```
print("Hello, World!")
```

```
import math
import os
import random
import re
import sys
```

```
if __name__ == '__main__':
    n = int(input().strip())
    if n%2!=0:
        print("Weird")
    else:
        if 2<=n<=5:
            print("Not Weird")
        elif 6<=n<=20:
            print("Weird")
        elif n>20:
             print("Not Weird")
```

Nuova sezione

```
if __name__ == '__main__':
    a = int(input())
    b = int(input())
    c= a+b
    print(c)
    d=a-b
    print(d)
    e= a*b
    print(e)
```

```
if __name__ == '__main__':
    a = int(input())
    b = int(input())
    c=a//b
    print(c)
    d=a/b
    print(d)
```

```
if __name__ == '__main__':
    n = int(input())
    for i in range(n):
        print(i**2)
```

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

```
if __name__ == '__main__':
    n = int(input())

    for i in range(1,n+1):
        print(i,end="")
```

```
n = int(input())
    arr = list(map(int, input().split()))
    scores=list(set(arr))
    scores.sort(reverse=True)
    runner=scores[1]
    print(runner)
```

```
students=[]
if __name__ == '__main__':
    for _ in range(int(input())):
        name = input()
        grade = float(input())
        students.append([name,grade])
        grades = sorted(set([student[1] for student in students]))

# the second lowest grade
second_lowest_grade = grades[1]
# names of students with the second lowest grade
second_lowest_students = [student[0] for student in students if student[1] == s
second_lowest_students.sort()


for student in second_lowest_students:
    print(student)



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()
    scores = student_marks[query_name]
    average = sum(scores) / len(scores)
    print(f"{average:.2f}")



if __name__ == '__main__':
    n = int(input())
    integer_list = map(int, input().split())
    t=tuple(integer_list)
    print(hash(t))



def swap_case(s):
    return s.swapcase()
```

```
def print_full_name(first, last):
     print(f"Hello {first} {last}! You just delved into python.")
```

```
def mutate_string(string, position, character):
    l=list(string)
    l[position]=character
    string=''.join(l)

    return string
```

```
def count_substring(string, sub_string):
    count=0
    for i in range(len(string) - len(sub_string) + 1):
        if string[i:i+len(sub_string)] == sub_string:
            count += 1
    return count
```

```
if __name__ == '__main__':
    x = int(input())
    y = int(input())
    z = int(input())
    n = int(input())

my_list=[[i,j,k] for i in range(x+1) for j in range(y+1) for k in range(z+1) if
print(my_list)
```

```
if __name__ == '__main__':
    N = int(input())
    lst=[]
    for _ in range(N):
        command = input().split()
        operation = command[0]
        if operation == 'insert':
            lst.insert(int(command[1]),int(command[2]))
        elif operation=='print':
             print(lst)
        elif operation=='remove':
             lst.remove(int(command[1]))
        elif operation=='append':
             lst.append(int(command[1]))
        elif operation=='sort':
             lst.sort()
        elif operation=='pop':
             lst.pop()
        elif operation=='reverse':
             lst.reverse()



def split_and_join(line):
    line=line.split(" ")
    line="-".join(line)

    return line
```

```
def minion_game(string):
    vocali='AEIOU'
    kev=0
    stu=0
    leng=len(string)
    for i in range(leng):
        if string[i] in vocali:
            kev += leng - i
        else:
            stu += leng - i

    if kev > stu:
        print(f"Kevin {kev}")
    elif stu > kev:
        print(f"Stuart {stu}")
    else:
        print("Draw")


if __name__ == '__main__':
  s = input()
  if any(i.isalnum() for i in s):
      print('True')
  else:
       print('False')
  if any(i.isalpha() for i in s):
        print('True')
  else:
        print('False')
  if any(i.isdigit() for i in s):
        print('True')
  else:
        print('False')
  if any(i.islower() for i in s):
        print('True')
  else:
        print('False')
  if any(i.isupper() for i in s):
        print('True')
  else:
        print('False')
```

```
#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(thi



def wrap(string, max_width):
     wrapped_string = ""
     for i in range(0, len(string), max_width):
         wrapped_string += string[i:i + max_width] + "\n"
     return wrapped_string.strip()






def solve(s):
    s=s.split(' ')
    for i in range(len(s)):
       s[i]=s[i].capitalize()

    return ' '.join(s)
```

```
N, M = map(int, input().split())
M= 3*N
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,"-")))
```

```
def print_formatted(number):
    width = len(bin(n)) - 2
    for i in range(1,n+1):
        decimal= str(i).rjust(width)
        octal= oct(i)[2:].rjust(width)
        hexadecimal=hex(i)[2:].upper().rjust(width)
        binary= bin(i)[2:].rjust(width)
        print(f"{decimal} {octal} {hexadecimal} {binary}")
```

```
def average(array):
    my_set=set(array)
    for i in array:
        average=sum(my_set)/len(my_set)
    return average
```

```
M=int(input())
l= set(map(int, input().split()))
N=int(input())
m=set(map(int, input().split()))
[print(i) for i in sorted( l ^ m)]
```

```
n=int(input())
my_set=set()
for i in range(n):
    country=input()
    my_set.add(country)
print(len(my_set))
```

```
n = int(input())
s = set(map(int, input().split()))
N=int(input())
for i in range(N):
    command = input().split()

    if command[0] == 'pop':
       s.pop()

    elif command[0] == 'remove':

         value = int(command[1])

         if value in s:

             s.remove(value)
    else:
         value = int(command[1])
         s.discard(value)
print(sum(s))
```

```
n=int(input())
english_sub= set(map(int,input().split()))
b=int(input())
french_sub=set(map(int,input().split()))
st_lea_one= english_sub.union(french_sub)
print (len((st_lea_one)))
```

```
n=int(input())
english_sub= set(map(int, input().split()))
b=int(input())
french_sub=set(map(int, input().split()))
diff= english_sub.difference(french_sub)
print(len(diff))
```

```
n=int(input())
english_sub= set(map(int, input().split()))
b=int(input())
french_sub=set(map(int, input().split()))
both_sub= english_sub.intersection(french_sub)
print(len(both_sub))
```

```
n=int(input())
english_sub= set(map(int, input().split()))
b=int(input())
french_sub=set(map(int, input().split()))
simm= english_sub.symmetric_difference(french_sub)
print(len(simm))
```

```
n=int(input())
A=set(map(int, input().split()))
for _ in range(int(input())):
    command= input().split()
    others=set(map(int, input().split()))
    if command[0]== "update":
        A.update(others)
    if command[0]=='intersection_update':
        A.intersection_update(others)
    if command[0]=='difference_update':
        A.difference_update(others)
    if command[0]=='symmertic_difference_update':
        A.symmetric_difference(others)
print(sum(A))
```

```
k=int(input())
l=list(map(int,input().split()))
l_set=set(l)
captain=(sum(l_set)*k - sum(l))// (k-1)
print(captain)
```

```
T=int(input())
for _ in range(T):
    n=int(input())
    a=set(map(int, input().split()))
    e=int(input())
    b=set(map(int, input().split()))
    if a.issubset(b):
       print('True')
    else:
        print('False')
```

```
A=set(map(int,input().split()))
is_superset=True
for i in range(int(input())):
    if not A.issuperset(set(map(int,input().split()))):
       is_superset=False
       break
print(is_superset)
```

```
n,m=map(int, input().split())
n=list(map(int,input().split()))
A=set(map(int,input().split()))
B=set(map(int,input().split()))
count=0
for i in n:
    if i in A:
        count+=1
    elif i in B:
        count-=1
print(count)
```

```
from collections import Counter
x=int(input())
shoes_size=list(map(int,input().split()))
inventory=Counter(shoes_size)
N=int(input())
count=0
for _ in range(N):
    size, price = map(int, input().split())
    if inventory[size] > 0:
        count += price
        inventory[size] -= 1
print(count)
```

```
from collections import defaultdict
n,m=map(int,input().split())

word = defaultdict(list)
for i in range(1,n+1):
    wo = input().strip()
    word[wo].append(i)
for _ in range(m):
    word_b = input().strip()
    if word_b in word:
        print(" ".join(map(str, word[word_b])))
    else:
        print(-1)
```

```
from collections import OrderedDict
N = int(input())
dict1 = OrderedDict()
for x in range(N):
    item, price = input().rsplit(' ', 1)
    dict1[item]= dict1.get(item, 0)+ int(price)

for item, price in dict1.items():
    print(f"{item} {price}")
```

```
from collections import namedtuple
marks=0
N=int(input())
Student=namedtuple('Student', input().split())
for x in range(N):
    s=Student(*input().split())
    marks+=int(s.MARKS)
print(f'{marks/N:.2f}')
```

```
import calendar
n=int()
m,d,y=input().split()
year=int(y)
month=int(m)
day=int(d)
out=calendar.weekday(year,month,day)
days=list(calendar.day_name)
print(days[out].upper())
```

```
import re
float_patt= r'^[+-]?\d*\.\d+$'
def valid_float(s):
    return bool(re.match(float_patt,s))
N=int(input())
for _ in range(N):
    s=input().strip()
    print(valid_float(s))
```

```
regex_pattern = r"[.,]" # Do not delete 'r'.
```

```
import re
s = str(input())
pattern = r'([a-zA-Z0-9])\1+'
stre = re.search(pattern, s)
if stre:
   print(stre.group(1))
else:
   print('-1')
```

```
regex_pattern = r"M{0,3}(CM|C{0,3}|CD|DC{0,3})(X[CLI]{0,1}|X{0,3}|LX{0,3})(I[XV
```

```
from re import match
n=int(input())
[print("YES" if match(r'^[789]\d{9}$', input()) else "NO") for _ in range(n)]
```

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

```
import numpy as np
n=input().split()
y=np.array(n,int)
z=y.reshape(3,3)
print(z)
```

```
import numpy as np
N,M=list(map(int,input().split()))
nm=np.array([list(map(int,input().split())) for _ in range(N)])
print(np.transpose(nm))
print(nm.flatten())
```

```
import numpy as np
n,m,p=list(map(int,input().split()))
arr1=np.array([list(map(int,input().split())) for i in range(n)])
arr2=np.array([list(map(int,input().split())) for j in range(m)])
print(np.concatenate((arr1,arr2), axis=0))
```

```
import numpy as np
n=list(map(int, input().split()))
print(np.zeros(n,int))
print(np.ones(n,int))
```

```
import numpy as np
np.set_printoptions(legacy='1.13')
n,m=map(int,input().split())
if (n==m):
    print(np.identity(n))
else:
    print(np.eye(n,m))
```

```
import numpy as np
n,m=input().split()
a = np.array([list(map(int, input().split())) for i in range(int(n))])
b = np.array([list(map(int, input().split())) for i in range(int(n))])
print(np.add(a,b))
print(np.subtract(a,b))
print(np.multiply(a,b))
print(np.floor_divide(a,b))
print(np.mod(a,b))
print(np.power(a,b))
```

```
import numpy as np
n,m=list(map(int,input().split()))
print(np.prod(np.sum(np.array([list(map(int,input().split())) for _ in range(n)
```

```
import numpy as np
n,m=map(int,input().split())
minore=np.min(np.array([list(map(int, input().split())) for _ in range(n)]),axi
print(np.max(minore))
```

```
import numpy as np
np.set_printoptions(legacy='1.13')
n=np.array(input().split(),float)
print(np.floor(n),np.ceil(n),np.rint(n),sep='\n')
```

```
import numpy as np
n,m=map(int,input().split())
array=np.array([list(map(int,input().split())) for _ in range(n)])
print(np.mean(array,axis=1),np.var(array,axis=0),np.around(np.std(array,axis=Nc
```

```
import numpy as np
n=int(input())
a=np.array([list(map(int,input().split())) for _ in range (n)])
b=np.array([list(map(int,input().split())) for _ in range (n)])

print(np.dot(a,b))
```

```
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))
```

```
import numpy as np
n=int(input())
a=np.array([list(map(float, input().split())) for i in range(n)])
print(round(np.linalg.det(a),2))
```

```
import numpy as np
p=list(map(float,input().split()))
x=int(input())
arr=np.array(p)
print(np.polyval(arr,x))
```

```
import re
s=input()
pattern = r'(?<=[bcdfghjklmnpqrstvwxyz])[aeiou]{2,}(?=[bcdfghjklmnpqrstvwxyz])'
match = re.findall(pattern, s, re.IGNORECASE)
if match:
	print('\n'.join(match))
else:
	print('-1')
```

```
import re

string = input()
k = input()
pattern = f"(?=({k}))"
matches = re.finditer(pattern, string)
indices = []

for match in matches:
    indices.append((match.start(), match.start() + len(k) - 1))

if not indices:
    print((-1, -1))
else:
    for x in indices:
        print(x)



cube = lambda x: x**3

def fibonacci(n):
    l=[]
    for i in range(n):
        if i == 0 or i == 1:
            l.append(i)
        else:
            x = l[i-1] + l[i - 2]
            l.append(x)
    return l



if __name__ == '__main__':


n,c=input().split()
arr=[]
for _ in range(int(c)):
    x=list(map(float,input().split()))
    arr.append(x)
colon=(list(zip(*arr)))
for i in colon:
    print(sum(i)/float(c))
```

```
def wrapper(f):
    def fun(l):
        for i in range(len(l)):
            l[i]="+91"+ " "+l[i][-10:-5]+" "+l[i][-5:]
        return f(l)
    return fun
```

ESERCIZIO 2

```
def birthdayCakeCandles(candles):
    tallest=max(candles)
    count_tallest = candles.count(tallest)
    return count_tallest


def kangaroo(x1, v1, x2, v2):
    result = (x2 - x1) / (v1 - v2)
    if result > 0 and result.is_integer():
       return "YES"
    elif v1-v2==0:
        return 'NO'
    else:
        return 'NO'


def viralAdvertising(n):
    share=5
    total_likes=0
    for _ in range(n):
        liked=share//2
        total_likes+=liked
        share= liked*3
    return total_likes


def superDigit(n, k):
    ini_sum = sum(int(digit) for digit in n) * k
    while ini_sum >= 10:
        temp = 0
        for digit in str(ini_sum):
            temp += int(digit)
        ini_sum = temp

    return ini_sum
```

```
def insertionSort1(n, arr):
     last_ele=arr[-1]
     i=n-2
     while i>=0 and arr[i]>last_ele:
         arr[i+1]=arr[i]
         print(" ".join(map(str, arr)))
         i -= 1
     arr[i + 1] = last_ele
     print(" ".join(map(str, arr)))
if __name__ == '__main__':
    n = int(input().strip())

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



def insertionSort2(n, arr):
    for i in range(1,n):
        start=arr[i]
        j=i-1
        while j >= 0 and arr[j] > start:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = start
        print(" ".join(map(str, arr)))
if __name__ == '__main__':
    n = int(input().strip())
```