**Say "Hello, World!" With Python**

print("Hello, World!")

**Python if-else**

import math

import os

import random

import re

import sys

n = int(input().strip())

if n%2==1:

print('Weird')

else:

if n>5 and n<21:

print('Weird')

else:

print('Not Weird')

**arithmetic operator**

a = int(input())

b = int(input())

print(a+b)

print(a-b)

print(a\*b)

**python: division**

if \_\_name\_\_ == '\_\_main\_\_':

    a = int(input())

    b = int(input())

    print(a//b)

    print(a/b)

**loops**

if \_\_name\_\_ == '\_\_main\_\_':

n = int(input())

i=0

while i<n:

print(i\*i)

i+=1

**tuples**

n=int(input())

t=tuple([int(x) for x in input().split()])

print(hash(t))

**write a function**

def is\_leap(year):

leap = False

if year%400==0:

leap=True

else:

if year%4==0 and year%100!=0:

leap =True

return leap

**print a function**

if \_\_name\_\_ == '\_\_main\_\_':

n = int(input())

a=''

i=1

while i<=n:

a=a+str(i)

i+=1

print(a)

**find the runner up score**

if \_\_name\_\_ == '\_\_main\_\_':

n = int(input())

arr = list(map(int, input().split()))

m=max(arr)

arr.remove(max(arr))

i=0

while i <n:

if max(arr)!=m:

break

else:

arr.remove(max(arr))

i+=1

print (max(arr))

**Finding the percentage**

if \_\_name\_\_ == '\_\_main\_\_':

n = int(input())

student\_marks = {}

for \_ in range(n):

name, \*line = input().split()

scores = list(map(float, line))

student\_marks[name] = scores

query\_name = input()

a=(student\_marks[query\_name])

b=0

i=0

while i < len(a):

b=b+a[i]

i+=1

print('%.2f'% (b/len(a)))

**Swap cases**

def swap\_case(s):

    t=list(s)

    for i in range(0, len(t)):

        if t[i].isupper() is True:

            t[i] = t[i].lower()

        elif t[i].islower() is True:

            t[i] = t[i].upper()

        a=''

        for i in range(0, len(t)):

            a=a+t[i]

    return a

if \_\_name\_\_ == '\_\_main\_\_':

    s = input()

    result = swap\_case(s)

    print(result)

**split and join**

def split\_and\_join(line):

    # write your code here

    a=line.split(" ")

    b="-".join(a)

    return b

if \_\_name\_\_ == '\_\_main\_\_':

    line = input()

    result = split\_and\_join(line)

    print(result)

**what’s your name**

def print\_full\_name(first, last):

    # Write your code here

    print('Hello '+ first + ' '+last+'! You just delved into python.')

if \_\_name\_\_ == '\_\_main\_\_':

    first\_name = input()

    last\_name = input()

    print\_full\_name(first\_name, last\_name)

**mutations**

def mutate\_string(string, position, character):

    a=list(string)

    a[position] = character

    string=''.join(a)

    return string

if \_\_name\_\_ == '\_\_main\_\_':

    s = input()

    i, c = input().split()

    s\_new = mutate\_string(s, int(i), c)

    print(s\_new)

**find a string**

def count\_substring(string, sub\_string):

    l=len(sub\_string)

    a=0

    for i in range(0, len(string)-l+1):

        if string[i: i+l]==sub\_string:

            a=a+1

    return a

if \_\_name\_\_ == '\_\_main\_\_':

    string = input().strip()

    sub\_string = input().strip()

    count = count\_substring(string, sub\_string)

    print(count)

**string validator**

if \_\_name\_\_ == '\_\_main\_\_':

    s = input()

    a=0

    b=0

    c=0

    d=0

    e=0

    t=list(s)

    for i in range(0,len(s)):

        if  t[i].isalnum() is True:

            a=1

        if  t[i].isalpha() is True:

            b=1

        if  t[i].isdigit() is True:

            c=1

        if  t[i].islower() is True:

            d=1

        if  t[i].isupper() is True:

            e=1

    print(a==1)

    print(b==1)

    print(c==1)

    print(d==1)

    print(e==1)

**text allignement**

#Replace all \_\_\_\_\_\_ with rjust, ljust or center.

thickness = int(input()) #This must be an odd number

c = 'H'

#Top Cone

for i in range(thickness):

    print((c\*i).rjust(thickness-1)+c+(c\*i).ljust(thickness-1))

#Top Pillars

for i in range(thickness+1):

    print((c\*thickness).center(thickness\*2)+(c\*thickness).center(thickness\*6))

#Middle Belt

for i in range((thickness+1)//2):

    print((c\*thickness\*5).center(thickness\*6))

#Bottom Pillars

for i in range(thickness+1):

    print((c\*thickness).center(thickness\*2)+(c\*thickness).center(thickness\*6))

#Bottom Cone

for i in range(thickness):

    print(((c\*(thickness-i-1)).rjust(thickness)+c+(c\*(thickness-i-1)).ljust(thickness)).rjust(thickness\*6))

**text wrap**

def wrap(s, w):

    t=len(s)

    for i in range(0,t//w):

        j=t//w-i

        s=s[:j\*w]+'\n'+s[j\*w:]

    return s

if \_\_name\_\_ == '\_\_main\_\_':

    string, max\_width = input(), int(input())

    result = wrap(string, max\_width)

    print(result)

**Capitalize**

# Complete the solve function below.

def solve(s):

    l=list(s)

    l[0]=l[0].upper()

    for i in range(1,len(s)):

        if l[i-1]==' ':

            l[i]=l[i].upper()

    return ''.join(l)

if \_\_name\_\_ == '\_\_main\_\_':

    fptr = open(os.environ['OUTPUT\_PATH'], 'w')

    s = input()

    result = solve(s)

    fptr.write(result + '\n')

**word score**

def is\_vowel(letter):

    return letter in ['a', 'e', 'i', 'o', 'u', 'y']

def score\_words(words):

    score = 0

    for word in words:

        num\_vowels = 0

        for letter in word:

            if is\_vowel(letter):

                num\_vowels += 1

        if num\_vowels % 2 == 0:

            score += 2

        else:

            score += 1

    return score

n = int(input())

words = input().split()

print(score\_words(words))

**arrays**

import numpy

def arrays(arr):

    a=numpy.array(arr,float)

    return a[::-1]

    # complete this function

    # use numpy.array

arr = input().strip().split(' ')

result = arrays(arr)

print(result)

**shape and reshape**

import numpy as np

arr = input().strip().split(' ')

a=np.array(arr,int)

print(np.reshape(a,(3,3)))

**polynomials**

import numpy as np

arr = input().strip().split(' ')

x=int(input())

a=np.array(arr,float)

p=a

print (np.polyval(p,x))

**linear algebra**

import numpy as np

N=int(input())

a=np.zeros((N,N))

for i in range(N):

   a[i]=input().strip().split(' ')

print (round(np.linalg.det(a),2))

**mean var and std**

import numpy as np

b=(input().strip().split(' '))

N=int(b[0])

M=int(b[1])

a=np.zeros((N,M))

for i in range(N):

   a[i]=input().strip().split(' ')

print(np.mean(a, axis = 1) )

print(np.var(a, axis = 0) )

print(round(np.std(a, axis = None),11) )

**dot and cross**

import numpy as np

N=int(input())

a=np.array([input().split()

for i in range(N)],int)

b=np.array([input().split()

for i in range(N)],int)

print(np.dot(a,b))

**inner and outer**

import numpy as np

a=np.array(input().split(' '),int)

b=np.array(input().split(' '),int)

print(np.inner(a,b))

print(np.outer(a,b))

**list comprension**

if \_\_name\_\_ == '\_\_main\_\_':

    x = int(input())

    y = int(input())

    z = int(input())

    n = int(input())

    out=[]

    a=[]

    for i in range(x+1):

        for j in range(y+1):

            for k in range(z+1):

                if i+j+k!=n:

                    a=[i,j,k]

                    out.append(a)

print(out)

**lists**

if \_\_name\_\_ == '\_\_main\_\_':

    N = int(input())

    l=[]

    for i in range(N):

        a=input().split()

        if a[0]=='insert':

            l.insert(int(a[1]),int(a[2]))

        elif a[0]=='print':

            print(l)

        elif a[0]=='append':

            l.append(int(a[1]))

        elif a[0]=='pop':

            l.pop()

        elif a[0]=='remove':

            l.remove(int(a[1]))

        elif a[0]=='sort':

            l.sort()

        elif a[0]=='reverse':

            l.reverse()

**nested list**

if \_\_name\_\_ == '\_\_main\_\_':

    i=0

    names=[]

    scores=[]

    n=int(input())

    while i<n:

        name = input()

        score = float(input())

        names+=[[name,score]]

        scores.append(score)

        i+=1

    a=sorted(list(set(scores)))[1]

    for i,j in sorted(names):

        if j==a:

            print(i)

**designer door mat**

n, m = map(int,input().split())

for i in range(1,n,2):

    print((i\*'.|.').center(m,'-'))

print('WELCOME'.center(m,'-'))

for i in range(n-2,-1,-2):

    print((i\*'.|.').center(m,'-'))

**string formatting**

def print\_formatted(number):

    # your code goes here

    w=len(bin(number)[2:])

    for i in range(1, number+1):

        d=str(i)

        o=oct(i)[2:]

        h=hex(i)[2:].upper()

        b=bin(i)[2:]

        print(d.rjust(w),o.rjust(w),h.rjust(w),b.rjust(w))

if \_\_name\_\_ == '\_\_main\_\_':

    n = int(input())

    print\_formatted(n)

**alphabet rangoli**

def print\_rangoli(size):

    # your code goes here

    import string as st

    a=st.ascii\_lowercase

    l=[]

    for i in range(0,n):

        s='-'.join(a[i:n])

        l.append((s[::-1]+s[1:]).center(4\*n-3,'-'))

    print('\n'.join(l[:0:-1]+l))

if \_\_name\_\_ == '\_\_main\_\_':

    n = int(input())

    print\_rangoli(n)

**minion game**

def minion\_game(string):

    # your code goes here

    vowel='AEIOU'

    n=len(string)

    pk=sum(n-i for i in range(n) if string[i] in vowel)

    ps=n\*(n+1)/2-pk

    if pk==ps:

        print('Draw')

    elif pk>ps:

        print('Kevin %d' % pk)

    else:

        print ('Stuart %d' % ps)

if \_\_name\_\_ == '\_\_main\_\_':

    s = input()

    minion\_game(s)

**merge the tools**

def merge\_the\_tools(string, k):

    # your code goes here

    a=[]

    la=0

    for i in string:

        la += 1

        if i not in a:

            a.append(i)

        if la==k:

            print(''.join(a))

            a=[]

            la=0

if \_\_name\_\_ == '\_\_main\_\_':

    string, k = input(), int(input())

    merge\_the\_tools(string, k)

**calendar module**

import datetime as dt

import calendar as cd

m,d,y = map (int,input().split())

date=dt.date(y,m,d)

print(cd.day\_name[date.weekday()].upper())

**time delta**

from datetime import datetime

# Complete the time\_delta function below.

def time\_delta(t1, t2):

    tformat='%a %d %b %Y %H:%M:%S %z'

    t1=datetime.strptime(t1, tformat)

    t2=datetime.strptime(t2, tformat)

    return str(int(abs((t1-t2).total\_seconds())))

**exceptions**

# Enter your code here. Read input from STDIN. Print output to STDOUT

n = int(input())

for i in range(n):

    try:

        a, b = map(int, input().split())

        print(a//b)

    except Exception as e:

        print("Error Code:",e)

if \_\_name\_\_ == '\_\_main\_\_':

    fptr = open(os.environ['OUTPUT\_PATH'], 'w')

    t = int(input())

    for t\_itr in range(t):

        t1 = input()

        t2 = input()

        delta = time\_delta(t1, t2)

        fptr.write(delta + '\n')

    fptr.close()

**map and lambda**

cube = lambda x: x \*\* 3

def fibonacci(n):

    a= 0

    b= 1

    c= 1

    for \_ in range(n):

        yield a

        a, b = b, a + b

if \_\_name\_\_ == '\_\_main\_\_':

    n = int(input())

    print(list(map(cube, fibonacci(n))))

**xml1**

import sys

import xml.etree.ElementTree as etree

def get\_attr\_number(node):

    # your code goes here

    a=len(node.attrib)

    b=sum(get\_attr\_number(i) for i in node)

    return a+b

if \_\_name\_\_ == '\_\_main\_\_':

    sys.stdin.readline()

    xml = sys.stdin.read()

    tree = etree.ElementTree(etree.fromstring(xml))

    root = tree.getroot()

    print(get\_attr\_number(root))

**xml2**

import xml.etree.ElementTree as etree

maxdepth = 0

def depth(elem, level):

    global maxdepth

    # your code goes here

    if (level == maxdepth):

        maxdepth += 1

    for i in elem:

        depth(i, level + 1)

if \_\_name\_\_ == '\_\_main\_\_':

    n = int(input())

    xml = ""

    for i in range(n):

        xml =  xml + input() + "\n"

    tree = etree.ElementTree(etree.fromstring(xml))

    depth(tree.getroot(), -1)

    print(maxdepth)

**standardize mobile numbers using decorators**

def wrapper(f):

    def fun(l):

        # complete the function

        f(['+91 ' + c[-10:-5] + ' '+ c[-5:]  for c in l])

    return fun

@wrapper

def sort\_phone(l):

    print(\*sorted(l), sep='\n')

if \_\_name\_\_ == '\_\_main\_\_':

    l = [input() for \_ in range(int(input()))]

    sort\_phone(l)

**decorators 2**

import operator

def person\_lister(f):

    def inner(people):

        # complete the function

        ordpeople=sorted(people, key = lambda x: (int(x[2])))

        return [f(i) for i in ordpeople]

    return inner

@person\_lister

def name\_format(person):

    return ("Mr. " if person[3] == "M" else "Ms. ") + person[0] + " " + person[1]

if \_\_name\_\_ == '\_\_main\_\_':

    people = [input().split() for i in range(int(input()))]

    print(\*name\_format(people), sep='\n')

**concatenate**

import numpy as np

a, b, c = map(int,input().split())

arr1 = np.array([input().split() for \_ in range(a)],int)

arr2 = np.array([input().split() for \_ in range(b)],int)

print(np.concatenate((arr1, arr2), axis = 0))

**zeros and ones**

import numpy as np

a= tuple(map(int,input().split(' ')))

print(np.zeros(a,int), np.ones(a,int), sep='\n')

**transpone and flatten**

import numpy as np

n,m = map(int,input().split())

a = []

for i in range(n):

    b = list(map(int,input().split()))

    a.append(b)

arr = np.array(a)

print(np.transpose(arr))

print(arr.flatten())

**eye and identity**

import numpy as np

n, m = map(int, input().split(' '))

a, b = (np.array([input().split(' ') for i in range(n)],int) for j in range(2))

print(a+b, a-b, a\*b, a//b, a%b, a\*\*b, sep='\n')

**array mathematics**

import numpy as np

n, m = map(int, input().split(' '))

a, b = (np.array([input().split(' ') for i in range(n)],int) for j in range(2))

print(a+b, a-b, a\*b, a//b, a%b, a\*\*b, sep='\n')

**floor ceil and rint**

import numpy as np

np.set\_printoptions(sign=' ')

a = np.array(input().split(),float)

np.set\_printoptions(sign=' ')

print(np.floor(a))

print(np.ceil(a))

print(np.rint(a))

**sum and prod**

import numpy as np

n,m = map(int, input().split())

a = np.array([input().split() for i in range(n)],int)

print(np.prod(np.sum(a, axis=0), axis=0))

**min and max**

import numpy as np

N, M = map(int, input().split())

a=np.array([input().split() for i in range(int(N))], int)

print(a.min(1).max())

**zipped**

# Enter your code here. Read input from STDIN. Print output to STDOUT

N, X = map(int,input().split())

l = []

for i in range(X):

    marks = map(float, input().split())

    l.append(marks)

for j in zip(\*l):

    print( sum(j)/len(j) )

**ginortS**

a=sorted(input(), key=lambda c: (c.isdigit() - c.islower(), c in '02468', c))

print(\*a, sep='')

**re.split**

regex\_pattern = r'[.,]'    # Do not delete 'r'.

import re

print("\n".join(re.split(regex\_pattern, input())))

**group groups**

import re

m = re.search(r'([a-zA-Z0-9])\1', input().strip())

print(m.group(1) if m else -1)

**re.findall**

import re

vowels = 'aeiou'

consonants = 'bcdfghjklmnpqrstvwxyz'

match = re.findall(r'(?<=[' + consonants + '])([' + vowels + ']{2,})(?=[' + consonants + '])', input(), flags=re.I)

print('\n'.join(match or ['-1']))

**introduction to set**

def average(array):

    arr=set(array)

    return sum(arr)/len(arr)

if \_\_name\_\_ == '\_\_main\_\_':

    n = int(input())

    arr = list(map(int, input().split()))

    result = average(arr)

    print(result)

**symmetric difference**

M = int(input().strip())

set\_m = set(map(int, input().strip().split(' ')))

N = int(input().strip())

set\_n = set(map(int, input().strip().split(' ')))

for j in sorted(set\_m ^ set\_n):

    print(j)

**no idea**

happiness = 0

n, m = map(int, input().strip().split())

arr = list(map(int, input().strip().split()))

like = set(map(int, input().strip().split()))

dislike = set(map(int, input().strip().split()))

for i in arr:

    if i in like:

        happiness += 1

    elif i in dislike:

        happiness -= 1

print(happiness)

**set.add**

n = int(input())

country= set([])

for i in range(n):

    country.add(input())

print(len(country))

**set.discard remove and pop**

n = int(input())

s = set(map(int, input().split()))

for i in range(int(input())):

    s1 = input().split()

    if s1[0] == 'pop':

        s.pop()

    elif s1[0] == 'remove':

        s.remove(int(s1[1]))

    elif s1[0] == 'discard':

        s.discard(int(s1[1]))

print(sum(s))

**set.union**

n= int(input())

s1 = set(map(int, input().split()))

b= int(input())

s2 = set(map(int, input().split()))

s3 = s1.union(s2)

print(len(s3))

**set.intersection**

n = int(input())

s1 = set(map(int, input().split()))

b = int(input())

s2 = set(map(int, input().split()))

print(len(s1 & s2))

**set.difference**

n = int(input())

s1 = set(map(int, input().split()))

b = int(input())

s2 = set(map(int, input().split()))

s3 = s1.difference(s2)

print(len(s3))

**set.symmetric difference**

n = int(input())

s1 = set(map(int, input().split()))

b = int(input())

s2 = set(map(int, input().split()))

s3=s1.symmetric\_difference(s2)

print(len(s3))

**set.mutations**

n = int(input())

s1 = set(map(int, input().split()))

m=int(input())

for \_ in range(m):

    operation = input().split()

    new\_set = set(map(int, input().split()))

    eval('s1.{}({})'.format(operation[0], new\_set))

print(sum(s1))

**captains room**

n= int(input())

l = input().split()

s = set(l)

for i in s:

    l.remove(i)

croom = s.difference(set(l)).pop()

print(croom)

**check subset**

n=int(input())

for i in range(n):

    a = int(input())

    s1 = set(map(int, input().split()))

    b = int(input())

    s2 = set(map(int, input().split()))

    if len(s1 - s2) == 0:

        print("True")

    else:

        print("False")

**check stric superset**

A = set(input().split())

n=int(input())

a = 0

for i in range(n):

    if A.issuperset(set(input().split()))==False:

        a += 1

if a != 0:

    print('False')

else:

    print('True')

**collection.counter**

from collections import Counter

X = int(input())

shoes = Counter(map(int, input().split()))

N = int(input())

tot = 0

for i in range(N):

    size, price = map(int, input().split())

    if shoes[size]:

        tot += price

        shoes[size] -= 1

print(tot)

**defaultdict tutorial**

from collections import defaultdict

d = defaultdict(list)

n, m = map(int, input().split())

for i in range(n):

    d[input()].append(str(i + 1))

for j in range(m):

    print(' '.join(d[input()]) or -1)

**collection.namedtuple**

from collections import namedtuple

n = int(input())

col = input().split()

tot= 0

for i in range(n):

    students = namedtuple('student', col)

    MARKS, CLASS, NAME, ID = input().split()

    student = students(MARKS, CLASS, NAME, ID)

    tot += int(student.MARKS)

print('{:.2f}'.format(tot/ n))

**ordered dict**

from collections import OrderedDict

od = OrderedDict()

n=int(input())

for \_ in range(n):

    item, space, price = input().rpartition(' ')

    od[item] = od.get(item, 0) + int(price)

for item, price in od.items():

    print(item, price)

**work order**

from collections import Counter

n = int(input())

l = []

for i in range(n):

    l.append(input().strip())

count = Counter(l)

print(len(count))

print(\*count.values())

**collection.deque**

from collections import deque

d = deque()

n=int(input())

for \_ in range(n):

    oper, val, \*args = input().split() + ['']

    eval(f'd.{oper} ({val})')

print(\*d)

**piling up**

ans = []

T = int(input())

for \_ in range(T):

    n = int(input())

    sl = list(map(int, input().split()))

    for \_ in range(n-1):

        if sl[0] >= sl[len(sl)-1]:

            a = sl[0]

            sl.pop(0)

        elif sl[0] < sl[len(sl)-1]:

            a = sl[len(sl)-1]

            sl.pop(len(sl)-1)

        else:

            pass

        if len(sl) == 1:

            ans.append("Yes")

        if((sl[0] > a) or (sl[len(sl)-1] > a)):

            ans.append("No")

            break

print("\n".join(ans))

**company logo**

from collections import Counter

if \_\_name\_\_ == '\_\_main\_\_':

    s = input()

    S = list(sorted(s))

    count=Counter(S)

    for i, j in count.most\_common(3):

        print(i, j)

**detect floating point number**

import re

t=int(input())

pattern = re.compile('^[-+]?[0-9]\*\.[0-9]+$')

for \_ in range(t):

    print(bool(pattern.match(input())))

**birthday cake candles**

def birthdayCakeCandles(candles):

    # Write your code here

    h=max(candles)

    c=0

    for i in range(candles\_count):

        if candles[i]==h:

            c +=1

    return c

if \_\_name\_\_ == '\_\_main\_\_':

    fptr = open(os.environ['OUTPUT\_PATH'], 'w')

    candles\_count = int(input().strip())

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

    result = birthdayCakeCandles(candles)

    fptr.write(str(result) + '\n')

**kangaroo**

#!/bin/python3

import math

import os

import random

import re

import sys

#

# Complete the 'kangaroo' function below.

#

# The function is expected to return a STRING.

# The function accepts following parameters:

#  1. INTEGER x1

#  2. INTEGER v1

#  3. INTEGER x2

#  4. INTEGER v2

#

#21 6 47 3

def kangaroo(x1, v1, x2, v2):

    # Write your code here

    if v1==v2 and x1==x2:

        return 'YES'

    elif v1==v2 and x1!=x2:

        return 'NO'

    else:

        x=(x2-x1)/(v1-v2)

        y=x1+v1\*x

        if float(x).is\_integer() and y>max(x1,x2):

            return 'YES'

        else:

            return 'NO'

if \_\_name\_\_ == '\_\_main\_\_':

    fptr = open(os.environ['OUTPUT\_PATH'], 'w')

    first\_multiple\_input = input().rstrip().split()

    x1 = int(first\_multiple\_input[0])

    v1 = int(first\_multiple\_input[1])

    x2 = int(first\_multiple\_input[2])

    v2 = int(first\_multiple\_input[3])

    result = kangaroo(x1, v1, x2, v2)

    fptr.write(result + '\n')

    fptr.close()

**viral advertising**

#!/bin/python3

import math

import os

import random

import re

import sys

#

# Complete the 'viralAdvertising' function below.

#

# The function is expected to return an INTEGER.

# The function accepts INTEGER n as parameter.

#

def viralAdvertising(n):

    # Write your code here

    a=[5]

    i=1

    b=2

    while i <n:

        nuovi=math.floor(a[i-1]/2)\*3

        a.append(nuovi)

        b+=math.floor(nuovi/2)

        i+=1

    return b

if \_\_name\_\_ == '\_\_main\_\_':

    fptr = open(os.environ['OUTPUT\_PATH'], 'w')

    n = int(input().strip())

    result = viralAdvertising(n)

**recursive digit sum**

#!/bin/python3

import math

import os

import random

import re

import sys

#

# Complete the 'superDigit' function below.

#

# The function is expected to return an INTEGER.

# The function accepts following parameters:

#  1. STRING n

#  2. INTEGER k

#

def superDigit(n, k):

    # Write your code here

    s=list(n)

    tot=0

    for i in range(len(s)):

        tot += int(s[i])

    tot=tot\*k

    if tot>=10:

        return superDigit(str(tot),1)

    else:

        return tot

if \_\_name\_\_ == '\_\_main\_\_':

    fptr = open(os.environ['OUTPUT\_PATH'], 'w')

    first\_multiple\_input = input().rstrip().split()

    n = first\_multiple\_input[0]

    k = int(first\_multiple\_input[1])

    result = superDigit(n, k)

    fptr.write(str(result) + '\n')

    fptr.close()

**insertion sort 1**

import math

import os

import random

import re

import sys

#

# Complete the 'insertionSort1' function below.

#

# The function accepts following parameters:

#  1. INTEGER n

#  2. INTEGER\_ARRAY arr

#

def insertionSort1(n, arr):

    # Write your code here

    e=arr[n-1]

    for i in range(2,n+1):

        if e>arr[-i]:

            arr[-i+1]=e

            break

        else:

            arr[-i+1]=arr[-i]

            print(\*arr)

    if arr[0]==arr[1]:

        arr[0]=e

    print(\*arr)

if \_\_name\_\_ == '\_\_main\_\_':

    n = int(input().strip())

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

    insertionSort1(n, arr)

**insertion sort 2**

#!/bin/python3

import math

import os

import random

import re

import sys

#

# Complete the 'insertionSort2' function below.

#

# The function accepts following parameters:

#  1. INTEGER n

#  2. INTEGER\_ARRAY arr

#

def insertionSort2(n, arr):

    # Write your code here

    for i in range(1,n):

        if arr[i]<arr[i-1]:

            for j in range(i):

                if arr[i]<arr[j]:

                    e=arr[i]

                    for k in range(i,j,-1):

                        arr[k]=arr[k-1]

                    arr[j]=e

                    break

        print(\*arr)

if \_\_name\_\_ == '\_\_main\_\_':

    n = int(input().strip())

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

    insertionSort2(n, arr)