



Nguyễn Tiến Đạt   NĐ

Bảng Điều khiển > Các khóa học của tôi > 2324II\_INT3404E\_20 > 6 March - 12 March > [Python] Bài tập Python

## Xử lý ảnh (2324II\_INT3404E\_20) [Python] Bài tập Python

Bắt đầu vào lúc Friday, 15 March 2024, 5:16 PM

Trạng thái Đã xong

Kết thúc lúc Saturday, 30 March 2024, 8:51 PM

Thời gian thực hiện 15 ngày 3 giờ

Điểm 330,00/330,00

Điểm 10,00 trên 10,00 (100%)

## Câu Hỏi 1

Đúng

Python là một ngôn ngữ lập trình thông dịch do Guido van Rossum tạo ra năm 1990. Python hoàn toàn tạo kiểu động và dùng cơ chế cấp phát bộ nhớ tự động; do vậy nó tương tự như Perl, Ruby, Scheme, Smalltalk, và Tcl. Python được phát triển trong một dự án mã mở, do tổ chức phi lợi nhuận Python Software Foundation quản lý.

Theo đánh giá của Eric S. Raymond, Python là ngôn ngữ có hình thức rất sáng sủa, cấu trúc rõ ràng, thuận tiện cho người mới học lập trình. Cấu trúc của Python còn cho phép người sử dụng viết mã lệnh với số lần gõ phím tối thiểu, như nhận định của chính Guido van Rossum trong một bài phỏng vấn ông.

Đầu tiên, chúng ta hãy làm quen với biến và lệnh `print`.

Dòng code dưới đây trong ngôn ngữ Python được dùng để in ra dòng chữ "Hello, world!".

```
print("Hello, World!")
```

Ngoài ra, chúng ta có thể lưu "Hello, World!" vào biến `my_string` như bên dưới và sau đó in ra `stdout`:

```
my_string = "Hello, World!"  
print(my_string)
```

## Bài tập

Bằng một trong hai cách trên, hãy hoàn thành đoạn code dưới đây để in ra dòng chữ "Hello, World!".

### Đầu vào

Không có dữ liệu đầu vào.

### Đầu ra

In dòng chữ "Hello, World!" ra đầu ra chuẩn `stdout`.

**Answer:** (penalty regime: 0 %)

```
1 | print("Hello, World!")
```

	Expected	Got	
✓	Hello, World!	Hello, World!	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 2

Đúng

Cho đầu vào gồm tên (`firstname`) và họ (`lastname`) của một người trên hai dòng khác nhau. Hãy viết chương trình đọc và in ra tên theo định dạng sau:

```
Hello firstname lastname! You just delved into python.
```

### Đầu vào

Đầu vào từ bàn phím gồm hai dòng (độ dài không vượt quá 10):

- Dòng đầu tiên chứa tên (`firstname`)
- Dòng thứ hai chứa họ (`lastname`)

### Đầu ra

In ra kết quả theo yêu cầu.

For example:

Input	Result
Ross Taylor	Hello Ross Taylor! You just delved into python.

Answer: (penalty regime: 0 %)

```
1 | firstname = input()
2 | lastname = input()
3 | print(f"Hello {firstname} {lastname}! You just delved into python.")
```

	Input	Expected	Got	
✓	Ross Taylor	Hello Ross Taylor! You just delved into python.	Hello Ross Taylor! You just delved into python.	✓
✓	Thong Thac	Hello Thong Thac! You just delved into python.	Hello Thong Thac! You just delved into python.	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

### Câu Hỏi 3

Đúng

Cho hai số nguyên nhập vào từ bàn phím, in ra màn hình các kết quả sau:

- 1. Dòng thứ nhất chứa tổng của hai số.
- 2. Dòng thứ hai chứa hiệu của số thứ nhất trừ số thứ 2.
- 3. Dòng thứ ba chứa tích của hai số.

### Đầu vào

Đầu vào từ bàn phím gồm hai dòng:

- Dòng thứ nhất chứa số nguyên đầu tiên,  $a$  ( $1 \leq a \leq 10^{10}$ ).
- Dòng thứ hai chứa số nguyên thứ hai,  $b$  ( $1 \leq b \leq 10^{10}$ ).

### Đầu ra

In ra màn hình 3 dòng chứa kết quả lần lượt là tổng, hiệu và tích của hai số như yêu cầu.

For example:

Input	Result
1	2
1	0
	1

Answer: (penalty regime: 0 %)

```
1 a = input()
2 b = input()
3 a = int(a)
4 b = int(b)
5
6 print(a + b)
7 print(a - b)
8 print(a * b)
```

	Input	Expected	Got	
✓	1 1	2 0 1	2 0 1	✓
✓	100 0	100 100 0	100 100 0	✓
✓	100 200	300 -100 20000	300 -100 20000	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 4

Đúng

Viết chương trình nhập vào hai số nguyên  $a$  và  $b$ , sau đó in ra các kết quả sau trên hai dòng.

- Dòng thứ nhất chứa kết quả của phép chia nguyên  $a//b$ .
- Dòng thứ hai chứa kết quả của phép chia số thực  $a/b$ . Làm tròn đến 3 chữ số sau dấu phẩy.

### Đầu vào

Gồm hai dòng nhập vào từ bàn phím:

- Dòng thứ nhất chứa số nguyên  $a$ .
- Dòng thứ hai chứa số nguyên  $b$ .

### Đầu ra

In ra màn hình các kết quả theo yêu cầu đã nêu ở trên.

For example:

Input	Result
4	1
3	1.333

Answer: (penalty regime: 0 %)

```
1 a = input()
2 b = input()
3 a = int(a)
4 b = int(b)
5
6 print(int(a / b))
7 print(format(float(a) / float(b), ".3f"))
```

	Input	Expected	Got	
✓	4 3	1 1.333	1 1.333	✓
✓	6564424525 323252462	20 20.307	20 20.307	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 5

Đúng

**Xét ví dụ sau:** Một chiếc xe tự lái đang lái trên đường, phía trước chiếc xe là một cột đèn tín hiệu giao thông. Dựa vào trạng thái là màu sắc của đèn tín hiệu, người lập trình hệ thống lái tự động của xe sẽ đưa ra các quyết định khác nhau dựa vào màu của đèn tín hiệu như sau :

```
if ( lights = red ) then:  
    stop the car;      # Nếu đèn màu đỏ thì dừng lại  
elif ( lights = yellow ):  
    slow down; # Nếu đèn màu vàng thi giảm tốc độ  
else:  
    keep moving; # Nếu đèn màu xanh thi điều khiển xe tiếp tục di chuyển.
```

Trong các ngôn ngữ lập trình, cách thức ra hành động như ví dụ trên được thực hiện bằng các khối lệnh rẽ nhánh.

Trong Python, có 3 loại khối lệnh rẽ nhánh như dưới đây:

### 1. Khối lệnh điều kiện IF

```
if bieu_thuc_dieu_kien:  
    hanh_dong_1  
    hanh_dong_2  
    hanh_dong_3
```

**Ví dụ:** Để kiểm tra một số có phải là số chẵn hay không, chúng ta sử dụng đoạn code sau:

```
n = 2  
if n%2 == 0:  
    print("n is an even number")
```

Khi chạy đoạn code trên, dòng chữ "n is an even number" sẽ được in ra màn hình. Tuy nhiên, nếu chúng ta thay n = 3, hàm `print()` trong khối lệnh `if` sẽ không được thực thi và không in ra chuỗi "n is an even number".

### 2. Khối lệnh điều kiện IF-ELSE

```
if bieu_thuc_dieu_kien:  
    hanh_dong_1  
    hanh_dong_2  
    hanh_dong_3  
else  
    hanh_dong_4  
    hanh_dong_5
```

**Ví dụ:**

```
n = 2  
if n%2 == 0:  
    print("n is an even number")  
else:  
    print("n is an odd number")
```

Đoạn code trên sẽ in ra chuỗi "n is an even number". Nếu thay thế `n = 1`, thì chương trình sẽ in ra "n is an odd number".

Ngoài ra, Python cũng cho phép chúng ta tạo ra các khối lệnh điều kiện với nhiều hơn một biểu thức điều kiện với cú pháp như sau:

```
if bieu_thuc_dieu_kien_1:  
    hanh_dong_1  
    hanh_dong_2  
    hanh_dong_3  
elif bieu_thuc_dieu_kien_2  
    hanh_dong_4  
    hanh_dong_5  
elif ...  
...  
...  
else:  
...  
...  
hanh_dong_n
```

### 3. Khối lệnh điều kiện chồng nhau

```

if bieu_thuc_dieu_kien_ngoai:
    if bieu_thuc_dieu_kien_trong:
        hanh_dong_1
    else:
        hanh_dong_2
else:
    hanh_dong_3

```

Ví dụ:

```

var = 100
if var < 200:
    print "Expression value is less than 200"
    if var == 150:
        print "Which is 150"
    elif var == 100:
        print "Which is 100"
    elif var == 50:
        print "Which is 50"
    elif var < 50:
        print "Expression value is less than 50"
    else:
        print "Could not find true expression"
print "Good bye!"

```

Đoạn code trong ví dụ trên sẽ in ra kết quả như sau:

```

Expression value is less than 200
Which is 100
Good bye!

```

## Bài tập

Cho một số nguyên  $n$ , hãy thực hiện các yêu cầu sau với điều kiện tương ứng:

- Nếu  $n$  là lẻ, in ra chữ "Weird"
- Nếu  $n$  chẵn và có giá trị nằm trong khoảng  $[2, 5]$ , in ra "Not Weird"
- Nếu  $n$  chẵn và có giá trị trong khoảng  $[6, 20]$ , in ra "Weird"
- Nếu  $n$  chẵn và lớn hơn 20, in ra "Not Weird"

### Đầu vào

Một dòng duy nhất từ bàn phím chứa số nguyên dương  $n$ . ( $1 \leq n \leq 100$ ).

### Đầu ra

In ra màn hình "Weird" hoặc "Not Weird" dựa vào điều kiện ở đầu bài.

For example:

Input	Result
3	Weird

Answer: (penalty regime: 0 %)

```

1 | n = int(input())
2 |
3 | if n % 2 != 0:
4 |     print("Weird")
5 | elif n in range(2, 6):
6 |     print("Not Weird")
7 | elif n in range(6, 21):
8 |     print("Weird")
9 | else:
10 |     print("Not Weird")

```

	Input	Expected	Got	
✓	3	Weird	Weird	✓
✓	24	Not Weird	Not Weird	✓
✓	4	Not Weird	Not Weird	✓
✓	18	Weird	Weird	✓
✓	29	Weird	Weird	✓
✓	5	Weird	Weird	✓
✓	100	Not Weird	Not Weird	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 6

Đúng

Trong Python, có hai kiểu vòng lặp như sau:

### 1. Vòng lặp for

```
for i in range(0, 5):
    print i
```

### 2. Vòng lặp while

```
i = 0
while i < 5:
    print i
    i += 1
```

Hai đoạn code trên đều in ra các số từ 0 đến 4, mỗi số trên một dòng.

## Bài tập

Viết chương trình nhập vào một số nguyên  $n$ , sau đó với mỗi số nguyên từ 0 đến  $n - 1$ , in ra giá trị  $i^2$ .

### Đầu vào

Đầu vào từ bàn phím gồm một dòng duy nhất chứa số nguyên,  $n$  ( $0 \leq n \leq 20$ ).

### Đầu ra

In ra màn hình  $n$  dòng, dòng thứ  $i$  (bắt đầu từ 0) chứa giá trị  $i^2$ .

For example:

Input	Result
5	0
	1
	4
	9
	16

Answer: (penalty regime: 0 %)

```
1 | n = int(input())
2 |
3 v for i in range(n):
4     print(i**2)
```

	Input	Expected	Got	
✓	5	0 1 4 9 16	0 1 4 9 16	✓

	Input	Expected	Got	
✓	10	0	0	✓
	1	1	1	
	4	4	4	
	9	9	9	
	16	16	16	
	25	25	25	
	36	36	36	
	49	49	49	
	64	64	64	
	81	81	81	

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 7

Đúng

Theo lịch Gregorius - loại lịch tiêu chuẩn hiện nay được dùng trên hầu khắp thế giới thì những năm nào chia hết cho 4 được và không chia hết cho 100 được coi là năm nhuận (ví dụ năm 2100 không phải là năm nhuận, 2104 là năm nhuận). Trong năm nhuận, tháng 2 có 29 ngày thay cho 28 ngày. Cứ 4 năm lại thêm 1 ngày vào lịch bởi vì một năm dương lịch dài khoảng 365 ngày và 6 giờ.

Ngoài ra, vẫn có một số ngoại lệ đối với nguyên tắc này, những năm chia hết cho 100 chỉ được coi là năm nhuận nếu chúng cũng chia hết cho 400. Ví dụ, 1600 và 2000 là các năm nhuận nhưng 1700, 1800 và 1900 không phải là năm nhuận. Tương tự như vậy, 2100, 2200, 2300, 2500, 2600, 2700, 2900 và 3000 không phải là năm nhuận nhưng 2400 và 2800 là các năm nhuận.

Hãy hoàn thành hàm dưới đây để kiểm tra một năm có phải là năm nhuận hay không:

### Đầu vào

Gồm một dòng duy nhất chứa  $y$  là năm cần kiểm tra. ( $1900 \leq y \leq 10^5$ ).

### Đầu ra

Hoàn thành hàm `is_leap(year)` trả về `True/False` thể hiện một năm là năm nhuận hoặc không.

For example:

Input	Result
2000	True

Answer: (penalty regime: 0 %)

Reset answer

```
1 def is_leap(year):
2     leap = False
3
4     # Write your logic here
5     if((year % 4 == 0 and year % 100 != 0) or year % 400 == 0):
6         leap = True
7
8     return leap
9
10 year = int(input())
11 print (is_leap(year))
```

	Input	Expected	Got	
✓	2000	True	True	✓
✓	2100	False	False	✓
✓	2400	True	True	✓
✓	3455	False	False	✓
✓	1990	False	False	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 8

Đúng

Trong Python, **list** là kiểu dữ liệu dùng để lưu một dãy các giá trị.

Để khởi tạo một list rỗng, chúng ta sử dụng các cách sau:

```
arr = list()  
# or simply  
arr = []
```

hoặc để khởi tạo danh sách list với một số phần tử cho trước:

```
arr = [1,2,3]
```

Các phần tử của list có thể được truy cập bằng cách các sau:

```
print arr[0]  
# in ra 1  
print arr[0] + arr[1] + arr[2]  
# in ra 6
```

Khác với nhiều ngôn ngữ khác, danh sách trong Python có thể chứa các phần tử có kiểu khác nhau như xâu (string), số nguyên(integer), số thực (float) hoặc là một danh sách khác (list). Ví dụ, dãy dưới đây chứa cả số nguyên, xâu ký tự và danh sách con.

```
a = [1,2,"hello",[2,3,"hi"]]
```

Kiểu dữ liệu **list** trong Python được xây dựng sẵn với các phương thức đi kèm sau:

### 1. append(x): Thêm phần tử x vào cuối danh sách

```
arr.append(9)  
print arr  
# [1, 2, 3 , 9]
```

### 2. extend(L): Nối một danh sách mới L vào cuối danh sách.

```
arr.extend([10, 11])  
print arr  
# [1, 2, 3, 9, 10, 11]
```

### 3. insert(i,x): Thêm phần tử x vào vị trí thứ i của danh sách

```
arr.insert(3,7)  
print arr  
# [1, 2, 3, 7, ,9 ,10, 11]
```

### 4. remove(x): Xóa phần tử x xuất hiện đầu tiên trong dãy

```
arr.remove(10)  
print arr  
# [1, 2,3, 7, 9, 11]
```

### 5. pop(): Xóa phần tử cuối cùng của dãy

```
temp = arr.pop()  
print temp  
# 11
```

### 6. index(x): Trả về vị trí của phần tử có giá trị là x trong dãy.

```
temp = arr.index(3)  
print temp  
# 2
```

### 7. count(x): Đếm số lần xuất hiện của x trong dãy.

```
temp = arr.count(1)  
print temp  
# 1
```

### 8. sort(): Sắp xếp danh sách theo thứ tự từ bé đến lớn

```
arr.sort()
print arr
# [1, 2, 3, 7, 9]
```

## 9. reverse(): Đảo ngược danh sách

```
arr.reverse()
print arr
# [9, 7, 3, 2, 1]
```

### Bài tập

Cho một danh sách rỗng (`list = []`). Chúng ta có thể thực hiện các loại câu lệnh sau:

- 1. insert i e: Thêm một số nguyên e vào vị trí i.
- 2. print: In ra danh sách
- 3. remove e: Xóa số nguyên e ở vị trí xuất hiện đầu tiên.
- 4. append e: Thêm số nguyên e vào cuối danh sách.
- 5. sort: Sắp xếp danh sách.
- 6. pop: Xóa phần tử cuối cùng ra khỏi danh sách.
- 7. reverse: Đảo ngược danh sách.

Khởi tạo một danh sách rỗng và nhập vào danh sách  $n$  câu lệnh, mỗi câu lệnh thuộc 1 trong 7 dạng ở trên. Hãy duyệt và thực thi các câu lệnh theo thứ tự.

### Đầu vào

Đầu vào từ bàn phím gồm  $n + 1$  dòng:

- Dòng thứ nhất chứa số nguyên  $n$  là số lượng câu lệnh.
- $n$  dòng tiếp theo, mỗi dòng chứa một câu lệnh.

### Đầu ra

Với mỗi câu lệnh dạng `print`, in ra danh sách trên một dòng.

For example:

Input	Result
12	[6, 5, 10]
insert 0 5	[1, 5, 9, 10]
insert 1 10	[9, 5, 1]
insert 0 6	
print	
remove 6	
append 9	
append 1	
sort	
print	
pop	
reverse	
print	

Answer: (penalty regime: 0 %)

```
1 my_list = []
2 no_command = int(input())
3 while no_command!=0:
4     command = input()
5
6     if command.startswith("insert"):
7         _, index, value = command.split(sep=" ")
8         index = int(index)
9         value = int(value)
10        my_list.insert(index, value)
11    elif command.startswith("remove"):
12        _, value = command.split(sep=" ")
13        value = int(value)
14        my_list.remove(value)
15    elif command == "print":
16        print(my_list)
17    elif command.startswith("append"):
18        _, value = command.split(sep=" ")
19        value = int(value)
20        my_list.append(value)
```

	<b>Input</b>	<b>Expected</b>	<b>Got</b>
✓	12 insert 0 5 insert 1 10 insert 0 6 print remove 6 append 9 append 1 sort print pop reverse print	[6, 5, 10] [1, 5, 9, 10] [9, 5, 1]	[6, 5, 10] [1, 5, 9, 10] [9, 5, 1]
✓	29 append 1 append 6 append 10 append 8 append 9 append 2 append 12 append 7 append 3 append 5 insert 8 66 insert 1 30 insert 6 75 insert 4 44 insert 9 67 insert 2 44 insert 9 21 insert 8 87 insert 1 75 insert 1 48 print reverse print sort print append 2 append 5 remove 2 print	[1, 48, 75, 30, 44, 6, 10, 44, 8, 9, 87, 75, 21, 2, 67, 12, 7, 66, 3, 5] [5, 3, 66, 7, 12, 67, 2, 21, 75, 87, 9, 8, 44, 10, 6, 44, 30, 75, 48, 1] [1, 2, 3, 5, 6, 7, 8, 9, 10, 12, 21, 30, 44, 44, 48, 66, 67, 75, 75, 87] [1, 3, 5, 6, 7, 8, 9, 10, 12, 21, 30, 44, 44, 48, 66, 67, 75, 75, 87, 2, 5]	[1, 48, 75, 30, 44, 6, 10, 44, 8, 9, 87, 75, 21, 2, 67, 12, 7, 66, 3, 5] [5, 3, 66, 7, 12, 67, 2, 21, 75, 87, 9, 8, 44, 10, 6, 44, 30, 75, 48, 1] [1, 2, 3, 5, 6, 7, 8, 9, 10, 12, 21, 30, 44, 44, 48, 66, 67, 75, 75, 87] [1, 3, 5, 6, 7, 8, 9, 10, 12, 21, 30, 44, 44, 48, 66, 67, 75, 75, 87, 2, 5]

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 9

Đúng

**Tuple** là cấu trúc dữ liệu tương tự như danh sách (**list**) nhưng các phần tử không thể bị thay đổi sau khi đã khởi tạo. **Tuples** ít được sử dụng trong khi lập trình bởi vì các phép thêm (**add**), xóa (**remove**) hay gán giá trị (**assign**) không tồn tại trên **tuples**. Tuy nhiên, **tuples** có một số lợi thế về thời gian và không gian tính toán.

Để khởi tạo một **tuple**, chúng ta chỉ cần đặt dấu phẩy giữa các phần tử của danh sách và đặt các phần tử vào giữa một cặp ngoặc tròn, nhưng điều này là không bắt buộc. Ví dụ:

```
>>> tup1 = ('physics', 'chemistry', 1997, 2000);
>>> tup2 = (1, 2, 3, 4, 5 );
>>> tup3 = "a", "b", "c", "d";
```

Một trường hợp phổ biến mà tuple thường được sử dụng đó là khi ta hoán đổi hai số như dưới đây:

```
>>> a = 3
>>> b = 2
>>> print a, b
3 2
>>> a, b = b, a
>>> print a, b
2 3
```

Ở trên (a, b) là một **tuple** và được khởi tạo bằng giá trị của b, a.

Một ứng dụng quan trọng khác của **tuple** nữa là nó có thể được sử dụng để làm khóa (**key**) trong từ điển bởi vì **tuple** có thể băm được (**hashable**).

## Bài tập

Cho một dãy  $n$  số nguyên cách nhau bởi khoảng trống. Hãy tạo một tuple  $t$ , chứa  $n$  số nguyên trên. Sau đó tính và in ra kết quả của hàm băm `hash(t)`.

Chú ý: Hàm `hash()` đã mặc định có sẵn trong các chương trình Python, chúng ta có thể dùng luôn mà không cần **import** thêm thư viện.

### Đầu vào

Gồm 2 dòng từ bàn phím:

- Dòng thứ nhất chứa số nguyên  $n$ .
- Dòng thứ hai chứa  $n$  số nguyên cách nhau bởi khoảng trống và các phần tử của tuple  $t$ .

### Đầu ra

In ra kết quả của hàm `hash(t)`.

For example:

Input	Result
2	-3550055125485641917
1 2	

Answer: (penalty regime: 0 %)

```
1 | a = input()
2 | numbers = input().split(" ")
3 | t = tuple(map(int, numbers))
4 | hash_value = hash(t)
5 | print(hash_value)
6 |
```

Input
✓ 2 1 2
✓ 50 387 38 498 988 434 282 467 641 464 682 341 586 222 736 187 415 330 323 109 818 78 469 560 623 748 782 352 398 196 39 603 344 630 841 75

Passed all tests! ✓

**Đúng**

Marks for this submission: 10,00/10,00.



## Câu Hỏi 10

Đúng

Từ điển là một kiểu cấu trúc lưu trữ dữ liệu theo cặp. Mỗi phần tử của từ điển có chứa một cặp khóa (**key**) và giá trị (**value**), chúng ta có thể truy cập **value** sử dụng **key** của nó.

**Ví dụ:** Danh sách dưới đây chứa bộ từ điển Anh-Việt.

```
ev_dict = {'one': 'một', 'two': 'hai', 'three': 'ba'}
```

Nếu muốn biết nghĩa của từ `'one'` trong tiếng Việt, chúng ta sử dụng câu lệnh `ev_dict['one']`.

**Chú ý:** Khóa của từ điển phải là các giá trị cố định - không thay đổi (ví dụ: tuple) sau khi đã khai báo (immutable). Vì vậy, chúng ta không thể sử dụng danh sách (list) để làm khóa bởi vì list cho phép thay đổi phần tử.

```
a_dict['four'] = "bốn" # Thêm khóa "four" với giá trị là "bốn"  
print a_dict['one']  
# Một  
if 'three' in a_dict:  
    # Kiểm tra xem một khóa có tồn tại trong từ điển hay không  
    print a_dict['three']  
else:  
    print "Three not there"  
# prints Three not there  
del a_dict['one']  
# Xóa khóa 'one' trong từ điển  
print a_dict # In ra tất cả khóa và giá trị tương ứng trong từ điển.
```

**Chú ý:** Khóa của từ điển không được lưu theo thứ tự theo thứ tự thêm vào.

## Bài tập

Cho danh sách thông tin về tên và điểm 3 môn học Toán, Vật Lý và Hóa học của  $N$  học sinh, điểm số có thể là số thực. Viết chương trình lần lượt đọc vào tên, các đầu điểm của từng sinh viên và lưu các thông tin đó sử dụng cấu trúc từ điển. Sau đó, nhập vào tên của một sinh viên bất kì trong danh sách, hãy in ra điểm trung bình 3 môn của sinh viên đó, làm tròn đến 2 số thập phân sau dấu phẩy.

### Đầu vào

Đầu vào gồm nhiều dòng từ bàn phím:

- Dòng đầu tiên chữ số nguyên  $N$ , là số lượng sinh viên. ( $2 \leq N \leq 10$ )
- $N$  dòng tiếp theo chứa tên và các đầu điểm (điểm lớn hơn 0 và bé hơn 100), các thông tin cách nhau bởi duy nhất một khoảng trắng.
- Dòng cuối cùng chứa tên của một sinh viên đã nhập.

### Đầu ra

In ra màn hình một dòng duy nhất chứa điểm trung bình của sinh viên có tên ở dòng cuối của đầu vào.

**For example:**

Input	Result
3	56.00
Krishna 67 68 69	
Arjun 70 98 63	
Malika 52 56 60	
Malika	

**Answer:** (penalty regime: 0 %)

```
1 no_student = int(input())  
2 empty_dict = {}  
3 while no_student > 0:  
4     student_name, score1, score2, score3 = input().split(" ")  
5     score1 = float(score1)  
6     score2 = float(score2)  
7     score3 = float(score3)  
8     avg = (score1 + score2 + score3) / 3  
9     empty_dict[student_name] = avg  
10    no_student -= 1  
11 search_student = input()  
12 print(f'{empty_dict[search_student]:.2f}')  
13
```

	<b>Input</b>	<b>Expected</b>	<b>Got</b>	
✓	3 Krishna 67 68 69 Arjun 70 98 63 Malika 52 56 60 Malika	56.00	56.00	✓
✓	2 Harsh 25 26.5 28 Anurag 26 28 30 Harsh	26.50	26.50	✓
✓	3 Riya 52 93 20 Rencho 69 65 62 Hbtg 52 60 68 Hbtg	60.00	60.00	✓
✓	2 Anurag 50 60 70 Prerna 100 99.5 99 Prerna	99.50	99.50	✓
✓	3 Gaurav 72 79 86 Bhejaria 59 60 61 Salman 59 59 60 Salman	59.33	59.33	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 11

Đúng

Cho 2 số phức  $A$  và  $B$ , nhiệm vụ của bạn là viết các hàm để tính tổng, hiệu và tích của hai số phức này.

### Đầu vào

Đầu vào từ bàn phím gồm hai dòng, mỗi xong chứa hai số nguyên là phần thực và phần ảo của  $A$  và  $B$ .

### Đầu ra

In ra màn hình gồm 3 dòng:

- $A + B$
- $A - B$
- $A * B$

**Answer:** (penalty regime: 0 %)

[Reset answer](#)

```
1 import math
2 class Complex(object):
3     def __init__(self, real, imaginary):
4         #code cua ban o day
5         self.real = real
6         self.imaginary = imaginary
7         return
8
9     def __add__(self, no):
10        #code cua ban o day
11        return Complex(self.real + no.real, self.imaginary + no.imaginary)
12
13    def __sub__(self, no):
14        #code cua ban o day
15        return Complex(self.real - no.real, self.imaginary - no.imaginary)
16
17    def __mul__(self, no):
18        #code cua ban o day
19        return Complex(self.real*no.real - self.imaginary*no.imaginary, self.real*no.imaginary + self.imaginary*no.real)
20
```

	Input	Expected	Got	
✓	1 2 3 4	4.00+6.00i -2.00-2.00i -5.00+10.00i	4.00+6.00i -2.00-2.00i -5.00+10.00i	✓
✓	5 6 4 5	9.00+11.00i 1.00+1.00i -10.00+49.00i	9.00+11.00i 1.00+1.00i -10.00+49.00i	✓
✓	0 1 2 0	2.00+1.00i -2.00+1.00i 0.00+2.00i	2.00+1.00i -2.00+1.00i 0.00+2.00i	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 12

Đúng

`defaultdict` là một công cụ trong lớp `collections` của Python. Cấu trúc `defaultdict` có chức năng tương tự như từ điển thông thường (`dict`), nhưng sự khác biệt duy nhất là `defaultdict` sẽ trả về một giá trị mặc định nếu khóa (key) không tồn tại khi truy vấn. Nếu không sử dụng `defaultdict`, chúng ta sẽ phải kiểm tra nếu khóa có tồn tại trong từ điển (`dict`) hay không, nếu không thì phải thêm khóa đó vào từ điển và liên kết khóa đó với một giá trị mới.

Ví dụ:

```
from collections import defaultdict
d = defaultdict(list)
d['python'].append("awesome")
d['something-else'].append("not relevant")
d['python'].append("language")
for i in d.items():
    print(i)
```

Đoạn code trên sẽ in ra:

```
('python', ['awesome', 'language'])
('something-else', ['not relevant'])
```

## Bài tập

Cho hai danh sách  $A$  gồm  $n$  xâu kí tự và  $B$  gồm  $m$  xâu kí tự. Với mỗi xâu kí tự trong  $B$ , kiểm tra xem xâu kí tự đó có xuất hiện trong  $A$  hay không. Nếu có xuất hiện, in ra các vị trí của xâu đó trong danh sách  $B$ . Nếu không, in ra  $-1$ .

### Đầu vào

Đầu vào gồm nhiều dòng từ bàn phím:

- Dòng thứ nhất chứa hai số  $n$  và  $m$  cách nhau bởi 1 khoảng trắng, là độ dài của danh sách  $A$  và  $B$ .
- $N$  dòng tiếp theo chứa các phần tử của danh sách  $A$ .
- $M$  dòng cuối cùng chứa các phần tử của danh sách  $B$ .

### Đầu ra

In ra màn hình  $m$  dòng khác nhau. Dòng thứ  $i$  chứa các vị trí xuất hiện của xâu  $B_i$  trong danh sách  $A$  hoặc  $-1$  nếu  $A$  không chứa  $B_i$ .

For example:

Input	Result
5 2	1 2 4
a	3 5
a	
b	
a	
b	
a	
b	

Answer: (penalty regime: 0 %)

```
1 from collections import defaultdict
2
3
4 # Read input
5 n, m = map(int, input().split())
6 A = [input() for _ in range(n)]
7 B = [input() for _ in range(m)]
8
9 # Create a defaultdict to store the positions of strings in A
10 positions = defaultdict(list)
11
12 # Iterate over A and store the positions of each string
13 for i, string in enumerate(A):
14     positions[string].append(i)
15
16 # Iterate over B and check if each string is in A
17 for string in B:
18     if string in positions:
19         # If the string is in A, print its positions
20         print(' '.join(map(str, [pos + 1 for pos in positions[string]])))
```

	Input	Expected	Got	
✓	5 2 a a b a b a b	1 2 4 3 5	1 2 4 3 5	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 13

Đúng

`collections.Counter()` là bộ đếm tần suất xuất hiện của các phần tử trong danh sách. `collections.Counter()` sử dụng cấu trúc từ điển, trong đó khóa (key) là phần tử của danh sách, và giá trị (value) là tần suất xuất hiện của phần tử đó.

Xem ví dụ sau:

```
>>> from collections import Counter
>>>
>>> myList = [1,1,2,3,4,5,3,2,3,4,2,1,2,3]
>>> print Counter(myList)
Counter({2: 4, 3: 4, 1: 3, 4: 2, 5: 1})
>>>
>>> print Counter(myList).items()
[(1, 3), (2, 4), (3, 4), (4, 2), (5, 1)]
>>>
>>> print Counter(myList).keys()
[1, 2, 3, 4, 5]
>>>
>>> print Counter(myList).values()
[3, 4, 4, 2, 1]
```

## Bài tập

Mr. A là chủ cửa hàng giày. Cửa hàng ông có  $X$  chiếc giày với kích cỡ khác nhau. Có  $N$  khách hàng tới cửa hàng, khách hàng thứ  $i$  muốn mua giày cỡ  $s_i$  với giá  $p_i$ .

Hãy viết chương trình tính xem ông A thu được bao nhiêu tiền từ việc bán giày cho khách.

## Đầu vào

Đầu vào gồm nhiều dòng nhập vào từ bàn phím:

- Dòng nhú nhất chứa số nguyên  $\backslash(X)$  là số lượng đôi giày trong cửa hàng.
- Dòng thứ 2 chứa  $X$  số nguyên là kích cỡ của từng đôi giày, các số cách nhau bởi một dấu cách.
- Dòng thứ 3 chứa số nguyên  $N$  là số lượng khách hàng
- Trong  $N$  dòng tiếp theo, dòng thứ  $i$  chứa  $s_i$  và  $p_i$  cách nhau bởi một khoảng trống thể hiện khách hàng thứ  $i$  muốn mua giày cỡ  $s_i$  với giá  $p_i$ .

## Đầu ra

In ra màn hình lượng tiền mà ông A thu được.

Lưu ý: ông A chỉ bán được cho khách thứ  $i$  nếu trong kho hàng còn giày với cỡ  $s_i$ .

For example:

Input	Result
10	200
2 3 4 5 6 8 7 6 5 18	
6	
6 55	
6 45	
6 55	
4 40	
18 60	
10 50	

Answer: (penalty regime: 0 %)

```
1 from collections import Counter
2 no_shoes = int(input())
3 shoes_sizes = list(map(int, input().split(" ")))
4 purchase = int(input())
5 shoes_remaain = Counter(shoes_sizes)
6 revenue = 0
7 for i in range(purchase):
8     shoe_size, price = map(int, input().split(" "))
9     if shoes_remaain[shoe_size] > 0:
10         shoes_remaain[shoe_size] -= 1
11         revenue += price
12 print(revenue)
13
```

	Input	Expected	Got	
✓	10 2 3 4 5 6 8 7 6 5 18 6 6 55 6 45 6 55 4 40 18 60 10 50	200	200 ✓	
✓	10 12 12 2 16 13 7 17 6 13 4 5 15 32 13 89 13 36 6 69 12 61	255	255 ✓	

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 14

Đúng

Như đã học ở bài trước, để khởi tạo một danh sách với một số phần tử cho trước, chúng ta sử dụng cú pháp sau:

```
list_a = [1, 3, 4]
```

Tuy nhiên, khi dách sách dài hơn với hàng triệu phần tử, rất khó để lập trình viên có thể gõ tất cả các phần tử vào danh sách. Khi đó chúng ta phải sử dụng phép duyệt để lặp lượt đọc và thêm các phần tử vào danh sách rỗng ban đầu thông qua phép `append`.

Ví dụ: Để tạo một danh sách chứa các giá trị từ 0 đến 9 ta có thể sử dụng đoạn code sau:

```
list_a = []
for i in range(0,10):
    list_a.append(i)
```

Ngoài ra, Python còn cho phép tạo ra danh sách như trên bằng nhiều cách ngắn gọn và tiện lợi hơn:

```
list_a = [x for x in range(0,10)]
print list_a
```

Hoặc, tạo ra danh sách hai chiều bằng cách sử dụng hai vòng lồng nhau:

```
# tạo ra một dách sách chứa các cặp chỉ số hàng và cột trong một ma trận vòng 10x10
list_b = [(i,j) for i in range(0,10) for j in range(0,10)]
print list_b
```

Hoặc, chúng ta có thể kết hợp lệnh điều kiện và vòng lặp để lọc bỏ bớt các phần tử của danh sách.

Ví dụ, để tạo một danh sách chứa các số chia hết 3 năm trong khoảng từ [0, 100], chúng ta làm như sau:

```
list_c = [x for x in range(0,101) if x%3 == 0]
```

## Bài tập

Cho 3 số nguyên  $X, Y, Z$  thể hiện số chiều của một hình lập phương và một số nguyên  $N$ . Hãy tạo và in ra một danh sách (list) các tọa độ  $[i, j, k]$  trong hình lập phương mà tổng  $i + j + k$  không bằng  $N$ . ( $0 \leq i \leq X, 0 \leq j \leq Y, 0 \leq k \leq Z$ ).

### Đầu vào

Đầu vào gồm 4 dòng từ bàn phím lần lượt chứa các giá trị  $X, Y, Z$  và  $N$ .

### Đầu ra

In ra các cặp  $(i, j, k)$  thỏa mãn bài toán theo thứ tự các chỉ số tăng dần.

For example:

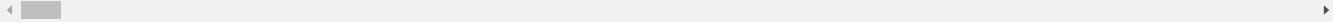
Input	Result
2	<pre>[[0, 0, 0], [0, 0, 1], [0, 1, 0], [0, 1, 2], [0, 2, 1], [0, 2, 2], [1, 0, 0], [1, 0, 2], [1, 1, 1], [1, 1, 2], [1, 2, 0], [1, 2, 1], [1, 2, 2]]</pre>
2	
2	
2	

Answer: (penalty regime: 0 %)

```
1 def asdsa(x, y, z,n):
2     result = [ [i, j, k] for i in range (0, x + 1) for j in range (0, y + 1) for k in range (0, z + 1) if ((i + j + k) != n) ]
3     return result
4 x = int(input())
5 y = int(input())
6 z = int(input())
7 n = int(input())
8 print(asdsa(x, y, z ,n))
```

	Input	Expected
✓	2 2 2 2	[[0, 0, 0], [0, 0, 1], [0, 1, 0], [0, 1, 2], [0, 2, 1], [0, 2, 2], [1, 0, 0], [1, 0, 2], [1, 1, 1], [1, 1, 2], [1, 2, 0], [1, 2,
✓	2 2 2 2	[[0, 0, 0], [0, 0, 1], [0, 1, 0], [0, 1, 2], [0, 2, 1], [0, 2, 2], [1, 0, 0], [1, 0, 2], [1, 1, 1], [1, 1, 2], [1, 2, 0], [1, 2,
✓	1 2 3 3	[[0, 0, 0], [0, 0, 1], [0, 0, 2], [0, 1, 0], [0, 1, 1], [0, 1, 3], [0, 2, 0], [0, 2, 2], [0, 2, 3], [1, 0, 0], [1, 0, 1], [1, 0,
✓	4 5 6 -10	[[0, 0, 0], [0, 0, 1], [0, 0, 2], [0, 0, 3], [0, 0, 4], [0, 0, 5], [0, 0, 6], [0, 1, 0], [0, 1, 1], [0, 1, 2], [0, 1, 3], [0, 1,
✓	5 5 5 8	[[0, 0, 0], [0, 0, 1], [0, 0, 2], [0, 0, 3], [0, 0, 4], [0, 0, 5], [0, 1, 0], [0, 1, 1], [0, 1, 2], [0, 1, 3], [0, 1, 4], [0, 1,

Passed all tests! ✓



Đúng

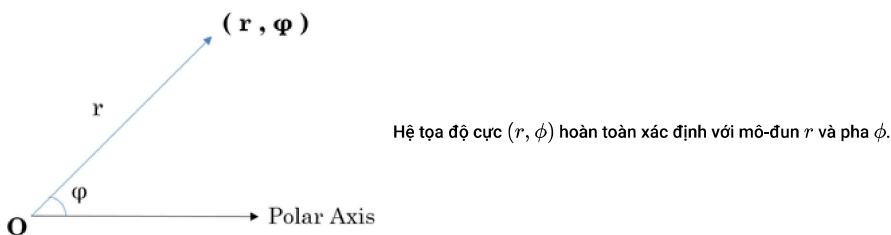
Marks for this submission: 10,00/10,00.

## Câu Hỏi 15

Đúng

Hệ tọa độ cực là một cách biểu diễn khác của hệ tọa độ Đè-các hoặc số phức.

Một số phức  $z = x + yj$  hoàn toàn xác định bởi phần thực  $x$  và phần ảo  $y$ . Ở đây  $j$  là đơn vị ảo.



Hệ tọa độ cực  $(r, \phi)$  hoàn toàn xác định với mô-đun  $r$  và pha  $\phi$ .

Nếu chúng ta đổi số phức  $z$  sang hệ tọa độ cực, ta có:

- $r$ : Khoảng cách từ  $z$  tới gốc tọa độ,  $\sqrt{x^2 + y^2}$
- $\phi$ : Góc tạo bởi trục  $x$  và đoạn thẳng nối từ  $z$  tới gốc tọa độ theo chiều ngược chiều kim đồng hồ.

Mô-đun `cmath` của Python cung cấp những công cụ toán học để xử lý số phức.

### cmath.phase

Công cụ này trả về pha của số phức  $z$  (còn gọi là argument của  $z$ )

```
>>> phase(complex(-1.0, 0.0))  
3.1415926535897931
```

### abs

Công cụ này trả về mô-đun (giá trị tuyệt đối) của số phức  $z$ .

```
>>> abs(complex(-1.0, 0.0))  
1.0
```

## Bài tập

Cho một số phức  $z$ . Nhiệm vụ của bạn là chuyển nó sang tọa độ cực.

### Đầu vào

Một dòng duy nhất chứa số phức  $z$ . Chú ý: hàm `complex()` có thể được dùng trong Python để chuyển đầu vào thành một số phức.

### Đầu ra

Gồm hai dòng: dòng 1 in giá trị của mô-đun  $r$ , dòng hai in giá trị của pha  $\phi$

For example:

Input	Result
1+2j	2.23606797749979 1.1071487177940904

Answer: (penalty regime: 0 %)

```
1 import cmath  
2 z = complex(input())  
3 print( abs(z))  
4 print(cmath.phase(z))
```

	Input	Expected	Got	
✓	1+2j	2.23606797749979 1.1071487177940904	2.23606797749979 1.1071487177940904	✓
✓	-1-5j	5.0990195135927845 -1.7681918866447774	5.0990195135927845 -1.7681918866447774	✓
✓	-80+25j	83.81527307120105 2.838707785214822	83.81527307120105 2.838707785214822	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 16

Đúng

Trong Python, một xâu có thể được tách thành một danh sách các xâu con bằng cách sử dụng hàm `split()`. Ví dụ:

```
>>> a = "this is a string"
>>> a = a.split(" ") # tách các từ trong xâu sử dụng khoảng cách trống.
>>> print a
# ['this', 'is', 'a', 'string']
```

Để nối các xâu lại với nhau, chúng ta dùng hàm `join` như ví dụ sau:

```
>>> a = "-".join(a) # Nối các xâu lại với nhau bằng dấu gạch ngang.
>>> print a
# this-is-a-string
```

## Bài tập

Nhập vào một chuỗi, hãy tách chuỗi đã nhập theo khoảng cách trống (" ") và nối vào với nhau sử dụng dấu gạch ngang ("-").

### Đầu vào

Gồm một dòng duy nhất từ bàn phím chứa chuỗi đầu vào.

### Đầu ra

In ra màn hình theo yêu cầu của đề bài.

For example:

Input	Result
this is a string	this-is-a-string

Answer: (penalty regime: 0 %)

```
1 | a = input().strip().split(' ')
2 | print("-".join(a))
```

	Input	Expected	Got	
✓	this is a string	this-is-a-string	this-is-a-string	✓
✓	quick brown fox jumps over a lazy dog	quick-brown-fox-jumps-over-a-lazy-dog	quick-brown-fox-jumps-over-a-lazy-dog	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 17

Đúng

Trong Python, `namedtuple` là một cách để tạo ra các mảng đối tượng một cách nhanh chóng và tiện lợi. Cấu trúc này giúp biến các tuples thành các đối tượng với các thuộc tính được đặt tên nhằm mang lại sự thuận lợi khi sử dụng.

Hãy xem xét ví dụ sau đây:

Code 01

```
>>> from collections import namedtuple  
>>> Point = namedtuple('Point','x,y')  
>>> pt1 = Point(1,2)  
>>> pt2 = Point(3,4)  
>>> dot_product = ( pt1.x * pt2.x ) +( pt1.y * pt2.y )  
>>> print dot_product  
11
```

Code 02

```
>>> from collections import namedtuple  
>>> Car = namedtuple('Car','Price Mileage Colour Class')  
>>> xyz = Car(Price = 100000, Mileage = 30, Colour = 'Cyan', Class = 'Y')  
>>> print xyz  
Car(Price=100000, Mileage=30, Colour='Cyan', Class='Y')  
>>> print xyz.Class  
Y
```

## Bài tập

Tiến sĩ X có một bảng tính Excel chứa danh sách mã sinh viên (IDS), điểm (marks), lớp (class) và tên (name) của tất cả các sinh viên. Hãy giúp tiến sĩ X tính điểm trung bình của cả lớp bằng công thức sau:  $\text{Điểm trung bình} = \frac{\text{Tổng điểm}}{\text{Số lượng sinh viên}}$ .

### Đầu vào

Đầu vào từ bàn phím gồm nhiều dòng từ bàn phím:

- Dòng thứ nhất chứa số nguyên  $N$  là số lượng sinh viên
- Dòng thứ hai chứa tên của các trường thông tin
- $N$  dòng tiếp theo chứa thông tin của sinh viên, thứ tự của các trường thông tin là thứ tự được xác định ở dòng thứ 2.

### Đầu ra

In ra điểm trung bình của cả lớp làm tròn đến hai chữ số thập phân sau dấu phẩy.

For example:

Input				Result
5				78.00
ID	MARKS	NAME	CLASS	
1	97	Raymond	7	
2	50	Steven	4	
3	91	Adrian	9	
4	72	Stewart	5	
5	80	Peter	6	

Answer: (penalty regime: 0 %)

```
1 from collections import namedtuple  
2  
3 input_ = int(input())  
4 my_fields = input().split()  
5 total_marks = 0  
6  
7 for _ in range(input_):  
8     students = namedtuple('my_student', my_fields)  
9     MARKS, CLASS, NAME, ID = input().split()  
10    my_student = students(MARKS, CLASS, NAME, ID)  
11    total_marks += int(my_student.MARKS)  
12  
13 average_marks = total_marks / input_  
14 average_marks_formatted = format(average_marks, '.2f')  
15 print(average_marks_formatted)  
16
```

	Input				Expected	Got		
✓	5	ID	MARKS	NAME	CLASS	78.00	78.00	✓
		1	97	Raymond	7			
		2	50	Steven	4			
		3	91	Adrian	9			
		4	72	Stewart	5			
		5	80	Peter	6			
✓	5	MARKS	CLASS	NAME	ID	81.00	81.00	✓
		92	2	Calum	1			
		82	5	Scott	2			
		94	2	Jason	3			
		55	8	Glenn	4			
		82	2	Fergus	5			

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 18

Đúng

Ở các bài học trước, chúng ta đã biết rằng, danh sách (**list**) có thể thay đổi được (**mutable**), còn **tuples** là không thể thay đổi được (**immutable**).

Hãy xem xét ví dụ sau với xâu:

Cho một xâu (xâu thuộc kiểu không thay đổi được - **immutable**). Hãy tìm cách để thay đổi một số kí tự trong xâu đó:

```
>>> string = "abracadabra"
```

Xem xét kí tự ở vị trí thứ 5:

```
>>> print string[5]
# a
```

Nếu sử dụng phép gán để thay đổi kí tự ở một vị trí bất kì trong xâu bằng một kí tự khác, chương trình sẽ sinh ra lỗi như sau:

```
>>> string[5] = 'k'
Traceback (most recent call last):
  File "", line 1, in 
TypeError: 'str' object does not support item assignment
```

Một cách để thực hiện được điều này là chuyển đổi xâu (**string**) sang danh sách (**list**) và sau đó thay đổi giá trị.

```
>>> string = "abracadabra"
>>> l = list(string)
>>> l[5] = 'k'
>>> string = ''.join(l)
>>> print string
abrackdabra
```

Một cách khác là tách xâu gốc ở vị trí cần thay đổi thành hai phần khác nhau và nối lại bằng kí tự cần thay thế:

```
>>> string = string[:5] + "k" + string[6:] # Thay thế kí tự ở vị trí thứ 5 bằng kí tự 'k'.
>>> print string
abrackdabra
```

## Bài tập

Nhập vào một xâu, hãy viết chương trình thay đổi kí tự ở một vị trí cho trước và in ra xâu đã sửa.

### Đầu vào

Gồm nhiều dòng nhập vào từ bàn phím:

- Dòng thứ nhất chứa xâu  $S$ .
- Dòng tiếp theo chứa số nguyên  $i$  và kí tự  $c$  cách nhau bởi duy nhất một khoảng trắng.

### Đầu ra

In ra xâu  $S$  sau khi đã thay thế kí tự ở vị trí  $i$  bằng kí tự  $c$ .

For example:

Input	Result
abracadabra 5 k	abrackdabra

Answer: (penalty regime: 0 %)

```
1 | S = input()
2 | i, c = input().split()
3 | i = int(i)
4 | S = S[:i] + c + S[i+1:]
5 | print(S)
```

	Input	Expected	Got	
✓	abracadabra 5 k	abrackdabra	abrackdabra	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 19

Đúng

Danh sách lồng là một danh sách chứa nhiều danh sách con khác nhau.

Ví dụ:

```
>>> nested_list = [['blue', 'green'], ['red', 'black'], ['blue', 'white']]
>>> print len(nested_list)
# [[['blue', 'green'], ['red', 'black'], ['blue', 'white']]]
>>> print nested_list[1]
# ['red', 'black']
>>> print nested_list[1][0]
# prints red
```

## Bài tập

Cho danh sách tên và điểm của  $N$  tất cả sinh viên trong lớp Lập trình Python. Hãy in ra tên tất cả những học sinh có điểm cao thứ nhì. Nếu có nhiều hơn một sinh viên có điểm cao thứ nhì, in ra tên của những sinh viên này theo thứ tự từ điển.

**Gợi ý:** Dùng danh sách lồng để lưu lại thông tin và tên và điểm của các sinh viên.

### Đầu vào

Đầu vào từ bàn phím gồm  $2N + 1$  dòng:

- Dòng thứ nhất chứa số  $N$  là số lượng học sinh. ( $2 \leq N \leq 5$ )
- Trong  $2N$  dòng tiếp theo, cứ mỗi 2 dòng chứa thông tin của một học sinh. Dòng thứ nhất chứa tên, dòng thứ hai chứa điểm.

### Đầu ra

In ra danh sách tên các học sinh có điểm số cao thứ hai theo thứ tự từ điển, mỗi sinh viên trên một dòng.

For example:

Input	Result
5	Berry
Harry	Harry
37.21	
Berry	
37.21	
Tina	
37.2	
Akriti	
41	
Harsh	
39	

**Answer:** (penalty regime: 0 %)

```
1
2 def main():
3     number_students = int(input())
4     students = []
5     grades = set()
6
7     for i in range(number_students):
8         name = input()
9         grade = float(input())
10        grades.add(grade)
11        students.append([name, grade])
12
13    second_lowest_grade = sorted(grades)[1]
14    second_lowest_students = sorted([student[0] for student in students if student[1] == second_lowest_grade])
15    for student in second_lowest_students:
16        print(student)
17
18 if __name__ == '__main__':
19     main()
```

	<b>Input</b>	<b>Expected</b>	<b>Got</b>	
✓	5 Harry 37.21 Berry 37.21 Tina 37.2 Akriti 41 Harsh 39	Berry Harry	Berry Harry	✓
✓	4 Prashant 32 Pallavi 36 Dheeraj 39 Shivam 40	Pallavi	Pallavi	✓
✓	5 Harsh 20 Beria 20 Varun 19 Kakunami 19 Vikas 21	Beria Harsh	Beria Harsh	✓
✓	4 Rachel -50 Mawer -50 Sheen -50 Shaheen 51	Shaheen	Shaheen	✓
✓	5 Prashant 52.22 Kush 52.223 Kant 52.222 Kanti 52.2222 Harshit 52.22222	Kant	Kant	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 20

Đúng

Hãy viết chương trình tìm số lần xuất hiện của một xâu con trong một xâu cho trước. (Xâu có chứa cả kí tự viết hoa và viết thường).

### Đầu vào

Đầu vào gồm hai dòng từ bàn phím (độ dài không quá 200):

- Dòng thứ nhất chứa xâu gốc.
- Dòng thứ hai chứa một xâu con của xâu gốc.

### Đầu ra

In ra màn hình một dòng duy nhất chứa số lần xuất hiện của xâu con trong xâu gốc.

Lưu ý: Trong Python, để biết độ dài của một xâu, chúng ta sử dụng hàm `len(s)`, trong đó s là một xâu. Để duyệt qua các kí tự của xâu, chúng ta có thể sử dụng vòng lặp `for` như sau:

```
for i in range(0, len(s)):  
    print (s[i])
```

For example:

Input	Result
ABCD CDC	2
CDC	

Answer: (penalty regime: 0 %)

```
1 #ex 19  
2 # Get the input strings  
3 original_string = input()  
4 substring = input()  
5  
6 # Count the occurrences of the substring in the original string  
7 count = original_string.count(substring)  
8  
9  
10  
11 # Initialize a counter variable  
12 count = 0  
13  
14 # Iterate through the original string  
15 for i in range(len(original_string) - len(substring) + 1):  
16     # Check if the substring matches the current slice of the original string  
17     if original_string[i:i+len(substring)] == substring:  
18         # Increment the counter if there is a match  
19         count += 1  
20
```

	Input	Expected	Got	
✓	ABCD CDC CDC	2	2	✓
✓	I am an Indian, by birth. Birth	0	0	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 21

Đúng

Cấu trúc `OrderedDict` là một kiểu từ điển mà thứ tự của các khóa (keys) trong từ điển được lưu theo thứ tự mà khóa đó được thêm vào. Nếu giá trị của một khóa được cập nhật, thì vị trí của khóa đó trong `OrderedDict` vẫn không thay đổi.

Ví dụ:

```
>>> from collections import OrderedDict
>>>
>>> ordinary_dictionary = {}
>>> ordinary_dictionary['a'] = 1
>>> ordinary_dictionary['b'] = 2
>>> ordinary_dictionary['c'] = 3
>>> ordinary_dictionary['d'] = 4
>>> ordinary_dictionary['e'] = 5
>>>
>>> print ordinary_dictionary
{'a': 1, 'c': 3, 'b': 2, 'e': 5, 'd': 4}
>>>
>>> ordered_dictionary = OrderedDict()
>>> ordered_dictionary['a'] = 1
>>> ordered_dictionary['b'] = 2
>>> ordered_dictionary['c'] = 3
>>> ordered_dictionary['d'] = 4
>>> ordered_dictionary['e'] = 5
>>>
>>> print ordered_dictionary
OrderedDict([('a', 1), ('b', 2), ('c', 3), ('d', 4), ('e', 5)])
```

## Bài tập

Cho danh sách tên và giá bán của N sản phẩm được siêu thị bán ra trong một ngày. Hãy in ra danh sách tên các sản phẩm và tổng số tiền thu được từ sản phẩm đó. Các sản được bắt đầu bán sớm hơn thì cần được in ra trước.

### Đầu vào

Đầu vào từ bàn phím gồm nhiều dòng:

- Dòng thứ nhất chứa số  $N$  là số lượng sản phẩm được bán ra.
- $N$  dòng tiếp theo chứa tên và giá của sản phẩm, cách nhau bởi một khoảng trống.

### Đầu ra

In ra tên và tổng số tiền thu được (cách nhau bởi 1 khoảng trống) của các sản phẩm trên các dòng khác nhau theo thứ tự đã quy định.

For example:

Input	Result
9	BANANA FRIES 12
BANANA FRIES 12	POTATO CHIPS 60
POTATO CHIPS 30	APPLE JUICE 20
APPLE JUICE 10	CANDY 20
CANDY 5	
APPLE JUICE 10	
CANDY 5	
CANDY 5	
POTATO CHIPS 30	

Answer: (penalty regime: 0 %)

```
1 from collections import OrderedDict
2
3 ordinary_dictionary = OrderedDict()
4
5 for _ in range(int(input())):
6
7     data = input().split()
8     price = int(data.pop())
9     name = ' '.join(data)
10
11 if name in ordinary_dictionary:
12     ordinary_dictionary[name] += price
13 else:
14     ordinary_dictionary[name] = price
15
16 [print(item[0] + ' ' + str(item[1])) for item in ordinary_dictionary.items()]
17
```

	Input	Expected	Got	
✓	9 BANANA FRIES 12 POTATO CHIPS 30 APPLE JUICE 10 CANDY 5 APPLE JUICE 10 CANDY 5 CANDY 5 CANDY 5 POTATO CHIPS 30	BANANA FRIES 12 POTATO CHIPS 60 APPLE JUICE 20 CANDY 20	BANANA FRIES 12 POTATO CHIPS 60 APPLE JUICE 20 CANDY 20	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 22

Đúng

Trong bài học này, chúng ta sẽ tìm hiểu một số hàm phổ biến khi làm việc với các ký tự trong xâu.

### 1. Hàm str.isalnum()

Hàm này kiểm tra nếu một xâu chỉ chứa các ký tự là chữ cái và số (a-z, A-Z, 0-9) hay không.

```
>>> print 'ab123'.isalnum()
True
>>> print 'ab123#'.isalnum()
False
```

### 2. Hàm str.isalpha()

Hàm này kiểm tra nếu một xâu chỉ chứa các ký tự là chữ cái (a-z, A-Z) hay không.

```
>>> print 'abcD'.isalpha()
True
>>> print 'abcd1'.isalpha()
False
```

### 3. Hàm str.isdigit()

Hàm này kiểm tra nếu xâu chỉ chứa các ký tự là số hay không.

```
>>> print '1234'.isdigit()
True
>>> print '123edsd'.isdigit()
False
```

### 4. Hàm str.islower()

Hàm này kiểm tra nếu một xâu chỉ chứa các ký tự viết thường.

```
>>> print 'abcd123#'.islower()
True
>>> print 'Abcd123#'.islower()
False
```

### 5. Hàm str.isupper()

Hàm này kiểm tra nếu một xâu chỉ chứa các ký tự viết hoa.

```
>>> print 'ABCD123#'.isupper()
True
>>> print 'Abcd123#'.isupper()
False
```

## Bài tập

Cho đầu vào là một xâu  $S$ . Hãy lân lượt kiểm tra xem xâu  $S$  có chứa ký tự là (1) chữ hoặc số, (2) ký tự chữ, (3) ký tự số, (4) ký tự viết hoa, (5) ký tự viết thường hay không.

### Đầu vào

Đầu vào gồm một dòng duy nhất từ bàn phím chứa xâu  $S$ . ( $0 < \text{len}(S) < 1000$ )

### Đầu ra

In ra màn hình các dòng sau:

- Dòng thứ nhất, in ra True nếu xâu  $S$  chứa ký tự chữ hoặc số. Ngược lại, in ra False.
- Dòng thứ hai, in ra True nếu xâu  $S$  chứa ký tự chữ. Ngược lại, in ra False.
- Dòng thứ ba, in ra True nếu xâu  $S$  chứa ký tự số. Ngược lại, in ra False.
- Dòng thứ tư, in ra True nếu xâu  $S$  chứa ký tự viết thường. Ngược lại, in ra False.
- Dòng thứ năm, in ra True nếu xâu  $S$  chứa ký tự viết hoa. Ngược lại, in ra False.

For example:

Input	Result
qA2	True True True True True

Answer: (penalty regime: 0 %)

```
 1 #!/usr/bin/python  
 2  
 3 S = input()  
 4  
 5 contains_alnum = any(char.isalnum() for char in S)  
 6  
 7 contains_alpha = any(char.isalpha() for char in S)  
 8  
 9 contains_digit = any(char.isdigit() for char in S)  
10  
11  
12 contains_lower = any(char.islower() for char in S)  
13  
14  
15 contains_upper = any(char.isupper() for char in S)  
16  
17  
18  
19 print(contains_alnum)  
20 print(contains_alpha)
```

Passed all tests! ✓

Đóng

Marks for this submission: 10,00/10,00.

Căn trái, căn phải hay căn giữa là các công cụ mà chúng ta thường dùng khi làm việc với văn bản. Trong Python, các chức năng này được thực hiện bằng các hàm xây dựng sẵn dưới đây:

## 1. str.ljust(width)

Trả về xâu đã được căn trái với độ dài xác định trước.

```
>>> width = 20 # Độ rộng dòng.  
>>> print 'HackerRank'.ljust(width,'-') # Căn trái sử dụng kí tự "-", độ rộng dòng = 20.  
HackerRank-----
```

## 2. str.center(width)

Trả về xâu đã được căn giữa với độ dài cho trước.

```
>>> width = 20
>>> print 'HackerRank'.center(width,'-')
-----HackerRank-----
```

### 3. str.rjust(width)

Trả về xâu đã được căn phải với độ dài cho trước.

```
>>> width = 20
>>> print 'HackerRank'.rjust(width,'-')
-----HackerRank
```

## Bài tập

Cho trước một phần của đoạn code để in ra logo HackerRank như ô bên dưới với biến độ dày nhập vào từ bàn phím. Hãy thay thế khoảng trống (\_\_\_\_) trong ô trả lời dưới đây với các hàm `rjust`, `ljust` hay `center` thích hợp.

## Đầu vào

Đầu vào từ bàn phím gồm một dòng duy nhất chứa biến độ lớn (thickness) của logo.

## Đầu ra

In ra logo HackerRank với độ lớn cho trước.

**For example:**

**Answer:** (penalty regime: 0 %)

[Reset answer](#)

```
1 #Replace all _____ with rjust, ljust or center.  
2
```

```
3 thickness = int(input()) #This must me an odd number
4 c = 'H'
5
6 #Top Cone
7 for i in range(thickness):
8     print((c*i).rjust(thickness-1)+c+(c*i).ljust(thickness-1))
9
10 #Top Pillars
11 for i in range(thickness+1):
12     print((c*thickness).center(thickness*2)+(c*thickness).center(thickness*6))
13
14 #Middle Belt
15 for i in range((thickness+1)//2):
16     print((c*thickness*5).center(thickness*6))
17
18 #Bottom Pillars
19 for i in range(thickness+1):
20     print((c*thickness).center(thickness*2)+(c*thickness).center(thickness*6))
```

Passed all tests! 

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 24

Đúng

Cho hai điểm  $A$  và  $B$  trên một đường thẳng có tọa độ là  $x_a$  và  $x_b$ . Hãy viết chương trình tính khoảng cách giữa hai điểm  $A$  và  $B$ .

Gợi ý: Sử dụng hàm `abs()` để tính giá trị tuyệt đối của một số.

### Đầu vào

Đầu vào gồm một dòng duy nhất chứa hai số nguyên cách nhau bởi một dấu cách là tọa độ của  $A$  và  $B$ .

### Đầu ra

In ra màn hình một dòng duy nhất chứa khoảng cách giữa hai điểm.

For example:

Input	Result
1 2	1

Answer: (penalty regime: 0 %)

```
1 import math
2
3 # Get the input
4 A, B = map(int, input().split())
5
6
7 # Calculate the distance between two points
8 distance = abs(A-B)
9
10 # Print the result
11 print(distance)
12
```

	Input	Expected	Got	
✓	1 2	1	1	✓
✓	1 3	2	2	✓
✓	3 1	2	2	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 25

Đúng

Cho 4 điểm  $A, B, C, D$  trên hệ tọa độ ba chiều. Bạn hãy tính góc giữa  $A, B, C$  và  $B, C, D$ .

Gọi góc này là  $P$ , ta có:

$$\cos(P) = \frac{(X \cdot Y)}{|X||Y|} \text{ với } X = AB \times BC \text{ và } Y = BC \times CD.$$

Ở đây,  $X, Y$  là tích vô hướng của  $X$  và  $Y$  và  $AB \times BC$  là tích có hướng của 2 vector  $AB, CD$ .

Hoàn thành các hàm trong lớp `Points` dưới đây để giải quyết bài toán.

### Đầu vào

Dữ liệu vào từ bàn phím gồm 4 dòng, mỗi dòng chứa 3 số nguyên cách nhau bởi một dấu phẩy là tọa độ của 4 điểm trên hệ tọa độ ba chiều.

### Đầu ra

In ra màn hình giá trị góc  $P$

For example:

Input	Result
0 4 5	8.19
1 7 6	
0 5 9	
1 7 2	

Answer: (penalty regime: 0 %)

Reset answer

```
1 import math
2
3 class Points(object):
4     def __init__(self, x, y, z):
5         self.x = x
6         self.y = y
7         self.z = z
8
9     def __sub__(self, no):
10        return Points(self.x - no.x, self.y - no.y, self.z - no.z)
11
12     def dot(self, no):
13        return self.x * no.x + self.y * no.y + self.z * no.z
14
15     def cross(self, no):
16        return Points(self.y * no.z - self.z * no.y, self.z * no.x - self.x * no.z, self.x * no.y - self.y * no.x)
17
18     def absolute(self):
19        return pow((self.x ** 2 + self.y ** 2 + self.z ** 2), 0.5)
20
```

	Input	Expected	Got	
✓	0 4 5 1 7 6 0 5 9 1 7 2	8.19	8.19 ✓	
✓	5 8.8 9 4 -1 3 7 8.7 3.3 4.4 5.1 6.3	5.69	5.69 ✓	

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 26

Đúng

Hàm `map()` trong Python sử dụng một hàm lén tất cả các phần tử của một mảng. Hàm này nhận vào hai tham số, tham số thứ nhất là hàm được thực thi, tham số thứ hai là mảng các phần tử.

Ví dụ, cho một danh sách tên các sinh viên, ta cần trả về một danh sách là độ dài của tên các sinh viên đó.

```
>> print (list(map(len, ['Tina', 'Raj', 'Tom'])))
[4, 3, 3]
```

`Lambda` là một hàm biểu thức thường được sử dụng như một hàm nội tuyến. Nó chứng minh rất tiện dụng trong lập trình hàm và lập trình GUI.

```
>> sum = lambda a, b, c: a + b + c
>> sum(1, 2, 3)
6
```

### Đề bài

Hãy tạo ra một mảng chứa  $N$  số Fibonacci đầu tiên và in ra lập phương của chúng.

#### Đầu vào

Dữ liệu vào từ bàn phím gồm một dòng chứa số nguyên dương  $N$  ( $N \leq 10$ ).

#### Đầu ra

In ra màn hình một dòng chứa lập phương của  $N$  số Fibonacci đầu tiên.

For example:

Input	Result
4	[1, 1, 8, 27]

Answer: (penalty regime: 0 %)

```
1 N = int(input())
2 fibonacci = [0,1]
3 for i in range(2, N+1):
4     fibonacci.append(fibonacci[i-1] + fibonacci[i-2])
5 fibonacci_cube = [num**3 for num in fibonacci]
6 fibonacci_cube = fibonacci_cube[1:] # Remove the first element
7 print(fibonacci_cube)
8
9
10
11
```

	Input	Expected	Got	
✓	4	[1, 1, 8, 27]	[1, 1, 8, 27]	✓
✓	5	[1, 1, 8, 27, 125]	[1, 1, 8, 27, 125]	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 27

Đúng

Hai hoặc nhiều mảng có thể nối lại với nhau bằng cách sử dụng hàm `concatenate`.

```
import numpy

array_1 = numpy.array([1,2,3])
array_2 = numpy.array([4,5,6])
array_3 = numpy.array([7,8,9])
print numpy.concatenate((array_1, array_2, array_3))

#Output
[1 2 3 4 5 6 7 8 9]
```

Nếu các mảng này là mảng nhiều chiều ta có thể chọn chiều nào sẽ là chiều được nối lại với nhau.

```
import numpy

array_1 = numpy.array([[1,2,3],[0,0,0]])
array_2 = numpy.array([[0,0,0],[7,8,9]])
print numpy.concatenate((array_1, array_2), axis = 1)

#Output
[[1 2 3 0 0 0]
 [0 0 0 7 8 9]]
```

### Đề bài

Cho hai mảng số nguyên có kích thước  $N \times P$  và  $M \times P$ . Nhiệm vụ của bạn là nối hai mảng này lại ở chiều số 0.

### Đầu vào

Đầu vào từ bàn phím. Dòng đầu tiên chứa 3 số nguyên  $M, N, P$  cách nhau bởi một dấu cách.  $N$  dòng tiếp theo mỗi dòng chứa  $P$  số nguyên cách nhau bởi một dấu cách là các phần tử của mảng thứ nhất.  $M$  dòng tiếp theo mỗi dòng chứa  $P$  số nguyên cách nhau bởi một dấu cách là các phần tử của mảng thứ hai.

### Đầu ra

In ra bàn phím mảng có kích thước  $(N + M) \times P$  là mảng sau khi đã nối.

For example:

Input	Result
4 3 2	[[1 2]
1 2	[1 2]
1 2	[1 2]
1 2	[1 2]
1 2	[3 4]
3 4	[3 4]
3 4	[3 4]
3 4	

Answer: (penalty regime: 0 %)

```
1 import numpy
2
3 n,m,p = map(int,input().strip().split())
4 A = numpy.array([[int(j) for j in input().strip().split()] for i in range(n)])
5 B = numpy.array([[int(j) for j in input().strip().split()] for i in range(m)])
6 print(numpy.concatenate((A,B),axis=0))
```

	<b>Input</b>	<b>Expected</b>	<b>Got</b>	
✓	4 3 2 1 2 1 2 1 2 1 2 1 2 3 4 3 4 3 4	[[1 2] [1 2] [1 2] [1 2] [1 2] [3 4] [3 4] [3 4]]	[[1 2] [1 2] [1 2] [1 2] [1 2] [3 4] [3 4] [3 4]]	✓
✓	3 3 2 1 2 1 2 1 2 1 2 3 4 3 4 3 4	[[1 2] [1 2] [1 2] [1 2] [3 4] [3 4] [3 4]]	[[1 2] [1 2] [1 2] [1 2] [3 4] [3 4] [3 4]]	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 28

Đúng

Hàm filter() sử dụng một hàm trả về giá trị True hoặc False lên một dãy số và trả về một danh sách các phần tử thuộc dãy số mà hàm được sử dụng trả về giá trị True.

Ví dụ, ta tạo một danh sách chứa các số nguyên từ 0 đến 9.

Sau đó, ta lấy ra những phần tử có giá trị lớn hơn 10 và nhỏ hơn 80.

Đề bài

Cho một danh sách  $N$  email. Bạn hãy in ra danh sách các email hợp lệ được sắp xếp theo thứ tự từ điển.

Một email là hợp lệ nếu nó thỏa mãn các điều kiện sau:

- Email phải có dạng : {ten\_nguo\_dung}@{ten\_website}.(duoi\_mo\_rong);
- ten\_nguo\_dung có thể chứa các kí tự là chữ cái, chữ số, dấu gạch ngang (-) và dấu gạch dưới (\_);
- ten\_website chỉ chứa chữ cái và chữ số;
- duoi\_mo\_rong có độ dài không vượt quá 3.

Đầu vào

Dữ liệu vào từ bàn phím gồm  $N + 1$  dòng. Dòng thứ nhất chứa số nguyên dương  $n$ .  $N$  dòng tiếp theo, mỗi dòng chứa một xâu biểu diễn một email.

Đầu ra

In ra màn hình danh sách các email hợp lệ được sắp xếp theo thứ tự từ điển.

For example:

Input	Result
2 vinh-ls@codepower.vn longtq@codepower.com	['longtq@codepower.com', 'vinh-ls@codepower.vn']

Answer: (penalty regime: 0 %)

```
1 import re
2
3 def validate_email(email):
4     pattern = r'^[a-zA-Z0-9_-]+@[a-zA-Z0-9]+\.[a-zA-Z]{1,3}$'
5     return re.match(pattern, email) is not None
6
7 # Get the number of emails
8 N = int(input())
9
10 # Initialize a list to store valid emails
11 valid_emails = []
12
13 # Iterate through the emails
14 for _ in range(N):
15     email = input()
16     if validate_email(email):
17         valid_emails.append(email)
18
19 # Sort the valid emails in alphabetical order
20 valid_emails.sort()
```

	Input	Expected	Got
✓	2 vinh-ls@codepower.vn longtq@codepower.com	['longtq@codepower.com', 'vinh-ls@codepower.vn']	['longtq@codepower.com', 'vinh-ls@codepower.vn']
✓	3 lara@codepower.com brian-23@codepower.com britts_54@codepower.com	['brian-23@codepower.com', 'britts_54@codepower.com', 'lara@codepower.com']	['brian-23@codepower.com', 'britts_54@codepower.com', 'lara@codepower.com']
✓	2 harsh@gmail iota_98@codepower.com	['iota_98@codepower.com']	['iota_98@codepower.com']

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 29

Đúng

Hàm `reduce()` lặp lượt áp dụng một hàm lên hai phần tử liên tiếp thuộc một danh sách từ trái sang phải để giảm danh sách đó về thành một giá trị.

Ví dụ, ta có một danh sách các số nguyên [1, 2, 3] và ta muốn tìm tổng các phần tử trong danh sách này.

```
>>> reduce(lambda x, y : x + y,[1,2,3])  
6
```

### Đề bài

Cho  $N$  phân số có tử số và mẫu số là các số nguyên. Tích tích của các phân số này.

### Đầu vào

Dữ liệu vào từ bàn phím gồm  $N + 1$  dòng. Dòng đầu tiên chứa số nguyên  $N$ .  $N$  dòng tiếp theo, mỗi dòng chứa 2 số nguyên cách nhau bởi một dấu cách là tử số và mẫu số của từng phân số.

### Đầu ra

In ra tích của các phân số trên dưới dạng phân số tối giản.

**Answer:** (penalty regime: 0 %)

```
1 from fractions import Fraction  
2 from functools import reduce  
3  
4 def product(fracs):  
5     t = reduce(lambda x,y : x * y, fracs)  
6     return t.numerator, t.denominator  
7  
8 if __name__ == '__main__':  
9     fracs = []  
10    for _ in range(int(input())):  
11        fracs.append(Fraction(*map(int, input().split())))  
12    result = product(fracs)  
13    print(*result)
```

	Input	Expected
✓	3 1 2 3 4 10 6	5 8
✓	21 684025282 932952183 349232934 278093065 778706161 742081687 374870211 874099626 849763633 211127281 566205501 508794028 814324820 443967409 402053385 120666811 261003218 519477752 134531387 354698174 790584924 879493679 520320652 475787580 480239279 930995305 465217671 718416348 270497402 819933293 216292652 688271440 76664269 6180883 802885525 457264574 194701183 725188300 63039953 241525609 576817633 832775273	747902331715715337881396435553672252898266314836662443238476051886467002406120347859760369550907496110583990580348

	Input	Expected
✓	44 375468635 987372137 950344768 254127968 931267940 752426822 299378749 104449522 988976655 207212805 554621848 95866287 611950359 750498230 452292779 385604851 902567758 424804876 459908304 678198353 429499173 900063044 484771040 141396209 767644590 868356601 963506573 628274031 17525296 135368811 751402882 269888345 527951951 33494700 890664542 83432735 94639301 903306636 312730609 963839832 579885681 650829059 142759020 50088950 31393026 489094467 436284731 128303059 108909603 580608955 921993295 846795960 518549629 567953655 825247248 481104810 445589293 536748940 353612111 71498328 22018601 137913232 772835111 58613521 215432052 109069122 321182563 134426361 151052683 416772427 750007722 443754342 116003610 343478950 207648092 73534447 883332282 404190441 530945004 591107521 277733003 22111421 111981816 882706740 835145704 340730336 463354250 612154306	164945660787156744972502827396672865972799577790953878368237418286201904669863005360251875403854931977561449790742

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

### Câu Hỏi 30

Đúng

Package Numpy trong python giúp chúng ta xử lý những dữ liệu kiểu mảng, ma trận có nhiều phần tử.

Để sử dụng package NumPy, ta cần import bằng câu lệnh sau:

```
import numpy
```

Một mảng trong numpy là một dãy các giá trị và tất cả các phần tử trong mảng phải cùng một kiểu.

```
import numpy
```

```
a = numpy.array([1,2,3,4,5])
print a[1]          #2

b = numpy.array([1,2,3,4,5],float)
print b[1]          #2.0
```

Ở ví dụ trên, hàm `numpy.array()` được dùng để chuyển một list sang một mảng numpy. Tham số thứ hai (`float`) được dùng để đặt kiểu dữ liệu cho các phần tử trong mảng.

#### Đề bài

Cho một dãy các số cách nhau bởi một dấu cách. In ra mảng theo thứ tự ngược của các số đã cho dưới kiểu `float`.

#### Đầu vào

Dữ liệu vào từ bàn phím gồm một dòng duy nhất chứa các số cách nhau bởi một dấu cách.

#### Đầu ra

In ra màn hình mảng theo thứ tự ngược của các số đã cho dưới kiểu `float`.

For example:

Input	Result
1 2 3 4 -8 -10	[ -10. -8. 4. 3. 2. 1.]

Answer: (penalty regime: 0 %)

Reset answer

```
1 import numpy
2
3 def rev(arr):
4     arr = numpy.array(arr, dtype=float)
5     return numpy.flipud(arr)
6
7 arr = input().strip().split(' ')
8 result = rev(arr)
9 print(result)
10
```

	Input	Expected	Got	
✓	1 2 3 4 -8 -10	[ -10. -8. 4. 3. 2. 1.]	[ -10. -8. 4. 3. 2. 1.]	✓
✓	1.1 -1.2	[ -1.2 1.1]	[ -1.2 1.1]	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 31

Đúng

### shape

`shape` biểu diễn kích thước của mảng và có thể dùng để thay đổi kích thước của mảng.

#### (a) Dùng `shape` để lấy kích thước của mảng

```
import numpy

my_1D_array = numpy.array([1, 2, 3, 4, 5])
print my_1D_array.shape      #(5,) -> 5 rows and 0 columns

my_2D_array = numpy.array([[1, 2],[3, 4],[6,5]])
print my_2D_array.shape     #(3, 2) -> 3 rows and 2 columns/code>
```

#### (b) Dùng `shape` để thay đổi kích thước mảng

```
import numpy

change_array = numpy.array([1,2,3,4,5,6])
change_array.shape = (3, 2)
print change_array

#Output
[[1 2]
 [3 4]
 [5 6]]/code>
```

### reshape

`reshape` dùng để tạo ra một bản sao của một mảng có sẵn với một kích thước khác mà không làm thay đổi kích thước và các giá trị của mảng gốc.

```
import numpy

change_array = numpy.array([1,2,3,4,5,6])
change_array.shape = (3, 2)
print change_array

#Output
[[1 2]
 [3 4]
 [5 6]]
```

### Đề bài

Cho một dãy số có 9 phần tử. Bạn hãy chuyển nó thành một mảng  $3 \times 3$  của numpy.

#### Đầu vào

Dữ liệu vào từ bàn phím gồm một dòng duy nhất chứa 9 số nguyên cách nhau bởi một dấu cách.

#### Đầu ra

In ra màn hình mảng  $3 \times 3$  của numpy.

For example:

Input	Result
1 2 3 4 5 6 7 8 9	[[1 2 3]  [4 5 6]  [7 8 9]]

Answer: (penalty regime: 0 %)

```
1 import numpy
2 arr = input().strip().split(' ')
3 change_array = numpy.array(arr,dtype=int)
4 change_array = change_array.reshape(3, 3)
5 print( change_array )
6
```

	Input	Expected	Got	
✓	1 2 3 4 5 6 7 8 9	[[1 2 3] [4 5 6] [7 8 9]]	[[1 2 3] [4 5 6] [7 8 9]]	✓
✓	1 2 3 4 4 6 7 2 9	[[1 2 3] [4 4 6] [7 2 9]]	[[1 2 3] [4 4 6] [7 2 9]]	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

## Câu Hỏi 32

Đúng

### transpose

Ta có thể tạo ra chuyển vị của mảng bằng cách sử dụng hàm `numpy.transpose`.

Hàm này sẽ tạo ra một mảng mới và không làm thay đổi giá trị mảng gốc.

```
import numpy

my_array = numpy.array([[1,2,3],
                      [4,5,6]])
print numpy.transpose(my_array)

#Output
[[1 4]
 [2 5]
 [3 6]]
```

### flatten

Hàm `flatten` tạo ra một bản sao của một mảng dưới dạng mảng một chiều

```
import numpy

my_array = numpy.array([[1,2,3],
                      [4,5,6]])
print my_array.flatten()

#Output
[1 2 3 4 5 6]
```

### Đề bài

Cho một mảng số nguyên có kích thước  $M \times N$ . In ra kết quả của hàm `transpose` và hàm `flatten` của mảng này.

### Đầu vào

Dữ liệu vào từ bàn phím. Dòng đầu tiên chứa 2 số nguyên  $M, N$  cách nhau bởi một dấu cách.  $M$  dòng tiếp theo, mỗi dòng chứa  $N$  số nguyên cách nhau bởi một dấu cách biểu diễn các phần tử của mảng.

### Đầu ra

In ra màn hình In ra kết quả của hàm `transpose` và hàm `flatten` của mảng đã nhập.

### For example:

Input	Result
2 2	[[1 3]]
1 2	[2 4]]
3 4	[1 2 3 4]

**Answer:** (penalty regime: 0 %)

```
1 import numpy as np
2 M, N = map(int, input().split())
3 array = []
4 for _ in range(M):
5     row = list(map(int, input().split()))
6     array.append(row)
7 print(np.transpose(array))
8 print(np.array(array).flatten())
```

	Input	Expected	Got	
✓	2 2 1 2 3 4	[[1 3] [2 4]] [1 2 3 4]	[[1 3] [2 4]] [1 2 3 4]	✓
✓	3 2 1 2 3 4 5 6	[[1 3 5] [2 4 6]] [1 2 3 4 5 6]	[[1 3 5] [2 4 6]] [1 2 3 4 5 6]	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

### Câu Hỏi 33

Đúng

#### zeros

Hàm `zeros` trả về một mảng chỉ chứa các phần tử có giá trị bằng 0.

```
import numpy

print numpy.zeros((1,2))           #Default type is float
#Output : [[ 0.  0.]]

print numpy.zeros((1,2), dtype = numpy.int) #Type changes to int
#Output : [[0 0]]
```

#### ones

Hàm `ones` trả về một mảng chỉ chứa các phần tử có giá trị bằng 1.

```
import numpy

print numpy.ones((1,2))           #Default type is float
#Output : [[ 1.  1.]]

print numpy.ones((1,2), dtype = numpy.int) #Type changes to int
#Output : [[1 1]]
```

#### Đề bài

Cho 2 số nguyên  $N, M$ , sử dụng hàm `zeros` và `ones` để in ra hai mảng có kích thước  $N \times M$  chỉ gồm các phần tử có giá trị bằng 0 và các phần tử có giá trị bằng 1.

#### Đầu vào

Đầu vào từ màn hình gồm 2 số nguyên  $N, M$  cách nhau bởi một dấu cách.

#### Đầu ra

In ra màn hình hai mảng có kích thước  $N \times M$ . Mảng đầu tiên chỉ gồm các phần tử có giá trị bằng 0, mảng thứ 2 chỉ gồm các phần tử bằng 1.

For example:

Input	Result
3 3	<pre>[[0 0 0]  [0 0 0]  [0 0 0]]  [[1 1 1]  [1 1 1]  [1 1 1]]</pre>

Answer: (penalty regime: 0 %)

```
1 import numpy
2
3 x,y = map(int, input().split())
4
5
6 # x = int(input())
7 # y = int(input())
8 print(numpy.zeros((x,y), dtype = int))
9 print(numpy.ones((x,y), dtype = int))
```

	<b>Input</b>	<b>Expected</b>	<b>Got</b>	
✓	3 3	<code>[[0 0 0] [0 0 0] [0 0 0]] [[1 1 1] [1 1 1] [1 1 1]]</code>	<code>[[0 0 0] [0 0 0] [0 0 0]] [[1 1 1] [1 1 1] [1 1 1]]</code>	✓
✓	3 2	<code>[[0 0] [0 0] [0 0]] [[1 1] [1 1] [1 1]]</code>	<code>[[0 0] [0 0] [0 0]] [[1 1] [1 1] [1 1]]</code>	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

◀ [Python + Google Colaboratory] Làm quen với Python và Google Colaboratory

Chuyển tới...

