• Introduction (all – total: 7 - max points: 75)

https://www.hackerrank.com/domains/python/py-introduction

1. Here is a sample line of code that can be executed in Python

   SOLUTION

   print("Hello, World!")

   ## Output

   Hello, World!

2. Given an integer, , perform the following conditional actions:

   If is odd, print Weird

   If is even and in the inclusive range of to , print Not Weird

   If is even and in the inclusive range of to , print Weird

   If is even and greater than , print Not Weird

   SOLUTION

```
n = int(raw_input())
if n % 2 == 1:
 print "Weird"
elif n % 2 == 0 and 2 <= n <= 5:
 print "Not Weird"
elif n % 2 == 0 and 6 <= n <= 20:
print "Weird"
else:
 print "Not Weird"
```

   OUTPUT

   input (stdin)
   3
   Output (stdout)
   Weird

   Expected Output
   Weird

3. Read two integers from STDIN and print three lines where:

   The first line contains the sum of the two numbers.

   The second line contains the difference of the two numbers (first - second).

   The third line contains the product of the two numbers.

   SOLUTION

```
a = int(raw_input())
b = int(raw_input())

print a + b
print a - b
print a * b
```

   OUTPUT

   Input (stdin)

   3
   2

   Your Output (stdout)

   5
   1
   6

   Expected Output

   5
   1
   6

4. Read two integers and print two lines. The first line should contain integer division, a// b.

   The second line should contain float division, a /b .

   You don't need to perform any rounding or formatting operations.

   SOLUTION

```
from __future__ import division
a = int(raw_input())
```

```
b = int(raw_input())

print a // b
print a / b
```

OUTPUT

Input (**stdin**)

4
3

Your **Output** (stdout)

1
1.33333333333

Expected **Output**

1
1.33333333333

5. **Task**

   Read an integer N . For all non-negative integers i<N, print i2 . See the sample for details.

SOLUTION

```
for i in range(int(raw_input())):
print i**2
```

OUTPUT

Input (stdin)

5

Your Output (stdout)

0
1
4
9
16

Expected Output

0

6. You are given the year, and you have to write a function to check if the year is leap or not.

Note that you have to complete the function and remaining code is given as template.

Input Format

Read y, the year that needs to be checked.

SOLUTION

```
def is_leap(n):
    if n % 400 == 0:
        return True
    if n % 100 == 0:
        return False
    if n % 4 == 0:
        return True
    return False

print is_leap(input())
```

OUTPUT

Input (stdin)

1990

Your Output (stdout)

False

Expected Output

False

7. Read an integer N.

Without using any string methods, try to print the following:

123….N

Note that "" represents the values in between.

Input Format

The first line contains an integer N .

SOLUTION

```
from __future__ import print_function
print(*range(1, input() + 1), sep="")
```

OUTPUT

Input (stdin)
3

Your Output (stdout)
123

Expected Output
123

• Data types (all – total: 6 - max points: 60)
https://www.hackerrank.com/domains/python/py-basic-data-types

1. Let's learn about list comprehensions! You are given three integers X,Yand Z representing the dimensions of a cuboid along with an integer N . You have to print a list of all possible coordinates given by(I,j,k) on a 3D grid where the sum of i+j+k is not equal to N. Here, 0<=i<=X; 0<=i<=X; 0<=j<=Y;0<=k<=Z.

Input Format

Four integers and each on four separate lines, respectively.

Constraints

Print the list in lexicographic increasing order.

SOLUTION:

```
a, b, c, n = [int(raw_input()) for _ in xrange(4)]
print [[x,y,z] for x in xrange(a + 1) for y in xrange(b + 1) for z in xrange(c + 1) if x + y + z != n]
```

OUTPUT:

Input (stdin)
1
1
1
2

Your Output (stdout)
[[0, 0, 0], [0, 0, 1], [0, 1, 0], [1, 0, 0], [1, 1, 1]]

Expected Output
[[0, 0, 0], [0, 0, 1], [0, 1, 0], [1, 0, 0], [1, 1, 1]]

2. Given the participants' score sheet for your University Sports Day, you are required to find the runner-up score. You are given n scores. Store them in a list and find the score of the runner-up.

Input Format

The first line contains n. The second line contains an array A [ ] of n integers each separated by a space.

Constraints

2 <= n <= 10

-100 <= A[i] <= 100

SOLUTION

```
if __name__ == '__main__':
    n = int(raw_input())
    arr = map(int, raw_input().split())
    m1 = max(arr)
    m2 = -9999999999
    for i in range(n):
        if arr[i] != m1 and arr[i] > m2:
            m2 = arr[i]
    print m2
```

OUTPUT

Input (stdin)

5
2 3 6 6 5

Your Output (stdout)

5

Expected Output

5

3. Given the names and grades for each student in a Physics class of N students, store them in a nested list and print the name(s) of any student(s) having the second lowest grade.

Note: If there are multiple students with the same grade, order their names alphabetically and print each name on a new line.

Input Format

The first line contains an integer, , the number of students.

The subsequent lines describe each student over lines; the first line contains a student's name, and the second line contains their grade.

Constraints

- 2 <= N <= 5
- There will always be one or more students having the second lowest grade.

SOLUTION

```
from __future__ import print_function
score_list = {}
for _ in range(input()):
    name = raw_input()
    score = float(raw_input())
    if score in score_list:
        score_list[score].append(name)
    else:
        score_list[score] = [name]
new_list = []
for i in score_list:
    new_list.append([i, score_list[i]])
new_list.sort()
result = new_list[1][1]
result.sort()
print (*result, sep = "\n")
```

OUTPUT:

Input (stdin)

5
Harry
37.21
Berry
37.21
Tina
37.2
Akriti
41
Harsh
39

Your Output (stdout)

Berry
Harry

Expected Output
Berry
Harry

4. You have a record of N students. Each record contains the student's name, and their percent marks in Maths, Physics and Chemistry. The marks can be floating values. The user enters some integer N followed by the names and marks for N students. You are required to save the record in a dictionary data type. The user then enters a student's name. Output the average percentage marks obtained by that student, correct to two decimal places.

Input Format

The first line contains the integer N, the number of students. The next N lines contains the name and

marks obtained by that student separated by a space. The final line contains the name of a particular

student previously listed.

Constraints

2<= N <= 10

0 <= Marks <= 100

Solution

```
d={}
for i in range(int(raw_input())):
        line=raw_input().split()
        d[line[0]]=sum(map(float,line[1:]))/3

print '%.2f' % d[raw_input()]
```

Output

Input (stdin)
3
Krishna 67 68 69
Arjun 70 98 63
Malika 52 56 60
Malika

Your Output (stdout)
56.00

Expected Output
56.00

5. Consider a list (list = []). You can perform the following commands:

1. insert i e: Insert integer e at position i.
2. print: Print the list.
3. remove e: Delete the first occurrence of integer e.
4. append e: Insert integer e at the end of the list.
5. sort: Sort the list.
6. pop: Pop the last element from the list.
7. reverse: Reverse the list.

Initialize your list and read in the value of n followed by n lines of commands where each command will be of the 7 types listed above. Iterate through each command in order and perform the corresponding operation on your list.

Input Format

The first line contains an integer, n, denoting the number of commands.

Each line i of the n subsequent lines contains one of the commands described above.

Constraints

The elements added to the list must be *integers*.

Solution:

```
arr = []
for i in range(int(raw_input())):
    s = raw_input().split()
    for i in range(1,len(s)):
        s[i] = int(s[i])

    if s[0] == "append":
        arr.append(s[1])
    elif s[0] == "extend":
        arr.extend(s[1:])
    elif s[0] == "insert":
        arr.insert(s[1],s[2])
    elif s[0] == "remove":
        arr.remove(s[1])
    elif s[0] == "pop":
        arr.pop()
    elif s[0] == "index":
```

```
        print arr.index(s[1])
    elif s[0] == "count":
        print arr.count(s[1])
    elif s[0] == "sort":
        arr.sort()
    elif s[0] == "reverse":
        arr.reverse()
    elif s[0] == "print":
        print arr
```

Output:

Input (stdin)

12
insert 0 5
insert 1 10
insert 0 6
print
remove 6
append 9
append 1
sort
print
pop
reverse
print

Your Output (stdout)

[6, 5, 10]
[1, 5, 9, 10]
[9, 5, 1]

Expected Output

[6, 5, 10]
[1, 5, 9, 10]
[9, 5, 1]

6. Given an integer, n, and n space-separated integers as input, create a tuple, t, of those n integers. Then compute and print the result of hash(t).

Note: hash() is one of the functions in the __builtins__ module, so it need not be imported.

Input Format

The first line contains an integer, n, denoting the number of elements in the tuple.

The second line contains n space-separated integers describing the elements in tuple t.

Output Format

Print the result of hash(t).

Solution:

```
n = raw_input()
print hash(tuple([int(i) for i in raw_input().split()]))
```

Input (stdin)

2
1 2

Your Output (stdout)

3713081631934410656

Expected Output

3713081631934410656

•Strings (all – total: 14 - max points: 220)

https://www.hackerrank.com/domains/python/py-strings
https://www.hackerrank.com/domains/python/py-strings/2

1. You are given a string and your task is to *swap cases*. In other words, convert all lowercase letters to uppercase letters and vice versa.

   Input Format

   A single line containing a string .

   Constraints

   0 < len(S) <= 1000

Output Format

Print the modified string S.

• Sets (all – total: 13 - max points: 170)

https://www.hackerrank.com/domains/python/py-sets
https://www.hackerrank.com/domains/python/py-sets/2

SETS

1. Given 2 sets of integers, M and N , print their symmetric difference in ascending order. The term *symmetric difference* indicates those values that exist in either M or N but do not exist in both.

SOLUTION

```
# Enter your code here. Read input from STDIN. Print output to STDOU
M = raw_input();m = raw_input().split()
N = raw_input();n = raw_input().split()

print "\n".join(sorted(list(set(m) ^ set(n)),key=int))
```

OUTPUT

Input (stdin)

```
4
2 4 5 9
4
2 4 11 12
```

Your Output (stdout)

```
5
9
11
12
```

Expected Output

```
5
9
11
12
```

2. Apply your knowledge of the *.add()* operation to help your friend Rupal.

Rupal has a huge collection of country stamps. She decided to count the total number of distinct country stamps in her collection. She asked for your help. You pick the stamps one by one from a stack of N country stamps.

Find the total number of distinct country stamps.

SOLUTION

```
stamps = set()
for _ in range(int(raw_input())):
   stamps.add(raw_input().strip())
print len(stamps)
```

OUTPUT

Input (stdin)

7
UK
China
USA
France
New Zealand
UK
France

Your Output (stdout)

5

Expected Output

5

3. You have a non-empty set , s and you have to execute N commands given in N lines.

The commands will be *pop, remove* and *discard*.

SOLUTION

```
n = int(raw_input())
s = set([int(x) for x in raw_input().strip().split()])
for _ in range(int(raw_input())):

    a = list(raw_input().strip().split())

    if a[0] == 'pop':
        s.pop()
    elif a[0] == 'discard':
        s.discard(int(a[1]))
    else:
        s.remove(int(a[1]))

print sum(s)
```

OUTPUT

Input (stdin)

9
1 2 3 4 5 6 7 8 9
10
pop
remove 9
discard 9
discard 8
remove 7
pop
discard 6
remove 5
pop
discard 5

Your Output (stdout)

4

Expected Output

4

4. The students of District College have subscriptions to *English* and *French* newspapers. Some students have subscribed only to *English*, some have subscribed to only *French* and some have subscribed to both newspapers.

You are given two sets of student roll numbers. One set has subscribed to the *English* newspaper, and the other set is subscribed to the *French* newspaper. The same student could be in both sets. Your task is to find the total number of students who have subscribed to *at least one* newspaper.

SOLUTION

```
 E = int(raw_input())
English = set(raw_input().split())

F = int(raw_input())
French = set(raw_input().split())

print len(English | French)
```

OUTPUT

Input (stdin)

```
9
1 2 3 4 5 6 7 8 9
9
10 1 2 3 11 21 55 6 8
```

Your Output (stdout)

```
13
```

Expected Output

```
13
```

5. The students of District College have subscriptions to *English* and *French* newspapers. Some students have subscribed only to *English*, some have subscribed only to *French*, and some have subscribed to both newspapers.

You are given two sets of student roll numbers. One set has subscribed to the *English* newspaper, one set has subscribed to the *French* newspaper. Your task is to find the total number of students who have subscribed to *both* newspapers.

SOLUTION

```
E = int(raw_input())
English = set(raw_input().split())

F = int(raw_input())
French = set(raw_input().split())

print len(English & French)
```

OUTPUT

Input (stdin)

```
9
1 2 3 4 5 6 7 8 9
9
10 1 2 3 11 21 55 6 8
```

Your Output (stdout)

```
5
```

Expected Output

```
5
```

6. Students of District College have a subscription to *English* and *French* newspapers. Some students have subscribed to only the *English* newspaper, some have subscribed to only the *French* newspaper, and some have subscribed to both newspapers.

You are given two sets of student roll numbers. One set has subscribed to the *English* newspaper, and one set has subscribed to the *French* newspaper. Your task is to find the total number of students who have subscribed to *only English* newspapers.

SOLUTION

```
E = int(raw_input())
English = set(raw_input().split())

F = int(raw_input())
French = set(raw_input().split())

print len(English - French)
```

OUTPUT

Input (stdin)

```
9
1 2 3 4 5 6 7 8 9
9
10 1 2 3 11 21 55 6 8
```

Your Output (stdout)

```
4
```

Expected Output

```
4
```

7. Students of District College have subscriptions to *English* and *French* newspapers. Some students have subscribed to *English* only, some have subscribed to *French* only, and some have subscribed to both newspapers.
   You are given two sets of student roll numbers. One set has subscribed to the *English* newspaper, and one set has subscribed to the *French* newspaper. Your task

is to find the total number of students who have subscribed to either the *English* or the *French* newspaper but *not both*.

SOLUTION

```
E = int(raw_input())
English = set(raw_input().split())

F = int(raw_input())
French = set(raw_input().split())

print len(English ^ French)
```

OUTPUT

Input (stdin)

```
9
1 2 3 4 5 6 7 8 9
9
10 1 2 3 11 21 55 6 8
```

Your Output (stdout)

```
8
```

Expected Output

```
8
```

Collections (all – total: 8 - max points: 220)

https://www.hackerrank.com/domains/python/py-collections

1. 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 xi amount of money only if they get the shoe of their desired size.

Your task is to compute how much money Raghu earned.

SOLUTION

```
from collections import Counter
X = input()
S = Counter(map(int,raw_input().split()))
N = input()
earnings = 0
for customer in range(N):
   size, x_i = map(int,raw_input().split())
   if size in S and S[size] > 0:
      S[size] -= 1
      earnings += x_i

print earnings
```

OUTPUT

Input (stdin)

```
10
2 3 4 5 6 8 7 6 5 18
6
6 55
6 45
6 55
4 40
18 60
10 50
```

Your Output (stdout)

```
200
```

Expected Output

```
200
```

1. You are the manager of a supermarket.

   You have a list of items together with their prices that consumers bought on a particular day.

   Your task is to print each item_name and net_price in order of its first occurrence.

SOLUTION

```
from collections import OrderedDict
my_order = OrderedDict()
for i in range(int(raw_input())):
    name,space,price = raw_input().rpartition(' ')
    if name not in my_order:
        my_order[name] = int(price)
    else:
        my_order[name] += int(price)
for item_name, net_price in my_order.items():
    print item_name,net_price
```

OUTPUT

Input (stdin)

```
9
BANANA FRIES 12
POTATO CHIPS 30
APPLE JUICE 10
CANDY 5
APPLE JUICE 10
CANDY 5
CANDY 5
CANDY 5
POTATO CHIPS 30
```

Your Output (stdout)

```
BANANA FRIES 12
POTATO CHIPS 60
APPLE JUICE 20
CANDY 20
```

Expected Output

```
BANANA FRIES 12
POTATO CHIPS 60
APPLE JUICE 20
CANDY 20
```

2. A *deque* is a double-ended queue. It can be used to add or remove elements from both ends.

Deques support thread safe, memory efficient appends and pops from either side of the deque with approximately the same O(1) performance in either direction.

Click on the link to learn more about **deque() methods**.

Click on the link to learn more about various approaches to working with deques: **Deque Recipes**.

SOLUTION

```
from collections import deque
d = deque()
for i in range(int(raw_input())):
    s = raw_input().split()
    if s[0] == 'append':
        d.append(s[1])
    elif s[0] == 'appendleft':
        d.appendleft(s[1])
    elif s[0] == 'pop':
        d.pop()
    else:
        d.popleft()
print " ".join(d)
```

OUTPUT

Input (stdin)

```
6
append 1
append 2
```

append 3
appendleft 4
pop
popleft

Your Output (stdout)

1 2

Expected Output

1 2

4 . A newly opened multinational brand has decided to base their company logo on the three most common characters in the company name. They are now trying out various combinations of company names and logos based on this condition. Given a string , which is the company name in lowercase letters, your task is to find the top three most common characters in the string.

- Print the three most common characters along with their occurrence count.
- Sort in descending order of occurrence count.
- If the occurrence count is the same, sort the characters in alphabetical order.

For example, according to the conditions described above,

GOOGLE would have it's logo with the letters G, O, E .

SOLUTION

```
S = raw_input()
letters = [0]*26
```

```
for letter in S:
    letters[ord(letter)-ord('a')] += 1

for _ in range(3):

    max_letter = max(letters)

    for index in range(26):
        if max_letter == letters[index]:
            print chr(ord('a')+index), max_letter
            letters[index] = -1
            break
```

OUTPUT

Input (stdin)

aabbbccde

Your Output (stdout)

b 3
a 2
c 2

Expected Output

b 3
a 2
c 2

Date and Time (all – total: 2 - max points: 40)

https://www.hackerrank.com/domains/python/py-date-time

1. Calendar Module

The calendar module allows you to output calendars and provides additional useful functions for them. class calendar.TextCalendar([firstweekday])

This class can be used to generate plain text calendars.

SOLUTION

```
import calendar

day = {0:'MONDAY', 1:'TUESDAY', 2:'WEDNESDAY', 3:'THURSDAY', 4:'FRIDAY', 5:'SATURDAY',
6:'SUNDAY'}
```

```
m,d,y = map(int,raw_input().split())
print day[calendar.weekday(y,m,d)]
```

OUTPUT

Input (stdin)

08 05 2015

Your Output (stdout)

WEDNESDAY

Expected Output

WEDNESDAY

Exceptions (only 1 - max points: 10)

https://www.hackerrank.com/challenges/exceptions

You are given two values a and b .

Perform integer division and print a/b .

Input Format

The first line contains ,T the number of test cases.

The next T lines each contain the space separated values of a and b .

SOLUTION:

```
for i in range(int(raw_input())):
    try:
        a,b = map(int, raw_input().split())
        print a/b
    except ZeroDivisionError as e:
        print 'Error Code:',e
    except ValueError as e:
        print 'Error Code:',e
```

OUTPUT

Input (stdin)

3
1 0
2 $
3 1

Your Output (stdout)

Error Code: integer division or modulo by zero
Error Code: invalid literal for int() with base 10: '$'
3

Expected Output

Error Code: integer division or modulo by zero
Error Code: invalid literal for int() with base 10: '$'
3

Built-ins (only 3 - max points: 80)

https://www.hackerrank.com/challenges/zipped

https://www.hackerrank.com/challenges/python-sort-sort

https://www.hackerrank.com/challenges/ginorts

1. This function returns a list of tuples. The i^{th} tuple contains the i^{th} element from each of the argument sequences or iterables.If the argument sequences are of unequal lengths, then the returned list is truncated to the length of the shortest argument sequence.

SOLUTION:

```
N, X = map(int, raw_input().split())
list_of_marks = []
for subject in xrange(X):
    list_of_marks += [map(float,raw_input().split())]

for student_marks in zip(*list_of_marks):
    print sum(student_marks)/X
```

OUTPUT:

Input (stdin)

5 3
89 90 78 93 80
90 91 85 88 86
91 92 83 89 90.5

Output (stdout)

90.0
91.0
82.0
90.0
85.5

Expected Output

90.0
91.0
82.0
90.0
85.5

2. You are given a spreadsheet that contains a list of athletes and their details (such as age, height, weight and so on). You are required to sort the data based on the th attribute and print the final resulting table. Follow the example given below for better understanding.

SOLUTION

```
n,m = map(int,raw_input().split())
lst = []
for i in range(n):
   lst.append(map(int,raw_input().split()))
k = int(raw_input())
print "\n".join(map(lambda x: " ".join(map(str, x)), sorted(lst, key = lambda x: x[k])))
```

OUTPUT

Input (stdin)

5 3
10 2 5
7 1 0
9 9 9
1 23 12
6 5 9
1

Your Output (stdout)

7 1 0
10 2 5
6 5 9
9 9 9
1 23 12

Expected Output

7 1 0
10 2 5
6 5 9
9 9 9
1 23 12

XML (all – total: 2 - max points: 40)

https://www.hackerrank.com/domains/python/xml

1. You are given a valid XML document, and you have to print the maximum level of nesting in it. Take the depth of the root as 0 .

SOLUTION

```
import xml.etree.ElementTree as etree

maxdepth = 0
def depth(elem, level):
    global maxdepth
    # your code goes here

if __name__ == '__main__':
    n = int(raw_input())
    xml = ""
    for i in range(n):
        xml =  xml + raw_input() + "\n"
    tree = etree.ElementTree(etree.fromstring(xml))
    depth(tree.getroot(), -1)
    print maxdepth
```

OUTPUT

Input (stdin)

```
6
<feed xml:lang='en'>
 <title>HackerRank</title>
 <subtitle lang='en'>Programming challenges</subtitle>
 <link rel='alternate' type='text/html' href='http://hackerrank.com/'/>
 <updated>2013-12-25T12:00:00</updated>
</feed>
```

Your Output (stdout)

```
0
```

Closures and Decorations (all – total: 2 - max points: 60)

https://www.hackerrank.com/domains/python/closures-and-decorators

1. Let's use decorators to build a name directory! You are given some information about people. Each person has a first name, last name, age and sex. Print their names in a specific format sorted by their age in ascending order i.e. the youngest person's name should be printed first. For two people of the same age, print them in the order of their input.

SOLUTION

```
import operator

def person_lister(f):
    def inner(people):
        # complete the function
    return inner

@person_lister
def name_format(person):
    return ("Mr. " if person[3] == "M" else "Ms. ") + person[0] +
" " + person[1]

if __name__ == '__main__':
    people = [raw_input().split() for i in
range(int(raw_input()))]
    print '\n'.join(name_format(people))
```

Problem: 2

• https://www.hackerrank.com/challenges/birthday-cake-candles

1. You are in charge of the cake for your niece's birthday and have decided the cake will have one candle for each year of her total age. When she blows out the candles, she'll only be able to blow out the tallest ones. Your task is to find out how many candles she can successfully blow out.For example, if your niece is turning 4 years old, and the cake will have 4 candles of height 4,4 1,3 she will be able to blow out 2 candles successfully, since the tallest candles are of height 4 and there are 2 such candles.

```
#!/bin/python3

import math
import os
import random
import re
import sys

# Complete the birthdayCakeCandles function below.
def birthdayCakeCandles(ar):

if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    ar_count = int(input())

    ar = list(map(int, input().rstrip().split()))

    result = birthdayCakeCandles(ar)

    fptr.write(str(result) + '\n')

    fptr.close()
```

OUTPUT

Input (stdin)

4
3 2 1 3

Your Output (stdout)

2

Expected Output

2

- https://www.hackerrank.com/challenges/kangaroo

You are choreographing a circus show with various animals. For one act, you are given two kangaroos on a number line ready to jump in the positive direction (i.e, toward positive infinity).

- The first kangaroo starts at location x1 and moves at a rate of v1 meters per jump.
- The second kangaroo starts at location x2 and moves at a rate of v2 meters per jump.

You have to figure out a way to get both kangaroos at the same location at the same time as part of the show. If it is possible, return YES, otherwise return NO.

For example, kangaroo 1 starts at x1=2 with a jump distance v1=1 and kangaroo 2 starts at x2=1 with a jump distance of v2=2. After one jump, they are both at x=3, (x1+v1=2+1, x2+v2=1+2), so our answer is YES.

Function Description

Complete the function *kangaroo* in the editor below. It should return YES if they reach the same position at the same time, or NO if they don't.

kangaroo has the following parameter(s):

- *x1, v1*: integers, starting position and jump distance for kangaroo 1
- *x2, v2*: integers, starting position and jump distance for kangaroo 2

Solution:

```
x1, v1, x2, v2 = map(int, raw_input().split())
X = [x1, v1]
Y = [x2, v2]
back = min(X, Y)
fwd = max(X, Y)
dist = fwd[0] - back[0]

while back[0] < fwd[0]:
   back[0] += back[1]
   fwd[0] += fwd[1]
   if fwd[0] - back[0] >= dist:
      break

print ["NO", "YES"][back[0] == fwd[0]]
```

Output:

Input (stdin)

0 3 4 2

Your Output (stdout)

YES

Expected Output

YES

- https://www.hackerrank.com/challenges/recursive-digit-sum

HackerLand Enterprise is adopting a new viral advertising strategy. When they launch a new product, they advertise it to exactly 5 people on social media.

On the first day, half of those 5 people (i.e., $floor(\frac{5}{2}) = 2$) like the advertisement and each shares it with 3 of their friends. At the beginning of the second day, $floor(\frac{5}{2}) \times 3 = 2 \times 3 = 6$ people receive the advertisement.

Each day, $floor(\frac{recipients}{2})$ of the recipients like the advertisement and will share it with 3 friends on the following day. Assuming nobody receives the advertisement twice, determine how many people have liked the ad by the end of a given day, beginning with launch day as day 1.

For example, assume you want to know how many have liked the ad by the end of the 5^{th} day.

Solution:

```
n = int(input())
shared = 5
total_opened = 0

for _ in range(0, n):
    opened = int(shared / 2)
    total_opened = total_opened + opened
    shared = opened * 3

print(total_opened)
```

Output:

Input (stdin)

3

Your Output (stdout)

9

Expected Output

9

- https://www.hackerrank.com/challenges/insertionsort1

Sorting

One common task for computers is to sort data. For example, people might want to see all their files on a computer sorted by size. Since sorting is a simple problem with many different possible solutions, it is often used to introduce the study of algorithms.

Insertion Sort

These challenges will cover Insertion Sort, a simple and intuitive sorting algorithm. We will first start with a nearly sorted list.

Insert element into sorted list

Given a sorted list with an unsorted number e in the rightmost cell, can you write some simple code to insert e into the array so that it remains sorted?

Since this is a learning exercise, it won't be the most efficient way of performing the insertion. It will instead demonstrate the brute-force method in detail.

Assume you are given the array $ar = [1, 2, 4, 5, 3]$ indexed $0 \ldots 4$. Store the value of $ar[4]$. Now test lower index values successively from 3 to 0 until you reach a value that is lower than $ar[4]$, $ar[1]$ in this case. Each time your test fails, copy the value at the lower index to the current index and print your array. When the next lower indexed value is smaller than $ar[4]$, insert the stored value at the current index and print the entire array.

The results of operations on the example array is:

Starting array: $[1, 2, 4, 5, 3]$

Store the value of $ar[4] = 3$ Do the tests and print interim results:

```
1 2 4 5 5
1 2 4 4 5
1 2 3 4 5
```

Solution:

```
n = int(raw_input())
test = raw_input()
a = test.split()
val = a[n-1]
for i in range(n-2,-2,-1):
    test=test.split()
    if i == -1:
        test[i+1] = val
        break
    elif int(test[i])>int(val):
        test[i+1]=test[i]

    else:
        test[i+1] = val
        break
    test = ' '.join(test)
    print test
test = ' '.join(test)
print test
```

Output:

Input (stdin)

```
5
2 4 6 8 3
```

Your Output (stdout)

```
2 4 6 8 8
2 4 6 6 8
2 4 4 6 8
2 3 4 6 8
```

Expected Output

```
2 4 6 8 8
2 4 6 6 8
2 4 4 6 8
2 3 4 6 8
```

- https://www.hackerrank.com/challenges/insertionsort2

In Insertion Sort Part 1, you inserted one element into an array at its correct sorted position. Using the same approach repeatedly, can you sort an entire array?

Guideline: You already can place an element into a sorted array. How can you use that code to build up a sorted array, one element at a time? Note that in the first step, when you consider an array with just the first element, it is already sorted since there's nothing to compare it to.

In this challenge, print the array after each iteration of the insertion sort, i.e., whenever the next element has been inserted at its correct position. Since the array composed of just the first element is already sorted, begin printing after placing the second element.

For example, there are $n = 7$ elements in $arr = [3, 4, 7, 5, 6, 2, 1]$. Working from left to right, we get the following output:

```
3 4 7 5 6 2 1
3 4 7 5 6 2 1
3 4 5 7 6 2 1
3 4 5 6 7 2 1
2 3 4 5 6 7 1
1 2 3 4 5 6 7
```

Solution:

```
#!/bin/python
def insertionSort(ar):
    for i in xrange(1, len(ar)):
        temp = ar[i]
        j = i
        while j > 0 and temp < ar[j-1]:
```

```
        ar[j] = ar[j-1]
        j -= 1
    ar[j] = temp
    print ' '.join(str(j) for j in ar)

m = input()
ar = [int(i) for i in raw_input().strip().split()]
insertionSort(ar)
```

Output:

Input (stdin)

```
6
1 4 3 5 6 2
```

Your Output (stdout)

```
1 4 3 5 6 2
1 3 4 5 6 2
1 3 4 5 6 2
1 3 4 5 6 2
1 2 3 4 5 6
```

Expected Output

```
1 4 3 5 6 2
1 3 4 5 6 2
1 3 4 5 6 2
1 3 4 5 6 2
1 2 3 4 5 6
```