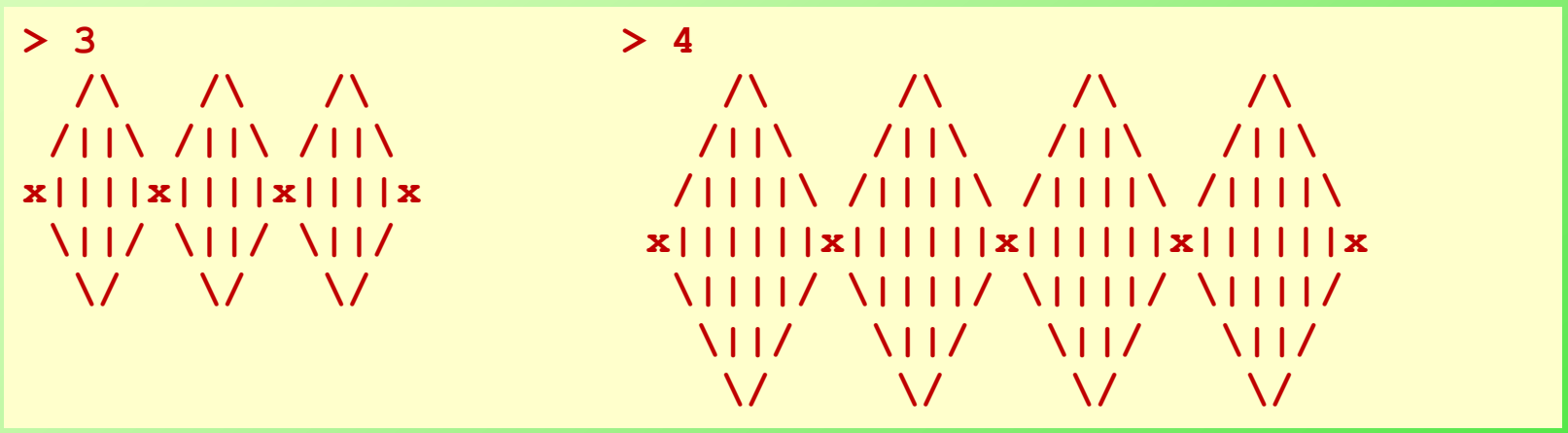
10.輸入最大數字 n ，印出以下菱形數字分佈圖案:

```
> 3
    1
  2 2 2
1 2 3 2 1
  2 2 2
    1
```

```
> 4
      1
    2 2 2
  2 3 3 3 2
1 2 3 4 3 2 1
  2 3 3 3 2
    2 2 2
      1
```

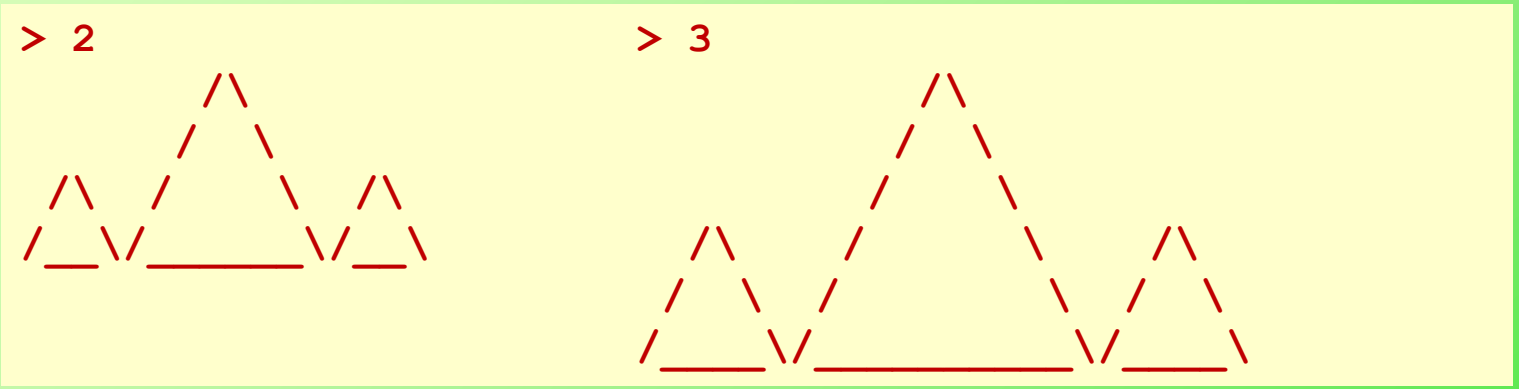
練習題 (十一)

11.輸入數字 n 印出 n 個連在一起鑽石形狀



練習題 (十二)

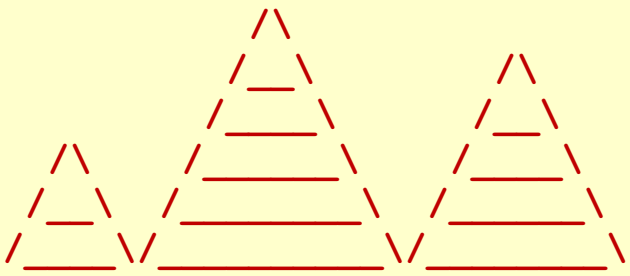
12.輸入小山的高，印出大小不等高的三座山：



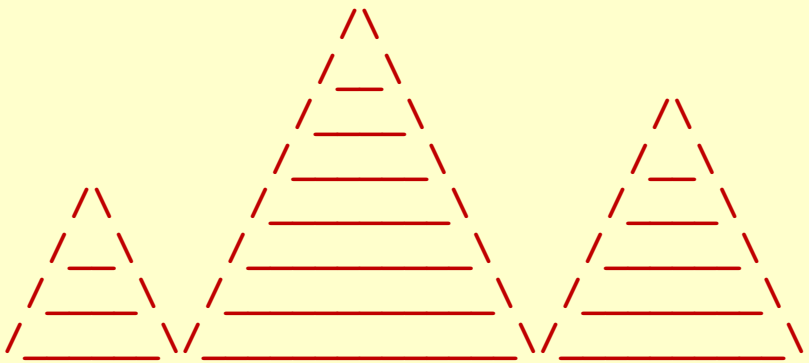
練習題 (十三)

13.輸入小山的高，印出大小不等高的三座山：

```
> 3
```



```
> 4
```



練習題 (十四)

14.設定一些 python 變數代表萬國碼中的一些字符:

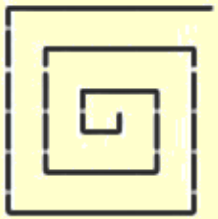
```
hh , vv = chr(9472) , chr(9474)
nw , ne , sw , se = chr(9484) , chr(9488) ,
chr(9492) , chr(9496)
```

各變數所對應的字符如下:

hh	─	vv	│	nw	┌	ne	┐	sw	└	se	┘

撰寫程式產生以下螺旋圖形。

> 6

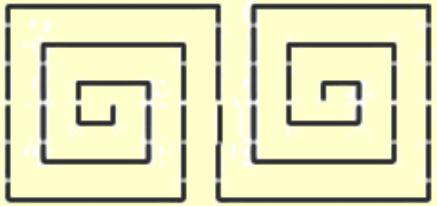


> 5

練習題 (十五)

15. 使用上題字符，產生以下雙螺旋圖形。

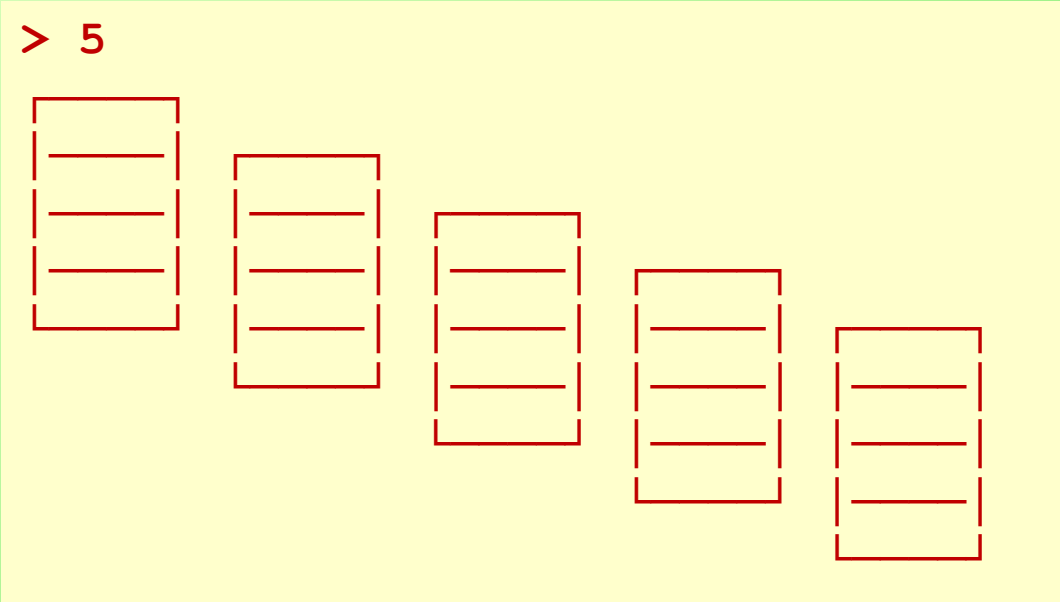
> 6



> 5

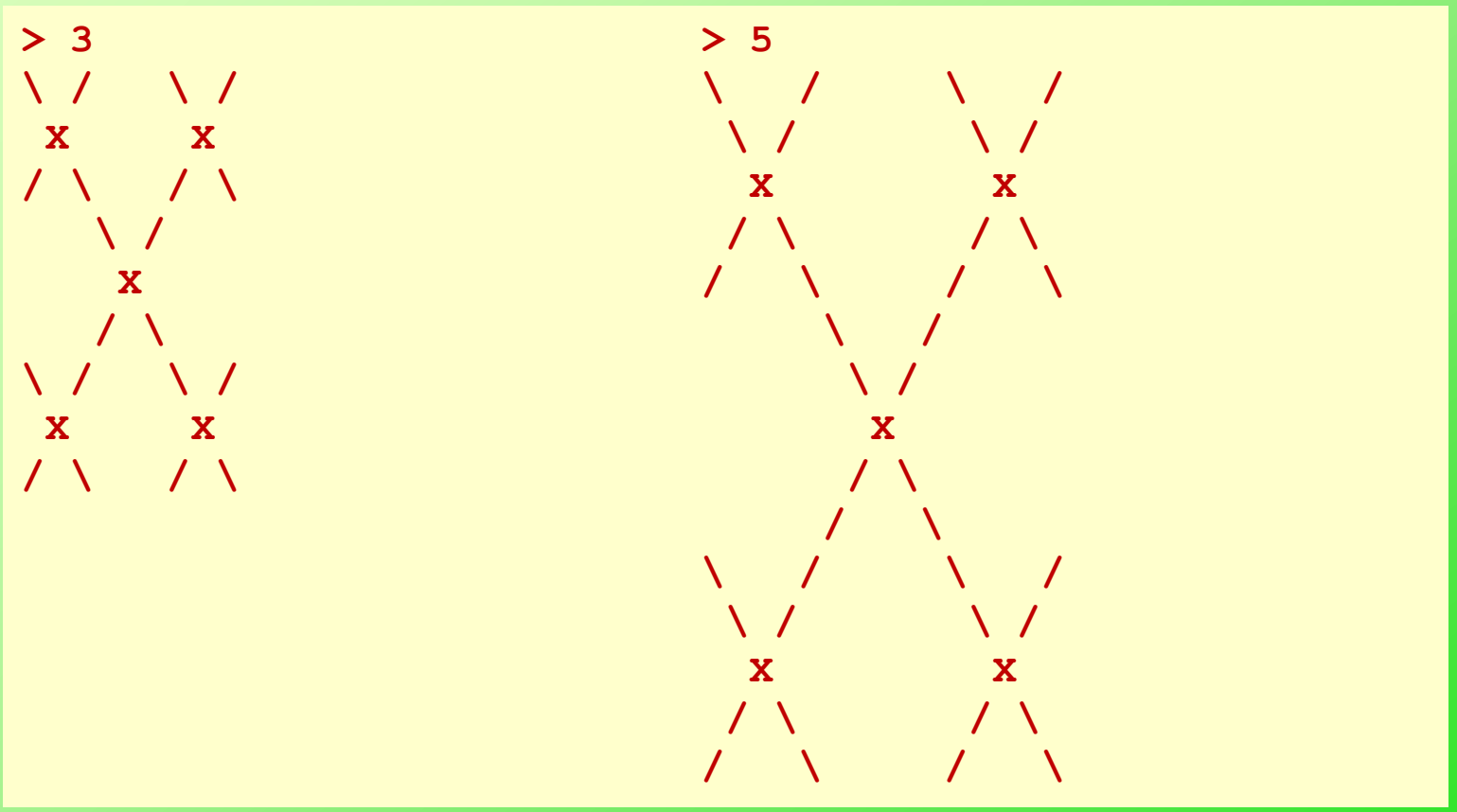
練習題 (十六)

16.使用第 14 題字符，輸入高 n ，產生以下傾斜排列的 n 個方塊圖案:



練習題 (十七)

17.輸入奇數，撰寫程式設計四層迴圈產生以下圖案:



練習題 (十八)

18.輸入小 X 的列數 n(奇數)，列印以下大 X 旁邊有兩個連在一起的小 X 圖案:

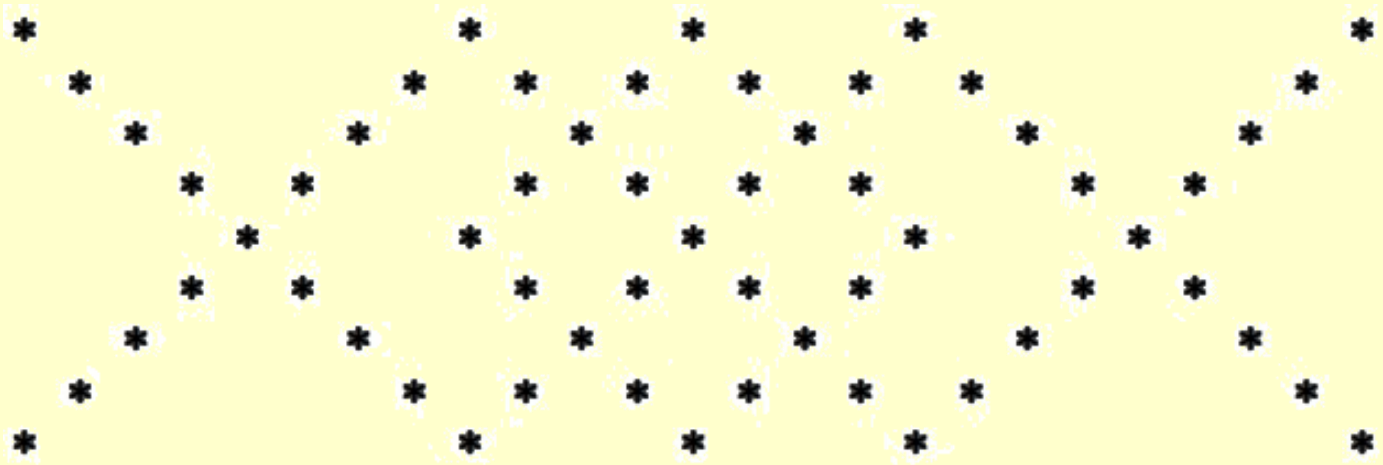
```
> 3
*       *   *
  *   *   *
    *   *   *
  *   *   *
*       *   *
```

```
> 5
*               *       *
  *           *   *   *
    *       *       *
      *   *   *   *
        *       *   *
      *   *       *   *
    *       *       *
  *           *   *   *
*               *       *
```

練習題 (十九)

19.同上輸入，列印兩個大 x 之間有兩組連在一起的小 x 圖案：

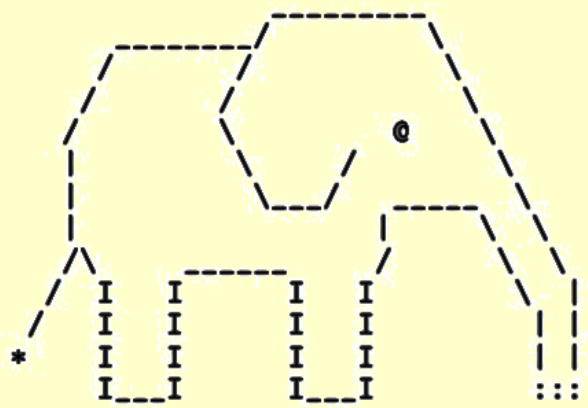
```
> 5
```



練習題 (二十)

20.撰寫程式，輸入數字 n 產生對應的大象圖案。

```
> 1
```



```
> 2
```