**Hackerrank infytq**

1. **Sieve\_of\_Eratosathene**

Generate all the prime numbers from 2 to N and print it.

**Input Format**

A single integer N is given

**Constraints**

2 <= N <= 10^12

**Output Format**

All the prime numbers space-seperated

**Sample Input 0**

10

**Sample Output 0**

2 3 5 7

**Code:**

**#Sieve\_of\_Eratosathene**

**import math**

**n=int(input())**

**lis=list(range(2,n+1))**

**i=0**

**while(lis[i]<=math.sqrt(n)):**

**if(lis[i]!=0):**

**for j in range(lis[i]\*\*2-2,n-1,lis[i]):**

**lis[j]=0**

**i+=1**

**lis=list(filter(lambda a:a!=0,lis))**

**for i in range(len(lis)):**

**print(lis[i],end=" ")**

**2.check this:**

Check whether a given number is a Prime Number or Not . If it is a prime number print "yes" or otherwise print "no"

**Input Format**

A single integer N

**Constraints**

2 <= N <= 10^12

**Output Format**

yes or no

**Sample Input 0**

7

**Sample Output 0**

yes

**code:**

**# check whether the number is prime or not.**

**n=int(input())**

**if n > 1:**

**for i in range(2,n):**

**if (n % i) == 0:**

**print('no')**

**break**

**else:**

**print('yes')**

**break**

**else:**

**print('no')**

**3.Theater square-6:**

Theatre Square in the capital city of Berland has a rectangular shape with the size n × m meters. On the occasion of the city's anniversary, a decision was taken to pave the Square with square granite flagstones. Each flagstone is of the size a × a.

What is the least number of flagstones needed to pave the Square? It's allowed to cover the surface larger than the Theatre Square, but the Square has to be covered. It's not allowed to break the flagstones. The sides of flagstones should be parallel to the sides of the Square.

**Input Format**

The input contains three positive integer numbers in the first line: n,  m and a (1 ≤  n, m, a ≤ 109).

**Constraints**

**(1 ≤  n, m, a ≤ 109)**

**Output Format**

Write the needed number of flagstones.

**Sample Input 0**

6 6 4

**Sample Output 0**

4

**Code:**

**# theater square problem.**

**n,m,a=map(int,input().split())**

**if m%a==0:**

**r1=m//a**

**else:**

**r1=m//a+1**

**if n%a==0:**

**r2=n//a**

**else:**

**r2=n//a+1**

**print(r1\*r2)**

**4.way too long words-6:**

Sometimes some words like "localization" or "internationalization" are so long that writing them many times in one text is quite tiresome.

Let's consider a word too long, if its length is strictly more than 10 characters. All too long words should be replaced with a special abbreviation.

This abbreviation is made like this: we write down the first and the last letter of a word and between them we write the number of letters between the first and the last letters. That number is in decimal system and doesn't contain any leading zeroes.

Thus, "localization" will be spelt as "l10n", and "internationalization» will be spelt as "i18n".

You are suggested to automatize the process of changing the words with abbreviations. At that all too long words should be replaced by the abbreviation and the words that are not too long should not undergo any changes.

**Input Format**

The first line contains an integer n (1 ≤ n ≤ 100). Each of the following n lines contains one word. All the words consist of lowercase Latin letters and possess the lengths of from 1 to 100 characters.

**Constraints**

\*\*(1 ≤ n ≤ 100) \*\*

**Output Format**

Print n lines. The i-th line should contain the result of replacing of the i-th word from the input data.

**Sample Input 0**

4

word

localization

internationalization

pneumonoultramicroscopicsilicovolcanoconiosis

**Sample Output 0**

word

l10n

i18n

p43s

**code:**

**# way too long words.**

**n=int(input())**

**l=[]**

**a=[]**

**for i in range(n):**

**s=input()**

**if len(s)>10:**

**print(s[0]+str(len(s)-2)+s[-1])**

**else:**

**print(s)**

**5.mix String:**

Given 2 strings, s1, and s2 return a new string made of the first, middle and last char each input string For example: – mixString("America", "Japan") = ""AJrpan"

**Input Format**

A single line with two space-seperated strings

**Constraints**

Length of the strings are greater than 7 String is of odd length

**Output Format**

print a single integer that combines the first,middle and last character of two characters

**Sample Input 0**

America Japan

**Sample Output 0**

AJrpan

**Explanation 0**

First , middle and last character of string 1 is A,r,a Similarly First, middle and last character of string 2 is J,p,n

Thus the new string becomes AJrpan

**Code:**

**# mix string.**

**def mix\_up(s1,s2):**

**l1=len(s1)**

**l2=len(s2)**

**print(s1[0]+s2[0]+s1[int(l1/2)]+s2[int(l2/2)]+s1[-1]+s2[-1])**

**s1,s2=map(str,input().split())**

**mix\_up(s1,s2)**

**6.product sum:**

Accept two int values from user and return their product. If the product is greater than 1000, then return their sum

**Input Format**

First Line consists of a number 't' - number of testcase Each of the next t lines consist two space-seperated integers a,b - whose product you need to find

**Constraints**

1

**Output Format**

print in a new line the product or either sum of the given two integers

**Sample Input 0**

3

100 9

203 100

23 23

**Sample Output 0**

900

303

529

**Explanation 0**

First a=100 , b=9 ----- product is 900 which is not greater than 1000. so print 900 Second a=203 , b=100 ------ product is 20300 which is greater than 1000. so print their sum 303 Third a=23 , b=23 ------ product is 529 which is not greater than 1000. so print 529

**Code:**

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

**n=int(input())**

**for i in range(n):**

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

**if a\*b>1000:**

**print(a+b)**

**else:**

**print(a\*b)**

**7.----Rangoli-----:**

You are given an integer, N. Your task is to print an alphabet rangoli of size N. (Rangoli is a form of Indian folk art based on creation of patterns.)

**Input Format**

INTEGER

**Constraints**

**First print the half of the Rangoli in the given way and save each line in a list. Then print the list in reverse order to get the rest.**

**Output Format**

sample output

**Sample Input 0**

5

**Sample Output 0**

--------e--------

------e-d-e------

----e-d-c-d-e----

--e-d-c-b-c-d-e--

e-d-c-b-a-b-c-d-e

--e-d-c-b-c-d-e--

----e-d-c-d-e----

------e-d-e------

--------e--------

**Code:**

**# ---Rangoli---**

**import string**

**alpha=string.ascii\_lowercase**

**n=int(input())**

**L=[]**

**for i in range(n):**

**s="-".join(alpha[i:n])**

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

**print('\n'.join(L[:0:-1]+L))**

**8.sum of a list:**

Given a range of numbers. Iterate from 0th number to the end number and print the sum of the current number and previous number

For example x=[10,4,7,2,5,8]

you need to take all the numbers

step 1 : Taking Element 10 , There is no previous element Thus sum of previous and current element is 10

step 2: Taking 4, previous elemnt is 10 , Thus sum is 4+10 = 14

step 3: Taking 7, previous element is 4 , Thus sum is 7+4 = 11

step 4: Taking 2, previous element is 7 , Thus sum is 2+7 = 9

step 5: Taking 5, previous element is 2 , Thus sum is 5+2 = 7

step 6: Taking 8, previous element is 5 , Thus sum is 8+5 = 13

Your Output is 10 14 11 9 7 13

**Input Format**

First line contains a integer n - size of list Second line consist of a list with n elements

**Constraints**

No Constraints

**Output Format**

print in each line the sum of previous and current element

**Sample Input 0**

5

4 8 1 6 9

**Sample Output 0**

4

12

9

7

15

**Code:**

**# sum of a list**

**n=int(input())**

**l=input().split()**

**for i in range(n):**

**l[i]=int(l[i])**

**if i==0:**

**print(l[i])**

**else:**

**print(l[i]+l[i-1])**

**9.even indexing string:**

Accept string from the user and display only those characters which are present at an even index For example str = "pynative" so you should display ‘p’, ‘n’, ‘t’, ‘v’.

**Input Format**

A Single Integer

**Constraints**

No Constraints

**Output Format**

print space seperated character in the even index of the given string

**Sample Input 0**

pynative

**Sample Output 0**

p n t v

**code:**

**# even idexed string.**

**s=input()**

**n=len(s)**

**for i in range(n):**

**if i%2==0:**

**print(s[i],end=' ')**

**10.Ice cream parlor:**

**Sample Input**

2

4

5

1 4 5 3 2

4

4

2 2 4 3

**Sample Output**

1 4

1 2

**Code:**

**#ice cream-parlor**

**import math**

**import os**

**import random**

**import re**

**import sys**

**def icecreamParlor(m,n, arr):**

**for i in range(n):**

**for j in range(i+1,n):**

**if arr[i]+arr[j]==m:**

**return(i+1,j+1)**

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

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

**t = int(input())**

**for t\_itr in range(t):**

**m = int(input())**

**n = int(input())**

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

**result = icecreamParlor(m,n, arr)**

**fptr.write(' '.join(map(str, result)))**

**fptr.write('\n')**

**fptr.close()**

**11.Project Euler #2: Even Fibonacci numbers**

**Sample Input 0**

2

10

100

**Sample Output 0**

10

44

**Code:**

#!/bin/python3

k=int(input())

for x in range(k):

n=int(input())

a=1

b=2

sumEven=2

while True:

c=3\*b+2\*a

if c>=n:

break

sumEven+=c

a=2\*b+a

b=c

print(sumEven)

**11.kangaroo:**

**Sample Input 0**

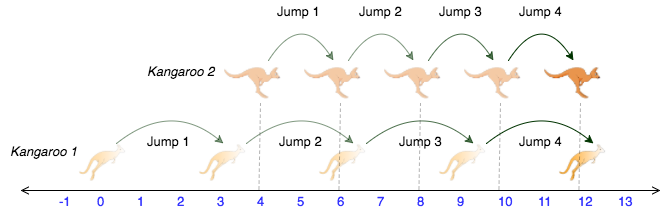
0 3 4 2

**Sample Output 0**

YES

**Explanation 0**

The two kangaroos jump through the following sequence of locations:



From the image, it is clear that the kangaroos meet at the same location (number12  on the number line) after same number of jumps (4 jumps), and we print YES.

**Code:**

**#kangaroo**

**import math**

**import os**

**import random**

**import re**

**import sys**

**def kangaroo(x1, v1, x2, v2):**

**for n in range(10000):**

**if((x1+v1)==(x2+v2)):**

**return "YES"**

**x1+=v1**

**x2+=v2**

**return "NO"**

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

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

**x1V1X2V2 = input().split()**

**x1 = int(x1V1X2V2[0])**

**v1 = int(x1V1X2V2[1])**

**x2 = int(x1V1X2V2[2])**

**v2 = int(x1V1X2V2[3])**

**result = kangaroo(x1, v1, x2, v2)**

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

**fptr.close()**

**12.find the runner up score:**

Given the participants' score sheet for your University Sports Day, you are required to find the runner-up score. You are given  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**

**Output Format**

Print the runner-up score.

**Sample Input 0**

5

2 3 6 6 5

**Sample Output 0**

5

**Code:**

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

**n = int(input())**

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

**print(sorted(list(set(arr)))[-2])**

**left rotation:**

**Sample Input**

5 4

1 2 3 4 5

**Sample Output**

5 1 2 3 4

**Code:**

**#Left rotation**

**import os**

**import sys**

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

**nd = input().split()**

**n = int(nd[0])**

**d = int(nd[1])**

**a = list(map(int, input().rstrip().split()))**

**b=[]**

**c=[]**

**for i in range(d,n):**

**b.append(a[i])**

**for i in range(d):**

**b.append(a[i])**

**for i in b:**

**c.append(str(i))**

**d=" ".join(c)**

**print(d)**

**13.String sorting-3:**

Write a program that accepts a comma separated sequence of words as input and prints the words in a comma-separated sequence after sorting them alphabetically. Suppose the following input is supplied to the program: **without,hello,bag,world** Then, the output should be: **bag,hello,without,world**

**Input Format**

A single line containing comma-seperated strings

**Constraints**

No Contraints

**Output Format**

Print all the sorted strings comma-seperatedly

**Sample Input 0**

without,hello,bag,world

**Sample Output 0**

bag,hello,without,world

**code:**

**#string sorting**

**s=list(map(str,input().split(",")))**

**s.sort()**

**l=len(s)**

**res=map(str,s)**

**print(','.join(res))**

**14.day of the programmer:**

A single integer denoting year .

**Constraints**

**Output Format**

Print the full date of *Day of the Programmer* during year  in the format dd.mm.yyyy, where dd is the two-digit day, mm is the two-digit month, and yyyy is .

**Sample Input 0**

2017

**Sample Output 0**

13.09.2017

**Code:**

**#day of the programmer**

**import math**

**import os**

**import random**

**import re**

**import sys**

**def dayOfProgrammer(year):**

**if(year==1918):**

**return '26.09.1918'**

**elif((year<=1917)&(year%4==0)or(year>1918)&(year%400==0 or(year%4==0)&(year%100!=0))):**

**return '12.09.%s'%year**

**else:**

**return '13.09.%s'%year**

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

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

**year = int(input().strip())**

**result = dayOfProgrammer(year)**

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

**fptr.close()**

**15. A very big sum:**

Calculate and print the sum of the elements in an array, keeping in mind that some of those integers may be quite large.

**Function Description**

Complete the *aVeryBigSum* function in the editor below. It must return the sum of all array elements.

aVeryBigSum has the following parameter(s):

* *ar*: an array of integers .

**Input Format**

The first line of the input consists of an integer n .  
The next line contains  n  space-separated integers contained in the array.

**Output Format**

Print the integer sum of the elements in the array.

**Sample Input**

5

1000000001 1000000002 1000000003 1000000004 1000000005

**Output**

5000000015

**Code:**

**#!/bin/python3**

**#very big sum.**

**import math**

**import os**

**import random**

**import re**

**import sys**

**# Complete the aVeryBigSum function below.**

**def aVeryBigSum(ar):**

**s=0**

**for i in range(n):**

**s=s+ar[i]**

**return(s)**

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

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

**n= int(input())**

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

**result = aVeryBigSum(ar)**

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

**fptr.close()**

**16. Electronic shop:**

**Function Description**

Complete the *getMoneySpent* function in the editor below. It should return the maximum total price for the two items within Monica's budget, or -1 if she cannot afford both items.

getMoneySpent has the following parameter(s):

* *keyboards*: an array of integers representing keyboard prices
* *drives*: an array of integers representing drive prices
* *b*: the units of currency in Monica's budget

**Input Format**

The first line contains three space-separated integers b, n and m, her budget, the number of keyboard models and the number of USB drive models.

The second line contains n  space-separated integers keyboard[i], the prices of each keyboard model.  
The third line contains m  space-separated integers drives, the prices of the USB drives.

**Output Format**

Print a single integer denoting the amount of money Monica will spend. If she doesn't have enough money to buy one keyboard and one USB drive, print -1 instead.

**Sample Input 0**

10 2 3

3 1

5 2 8

**Sample Output 0**

9

**Code:**

**#!/bin/python3**

**import os**

**import sys**

**def getMoneySpent(keyboards, drives, b):**

**return max([sum([x,y])for x in keyboards for y in drives if sum([x,y])<=b]+[-1])**

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

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

**bnm = input().split()**

**b = int(bnm[0])**

**n = int(bnm[1])**

**m = int(bnm[2])**

**keyboards = list(map(int, input().rstrip().split()))**

**drives = list(map(int, input().rstrip().split()))**

**moneySpent = getMoneySpent(keyboards, drives, b)**

**fptr.write(str(moneySpent) + '\n')**

**fptr.close()**

**17. Div times mod:**

Vasya likes to solve equations. Today he wants to solve (x div k)⋅(xmodk)=n, where div and mod stand for integer division and modulo operations (refer to the Notes below for exact definition). In this equation, k and n are positive integer parameters, and x is a positive integer unknown. If there are several solutions, Vasya wants to find the smallest possible x. Can you help him?

**Input Format**

The first line contains two integers n and k (1≤n≤106, 2≤k≤1000).

**Constraints**

nill

**Output Format**

Print a single integer x — the smallest positive integer solution to (x div k)⋅(xmodk)=n. It is guaranteed that this equation has at least one positive integer solution.

**Sample Input 0**

6 3

**Sample Output 0**

11

**Code:**

**# Div times mod.**

**n,k=map(int,input().split())**

**for i in range(1,((n\*k)+(n//k)+2)):**

**if (i//k)\*(i%k)==n:**

**print(i)**

**break**

**18. Duplicate sorting.**

Write a program that accepts a sequence of whitespace separated words as input and prints the words after removing all duplicate words and sorting them alphanumerically. Suppose the following input is supplied to the program: hello world and practice makes perfect and hello world again Then, the output should be: again and hello makes perfect practice world

**Input Format**

A single line with space seperated strings

**Constraints**

No Constraints

**Output Format**

Print the non-duplicate in sorted order with spaces inbetween

**Sample Input 0**

hello world and practice makes perfect and hello world again

**Sample Output 0**

again and hello makes perfect practice world

**code:**

**# Duplicate\_sorting**

**l=input().split()**

**m=set(l)**

**j=list(m)**

**j.sort()**

**s=""**

**for i in range(len(j)):**

**if i==(len(j)-1):**

**s=s+j[i]**

**else:**

**s=s+j[i]+" "**

**print(s)**

**19.camel scale:**

**Input Format**

A single line containing string .

**Output Format**

Print the number of words in string .

**Sample Input**

saveChangesInTheEditor

**Sample Output**

5

**Explanation**

String s contains five words:

1. save
2. Changes
3. In
4. The
5. Editor

Thus, we print 5 on a new line.

**Code:**

**# Camel case.**

**s=input()**

**c=1**

**for i in s:**

**if(ord(i)>=65 and ord(i)<=90):**

**c=c+1**

**print(c)**

**20. Migratory-birds:**

**Function Description**

Complete the *migratoryBirds* function in the editor below. It should return the lowest type number of the most frequently sighted bird.

migratoryBirds has the following parameter(s):

* *arr*: an array of integers representing types of birds sighted

**Input Format**

The first line contains an integer denoting  n, the number of birds sighted and reported in the array arr.  
The second line describes arr  as n  space-separated integers representing the type numbers of each bird sighted.

**Output Format**

Print the type number of the most common bird; if two or more types of birds are equally common, choose the type with the smallest ID number.

**Sample Input 0**

6

1 4 4 4 5 3

**Sample Output 0**

4

**Explanation 0**

The different types of birds occur in the following frequencies:

* Type 1:1  bird
* Type 2:0  birds
* Type 3:1  bird
* Type 4:3  birds
* Type 5:1  bird

The type number that occurs at the highest frequency is type 4, so we print 4 as our answer.

**Code:**

**#migratory Birds**

**import math**

**import os**

**import random**

**import re**

**import sys**

**def migratoryBirds(arr):**

**fr=[0,0,0,0,0,0]**

**for i in range(len(arr)):**

**fr[arr[i]]+=1**

**return fr.index(max(fr))**

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

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

**arr\_count = int(input().strip())**

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

**result = migratoryBirds(arr)**

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

**fptr.close()**

**21.papaya-1**

One hot summer day Sachin and his friend Abu decided to buy a Papaya. They chose the biggest and the ripest one, in their opinion. After that the Papaya was weighed, and the scales showed w kilos. They rushed home, dying of thirst, and decided to divide the berry, however they faced a hard problem.

Sachin and Abu are great fans of even numbers, that's why they want to divide the Papaya in such a way that each of the two parts weighs even number of kilos, at the same time it is not obligatory that the parts are equal. The boys are extremely tired and want to start their meal as soon as possible, that's why you should help them and find out, if they can divide the Papaya in the way they want. For sure, each of them should get a part of positive weight.

**Input Format**

The first (and the only) input line contains integer number w (1 ≤ w ≤ 100) — the weight of the Papaya bought by the boys.

**Constraints**

\*\* w (1 ≤ w ≤ 100)\*\*

**Output Format**

Print YES, if the boys can divide the Papaya into two parts, each of them weighing even number of kilos; and NO in the opposite case.

**Sample Input 0**

8

**Sample Output 0**

YES

**Code:**

**# PAPAYA**

**w=int(input())**

**if (w/2)%2==0:**

**print('YES')**

**else:**

**print('NO')**