15 June 2019

In [1]:
```
1 dir(list())
```

Out[1]:
```
['__add__',
 '__class__',
 '__contains__',
 '__delattr__',
 '__delitem__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__gt__',
 '__hash__',
 '__iadd__',
 '__imul__',
 '__init__',
 '__init_subclass__',
 '__iter__',
 '__le__',
 '__len__',
 '__lt__',
 '__mul__',
 '__ne__',
 '__new__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__reversed__',
 '__rmul__',
 '__setattr__',
 '__setitem__',
 '__sizeof__',
 '__str__',
 '__subclasshook__',
 'append',
 'clear',
 'copy',
 'count',
 'extend',
 'index',
 'insert',
 'pop',
 'remove',
 'reverse',
 'sort']
```

In [2]:
```
1 lis = [ 1, 2, 3, 4]
2 lis.append(22)
```

In [7]:
```
l = [ 1, 2, 3, 4, 5]
l.append([1, 2, 3])
l
[1,2,3,4,5,[1,2,3],1,2]
l.extend([1,2])
```

Problem Statement - ClosestZero

Explanation

- li = [3, 3, -1, -2, -3]
- sort the data
- li = [-3, -2, -1, 2, 3] (Sorted List)
- pl = [1, 2, 2, 3, 3] (Positive Sorted List)
- pl[0] --> Check if this number is -ve or +ve in the original list
- if pl[0] in li:
  - return pl[0]
- else
  - return -pl[0]

In [41]:
```
li = [-1, -2, 2, 3, 1]

li.sort()

pl = []

for i in li:
    pl.append(abs(i))

#pl   positive list
pl.sort()
if pl[0] in li:
    print(pl[0])
else:
    print(-pl[0])

```

```
1
```

In [23]:
```
# FarthestFromZero

li = [-1, -2, 1, -10, 9]

#li = [-1, -2, 2, 3, 1]

li.sort()

pl = []

for i in li:
    pl.append(abs(i))

#pl   positive list
pl.sort()
if pl[-1] in li:
    print(pl[-1])
else:
    print(-pl[-1])
```

```
-10
```

Problem - 3

- You are given three numbers a, b and c
- Write a program to find the largest nuber
- which is less than or equal to c and leaves
- remainder b when divided by a.

In [38]:
```
def largestNumber(a,b,c):
    for i in range (c, b-1,-1): # c, c-1, c-2, ..... b or a-1 will also be
        if(i<=c and i%a==b):
            return i
    else:
        return -1

largestNumber(1,2,4)
```

Out[38]:
```
-1
```

In [40]:
```
def cal(a,b,c):
    for i in range(c,a-1,-1):   # c, c-1, c-2, ..... a
        if i % a == b:
            return i
    return -1
cal(3, 2, 100)
```

Out[40]:
```
98
```

In [2]:
```
# Function to generate data list
# in data list even position contains prime number
# Odd positions fibano series
n = int(input())
s = input().split()[:n]
li = s
li
```

```
2
1 2 3
```

Out[2]:
```
['1', '2']
```

In [46]:
```
s = " 1 2 3 4 5 6"
li = s.split()
numberlist = []
for i in li:
    numberlist.append(int(i))
numberlist
```

Out[46]:
```
[1, 2, 3, 4, 5, 6]
```

In [5]:
```
n = int(input())
lis = [int(i) for i  in input().split()][:n]
lis
```

```
5
1 2 3 4 5 6 7 8 9
```

Out[5]:
```
[1, 2, 3, 4, 5]
```

You have been given 3 integers - l, r and k. Find how many numbers between l and r (both inclusive) are divisible by k. You do not need to print these numbers, you just have to find their count.

Input Format The first and only line of input contains 3 space separated integers l, r and k.

Output Format Print the required answer on a single line.

Constraints

In [6]:
```
# Count the
def countDivisors(i,j,k):
    count = 0
    for n in range(i,j+1):
        if n% k == 0:
            count = count+1
    return count

s = input()

s=s.split()
i = int(s[0])

j = int(s[1])

k = int(s[2])

print(countDivisors(i,j,k))


```

```
1 10 1
10
```

In [9]:
```
# same above solved by me
def countDivisors(l,r,k):
    count = 0
    for i in range(l,r+1):
        if(i%k==0):
            count = count+1
    return count

li=input().split()
l=int(li[0])
r=int(li[1])
k=int(li[2])
print(countDivisors(l,r,k))

```

```
1 10 1
10
```

Problem Statement

You have been given a positive integer N. You need to find and print the Factorial of this number. The Factorial of a positive integer N refers to the product of all number in the range from 1 to N. You can read more about the factorial of a number here.

Input Format: The first and only line of the input contains a single integer N denoting the number whose factorial you need to find.

Output Format Output a single line denoting the factorial of the number N.

Constraints 1<=N<=10

In [10]:
```
def factorial(N):
    f = 1
    for i in range(1,N+1):
        f = f*i
    return f


N=int(input())
print(factorial(N))
```

```
5
120
```

In [20]:
```
print(ord('Z'))
```

```
90
```

In [12]:
```
print(chr(99))
```

```
c
```

Problem Statement

You have been given a String S consisting of uppercase and lowercase English alphabets. You need to change the case of each alphabet in this String. That is, all the uppercase letters should be converted to lowercase and all the lowercase letters should be converted to uppercase. You need to then print the resultant String to output.

Input Format The first and only line of input contains the String S

Output Format Print the resultant String on a single line.

Constraints 1<=|S|<=100 where S denotes the length of string S.

In [21]:
```
# ToggleString i.e changing Case
def ToggleString(s):
    a=''
    for i in s:
        if (i>='A' and i<='Z'):
            a = a+chr(ord(i)+32)
        elif (i>='a' and i<='z'):
            a = a+chr(ord(i)-32)
    print(a)

s = input()
ToggleString(s)


```

```
SiResHa
sIrEShA
```

Problem Statement

You have been given a String S. You need to find and print whether this string is a palindrome or not. If yes, print "YES" (without quotes), else print "NO" (without quotes).

Input Format The first and only line of input contains the String S. The String shall consist of lowercase English alphabets only.

Output Format Print the required answer on a single line.

Constraints

Note String S consists of lowercase English Alphabets only.

In [16]:
```
# isPolindromem or not
def palindrome(s):
    if(s==s[::-1]):
        return 'YES'
    else:
        return 'NO'

s=input()
print(palindrome(s))
```

```
Madam
NO
```

In [22]:
```
# Prime Number upto N

def NisPrime(n):
    #count = 0
    for i in range(1,n+1):
        count =0
        for j in range(1,i+1):
            if i%j==0:
                count= count+1
        if(count==2):
            print(i,end=" ")

n=int(input())
NisPrime(n)
```

```
9
2 3 5 7
```

In [6]:
```
n= int(input())
for i in range(n):
    st=input().split()
    s=st[0]
    t=st[1]


```

```
4
jckljd dkjfkd jdkfj
kdjf jdkfj jdkfj
jkdjf jdkjf jkdjf
jkdj iudij jdid
```

In [ ]:
```

```

Problem Statement

Given two strings of equal length, you have to tell whether they both strings are identical.

Two strings S1 and S2 are said to be identical, if any of the permutation of string S1 is equal to the string S2. See Sample explanation for more details.

Input :

First line, contains an intger 'T' denoting no. of test cases. Each test consists of a single line, containing two space separated strings S1 and S2 of equal length. Output:

For each test case, if any of the permutation of string S1 is equal to the string S2 print YES else print NO. Constraints:

1<= T <=100 1<= |S1| = |S2| <= 10^5 String is made up of lower case letters only. Note : Use Hashing Concept Only . Try to do it in O(string length) .

SAMPLE INPUT 3 sumit mitsu ambuj jumba abhi hibb

SAMPLE OUTPUT YES YES NO

Explanation For first test case,

mitsu can be rearranged to form sumit .

For second test case,

jumba can be rearranged to form ambuj .

For third test case,

hibb can not be arranged to form abhi.

In [8]:

```
# Two Strings   All test cases not satisfied
T = int(input())
def twoStrings(S1,S2):
    X = 0
    if len(S1)!=len(S2):
        return "NO"
    else:
        for i in range(len(S1)):
            if(S1.count(S1[i])==S2.count(S1[i])):
                X+=1
        if(X==len(S1)):
            return "YES"
        else:
            return "NO"


#T = int(input())
for i in range(T):
    string =input().split()
    S1=string[0]
    S2=string[1]
    print(twoStrings(S1,S2))

```

```
3
siri irsi
YES
satya tyaas
YES
jhsk jkdjd
NO
```

In [9]:
```
T = int(input())
def twoStrings(S1,S2):
    flag = 0
    if len(S1)!=len(S2):
        return "NO"
    else:
        for i in range(len(S1)):
            if(S1.count(S1[i])!=S2.count(S1[i])):
                return "NO"
        if(flag==0):
            return "YES"
#T = int(input())
for i in range(T):
    string =input().split()
    S1=string[0]
    S2=string[1]
    print(twoStrings(S1,S2))

```

```
3
syaman djjdu
NO
shd djksj
NO
syamala syamala
YES
```

In [7]:
```
 ord('a')

```

Out[7]: 97

In [10]:
```
a = ord('a') - 96
a

```

Out[10]: 1

In [ ]:
```

```