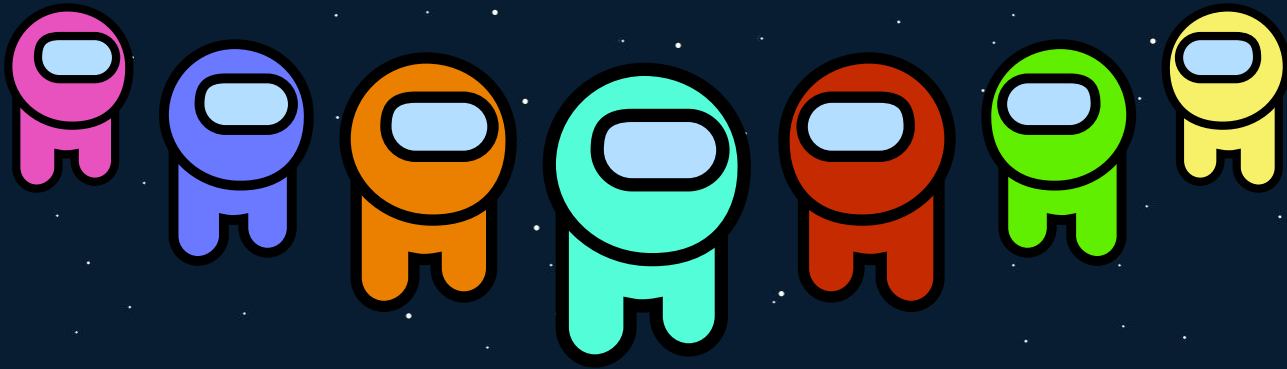


<http://gg.gg/scist-python>

Happy Python Day

高英耀

joseph@gm.tnfsh.tn.edu.tw



本課程由以下贊助商贊助辦理



奧義智慧科技™
Powered by CyCraft

DEV✓CORE



少年圖靈計畫
Young Turing Program



TEAM T5
Persistent Cyber Threat Hunters

Happy Python Day上午場

APCS實作2級分應具備程式設計能力

A.變數與資料型態

B.運算式與運算子

C.輸入與輸出

D.條件判斷與迴圈

E.列表(list)

F.內建函數與物件方法

G.常用輸入技巧

H.字串(str)

I.解題範例

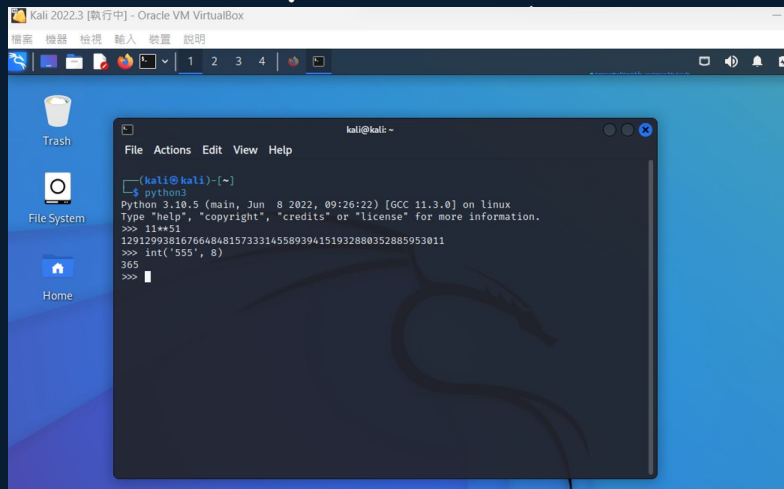
課程相關資訊

- Python語法 : Python 3.X
- 解題平台 : SecurityFocusOnline(<http://140.110.112.224/>)
 - [Python101]:全、[Crypto102]:CRY11、CRY14
- 自主練習 : 高中生程式解題系統(<https://zerojudge.tw/>)
 - [d483](#), [a001](#), [d827](#), [d064](#), [a058](#), [a009](#), [a147](#), [a149](#), [a003](#), [a005](#), [a004](#), [e189](#),
[a024](#), [a022](#), [a038](#), [a148](#), [a104](#),
[APCS實作第1題] [g595](#), [h081](#), [g275](#), [f605](#), [f312](#), [f579](#).

Python 撰寫程式模式

[交互模式] Terminal 執行 Python3 指令

註：離開交互模式，請輸入 `exit()`

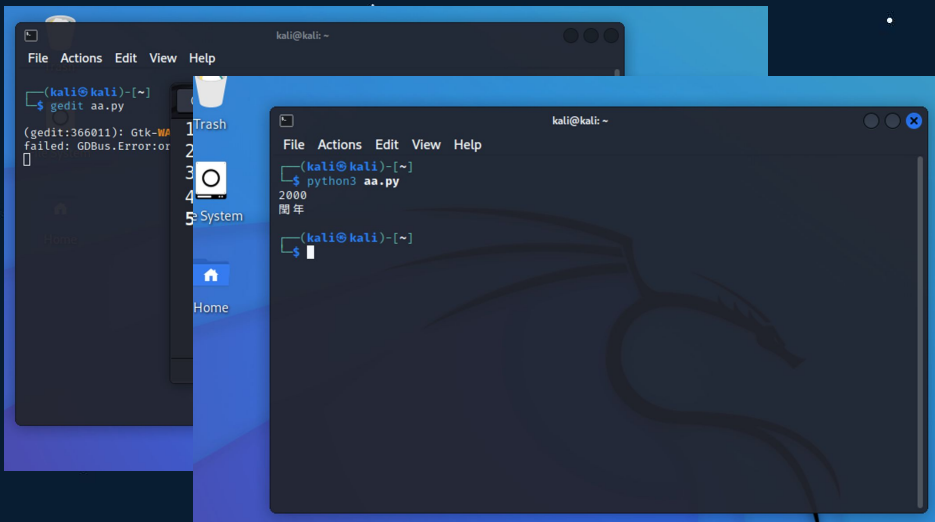


[標準模式] 撰寫一份完整程式

(1) 文書編輯軟體編寫 Python 程式:

\$ `gedit xx.py` (主檔名可自己命名, 副檔名需為 `.py`)

(2) 編譯 Python 程式: \$ `python3 xx.py`



A

變數與資料型態

A-1.變數命名規則

程式中的變數主要用於儲存使用者輸入資料，並隨程式執行，其儲存內容亦會隨之改變，故稱之為變數。

- 變數命名規則
 - 由英文、數字、底線組成
 - 不得以數字開頭
 - 不能與Python內建的保留字相同

A-2. 變數指派

- Python變數**不需宣告**，依指派值自動設定資料型態。

- 指派語法：

變數名稱 = 指派值

- 範例：

a = 5 # a為整數，且值為5

b = 3.14 # b為浮點數，且值為3.14

c = 'Taiwan' # c為字串，且值為Taiwan, 亦可使用雙引號"含括

A-2. 變數指派

- 多個變數指派相同值 `a = b = c = 10`
- 同一列指派多個變數 `name, number = 'Joseph', 35`
- 兩個變數內容互換 `a, b = b, a`

A-3. 資料型態

- 數值型態(Numeric Data Types) : `int`, `float`, `bool`
- 字串型態(String Data Types) : `str`, `chr`
- 容器型態(Container Types) : `list`, `dict`, `tuple`, `set`

A-4. 資料型態轉換

- Python 自動轉換

- `score = 60`

- `score = score + 3.5` # 自動轉換為浮點數, 結果為63.5

- 函數強制轉換

- `int()`

- `# 將括弧內的資料轉換為整數`

- `float()`

- `# 將括弧內的資料轉換為浮點數`

- `str()`

- `# 將括弧內的資料轉換為字串`



B



運算式與運算子



B-1. 運算式

- 運算式規則: 變數 = 運算式
- 範例: $\text{sum} = ((a+b)*3+c*4) / 10$
- 說明: 將等號右邊結果存於變數 sum

B-2. 運算子

- 運算子俗稱運算符號，可區分為
 - 指派運算子：`=`
 - 算術運算子：`+`, `-`, `*`, `/`, `%`, `//`, `**`
 - 關係運算子：`>`, `<`, `>=`, `<=`, `==`, `!=`
 - 邏輯運算子：`and`, `or`, `not`
 - 成員運算子：`in`
 - 位元運算子：`&`(且), `|`(或), `^`(互斥或), `~`(反相)
 - 位移運算子：`<<`(向左位移), `>>`(向右位移)

B-2-1. 算術運算子

算術運算子	意義	範例	運算結果
+	加	5+3	8
-	減	5-3	2
*	乘	5*3	15
/	除	5/3	1.6666666666666667
%	餘數運算	5%3	2
//	商數運算	5//3	1
**	指數運算	5**3	125

B-2-2. 關係運算子

算術運算子	意義	範例	運算結果
>	大於	5>2	True
<	小於	5<2	False
>=	大於等於	5>=2	True
<=	小於等於	5<=2	False
==	等於（比較）	5==2	False
!=	不等於	5!=2	True
>	大於	5>2	True

B-2-3. 邏輯運算子

算術運算子	意義	範例	運算結果
and	和	$(5 > 3) \text{ and } (3 > 2)$	True
		$(5 > 3) \text{ and } (3 < 2)$	False
		$(5 < 3) \text{ and } (3 > 2)$	False
		$(5 < 3) \text{ and } (3 < 2)$	False
or	或	$(5 > 3) \text{ or } (3 > 2)$	True
		$(5 > 3) \text{ or } (3 < 2)$	True
		$(5 < 3) \text{ or } (3 > 2)$	True
		$(5 < 3) \text{ or } (3 < 2)$	False
not	反相	$\text{not}(5 > 3)$	False
		$\text{not}(5 < 3)$	True

B-2-4. 複合指定運算子

複合指定運算子	意義	原運算式	縮寫後運算式
+=	加法	$A=A+3$	$A += 3$
-=	減法	$A=A-3$	$A -= 3$
*=	乘法	$A=A*3$	$A *= 3$
/=	除法	$A=A/3$	$A /= 3$
%=	求餘數	$A=A\%3$	$A \% = 3$
//=	求商數	$A=A//3$	$A //= 3$
=	指數	$A=A3$	$A ** = 3$

B-2-5. 成員運算子

`in` 用來判斷`list`, `string`等資料型態是否包含指定元素、`dict`是否包含指定的`key`, 亦可與`for`連用。

範例	輸出	範例	輸出
<pre>a = [1, 2, 3] print(1 in a) print("1" in a)</pre>	True False	<pre>d = {1:2, 3:4} print(1 in d) print(2 in d)</pre>	True False
<pre>b = "123" print("23" in b)</pre>	True	<pre>a = [1, 2, 3] print(1 not in a) print("1" not in a)</pre>	False True

運算式與運算子練習

SecurityFocusOnline [Python101]

1. 11的51次方



輸入與輸出



C-1. 輸出函數

- print 語法

`print([項目1, 項目2, ... , sep = 分隔字元 , end = 結束字元])`

- 若輸出變數內容, 直接放變數名稱
- 若要輸出字串, 前後加上' 或"
- sep (分隔字元) 預設為空白字元
- end (結束字元) 預設為換行字元\n

C-2. 輸出函數範例

範例	輸出	範例	輸出
<pre>a, b = 5, 10 print(a, b)</pre>	5 10	<pre>a, b = 5, 10 print(a, end='') print(b)</pre>	510
<pre>a, b = 5, 10 print(a, b, sep=',')</pre>	5,10		

C-3. 輸出格式化字串

Python 有 3 種不同的方式來達成字串格式化 (String format).

若輸出較複雜的字串, 可採用以下方式:

- %-formatting
- str.format (Python 2.6+)
- f-string (Python 3.6+)

C-3-1. %-formatting

- 最早的Python 格式化字串延用C語言的字串格式化
- 透過% 運算符號, 將在元組(tuple)中的各個元素依照指定的格式化方式輸出。
 - %s(字串)、%d(十進位整數)、%f(浮點數)

```
>>> "I am %s %s. %s" % ("Monkey", "D", "Luffy")  
'I am Monkey D. Luffy'
```

C-3-2.str.format()

- Python2.6 新增格式化字串函數str.format(·)
- 解決 %-formatting 不易閱讀的困擾。
- 透過 { } 和 format 來代替 % 符號

```
>>> s = 'I am {first_name} {middle_name}. {last_name}'  
>>> s.format(first_name='Monkey', middle_name='D', last_name='Luffy')  
'I am Monkey D. Luffy'
```

C-3-3.f-string

- Python3.6 新增f-string格式化字串功能, 只需於字串前加上前綴字**f**
- 解決 %-formatting 不易閱讀的困擾。
- 解決str.format() 接變數後程式碼超長的問題

```
first_name = "Monkey"  
middle_name = "D"  
last_name = "Luffy"  
print(f"I am {first_name} {middle_name}. {last_name}")
```

Output:

I am Monkey D. Luffy

C-4. 輸入函數

- input: 以字串型態讀入整行資料，直至換行為止。語法如下：

變數 = input([提示字串])

範例	輸出
<pre>a = input('輸入國文成績:') b = input('輸入數學成績:') c = input() print('三科成績為%5s %5s %5s' %(a,b,c))</pre>	<pre>輸入國文成績:75 輸入數學成績:55 80 三科成績為75 55 80</pre>

C-5. 輸入強制轉資料型態

- input()搭配int()將輸入的字串型態轉成整數型態

變數 = int(input())

範例	輸出
<pre>a = int(input('輸入國文成績:')) b = int(input('輸入數學成績:')) c = int(input()) print('三科總分為%5d' %(a+b+c))</pre>	<pre>輸入國文成績:75 輸入數學成績:55 80 三科總分為210</pre>



條件判斷與迴圈



D-1-1.if敘述

if 語法

if 條件式:

程式區塊

- 條件式可不用括號(), 條件式後需搭配冒號 :
- 程式區塊以縮排方式處理, 同一層縮排視為同一程式區塊



奇偶數判斷

```
num = int(input( ))
```

```
if num%2 == 0:
```

```
    print('%d is even' %(num))
```

```
if num%2 != 0:
```

```
    print('%d is odd' %(num))
```



```
60
```

```
60 is even
```


D-1-2.if-else敘述

if-else 語法

if 條件式:

程式區塊1

else:

程式區塊2



奇偶數判斷

```
num = int(input( ))
```

```
if num%2 == 0:
```

```
    print('%d is even' %(num))
```

```
else:
```

```
    print('%d is odd' %(num))
```



77

77 is odd

D-1-3.if-elif-else敘述

if-elif-else 語法

```
if 條件式1:  
    程式區塊1  
  
elif 條件式2:  
    程式區塊2  
  
....  
  
else:  
    程式區塊N
```



```
# 成績等第判斷  
score = int(input( ))  
if score >= 90:  
    print('A')  
elif score >= 80:  
    print('B')  
elif score >= 70:  
    print('C')  
elif score >= 60:  
    print('D')  
else:  
    print('F')
```



55
F

D-2-1.while敘述

while 語法

while 條件式:

程式區塊

```
▶ m, n = int(input()), int(input())  
  while(n>0):  
    m, n = n, m%n  
  print(m)
```

```
↳ 42  
   75  
   3
```

```
▶ n = int(input())  
  while(n > 0):  
    print(n%10)  
    n = n//10
```

```
↳ 12345  
   5  
   4  
   3  
   2  
   1
```

D-2-2.if與while的差異

基本結構名稱	流程圖表示法
判斷結構	<pre>graph TD; Entry(()) --> Cond{條件式}; Cond -- False --> A[程式敘述 A]; Cond -- True --> B[程式敘述 B]; A --> ExitA(()); B --> ExitB(())</pre>
重複結構	<pre>graph TD; Entry(()) --> Cond{條件式}; Cond -- True --> S[程式敘述]; S --> Entry; Cond -- False --> Exit(())</pre>

D-3-1.for敘述

for 語法

```
for 變數 in 序列:
```

```
    程式區塊
```

- for迴圈的變數會依序走訪序列中的元素
- 序列可為range函式、字串(string)、表列(list)、元組(tuple)、字典(dict)、集合(set)

D-3-2.for+range()

for 語法

```
for 變數 in range([起始值,] 終止值[, 遞增值]):  
    程式區塊
```

- 起始值預設為0, 遞增值預設為1
- 起始值 \leq range()的範圍 $<$ 終止值 (左閉右開)

D-3-3.for+range()範例



```
for x in range(6):  
    print(x, end=' ')
```



0 1 2 3 4 5



```
for x in range(1, 11):  
    print(x, end=' ')
```



1 2 3 4 5 6 7 8 9 10



```
for x in range(3, 10, 2):  
    print(x, end=' ')
```



3 5 7 9

D-3-4. 序列為字串(string)範例

```
▶ s = 'From A to A+'  
  for x in s:  
    print(x)
```

```
☞ F  
   r  
   o  
   m  
  
   A  
  
   t  
   o  
  
   A  
   +
```

```
▶ s = input()  
  for x in s:  
    print(chr(ord(x)+3), end='')
```

```
☞ TNFSH  
   WQIVK
```

註: ord() ⇒ 將字元轉為ASCII編碼(整數)
chr() ⇒ 將ASCII編碼(整數)轉為字元

D-4.break敘述

break語法：跳出當前迴圈



#break範例

```
pre = post = 0
while(True):
    m, n = int(input()), int(input())
    if (m == n): break
    elif (m > n): pre += 1
    else: post += 1
print('前者大:%d、後者大:%d' %(pre, post))
```



```
30
10
40
10
20
20
```

前者大:2、後者大:0

D-5.continue敘述

continue語法：略過下方語法，直接執行下一輪迴圈



#印出區間所有7的倍數

```
m, n = int(input()), int(input())
```

```
if (m > n):
```

```
    m, n = n, m
```

```
for i in range(m, n+1):
```

```
    if (i%7 != 0):
```

```
        continue
```

```
    print(i, end=' ')]
```

英



50

150

56 63 70 77 84 91 98 105 112 119 126 133 140 147

D-6. 巢狀迴圈範例



#印出九九乘法表

```
for i in range(1,10):  
    for j in range(1,10):  
        print(' %d*%d=%2d' %(i,j,i*j), end='  ' )  
    print() #換行
```



1*1= 1	1*2= 2	1*3= 3	1*4= 4	1*5= 5	1*6= 6	1*7= 7	1*8= 8	1*9= 9
2*1= 2	2*2= 4	2*3= 6	2*4= 8	2*5=10	2*6=12	2*7=14	2*8=16	2*9=18
3*1= 3	3*2= 6	3*3= 9	3*4=12	3*5=15	3*6=18	3*7=21	3*8=24	3*9=27
4*1= 4	4*2= 8	4*3=12	4*4=16	4*5=20	4*6=24	4*7=28	4*8=32	4*9=36
5*1= 5	5*2=10	5*3=15	5*4=20	5*5=25	5*6=30	5*7=35	5*8=40	5*9=45
6*1= 6	6*2=12	6*3=18	6*4=24	6*5=30	6*6=36	6*7=42	6*8=48	6*9=54
7*1= 7	7*2=14	7*3=21	7*4=28	7*5=35	7*6=42	7*7=49	7*8=56	7*9=63
8*1= 8	8*2=16	8*3=24	8*4=32	8*5=40	8*6=48	8*7=56	8*8=64	8*9=72
9*1= 9	9*2=18	9*3=27	9*4=36	9*5=45	9*6=54	9*7=63	9*8=72	9*9=81

條件判斷與迴圈練習

SecurityFocusOnline [Python101]

1. FOR your summation
2. IF 潤年
3. IF幸福企業的獎金制度 -> 使用round()函數

語法練習

語法練習1：輸入輸出

語法練習2：條件判斷

語法練習3：迴圈

程式練習：zerojudge平台(採google登入)

1. d483: hello, world：基本輸出
2. a001.哈囉：基本輸入輸出
3. d827.買鉛筆：基本輸入、運算子
4. d064: < | 數?：條件判斷
5. a058.MOD3：for + range、if
6. a009: 解碼器：for + string
7. a147: Print it all：for + range、while、break、if
8. a149: 乘乘樂：for + range、for + string



列表(list)



E-1-1. 列表(list)

- 資料格式：以 `[]` 將不同型態的資料含括起來，以 `,` 分隔
- 列表中的資料(稱之為元素)是有序排列，從0開始編號
- 格式：表列名稱 = `[元素0, 元素1, ...]`
- 範例：

取用列表元素	替換列表中元素	宣告空的列表
<pre>data = ['John', [95, 118], 'May', 100] print(data[1]) ⇒ [95, 118]</pre>	<pre>data[1] = 20 print(data) ⇒ ['John', 20, 'May', 100]</pre>	<pre>a = []</pre>

E-1-2. 列表的索引值

- 範例: num = [11, 22, 33, 44, 55, 66, 77, 88, 99, 10]
- 索引值(index)編號如下

index 左至右	0	1	2	3	4	5	6	7	8	9
列表	11	22	33	44	55	66	77	88	99	10
index 右至左	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

E-2-1. 列表list運算子: +, *

- +: list 串接
- *: list 重複
- 範例:

```
>>> a, b = [1, 2, 3], [10, 20, 30]
>>> a += b
>>> print(a)
[1, 2, 3, 10, 20, 30]
```

```
>>> c = ['A', 'B', 'C']
>>> c *= 2
>>> print(c)
['A', 'B', 'C', 'A', 'B', 'C']
```

E-2-2. 列表list運算子: []

- []: 截取 list 部份元素
- 用法1 `[index]`: 取用 *index* 所在元素
- 用法2 `[start : stop : step]`: 截取索引值為 *start* 到 *stop-1* 的元素, 並依 *step* 值跳躍取用 (*step* 值-1 代表反轉取用)

- 範例:

```
>>> a = [0, 1, 2, 3, 4, 5]
>>> print(a[0])
0
>>> print(a[1:3])
[1, 2]
```

```
>>> print(a[:4])
[0, 1, 2, 3]
>>> print(a[2:])
[2, 3, 4, 5]
>>> print(a[:4:2])
[0, 2]
```

E-3. 列表(list)常用內建函數

a = ['John', [95, 118], 'May', 100]

b = [89, 119, 55]

通用函數	描述	範例	結果
len(<list>)	計算表列元素個數	len(a)	4
list(<str>)	將<str>轉成列表(list)	list('TNFSH')	['T','N','F','S','H']
sum(<list>)	將列表中所有元素加總 註: 僅限列表元素皆為數字	sum(b)	263
max(<list>) min(<list>)	找出列表中最大(小)值 若為字串, 則依字典序排列	max(b) min(b)	119 55
sorted(<list>)	回傳依遞增排序的列表	c = sorted(b) print(c)	[55, 89, 119]

E-4. 列表(list)專用物件方法

a = ['John', [95, 118], 'May', 100]

b = [89, 119, 55]

列表物件方法	描述	範例	結果
<code><list>.clear()</code>	清除列表中所有元素	<code>a.clear()</code>	<code>[]</code>
<code><list>.append(<obj>)</code>	將<obj>加到<list>尾端	<code>b.append(35)</code>	<code>[89, 119, 55, 35]</code>
<code><list>.sort()</code>	將列表中元素遞增排序 (直接於原列表操作)	<code>b.sort()</code> <code>print(b)</code>	<code>[55, 89, 119]</code>
<code><list>.reverse()</code>	反轉列表元素	<code>b.reverse()</code> <code>print(b)</code>	<code>[55, 119, 89]</code>
<code><list>.extend(<list1>)</code>	將<list1>合併至<list>尾端	<code>a.extend(b)</code>	<code>['John', [95, 118], 'May', 100, 89, 119, 55]</code>

E-4-1. 列表(list)常用物件方法append()

- 功用: 將物件加至列表(list)尾端
- 語法: `list.append(x)`
- 參數說明: x 為欲加至列表尾端的物件
- 範例:

```
>>> a = [0, 1, 2, 3, 4, 5]
>>> a.append(100)
>>> print(a)
[0, 1, 2, 3, 4, 5, 100]
```

E-4-2. 列表(list)常用物件方法clear()

- 功用: 將列表(list)清空

- 語法: `list.clear()`

- 參數說明:

- 範例:

```
>>> a = [0, 1, 2, 3, 4, 5]
>>> a.clear()
>>> print(a)
[]
```

E-4-3. 列表(list)常用物件方法copy()

- 功用: 複製列表(list) *注意與=的差異
- 語法: list.copy()
- 範例:

```
>>> a = [0, 1, 2, 3, 4, 5]
>>> b = a
>>> a[0] = -1
>>> print(a, b)
[-1, 1, 2, 3, 4, 5] [-1, 1, 2, 3, 4, 5]
```

b = a,
則a, b兩變數記憶體位置相同
a 做任何改變, b 也會變

```
>>> a = [0, 1, 2, 3, 4, 5]
>>> c = a.copy()
>>> a[0] = -1
>>> print(a, c)
[-1, 1, 2, 3, 4, 5] [0, 1, 2, 3, 4, 5]
```

c = a.copy(),
則a, c兩變數記憶體位置不同
a 做改變, c 不影響

E-4-4. 列表(list)常用物件方法extend()

- 功用: 在列表(list)末尾添加另一個序列
- 語法: `list.extend(seq)`
- 參數說明: *seq* 為添加的序列, 可以為list、tuple、set

- 範例:

```
>>> a, b = [0, 1, 2, 3, 4, 5], ['AAA', 'BBB']
>>> a.extend(b)
>>> print(a)
[0, 1, 2, 3, 4, 5, 'AAA', 'BBB']
```

E-4-5. 列表(list)常用物件方法reverse()

- 功用: 反轉列表(list)元素
- 語法: `list.reverse()`
- 參數說明:
- 範例:

```
>>> a = [0, 1, 2, 3, 4, 5]
>>> a.reverse()
>>> print(a)
[5, 4, 3, 2, 1, 0]
```

E-4-6. 列表(list)常用物件方法sort()

- 功用: 將列表(list)元素排序
- 語法: `list.sort(key=None, reverse=False)`
- 參數說明: *reverse* 為排序規則(*False*為遞增、*True*為遞減)

- 範例:

```
>>> a = [7, 2, 5, 3, 8]
>>> a.sort()
>>> print(a)
[2, 3, 5, 7, 8]
```

```
>>> a = [7, 2, 5, 3, 8]
>>> a.sort(reverse=True)
>>> print(a)
[8, 7, 5, 3, 2]
```

E-5. 列表建構 list comprehension

- 建構 list 的方法, 語法分成三部分
- **語法** [*expression* **for** *item* **in** *list* (**if** *condition*)]
- **說明** *expression* 可為函數或運算式、*condition* 為條件式(可省略)
- 範例:

```
>>> numbers = []
>>> for x in range(10):
...     numbers.append(x * 3)
...
...
>>> print(numbers)
[0, 3, 6, 9, 12, 15, 18, 21, 24, 27]
```

```
>>> numbers = [x*3 for x in range(10)]
>>> print(numbers)
[0, 3, 6, 9, 12, 15, 18, 21, 24, 27]
```

E-6. 列表(list)範例



#輸入成績直到-1為止，並將成績由高到低排序輸出

```
score = []  
while(True):  
    stu = int(input())  
    if (stu != -1): score.append(stu)  
    else: break  
score.sort(reverse=True)  
print(score)
```



78

9

66

45

-1

[78, 66, 45, 9]

列表練習

SecurityFocusOnline [Python101]

1. 黃金比例與費氏數列



內建函數與物件方法



F-1.內建函數與物件方法差別

- 內建函數：`print()`、`input()`

一段「被命名」的程式碼，這段程式碼可以用於執行某個特定任務

- 物件：`str`、`list`

Python為物件導向程式語言，物件包含資料型態、類別、模組

- 物件方法：`str.split()`、`list.append()`

附屬於物件之下的函數，需以物件變數後接，再接函數名稱

F-2. 常用內建函數(Built-in Functions)

<code>abs()</code>	<code>chr()</code>	<code>eval()</code>	<code>float()</code>
<code>hex()</code>	<code>input()</code>	<code>int()</code>	<code>len()</code>
<code>list()</code>	<code>map()</code>	<code>max()</code> 、 <code>min()</code>	<code>ord()</code>
<code>pow()</code>	<code>print()</code>	<code>range()</code>	<code>sorted()</code>
<code>str()</code>	<code>sum()</code>	<code>type()</code>	

F-2-1.abs()

- 功用：回傳絕對值
- 語法：abs(x)
- 參數說明： x 為整數
- 範例：

```
>>> abs(-5)
5
>>> abs(66)
66
```

F-2-2.chr()

- 功用: ASCII值轉字元
- 語法: `abs(i)`
- 參數說明: *i* 為ASCII值
- 備註: 與chr()函數相反者為ord()
- 範例:

```
>>> chr(97)
'a'
>>> chr(65)
'A'
```

二進位	十進位	十六進位	圖形	二進位	十進位	十六進位	圖形
0100 0000	64	40	@	0110 0000	96	60	^
0100 0001	65	41	A	0110 0001	97	61	a
0100 0010	66	42	B	0110 0010	98	62	b
0100 0011	67	43	C	0110 0011	99	63	c
0100 0100	68	44	D	0110 0100	100	64	d
0100 0101	69	45	E	0110 0101	101	65	e

F-2-3.eval()

- 功用：執行字串運算式，並回傳結果
- 語法：`eval(expression)`
- 參數說明：`expression` 為字串運算式
- 範例：

```
>>> x = 7
>>> eval('3*x')
21
>>> eval('pow(x, 2)')
49
```

F-2-4.float()

- 功用：將整數或字串轉成浮點數資料型態
- 語法：float(*x*)
- 參數說明：*x* 為整數或字串
- 範例：

```
>>> float(100)
100.0
>>> float('-123.45\n')
-123.45
>>> float('1e-3')
0.001
```

F-2-5.hex()

- 功用：將10進制整數轉成16進制表達字串(含前綴0x)
- 語法：`hex(i)`
- 參數說明：*i* 為10進制整數
- 備註：`bin()`函數為轉成2進制表達字串(含前綴0b)
- 範例：

```
>>> hex(345)
'0x159'
```

F-2-6.input()

- 功用: 以字串格式讀入整行資料
- 語法: `input([prompt])`
- 參數說明: *prompt* 為輸入提示字, 可省略
- 範例:

```
>>> s = input('Please input your name ---> ')
Please input your name ---> Elon Mask
>>> s
'Elon Mask'
```

F-2-7.int()

- 功用：將不同進制表達字串轉成10進制整數
- 語法：`int(x, base=10)`
- 參數說明：*x* 為字串, *base* 為進制(預設為10進制)
- 範例：

```
>>> int('12345')
12345
>>> int('10101', 2)
21
>>> int('FF', 16)
255
```


F-2-8.len()

- 功用：回傳物件的長度(元素個數)
- 語法：len(*s*)
- 參數說明：*s* 為序列，如：string, list, range, dict

- 範例：

```
>>> len('Taiwan can help')
15
>>> len([1, 2, 3, 4, 5, 6])
6
```

F-2-9.list()

- 功用：將字串、元組等可迭代序列轉換成列表(list)
- 語法：`list([iterable])`
- 參數說明：*iterable* 為可迭代序列，如：string, range, tuple

- 範例：

```
>>> a = list()
>>> print(a)
[]
>>> print(list('Taiwan can help'))
['T', 'a', 'i', 'w', 'a', 'n', ' ', 'c', 'a', 'n', ' ', 'h', 'e', 'l', 'p']
>>> print(list(range(7)))
[0, 1, 2, 3, 4, 5, 6]
```

F-2-10.map()

- 功用：將可迭代序列丟入函數運算，回傳一個迭代器
- 語法：`map(function, iterable, ...)`
- 參數說明：*function* 為函數名，*iterable* 為可迭代序列
- 範例：

```
>>> print(map(int, ['20', '30', '40']))  
<map object at 0x0000018ABB35E590>  
>>> a, b, c = map(int, ['20', '30', '40'])  
>>> print(a, b, c)  
20 30 40
```

F-2-11.max()

- 功用: 回傳最大值
- 語法: `max(arg1, arg2, ...)`、`max([iterable])`
- 參數說明: *iterable* 為可迭代序列

- 範例:

```
>>> max(50, -30, 80, -100)
80
```

```
>>> max("Taiwan can help")
'w'
```

```
>>> max([10, 20, 30, 40, 50])
50
```

```
>>> max([[1,0], [1,1], [2,1], [2, 0]])
[2, 1]
```

F-2-11.min()

- 功用: 回傳最小值
- 語法: `min(arg1, arg2, ...)`、`min([iterable])`
- 參數說明: *iterable* 為可迭代序列
- 範例:

```
>>> min(50, -30, 80, -100)
-100
>>> min("Taiwan can help")
' '
>>> min([10, 20, 30, 40, 50])
10
>>> min([[1,0], [1,1], [2,1], [2, 0]])
[1, 0]
```

F-2-12.ord()

- 功用：字元轉ASCII值
- 語法：ord(*c*)
- 參數說明：*c* 為字元
- 備註：與ord()函數相反者為chr()
- 範例：

```
>>> ord('A')
65
>>> ord('z')
122
```

二進位	十進位	十六進位	圖形	二進位	十進位	十六進位	圖形
0100 0000	64	40	@	0110 0000	96	60	^
0100 0001	65	41	A	0110 0001	97	61	a
0100 0010	66	42	B	0110 0010	98	62	b
0100 0011	67	43	C	0110 0011	99	63	c
0100 0100	68	44	D	0110 0100	100	64	d
0100 0101	69	45	E	0110 0101	101	65	e

F-2-13.pow()

- 功用: 回傳指數值
- 語法: $\text{pow}(\text{base}, \text{exp}[, \text{mod}]) \Rightarrow \text{base}^{\text{exp}} \% \text{mod}$
- 參數說明: *base* 為底數、*exp* 為指數、*mod* 為模數
- 範例:

```
>>> pow(10, 5)
100000
>>> pow(7, 5, 11)
10
```

F-2-14.print()

- 功用:輸出
- 語法: `print(objects, sep=' ', end='\n')`
- 參數說明: *objects* 為python物件、*sep* 為分隔字元(預設空白)、*end* 為結束字元(預設換行)

- 範例:

```
>>> a, b, c = 50, 'Taiwan', [1, 2, 'AA']
>>> print(a, b, c)
50 Taiwan [1, 2, 'AA']
```


F-2-15.range()

- 功用：回傳一個計數範圍的迭代類別
- 語法：`range(stop)`、`range(start, stop[, step])`
- 參數說明：`start` 為計數起始值(預設為0)、`stop` 為計數終止值(不含`stop`)、`step` 為遞增值(預設為1)

- 範例：

```
>>> list(range(5))  
[0, 1, 2, 3, 4]  
>>> list(range(1, 10, 2))  
[1, 3, 5, 7, 9]
```

F-2-16.sorted()

- 功用：回傳一個排序好的列表
- 語法：`sorted(iterable, key=None, reverse=False)`
- 參數說明：*iterable* 為可迭代序列、*reverse* 為排序規則(*False*為遞增、*True*為遞減)

● 範例：

```
>>> a = [5, 2, 3, 1, 4]
>>> b = sorted(a, reverse=True)
>>> print(b)
[5, 4, 3, 2, 1]
```

F-2-17.str()

- 功用：將資料轉成字串
- 語法：`str(object)`
- 參數說明：*objects* 為python物件

- 範例：

```
>>> a, b = 13579, [1, 3, 5, 7, 9]
>>> print(str(a), str(b))
13579 [1, 3, 5, 7, 9]
```

F-2-18.sum()

- 功用：對可迭代的序列(必須為整數資料)求總和
- 語法：`sum(iterable[, start=0])`
- 參數說明：*iterable* 為可迭代序列、*start* 為總和起始值(預設0)
- 範例：

```
>>> sum([1, 2, 3, 4, 5])  
15  
>>> sum([1, 2, 3, 4, 5], 100)  
115
```

F-2-19.type()

- 功用: 回傳物件類型
- 語法: `type(object)`
- 參數說明: *objects* 為python物件

- 範例:

```
>>> type(12345)
<class 'int'>
>>> type('12345')
<class 'str'>
>>> type([1, 2, 3, 4, 5])
<class 'list'>
```

內建函數練習

SecurityFocusOnline [Python101]

1. 16進位制的計算
2. 8進位的閱讀



常用輸入技巧



G-1.輸入格式:每行只有一個數字

- 因input()讀入的資料型態為字串, 無法算術運算, 需搭配int()
- 語法: `a = int(input())`
- 範例: 輸入有兩行, 每行一個數字, 請輸出這兩個數字和。
- 輸入範例: 60
40
- 輸出範例: 100


```
a, b = int(input()), int(input())  
print(a+b)
```


G-2.輸入格式：每行有多個數字

- 語法： $a, b = \text{map}(\text{int}, \text{input}().\text{split}())$
- 範例：輸入只有1行，有3個數字a, b, c，請輸出 $(a*b)\%c$ 。
- 輸入範例：11 23 19
- 輸出範例：6

```
a, b, c = map(int, input().split())  
print((a*b) % c)
```

G-3. 輸入格式：每行有多個數字

- 語法： $a, b = [\text{int}(x) \text{ for } x \text{ in } \text{input().split()}]$  list comprehension
- 範例：輸入只有1行，有3個數字a, b, c，請輸出 $(a*b)\%c$ 。
- 輸入範例：11 23 19
- 輸出範例：6

```
a, b, c = [int(x) for x in input().split()]  
print((a*b) % c)
```

G-4. 輸入格式：多個數字存成一維陣列

- 語法：`a = [int(x) for x in input().split()]`

list comprehension

- 範例：輸入只有1行，有n個數字代表成績，請輸出最高分。
- 輸入範例：89 93 69 77 56 97 66 71 91
- 輸出範例：97

```
a = [int(x) for x in input().split()]  
print(max(a))
```

G-5. 輸入格式：輸入到EOF結束

```
while True:  
    try:
```

正常程式區段

```
except EOFError:  
    break
```

範例：輸入a,b兩數，輸出兩數和

```
while True:
```

```
    try:
```

```
        a, b = map(int, input().split())
```

```
        print(a+b)
```

```
    except EOFError:
```

```
        break
```

程式練習：zerojudge平台(採google登入)


1. a003.兩光法師占卜術：輸入(一行2個整數)、if
2. a005: Eva 的回家作業：輸入(一行4個整數)、for + range、if
3. a004.文文的求婚：輸入(到EOF)、if
4. e189.三的倍數：輸入(到EOF)、if
5. a024: 最大公因數(GCD)：輸入(一行2個整數)、while



字符串(str)



H-1-1. 列表list與字串str比較

比較項目	列表list	字串str
範例	a = ['AB', 55, [0,1], 7]	b = 'I love Python'
含括各元素的符號	[]	' ' 或 " "
元素的資料型態	不限資料型態	字元
各元素是否有序排列	是 (索引值從0開始)	是 (索引值從0開始)
修改元素內容	可 (a[2] = 99)	不可 (b[2]  'L')

H-1-2. 字串的索引值

- 範例: tel = '06-2371206'
- 索引值(index)編號如下

index 左至右	0	1	2	3	4	5	6	7	8	9
字串	0	6	-	2	3	7	1	2	0	6
index 右至左	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

H-2-1. 字串str運算子: +, *

- +: str 串接
- *: str 重複
- 範例:

```
>>> a = 'Happy'
>>> b = a + ' ' + 'Python'
>>> print(b)
Happy Python
```

```
>>> c = 'ABC' * 3
>>> print(c)
ABCABCABC
```

H-2-2. 字串str運算子: []

- []: 截取 str 部份字元
- 用法1 `[index]`: 取用 *index* 所在字元
- 用法2 `[start : stop : step]`: 截取索引值為 $start$ 到 $stop-1$ 的字元, 並依 $step$ 值跳躍取用($step$ 值-1代表反轉取用)

- 範例:

```
>>> a = 'I love Python'
>>> print(a[3])
o
>>> print(a[2:6])
love
```

```
>>> print(a[:5])
I lov
>>> print(a[:5:2])
Ilv
>>> print(a[:5:-1])
nohtyP
```

H-3. 字串(str)專用物件方法

name = 'TNFSH 109'

school = 'TNFSH'

通用函數 / 物件方法	描述	範例	結果
<code>len(<str>)</code>	計算字串長度	<code>len(name)</code>	9
<code><str>.lower()</code> <code><str>.upper()</code>	字串轉小寫 字串轉大寫	<code>school.lower()</code>	'tnfsh'
<code><str>.islower()</code> <code><str>.isupper()</code>	字串裡英文是否全小寫 字串裡英文是否全大寫	<code>school.isupper()</code>	True
<code><str>.replace(<str1>, <str2>)</code>	將<str>中的<str1>以<str2>取代	<code>name.replace('FSH', 'GS')</code>	'TNGS 109'
<code><str>.split([sep])</code>	字串以sep分割, 分割完回傳列表 註: sep預設值為空白	<code>name.split()</code>	['TNFSH', '109']

H-3-1. 字串(str)常用物件方法split()

- 功用: 對字串分割, 回傳分割後的列表list
- 語法: `str.split(sep=' ')`
- 參數說明: *sep* 為分割字元(可省略, 預設為空白)

- 範例:

```
>>> a, b = '10 20 30 40', 'AA,BB,CC,DD'
>>> print(a.split())
['10', '20', '30', '40']
>>> print(b.split(','))
['AA', 'BB', 'CC', 'DD']
```

字串練習

SecurityFocusOnline

1. [Python101] 字串的逆向工程
2. [Crypto102] CRY11_PythonCrypto
3. [Crypto102] CRY14_加密的奧義

程式練習：zerojudge平台(採google登入)

語法練習：列表與字串

1. a022: 迴文：string、for(亦可不用)
2. a038: 數字翻轉：string、int()
3. a148: You Cannot Pass?!：list、輸入(到EOF)
4. a104: 排序：list、sort()、輸入(到EOF)



解題範例



I-1.2021.11_P1-修補圍籬

- <https://zerojudge.tw/ShowProblem?problemid=g595>

題目

有一個農場有寬度為 n 的圍籬，每個圍籬都有各自的高度 $h[1], h[2], \dots, h[n]$

有些圍籬被吹斷了(以高度0代表)，農場主人想修補這些圍籬，但他忘記這些壞掉的圍籬原本高度是多少，為了減少成本，他會取斷掉的圍籬位置相鄰左邊和右邊較小的那個高度補上去，問需要多少成本

題目保證不會有兩個相鄰的吹斷圍籬，而穿斷的圍籬有可能位在邊界

輸入說明

輸入包含兩行

第一行有一個正整數 n

第二行有 n 個以空格分隔的整數 $h[1], h[2], \dots, h[n]$

輸出說明

輸出一個正整數表示新增的圍籬高度總和

範例輸入 #1

3

2 0 4

範例輸出 #1

2

範例輸入 #2

9

0 5 3 0 6 4 0 1 0

範例輸出 #2

10

I-1.2021.11_P1-修補圍籬(題解1)

1. 將 n 和 $h[1], h[2], \dots, h[n]$ 讀入
2. 走訪每個 $h[i]$, 若 $h[i]$ 為0, 先判斷其是否為邊界(特判), 若非邊界, 則取相鄰左右的較小值

```
n = int(input())
h = [int(x) for x in input().split()]
ans = 0
for i in range(n):
    if h[i] == 0:
        if i == 0: ans += h[1]
        elif i == n-1: ans += h[n-2]
        else: ans += min(h[i-1], h[i+1])
print(ans)
```

I-1.2021.11_P1-修補圍籬(題解2)

1. 將 n 和 $h[1], h[2], \dots, h[n]$ 讀入, 並補上 $h[0]$ 和 $h[n+1]$ 兩個較大值代表邊界
2. 走訪 $h[1] \sim h[n]$, 若 $h[i]$ 為 0, 則取相鄰左右的較小值

```
n = int(input())
h = [10000]+[int(x) for x in input().split()]+[10000]
ans = 0
for i in range(1, n+1):
    if h[i] == 0:
        ans += min(h[i-1], h[i+1])
print(ans)
```

I-2.2022.01_P1 - 程式交易

<https://zerojudge.tw/ShowProblem?problemid=h081>

題目

小明最近想要用程式做股票交易，給一個股票的歷史價格 $a[1], a[2], \dots, a[n]$ 他的投資策略如下

1. 同一個時間最多只會持有一張股票，並會在時間點 1 花 $a[1]$ 買進。
2. 若當下持有股票且該股票買進價格為 x ，當遇到價格 $y \geq x + D$ 時即賣出，並轉得利潤 $y - x$ 。
3. 若當下沒有持有股票且上一次的賣出價格為 x ，當遇到價格 $y \leq x - D$ 時則會買進股票。

輸出依照上述規則買賣後所得到的利潤和，若交易結束仍持有股票，則不考慮該股票買進的成本，直接無視該股票即可。

輸入說明

第一行輸入兩個正整數 n, D ，接下來有 n 個正整數，代表每個時間點股票的價格。

輸出說明

輸出一個正整數，代表總利潤。

範例輸入 #1

3 10

50 20 45

範例輸出 #1

0

範例輸入 #2

6 10

30 20 45 38 10 20

範例輸出 #2

25

I-2.2022.01_P1-程式交易(題解)

1. 讀入 n 、 D 和每個時間點股票價格(存入stock列表)
2. 設定三個變數: ans代表總利潤(初始0)、price代表持有或賣出價格(初始stock[0])、hold代表當下是否持有股票(初始1, 代表持有股票)
3. 走訪 stock[1] ~ stock[n-1],
 - 3.1. 若hold為1且 $\text{stock}[i] \geq \text{price} + D$, 則賣出股票, 計算利潤, hold設為0(代表未持有)、price設為stock[i](記錄賣出價格)
 - 3.2. 若hold為0且 $\text{stock}[i] \leq \text{price} - D$, 則買進股票, hold設為1(代表持有)、price設為stock[i](記錄買入價格)

I-2.2022.01_P1 - 程式交易(程式碼)

```
n, D = map(int, input().split())
stock = [int(x) for x in input().split()]
ans, price, hold = 0, stock[0], 1
for i in range(1, n):
    if hold == 1 and stock[i] >= price + D:
        ans += stock[i] - price
        hold, price = 0, stock[i]
    elif hold == 0 and stock[i] <= price - D:
        hold, price = 1, stock[i]

print(ans)
```

APCS實作題第1題題單

1. 2021.09_P1-七言對聯

<https://zerojudge.tw/ShowProblem?problemid=g275>

2. 2021.01_P1-購買力

<https://zerojudge.tw/ShowProblem?problemid=f605>

3. 2020.10_P1-人力分配

<https://zerojudge.tw/ShowProblem?problemid=f312>

4. 2020.07_P1-購物車

<https://zerojudge.tw/ShowProblem?problemid=f579>

Happy Python Day下午場

Python 在資訊安全的應用

J.函數、模組、套件

K.自定義函數

L.CTF - PPC

M.Python進行nc連線

課程相關資訊

- Python語法 : Python 3.X
- 撰寫環境 : Linux + gedit + python3編譯
- 解題平台 : SCIST CTF (<https://misc.ctf.scist.org/>)

PPC

Welcome 500	Count 500	Nth 500	Guess 500
Calculator 500	Alphabet 500	Digit 500	PI 500
Hanoi 500			

資安倫理宣傳

本課程目的在提升學員對資訊安全之認識及資安實務能力，深刻體認到資安的重要性！所有課程學習內容不得從事非法攻擊或違法行為，**所有非法行為將受法律規範**，提醒學員不要以身試險。



函數、模組、套件

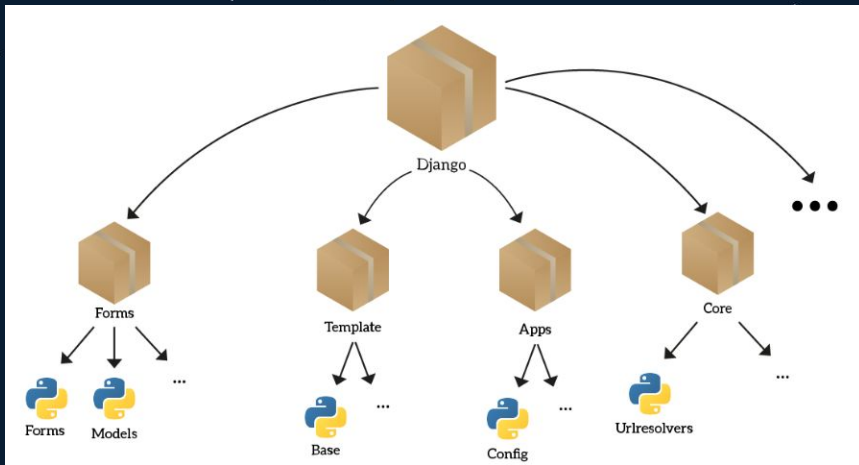


J-1. 函數、模組、套件的差別

函數(function): 一段程式碼

模組(module): 一個 python 檔案, 包含 class、variable、function

套件(package): 一個目錄, 包含 subpackage、module



J-2. 函數(function)

- 函數為一段程式碼程式碼的集合，可視為一個獨立的程式區段
- 函數可重覆呼叫使用，為結構化程式語言的重要元素
- python 函數可區分為三大類：
 - 內建函數: print()、input()、int()、len()、str()...
 - 第三方公司所開發的模組module
 - 自定義函數

J-3. 模組(module)

- 模組為一個python程式檔案, 包含了變數、類別、函數等
- 範例:random模組
- 匯入模組語法: import 模組名稱
 - `import random` # 匯入random模組
- 使用模組內函數語法: 模組名稱.函數名稱
 - `random.randint(1, 100)` # 產生一個1-100間的隨機數

J-4. 套件(package)

- 套件為一個python程式資料夾, 包含了多個python檔案(模組)
- 範例: pwnlib 套件
- 匯入套件語法: import 套件名稱. 模組名稱
 - `import pwnlib.tubes.ssh` # 匯入ssh模組

J-5.import 語法

1. import [套件.]模組

```
import random  
random.randint(1, 100)
```

2. from 模組/套件 import 函數/模組

```
from random import randint  
randint(1, 100)
```

3. import 模組 as 別名

```
import random as rd  
rd.randint(1, 100)
```

4. from 模組/套件 import * (較不建議)

```
from random import *  
randint(1, 100)
```



函數、模組、套件



K-1. 自定義函數宣告

★ 宣告語法:

```
def 函數名稱 ( [參數1, 參數2, ...] ):
    程式區塊
    [ return 值 ]
```

★ 說明:

- 參數可省略, 亦即呼叫function不需傳入任何資料(引數)。
- 若無回傳值, 則無需return。若多個回傳值可用逗號隔開。

K-2. 呼叫自定義函數

★ 呼叫函數語法： [變數 =] 函數名稱([引數1, 引數2, ...])

```
import random

def isPrime(x):
    flag = True
    if x%2 == 0: flag = False
    else:
        for j in range(3, int(x**0.5)+1, 2):
            if x%j == 0: flag = False; break
    return flag

n = random.randint(1, 1<<30)
ans = isPrime(n)
print(f'{n} is prime? {ans}')
```

自定義函數練習

SecurityFocusOnline [Python101]

1. 函數的虛擬碼實作



CTF - PPC



L-1. 解謎式CTF - PPC

- PPC 全名 Professionally Program Coder
- 部份CTF初賽會新增 PPC 類型題目
- 主要以程式解題，與一般程式設計競賽不同
 - **程式競賽**: 程式碼送至評分主機，評分主機會將測試資料輸入給程式，比對程式執行結果與答案是否相同
 - **CTF競賽**: 透過程式與伺服器連線，以即時互動方式給出答案，若答對，則伺服器端會給出 flag

L-2.PPC 題目範例

Challenge


0 Solves

×

Count
500

Author : Curious

`nc lab.scist.org 33342`

 server.py


Flag

Submit

題目會給定一個連線指令

L-3.PPC 題目解題步驟

1. 先於 Shell 輸入連線指令看看回傳什麼訊息
2. 再依回傳訊息撰寫程式連線

```
(kali@kali)-[~]  
$ nc lab.scist.org 33342  
===== Welcome To Counting Game =====  
You just need to count from 1 to 100 !  
----- wave 1/100 -----  
> 1  
----- wave 2/100 -----  
> 2  
----- wave 3/100 -----  
> 3  
----- wave 4/100 -----  
> 
```

連線後，

伺服主機回傳訊息



M



python 進行 nc 連線



M-1.pwntools 套件

- pwntools是一个CTF框架和漏洞利用開發套件
- 用Python開發
- 旨在讓使用者簡單快速的編寫exploit (漏洞利用)

M-2.匯入 pwntools 套件

依據官方說明文件

```
from pwn import *
```

M-3.pwntools 進行 nc 連線

格式：

```
連線名稱 = remote('主機位址', port)
```

範例：`r = remote('120.114.62.214', 2403)`

M-4-1.remote連線後接收資料

- `recv(N)`: 接受 N 個字元
- **`recvline()`**: 接收一行資料
- **`recvlines(N)`**: 接收 N 行資料
- **`recvuntil(some_string)`**: 接收到 `some_string` 為止

M-4-2.remote連線後接收資料(範例)

```
from pwn import *      # 匯入pwntools套件  
  
r = remote('120.114.62.214', 2403)    # 進行 nc 連線  
  
r.recvline()           # 從連接端接收一行資料  
  
r.recvlines(7)         # 從連接端接收七行資料  
  
r.recvuntil(b'answer: ') # 從連接端接收資料; 直到 'answer: ' 字串  
  
# 字串 'answer: ' 加上前綴b, 代表此為Bytes字串(python3字串格式)
```

M-5-1.remote連線後傳送資料

- `send(payload)`: 發送payload(需為字串)
- **`sendline(payload)`**: 發送payload, 並換行(末尾\n)
- `sendafter(some_string, payload)`: 接收到 `some_string` 後, 發送 payload
- **`sendlineafter(some_string, payload)`**: 接收到 `some_string` 後, 發送 payload並換行

M-5-2.remote連線後傳送資料(範例)

```
from pwn import *          # 匯入pwntools套件
r = remote('120.114.62.214', 2403) # 進行 nc 連線
for i in range(100):       # 跑 100 次迴圈
    r.recvlines(2)          # 從連接端接收2行資料
    r.sendline(str(i+1).encode()) # 將i+1轉字串傳送至連接端
# encode()函數: 將文字字串轉換成Bytes字串
```

M-6.remote連線取得最後flag方法

1. interactive():與 shell 互動, 輸入 logout 離開連接端 shell

範例: **r.interactive()** # 回到 shell, 可取得 flag

2. 用 recvline() 接收 flag 後再印出

範例: **a = r.recvline()** # 將接收的 flag 存到變數a

print(a.decode()) # 將a轉成字元字串再輸出至螢幕

M-7-1. 題目count - 解題範例

1. 先以 shell 連線，
看看從連接端接收什麼訊息
2. 首先有 2 行題目敘述，
接著有 100 個詢問，
每個詢問於 **>** 後回答
3. 改以 python 程式碼，
透過 remote() 來進行 nc 連線

```
(kali㉿kali)-[~]  
$ nc lab.scist.org 33342  
===== Welcome To Counting Game =====  
You just need to count from 1 to 100 !  
----- wave 1/100 -----  
> 1  
----- wave 2/100 -----  
> 2  
----- wave 3/100 -----  
> 3  
----- wave 4/100 -----  
> █
```

M-7-2. 題目count - 解題範例

1. gedit 編修一個python檔案

```
(ksu@Kali-20212-64)-[~]  
$ gedit count.py
```

2. 存檔後，執行該檔案

```
(ksu@Kali-20212-64)-[~]  
$ python3 count.py
```

3. 執行完畢，
flag 會出現在螢幕上

```
from pwn import *  
r = remote('lab.scist.org', 33342)  
  
r.recvlines(2)  
for i in range(100):  
    r.sendlineafter(b'> ', str(i+1).encode())  
  
r.interactive()  
r.close()
```

M-8.remote()連線注意事項

1. 透過 python 的 remote() 函數連線，傳送接收資料皆為本機程式與連接端之間傳遞，**不會於螢幕上顯示過程**。

2. **若要於螢幕上顯示過程**，可於 remote() 函數加上第三個參數

```
r = remote('120.114.62.214', 2409, level='debug')
```

或者使用 print() 函數，將資料顯示於螢幕上

```
r.sendline(ans.encode())  
print(ans)
```

3. 程式結尾，請將 remote() 連線結束

```
r.close()
```

SCIST CTF 練習

<https://misc.ctf.scist.org/>