UNIVERSIDAD GALILEO INSTITUTO DE INVESTIGACIÓN DE OPERACIONES MAESTRÍA EN DATA SCIENCE ESTADÍSTICA APLICADA A LA CIENCIA DE DATOS I



TAREA 4

EDGAR RUBÉN ROJAS OVANDO

CARNÉ: 21000567

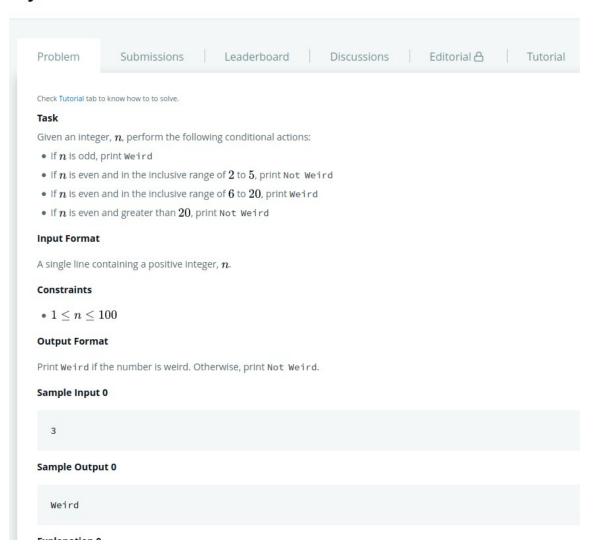
CIUDAD DE GUATEMALA, GUATEMALA, MARZO DE 2021

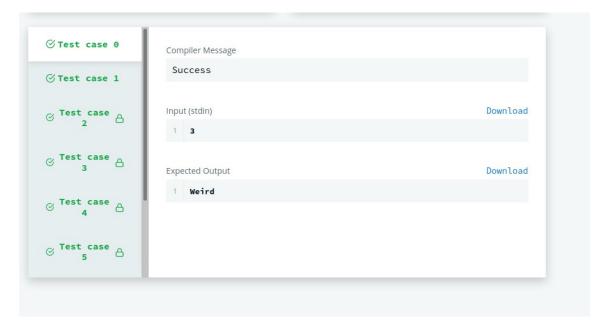
USUARIO: rbencho

PY IF ELSE PROBLEM

Practice > Python > Introduction > Python If-Else

Python If-Else *



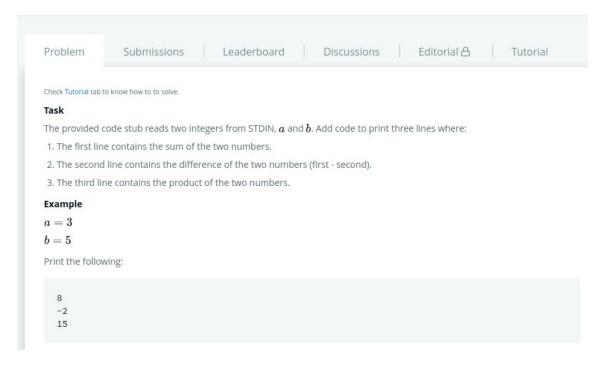


⊗ Accepted		10.0	Python 3	a minute ago	View Results
RESULT		SCORE	LANGUAGE	TIME	
Problem	Submissions	Leaderbo	pard Discuss	sions Editorial 🖰	Tutorial

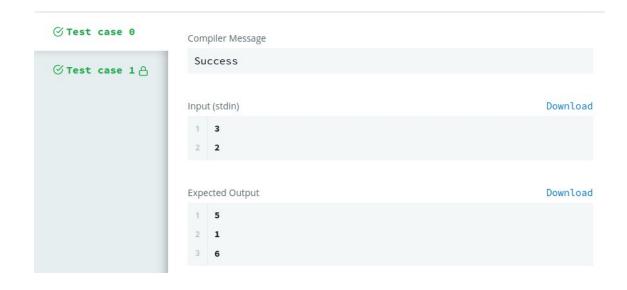
PYTHON ARITHMETIC OPERATORS

Practice > Python > Introduction > Arithmetic Operators

Arithmetic Operators *



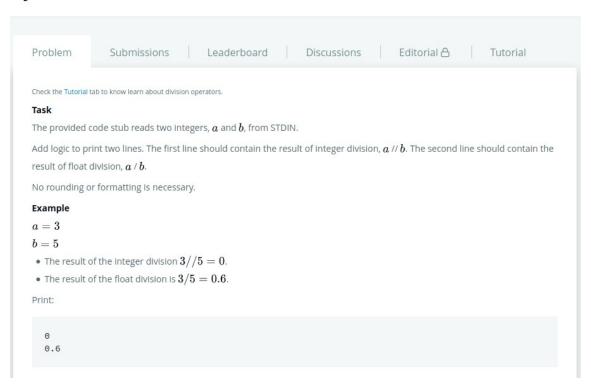
```
1 \vee def operar(a,b):
 2
 4
         diferencia = a - b
 5
         producto = a * b
 6
         print(suma)
         print(diferencia)
 8
 9
         print(producto)
10
11 v if __name__ == '__main__':
         a = int(input())
12
         b = int(input())
13
         operar(a,b)
14
15
16
```

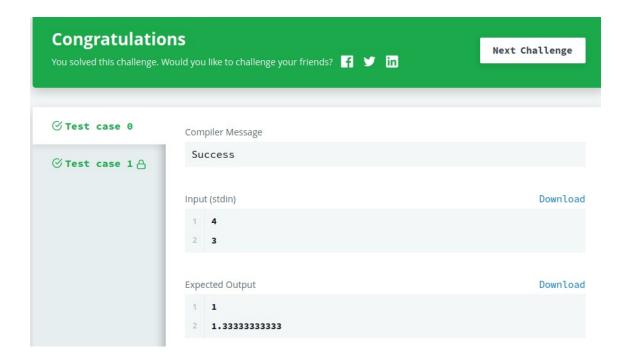


⊗ Accepted		10.0	Python 3	a few seconds ago	View Results
RESULT		SCORE	LANGUAGE	TIME	
Problem	Submissions	Leaderbo	oard Discussi	ions Editorial 🖰	Tutorial

Python Division

Python: Division *



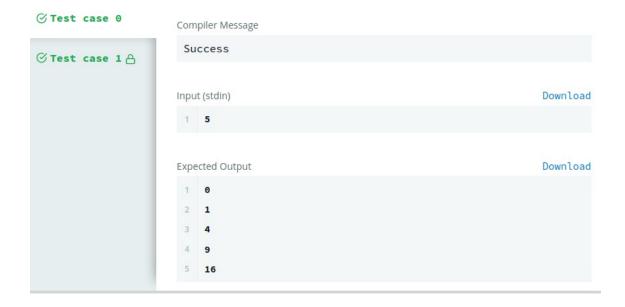


Problem	Submissions	Leaderb	oard Discussi	ons Editorial 🖰	Tutorial
RESULT		SCORE	LANGUAGE	TIME	
⊗ Accepted		10.0	Python 3	a few seconds ago	View Results

Loops

Loops *





⊗Accepted		10.0	Python 3	a minute ago	View Results
RESULT		SCORE	LANGUAGE	TIME	
Problem	Submissions	Leaderb	oard Discuss	ions Editorial 🖰	Tutorial

Write a Function

Write a function *

Problem Submissions Leaderboard Discussions Editorial △

An extra day is added to the calendar almost every four years as February 29, and the day is called a leap day. It corrects the calendar for the fact that our planet takes approximately 365.25 days to orbit the sun. A leap year contains a leap day.

In the Gregorian calendar, three conditions are used to identify leap years:

- The year can be evenly divided by 4, is a leap year, unless:
 - o The year can be evenly divided by 100, it is NOT a leap year, unless:
 - \blacksquare The year is also evenly divisible by 400. Then it is a leap year.

This means that in the Gregorian calendar, the years 2000 and 2400 are leap years, while 1800, 1900, 2100, 2200, 2300 and 2500 are NOT leap years. Source

Task

Given a year, determine whether it is a leap year. If it is a leap year, return the Boolean True, otherwise return False.

Note that the code stub provided reads from STDIN and passes arguments to the is_leap function. It is only necessary to

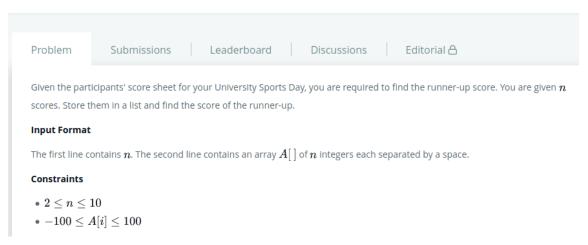
```
def is_leap(year):
         leap = False
         if(year % 4 == 0):
             if(year % 100 == 0):
                  if (year % 400 ==0):
                      leap = True
                      return leap
L0
                 leap = False
11
                 return leap
L2
L3
             leap = True
             return leap
L4
L5
         return leap
L7 > year = int(input()) --
```





Find Second Maximum Number in a List

Find the Runner-Up Score! 🖈

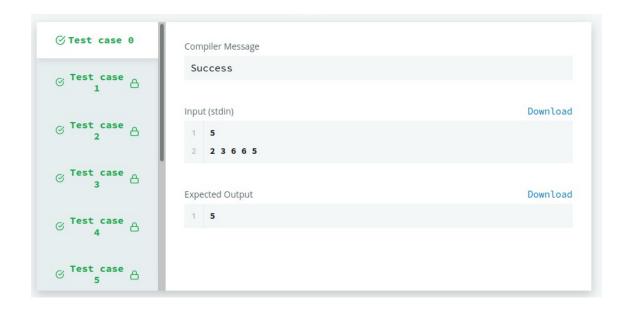


```
vif __name__ == '__main__':
    n = int(input())
    arr = list(map(int, input().split()))

maxNumber = max(arr)
    lista = []

v    for i in arr:
        if(i < maxNumber):
             lista.append(i)

print(max(lista))</pre>
```



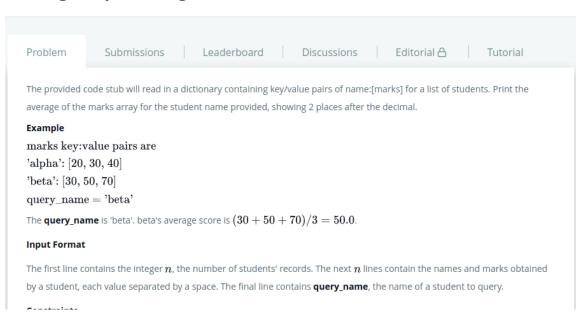
Find the Runner-Up Score! 🖈

Rank: 6

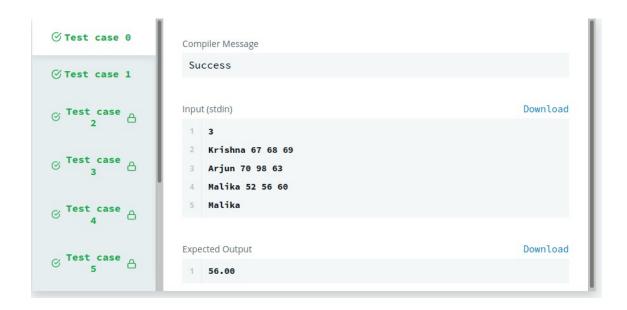
⊗ Accepted		10.0	Python 3	2 minutes ago	View Results
RESULT		SCORE	LANGUAGE	TIME	
Problem	Submissions	Leaderb	oard Discuss	ions Editorial 🛆	

Finding the porcentage

Finding the percentage *



Ran

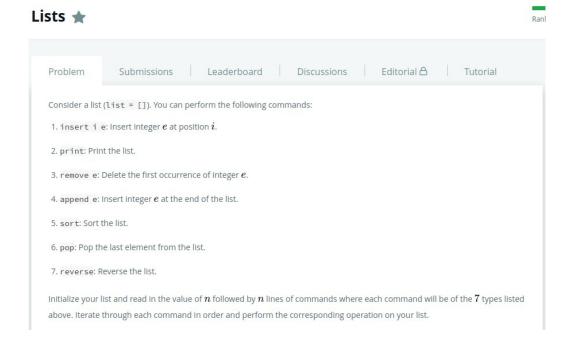


Finding the percentage *



Rank:

Python Lists



```
__name__ == '__main__':
        N = int(input())
         arreglo = []
6 ~
         for i in range(N):
             entrada = input().split()
8 ~
             for i in range(1,len(entrada)):
                 entrada[i] = int(entrada[i])
11 ~
             if entrada[0] == "append":
                 arreglo.append(entrada[1])
             elif entrada[0] == "extend":
14
                 arreglo.extend(entrada[1:])
             elif entrada[0] == "insert":
                 arreglo.insert(entrada[1],entrada[2])
19 ~
             elif entrada[0] == "pop":
                 arreglo.pop()
             elif entrada[0] == "index":
                 print(arreglo.index(entrada[1]))
             elif entrada[0] == "count":
24
                 print(arreglo.count(entrada[1]))
25 ~
             elif entrada[0] == "sort":
26
             elif entrada[0] == "reverse":
                arreglo.reverse()
29 ~
             elif entrada[0] == "print":
32
```

```
P1 1115
⊘Test case 0
                            remove 6
                            append 9
⊘Test case 1 A
                            append 1
                            sort
                            print
                            pop
                            reverse
                            print
                       Expected Output
                                                                                     Download
                        [6, 5, 10]
                         2 [1, 5, 9, 10]
                            [9, 5, 1]
```

Lists 🖈

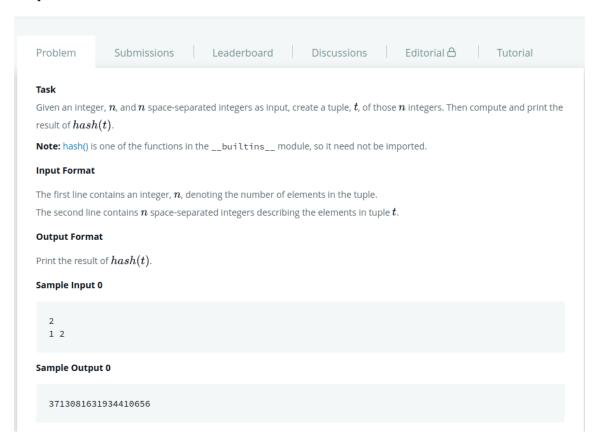
 Problem
 Submissions
 Leaderboard
 Discussions
 Editorial △
 Tutorial

 RESULT
 SCORE
 LANGUAGE
 TIME

 ⊗ Accepted
 10.0
 Python 3
 a few seconds ago
 View Results

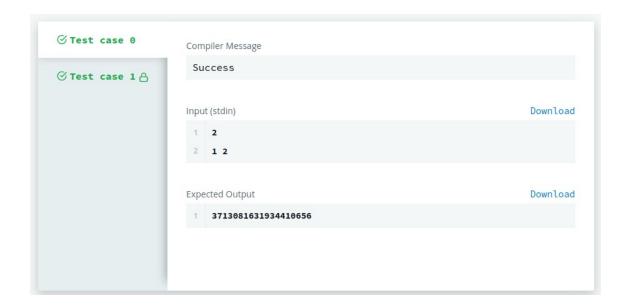
Python Tuples

Tuples ★



```
import builtins

i
```



Tuples ★



DefaultDict tutorial

DefaultDict Tutorial *

Ran

Problem Submissions Leaderboard Discussions Editorial △

The defaultdict tool is a container in the collections class of Python. It's similar to the usual dictionary (dict) container, but the only difference is that a defaultdict will have a default value if that key has not been set yet. If you didn't use a defaultdict you'd have to check to see if that key exists, and if it doesn't, set it to what you want.

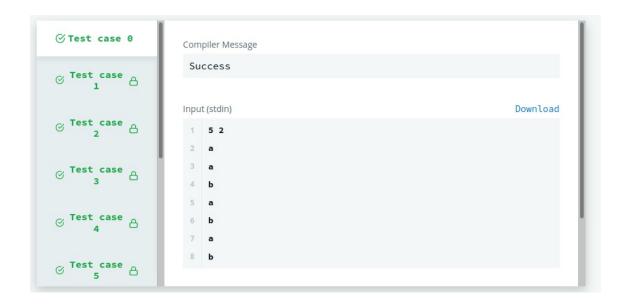
For example:

```
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
```

This prints:

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

In this challenge, you will be given 2 integers, n and m. There are n words, which might repeat, in word group A. There are m words belonging to word group B. For each m words, check whether the word has appeared in group A or not. Print the indices of each occurrence of m in group A. If it does not appear, print -1.



DefaultDict Tutorial ★



Problem	Submissions	Leaderb	oard Discuss	sions Editorial 🛆	
RESULT		SCORE	LANGUAGE	TIME	

Collections – Counter

collections.Counter() *

Editorial 🖰 Discussions Problem Submissions Leaderboard collections.Counter() A counter is a container that stores elements as dictionary keys, and their counts are stored as dictionary values. Sample Code >>> 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]

Task

 ${\it Raghu}$ is a shoe shop owner. His shop has X number of shoes.

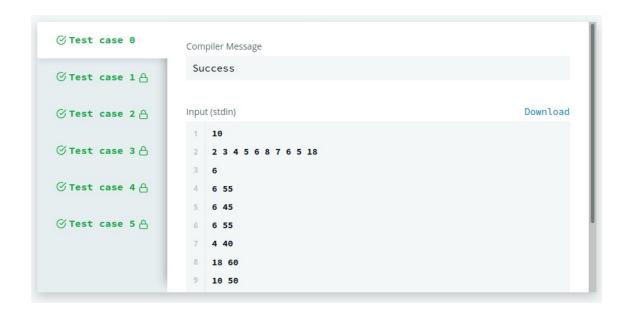
He has a list containing the size of each shoe he has in his shop.

There are N number of customers who are willing to pay x_i amount of money only if they get the shoe of their desired size.

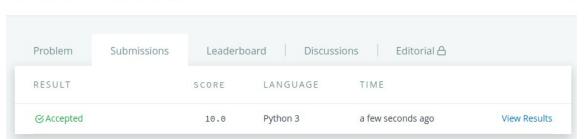
Your task is to compute how much money Raghu earned.

```
2
    import collections
    X = int(input())
    tamanios = collections.Counter(map(int, input().split()))
    N = int(input())
10
11 \vee for i in range(N):
         (tamanio, precio) = map(int, input().split())
12
13
         if tamanios[tamanio] > 0:
14 ~
15
             tamanios[tamanio] -= 1
16
             dinero += precio
17
18
     print(dinero)
```

Rai



collections.Counter() *



Minimum Swaps 2

Minimum Swaps 2 🖈

Problem Submissions | Leaderboard | Discussions | Editorial △

You are given an unordered array consisting of consecutive integers \in [1, 2, 3, ..., n] without any duplicates. You are allowed to swap any two elements. You need to find the minimum number of swaps required to sort the array in ascending order.

For example, given the array arr = [7, 1, 3, 2, 4, 5, 6] we perform the following steps:

```
i arr swap (indices)
0 [7, 1, 3, 2, 4, 5, 6] swap (0,3)
1 [2, 1, 3, 7, 4, 5, 6] swap (0,1)
2 [1, 2, 3, 7, 4, 5, 6] swap (3,4)
3 [1, 2, 3, 4, 7, 5, 6] swap (4,5)
4 [1, 2, 3, 4, 5, 7, 6] swap (5,6)
5 [1, 2, 3, 4, 5, 6, 7]
```

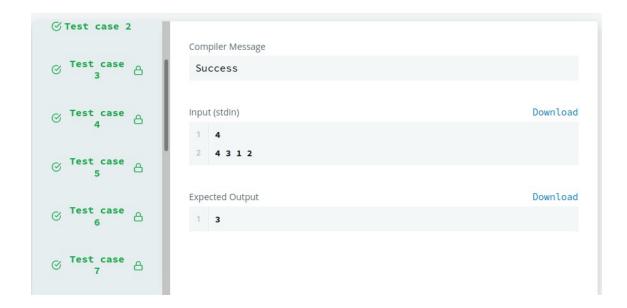
It took 5 swaps to sort the array.

Function Description

Complete the function minimumSwaps in the editor below. It must return an integer representing the minimum number of swaps to sort the array.

minimumSwaps has the following parameter(s):

arr: an unordered array of integers



Minimum Swaps 2 🖈



• Floor, Ceil and Rint

Floor, Ceil and Rint *

Rar

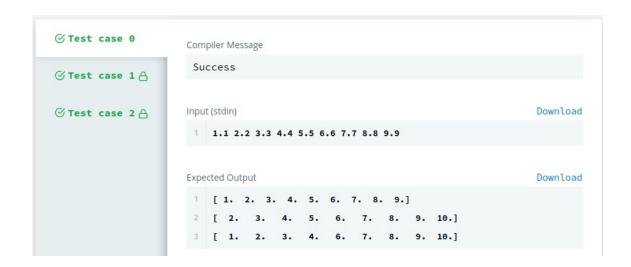
```
Editorial A
                Submissions
                                   Leaderboard
                                                        Discussions
Problem
The tool floor returns the floor of the input element-wise.
The floor of x is the largest integer i where i \leq x.
  import numpy
  my_array = numpy.array([1.1, 2.2, 3.3, 4.4, 5.5, 6.6, 7.7, 8.8, 9.9])
  print numpy.floor(my_array) #[ 1. 2. 3. 4. 5. 6. 7. 8. 9.]
ceil
The tool ceil returns the ceiling of the input element-wise.
The ceiling of x is the smallest integer i where i \geq x.
  import numpy
  my_array = numpy.array([1.1, 2.2, 3.3, 4.4, 5.5, 6.6, 7.7, 8.8, 9.9])
  print numpy.ceil(my_array) #[ 2. 3. 4. 5. 6. 7. 8. 9. 10.]
rint
The rint tool rounds to the nearest integer of input element-wise.
  import numpy
  \label{eq:my_array} \verb"my_array" = \verb"numpy.array" ([1.1, 2.2, 3.3, 4.4, 5.5, 6.6, 7.7, 8.8, 9.9])
  print numpy.rint(my_array) #[ 1. 2. 3. 4. 6. 7. 8. 9. 10.]
```

```
import numpy
numpy.set_printoptions(legacy="1.13")

arreglo = numpy.array(list(map(float, input().split())))

print(numpy.floor(arreglo))
print(numpy.ceil(arreglo))
print(numpy.rint[arreglo])
```

You are given a 1-D array, A. Your task is to print the floor, ceil and rint of all the elements of A.







Sorting: Bubble Sort

Sorting: Bubble Sort *

Problem Submissions | Leaderboard | Discussions | Editorial △

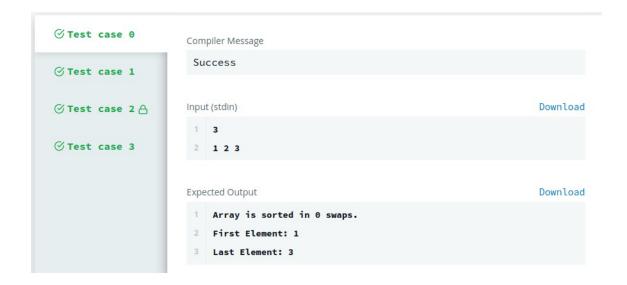
Consider the following version of Bubble Sort:

```
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n - 1; j++) {
        // Swap adjacent elements if they are in decreasing order
        if (a[j] > a[j + 1]) {
            swap(a[j], a[j + 1]);
        }
    }
}
```

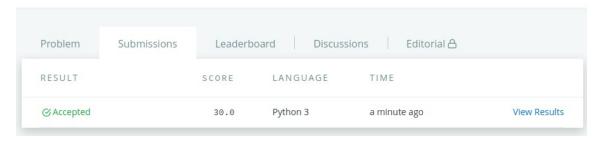
Given an array of integers, sort the array in ascending order using the Bubble Sort algorithm above. Once sorted, print the following three lines:

- 1. Array is sorted in numSwaps swaps., where numSwaps is the number of swaps that took place.
- 2. First Element: firstElement, where firstElement is the first element in the sorted array.
- 3. Last Element: lastElement, where lastElement is the last element in the sorted array.

Hint: To complete this challenge, you must add a variable that keeps a running tally of all swaps that occur during execution.



Sorting: Bubble Sort 🖈



Arrays

Arrays *

Submissions Leaderboard Discussions Editorial A Problem The NumPy (Numeric Python) package helps us manipulate large arrays and matrices of numeric data. To use the NumPy module, we need to import it using: import numpy Arrays A NumPy array is a grid of values. They are similar to lists, except that every element of an array must be the same type. 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 In the above example, numpy.array() is used to convert a list into a NumPy array. The second argument (float) can be used to set the type of array elements. Task

You are given a space separated list of numbers.

Your task is to print a reversed NumPy array with the element type float.

```
import numpy

def arrays(arr):
    arrnumpy = numpy.array(arr, float)
    return [arrnumpy[::-1]])

7 varr = input().strip().split(' ')
    result = arrays(arr)
    print(result)
```



· Zeros and Ones

Zeros and Ones *

Problem Submissions Leaderboard Discussions Editorial 合

zeros

The zeros tool returns a new array with a given shape and type filled with $\mathbf{0}$'s.

```
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

The ones tool returns a new array with a given shape and type filled with 1's.

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

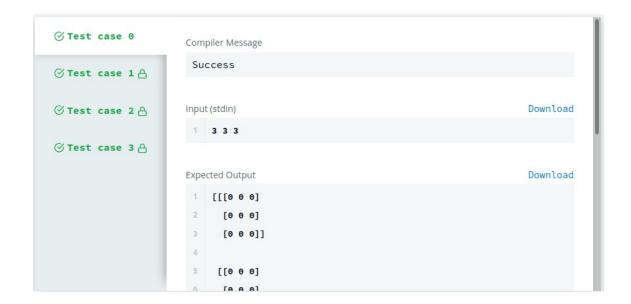
print numpy.ones((1,2), dtype = numpy.int) #Type changes to int
#Output : [[1 1]]
```

Task

You are given the shape of the array in the form of space-separated integers, each integer representing the size of different dimensions, your task is to print an array of the given shape and integer type using the tools numpy.zeros and numpy.ones.

```
import numpy
dimensiones = [int(x) for x in input().strip().split()]

print(numpy.zeros(tuple(dimensiones), dtype = numpy.int))
print(numpy.ones(tuple(dimensiones), dtype = numpy.int))
```



Zeros and Ones 🖈

Rank

RESULT		SCORE	LANGUAGE	TIME	
Problem	Submissions	Leaderboard Discussions			