Python-part2

May 17, 2024

1 More Python

1.1 Input Reading

[1]:
```
name = input()
print("Hello", name, "!")
```

```
 Zoe Paraskevopoulou
```

```
Hello Zoe Paraskevopoulou !
```

[2]:
```
n = int(input())
print("The number is: ", n)
```

```
 42
```

```
The number is:  42
```

[3]:
```
"42 is the answer to everything".split()
```

[3]:
```
['42', 'is', 'the', 'answer', 'to', 'everything']
```

[4]:
```
"42, is, the, answer, to, everything".split(", ")
```

[4]:
```
['42', 'is', 'the', 'answer', 'to', 'everything']
```

[5]:
```
name = input()
words = name.split()
first = words[0]
last = words[1]
print("First:", first, "Last:", last)
```

```
 Zoe Paraskevopoulou
```

```
First: Zoe Last: Paraskevopoulou
```

[6]:
```
first,last = input().split()
print("First:", first, "Last:", last)
```

```
 Zoe Paraskevopoulou
```

```
First: Zoe Last: Paraskevopoulou
```

[7]:
```
x, y = [2,17]
print(x,y)
```

```
2 17
```

[8]:
```
x,y,z=[2,17,42]
```

[9]:
```
# In C: printf("The answer to everything is %d", 42)
"The answer to everything is {}".format(42)
```

[9]:
```
'The answer to everything is 42'
```

[10]:
```
"The answer to everything is {} and {}".format(42, 17)
```

[10]:
```
'The answer to everything is 42 and 17'
```

[11]:
```
"The answer to everything is {} and {}".format(42)
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[11], line 1
----> 1 "The answer to everything is {} and {}".format(42)

IndexError: Replacement index 1 out of range for positional args tuple
```

[12]:
```
"The answer to everything is {} and".format(42, 17)
```

[12]:
```
'The answer to everything is 42 and'
```

[13]:
```
"The answer to everything is not {}".format("3.14")
```

[13]:
```
'The answer to everything is not 3.14'
```

[14]:
```
"The answer to everything  is not {:0.2f}".format(3.14159)
```

[14]:
```
'The answer to everything  is not 3.14'
```

[15]:
```
first,last = input().split()
print("First: {} Last: {}".format(first, last))
```

```
 Zoe Paraskevopoulou
```

```
First: Zoe Last: Paraskevopoulou
```

1.2 Exercise 1

Read two numbers from the input and print their sum.

[ ]:
```
# TODO
```

1.2.1 Solution

[18]:
```
x,y = input().split()
print(int(x)+int(y))
```

```
 21 21
```

```
42
```

1.3 Map

[20]:
```
def f(x): return 2*x
```

[21]:
```
map(f, [1,2,3,4])
```

[21]:
```
<map at 0x1093adf30>
```

[22]:
```
type(map(f, [1,2,3,4]))
```

[22]:
```
map
```

[23]:
```
i = map(f, [1,2,3,4])
print(next(i))
print(next(i))
```

```
2
4
```

[24]:
```
next(i)
```

[24]:
```
6
```

[25]:
```
next(i)
```

[25]:
```
8
```

[26]:
```
next(i)
```

```
---------------------------------------------------------------------------
StopIteration                             Traceback (most recent call last)
Cell In[26], line 1
----> 1 next(i)

StopIteration:
```

[27]:
```
list(map(f, [1,2,3,4]))
```

[27]:
```
[2, 4, 6, 8]
```

[28]:
```
for y in map(f, [1,2,3,4]): print(y)
```

```
2
4
6
8
```

[29]:
```
list(map(f, {1,2,3,4,100}))
```

[29]:
```
[2, 4, 6, 200, 8]
```

1.4 Anonymous functions

[30]:
```
list(map(lambda x: 2*x, [1,2,3,4]))
```

[30]:
```
[2, 4, 6, 8]
```

1.5 Exercise 2

Write an **one-liner program** that reads two numbers from the input and prints their sum.

[ ]:
```
# TODO
```

1.5.1 Solution

[32]:
```
print(sum(map(int,input().split())))
```

```
 21 10 11
```

```
42
```

1.6 Exercise 3

Πρόβλημα “exclude”

Δίνονται δύο ακολουθίες $a(1), ..., a(N)$ και $b(1), ..., b(M)$, αποτελούμενες από φυσικούς αριθμούς. Ζητείται να βρεθούν οι αριθμοί της πρώτης ακολουθίας που δεν ανήκουν στη δεύτερη.

Δεδομένα εισόδου

Η πρώτη γραμμή της εισόδου θα περιέχει δύο αριθμούς χωρισμένους μεταξύ τους με ένα κενό διάστημα: τις τιμές των N και M. Η δεύτερη γραμμή της εισόδου θα περιέχει N φυσικούς αριθμούς, που αντιστοιχούν στους όρους της πρώτης ακολουθίας, χωρισμένους ανά δύο με ένα κενό διάστημα. Ομοίως, η τρίτη γραμμή της εισόδου θα περιέχει τους M φυσικούς αριθμούς της δεύτερης ακολουθίας. Να θεωρήσετε ως δεδομένο ότι η είσοδος θα είναι έγκυρη και ότι οι αριθμοί δε θα υπερβαίνουν τα όρια που αναγράφονται παρακάτω.

Δεδομένα εξόδου

Η έξοδος πρέπει να αποτελείται από τόσες γραμμές όσοι όροι της πρώτης ακολουθίας δεν εμφανίζονται στη δεύτερη. Κάθε γραμμή θα περιέχει ακριβώς έναν όρο της πρώτης ακολουθίας. Η σειρά εμφάνισης των όρων θα είναι η ίδια με τη σειρά που αυτοί εμφανίζονται στην είσοδο.

Περιορισμοί

- $1 \leq N, M \leq 1.000.000$
- $0 \leq a(i), b(j) \leq 1.000.000$

Παράδειγμα εισόδου 1

```
5 5
4 9 5 1 10
5 7 2 4 1
```

Παράδειγμα εξόδου 1

```
9
10
```

Παράδειγμα εισόδου 2

```
10 7
5 17 15 11 13 10 5 1 4 9
14 1 8 11 19 13 9
```

[ ]:
```
# TODO
```

1.6.1 Solution 1

[34]:
```
N, M = map(int, input().split())
A = map(int, input().split())
B = list(map(int, input().split()))
for a in A:
    if a not in B:
        print(a)
```

```
5 5
4 9 5 1 10
5 7 2 4 1
```

```
9
10
```

1.6.2 Solution 2

[35]:
```
N, M = map(int, input().split())
A = map(int, input().split())
B = set(map(int, input().split()))
for a in A:
    if a not in B:
        print(a, end=' ')
```

```
5 5
4 9 5 1 10
5 7 2 4 1
```

```
9 10
```

1.6.3 Solution 3

Print the answer in one line:

[71]:
```
N, M = map(int, input().split())
A = map(int, input().split())
B = list(map(int, input().split()))
for a in A:
    if a not in B:
        print(a, end=' ')
```

```
5 5
4 9 5 1 10
5 7 2 4 1
```

```
9 10
```

[53]:
```
N, M = map(int, input().split())
A = list(map(int, input().split())) # use a list because maps are consumed and␣
 ↪we want to reuse A
B = set(map(int, input().split()))
```

```
5 5
4 9 5 1 10
5 7 2 4 1
```

[54]:
```
L = []
for a in A:
    if a not in B:
```

```
        L.append(a)
```

[55]:
```
L
```

[55]: [9, 10]

[56]:
```
print(*L, sep=' ')
```

```
9 10
```

[57]:
```
L = [a for a in A if a not in B]
print(L)
```

```
[9, 10]
```

[58]:
```
" and ".join(["dogs","cats"])
```

[58]: 'dogs and cats'

[59]:
```
" ".join(["dogs","cats"])
```

[59]: 'dogs cats'

[60]:
```
" ".join(map(str,L))
```

[60]: '9 10'

[61]:
```
L = [str(a) for a in A if a not in B]
print(" ".join(L))
```

```
9 10
```

[62]:
```
print(" ".join([str(a) for a in A if a not in B]))
```

```
9 10
```

[63]:
```
type(str(a) for a in A if a not in B)
```

[63]: generator

[64]:
```
print(" ".join(str(a) for a in A if a not in B))
```

```
9 10
```

[65]:
```
L = (1,2,3)
print(L)
```

```
(1, 2, 3)
```

[66]:
```
print(*L)
```

```
1 2 3
```

[67]:
```
print(*(str(a) for a in A if a not in B))
```

```
9 10
```

finally...

[68]:
```
N, M = map(int, input().split())
A = map(int, input().split())
B = set(map(int, input().split()))
print(*(str(a) for a in A if a not in B))
```

```
 5 5
 4 9 5 1 10
 5 7 2 4 1
```

```
9 10
```

1.7 The Python Standard Library

Here!

1.7.1 Random

Useful for fuzzing (e.g., generate random test cases and test that problem set solutions produce the sae output as brute force solutions).

[72]:
```
import random
```

[73]:
```
random.randrange(10)
```

[73]: 9

[74]:
```
random.randrange(10)
```

[74]: 9

[75]:
```
L = [1,2,3,4,5]
```

[76]:
```
random.shuffle(L)
```

[77]:
```
L
```

[77]: [4, 2, 1, 5, 3]

Shrinking https://courses.softlab.ntua.gr/q2a/722/extra-test-cases

1.7.2 Itertools

[78]:
```
import itertools
```

[79]:
```
animals = ["cats", "dogs", "goats", "bunnies"]
```

[80]:
```
for animal1, animal2 in itertools.combinations(animals,2):
    print(animal1, "love", animal2)
```

```
cats love dogs
cats love goats
cats love bunnies
dogs love goats
dogs love bunnies
goats love bunnies
```

[81]:
```
for c in itertools.permutations(animals):
    print(c)
```

```
('cats', 'dogs', 'goats', 'bunnies')
('cats', 'dogs', 'bunnies', 'goats')
('cats', 'goats', 'dogs', 'bunnies')
('cats', 'goats', 'bunnies', 'dogs')
('cats', 'bunnies', 'dogs', 'goats')
('cats', 'bunnies', 'goats', 'dogs')
('dogs', 'cats', 'goats', 'bunnies')
('dogs', 'cats', 'bunnies', 'goats')
('dogs', 'goats', 'cats', 'bunnies')
('dogs', 'goats', 'bunnies', 'cats')
('dogs', 'bunnies', 'cats', 'goats')
('dogs', 'bunnies', 'goats', 'cats')
('goats', 'cats', 'dogs', 'bunnies')
('goats', 'cats', 'bunnies', 'dogs')
('goats', 'dogs', 'cats', 'bunnies')
('goats', 'dogs', 'bunnies', 'cats')
('goats', 'bunnies', 'cats', 'dogs')
('goats', 'bunnies', 'dogs', 'cats')
('bunnies', 'cats', 'dogs', 'goats')
('bunnies', 'cats', 'goats', 'dogs')
('bunnies', 'dogs', 'cats', 'goats')
('bunnies', 'dogs', 'goats', 'cats')
('bunnies', 'goats', 'cats', 'dogs')
('bunnies', 'goats', 'dogs', 'cats')
```