Задачи курсера по Python:

**Целые числа, ввод-вывод, простые операции со строками**

1. Напишите программу, которая приветствует пользователя, выводя слово Hello, введенное имя и знаки препинания по образцу (см. пример входных и выходных данных). Программа должна считывать в строковую переменную значение и писать соответствующее приветствие. Обратите внимание, что после запятой должен обязательно стоять пробел, а перед восклицательным знаком пробела нет. Операцией конкатенации строк (+) пользоваться нельзя.

**Формат ввода**

Вводится строка, длина которой не превышает 100 символов.

**Формат вывода**

Выведите ответ на задачу.

Ответ: = input()  
print('Hello, ', a, '!', sep='')

2. Напишите программу, которая по данному числу N от 1 до 9 выводит на экран N пингвинов. Изображение одного пингвина имеет размер 5×9 символов, между двумя соседними пингвинами также имеется пустой (из пробелов) столбец. Разрешается вывести пустой столбец после последнего пингвина. Для упрощения рисования скопируйте пингвина из примера в среду разработки.

**Формат ввода**

Вводится натуральное число.

**Формат вывода**

Выведите ответ на задачу.

**Примечания**

* В задачах нашего курса не нужно проверять ограничения входных данных: гарантируется, что введут данные, соответствующие условию. Т. е. например в этой задаче введенное число N точно будет не меньше 1 и не больше 9.
* Напоминаем, что во всех задачах этого курса решения должны выдавать в точности требуемый ответ. В частности, не надо выводить призыв к вводу вроде "Введите количество пингвинов". Пожалуйста, протестируйте свое решение на примерах ввода/вывода, а также придумайте свои удовлетворяющие условию примеры и протестируйте свое решение на них.
* Учтите, что вывод данных на экран производится построчно, а не попингвинно.
* Не забудьте, что для вывода бекслеша надо написать два бекслеша подряд.
* Эта задача относительно сложная. Вы можете пока что ее пропустить, и вернуться позже.

\_~\_

(o o)

/ V \

/( \_ )\

^^ ^^

n = int(input())  
print(' \_~\_ ' \* n, sep='')  
print(' (o o) ' \* n, sep='')  
print(' / V \ ' \* n, sep='')  
print('/( \_ )\ ' \* n, sep='')  
print(' ^^ ^^ ' \* n, sep='')

3. N школьников делят K яблок поровну, не делящийся остаток остается в корзинке. Сколько яблок достанется каждому школьнику?

**Формат ввода**

Программа получает на вход числа N и K — натуральные, не превышают 10000.

**Формат вывода**

Выведите ответ на задачу.

Ответ:

n = int(input())  
k = int(input())  
print(k // n)

4. N школьников поделили K яблок поровну, не делящийся остаток остался в корзинке. Сколько яблок осталось в корзинке?

**Формат ввода**

Программа получает на вход числа N и K — натуральные, не превышают 10000.

**Формат вывода**

Выведите ответ на задачу.

Ответ:

n = int(input())  
k = int(input())  
print(k % n)

5. **Формат ввода**

Вводится целое неотрицательное число N (N≤100).

**Формат вывода**

Выведите 2ᴺ.

n = int(input())  
print(2 \*\* n)

6. Дано натуральное число. Выведите его последнюю цифру.

**Формат ввода**

Вводится единственное целое положительное число (гарантируется, что оно не превышает 10000).

**Формат вывода**

Выведите ответ на задачу.

n = int(input())  
print(n % 10)

7. Дано положительное двузначное число. Найдите число десятков в нем.

**Формат ввода**

Вводится единственное число (гарантируется, что оно соответствует условию задачи).

**Формат вывода**

Выведите ответ на задачу.

n = int(input())  
print(n // 10)

8. Дано натуральное число. Найдите цифру, стоящую в разряде десятков в его десятичной записи (вторую справа цифру или 0, если число меньше 10).

**Замечание**

Предполагается решение этой задачи без использования строковых методов. Пожалуйста, пользуйтесь арифметикой.

**Формат ввода**

Вводится единственное число.

**Формат вывода**

Выведите ответ на задачу.

n = int(input())  
print(n % 100 // 10)

9. Дано трехзначное число. Найдите сумму его цифр.

**Формат ввода**

Вводится целое положительное число. Гарантируется, что оно соответствует условию задачи.

**Формат вывода**

Выведите ответ на задачу.

n = int(input())  
print(n // 100 + n % 100 // 10 + n % 10)

10. Запишите букву 'A' (латинскую, заглавную) 100 раз подряд. Сдайте на проверку программу, которая выводит эту строчку (только буквы, без кавычек или пробелов).

print('A'\*100)

11. Дано число N. С начала суток прошло N минут. Определите, сколько часов и минут будут показывать электронные часы в этот момент.

**Формат ввода**

Вводится число N — целое, положительное, не превышает 10⁷.

**Формат вывода**

Программа должна вывести два числа: количество часов (от 0 до 23) и количество минут (от 0 до 59).

Учтите, что число N может быть больше, чем количество минут в сутках.

**Примеры**

Тест 1

Входные данные:

150

Вывод программы:

2 30

Ответ

t = int(input())  
print(t // 60 % 24, t % 60)

12. Пирожок в столовой стоит A рублей и B копеек. Определите, сколько рублей и копеек нужно заплатить за N пирожков.

**Формат ввода**

Программа получает на вход три числа: A, B, N — целые, неотрицательные, не превышают 10000.

**Формат вывода**

Программа должна вывести два числа: стоимость покупки в рублях и копейках.

**Ответ**

a = int(input())  
b = int(input())  
n = int(input())  
total = (a \* 100 + b) \* n  
print(total // 100, total % 100)

12. Напишите программу, которая считывает целое число и выводит текст, аналогичный приведенному в примере (важно в точности соблюдать вывод программы: обратите внимание на пробелы и на точки). Нельзя пользоваться конкатенацией строк, используйте print с несколькими параметрами.

**Формат ввода**

Вводится целое число (гарантируется, что число находится в диапазоне от -1000 до +1000).

**Формат вывода**

Выведите две строки, согласно образцу.

Тест 1

Входные данные:

179

Вывод программы:

The next number for the number 179 is 180.

The previous number for the number 179 is 178.

Ответ

a = int(input())  
print('The next number for the number', a, 'is', a + 1, end='.\n')  
print('The previous number for the number', a, 'is', a - 1, end='.')

13. Вводится число 0 или 1, необходимо вывести 1 или 0 соответственно.

**Формат ввода**

Число 0 или 1.

**Формат вывода**

Число 0 или 1.

Ответ

a = int(input())  
print((a - 1) \*\* a)

14. Дано целое число N. Выведите следующее за ним четное число.

**Формат ввода**

Вводится целое положительное число, не превышающее 1000.

**Формат вывода**

Выведите ответ на задачу.

a = int(input())  
print((a // 2 \* 2) + 2)

15. Заданное число N записали 100 раз подряд и затем возвели в квадрат. Что получилось?

**Формат ввода**

Вводится целое неотрицательное число N не превышающее 1000.

**Формат вывода**

Выведите ответ на задачу.

a = input()  
print(int(a \* 100) \*\* 2)

16. Длина Московской кольцевой автомобильной дороги — 109 километров. Байкер Вася стартует с нулевого километра МКАД и едет со скоростью v километров в час. На какой отметке он остановится через t часов?

**Формат ввода**

Программа получает на вход значение v и t. Если v>0, то Вася движется в положительном направлении по МКАД, если же значение v<0, то в отрицательном. (Гарантируется, что исходные числа — целые и находятся в промежутке от -1000 до +1000).

**Формат вывода**

Программа должна вывести целое число от 0 до 108 — номер отметки, на которой остановится Вася.

v = int(input())  
t = int(input())  
print(v \* t % 109)

17. Электронные часы показывают время в формате h:mm:ss, то есть сначала записывается количество часов (число от 0 до 23), потом обязательно двузначное количество минут, затем обязательно двузначное количество секунд. Количество минут и секунд при необходимости дополняются до двузначного числа нулями.

С начала суток прошло N секунд. Выведите, что покажут часы.

**Формат ввода**

Вводится число N — целое, неотрицательное.

**Формат вывода**

Выведите показания часов, соблюдая формат.

**Примечания**

Вывести числа можно поциферно.

Ответ:

1.

s = int(input())  
print(s // 3600 % 24, str(s % 3600 // 600) + str(s % 3600 // 60 % 10), str(s % 60 // 10) + str(s % 60 % 10), sep=':')

не правильно: Precompile check failed:

/solution//solution.py:2:80: E501 line too long (117 > 79 characters)

2.

s = int(input())  
print(s // 3600 % 24, end=':')  
print(str(s % 3600 // 600) + str(s % 3600 // 60 % 10), end=':')  
print(str(s % 60 // 10) + str(s % 60 % 10))

Правильно.

18. Даны два момента времени в пределах одних и тех же суток. Для каждого момента указан час, минута и секунда. Известно, что второй момент времени наступил не раньше первого.

Определите сколько секунд прошло между двумя моментами времени.

**Формат ввода**

Программа на вход получает шесть целых чисел через перевод строки. Первые три целых числа соответствуют часам, минутам и секундам первого момента, следующие три числа соответствуют второму моменту.

Часы задаются числом от 0 до 23 включительно. Минуты и секунды — от 0 до 59.

**Формат вывода**

Выведите число секунд между этими моментами времени.

h1 = int(input()) \* 3600  
m1 = int(input()) \* 60  
s1 = int(input())  
h2 = int(input()) \* 3600  
m2 = int(input()) \* 60  
s2 = int(input())  
print((h2 + m2 + s2) - (h1 + m1 + s1))

19. За день машина проезжает N километров. Сколько дней нужно, чтобы проехать маршрут длиной M километров?

**Формат ввода**

Программа получает на вход числа N и M (целые, положительные).

**Формат вывода**

Выведите ответ на задачу.

n = int(input())  
m = int(input())  
print((m + n - 1) // n)

20. Улитка ползет по вертикальному шесту высотой H метров, поднимаясь за день на A метров, а за ночь спускаясь на B метров. На какой день улитка доползет до вершины шеста?

**Формат ввода**

Программа получает на вход целые H, A, B. Гарантируется, что A > B ≥ 0.

**Формат вывода**

Программа должна вывести одно натуральное число.

\*\*Первое, что нам надо уяснить, это то, что улитка достигнет вершины днем. То есть в последний день он сделает шаг A вверх и вниз уже не пойдет. То есть кол-во дней будет на 1 меньше количества ночей.

А теперь составим уравнение, где x - это кол-во дней. А - высота подъема, В - спуск, H - высота шеста.

A\*x - B\*(x-1) = H

B умножается на x - 1, потому что ночей на один меньше

Из уравнения выше находим x

x = (H - B) / (A - B)

В уравнении может получится дробное число, поэтому надо округлить результат вверх. На видео выше был такой пример. Окончательный результат на питоне:

h = int(input())  
a = int(input())  
b = int(input())  
print((h - b - 1) // (a - b) + 1)

\*\*\*2 вариант

Первое решение задачи, что приходит на ум это Высоту H разделить на A - B: H // (A - B)

Но тогда мы закончим спуском. То есть, если при делении у нас получится 2, это значит, что мы поднялись, спустились. Еще раз поднялись и спустились. Но как мы уже знаем, последним должен быть шаг вверх.

Поэтому поступим так(нарисуйте, чтобы представить лучше):

Сделаем шаг A вверх и начнем считать оттуда. Тогда у нас первым шагом будет спуск, а последним подъем, как нам и надо.

Запишем это формулой. Так как мы сделали шаг вверх вначале, высота шеста уменьшилась на A: (H - A). Теперь делим это на A - B. Так как мы сначала сделали шаг вверх и начали считать оттуда, надо к результату добавить 1.

x = (H - A) / (A - B) + 1

h = int(input())  
a = int(input())  
b = int(input())  
print((h – a - 1) // (a - b) + 2)

21. Дано четырехзначное число. Определите, является ли его десятичная запись симметричной. Если число симметричное, то выведите 1, иначе выведите любое другое целое число. Число может иметь меньше четырех знаков, тогда нужно считать, что его десятичная запись дополняется слева нулями.

**Формат ввода**

Вводится единственное число.

**Формат вывода**

Выведите ответ на задачу.

**Примечание**

Десятичная запись числа симметрична, если при прочтении слева направо и справа налево получается одно и то же число.

n = int(input())  
n1 = n % 100  
n2 = str(n % 1000 // 100) + str(n // 1000)  
print(n1 - int(n2) + 1)

22. Напишите программу, которая считывает два целых числа A и B и выводит наибольшее значение из них. Числа — целые от 1 до 1000.

При решении задачи можно пользоваться только целочисленными арифметическими операциями. Нельзя пользоваться нелинейными конструкциями: ветвлениями, циклами, функциями.

**Формат ввода**

Вводятся два числа.

**Формат вывода**

Выведите ответ на задачу.

a = input()  
b = input()  
c = int(a) // int(b)  
d = 0 \*\* c  
print(a \* (0 \*\* d), b \* (0 \*\* c), sep='')

23. В этой задаче необходимо проверить, делится ли число A на число B нацело. Использовать можно только арифметические операции, использование любых видов ветвлений, функций и т.п. запрещено.

**Формат ввода**

Вводятся два натуральных числа A и B.

**Формат вывода**

Выведите "YES", если A кратно B и "NO" в противном случае.

a = int(input())  
b = int(input())  
c = a % b  
d = 0 \*\* c  
print('YES' \* (0 \*\* c), 'NO' \* 0 \*\* d, sep='')

\*\* c = 0, если число делится нацело

0 \*\* 0 = 1, а 0 в любой другой степени = 0

d может быть равно только 0 или 1

Если d = 1, то YES пишется 1 раз, а NO 0 раз ( так как 0 в степени 1 = 0)

Если d = 0, то YES пишется 0 раз, а NO 1 ( так как 0 в степени 0 = 1)

### Условный оператор и цикл while

1. Напишите программу, которая считывает два целых числа A и B и выводит наибольшее значение из них. Числа — целые от 1 до 1000.

**Формат ввода**

Вводятся два числа.

**Формат вывода**

Выведите ответ на задачу.

a = int(input())  
b = int(input())  
if a >= b:  
 print(a)  
else:  
 print(b)

2. Даны два целых числа. Программа должна вывести число "1", если первое число больше второго, число "2", если второе больше первого или число "0", если они равны.

**Формат ввода**

Вводятся два целых числа.

**Формат вывода**

Выведите ответ на задачу.

**Примечания**

Эту задачу желательно решить с использованием каскадных инструкций if... elif... else.

a = int(input())  
b = int(input())  
if a > b:  
 print(1)  
elif a == b:  
 print(0)  
else:  
 print(2)

3. Даны три целых числа. Найдите наибольшее из них (программа должна вывести ровно одно целое число).

Какое наименьшее число операторов сравнения (>, <, >=, <=) необходимо для решения этой задачи?

**Формат ввода**

Вводится три целых числа.

**Формат вывода**

Выведите ответ на задачу.

a = int(input())  
b = int(input())  
c = int(input())  
if a >= b:  
 if a >= c:  
 print(a)  
 else:  
 print(c)  
elif b >= c:  
 print(b)  
else:  
 print(c)

4. Дано натуральное число. Требуется определить, является ли год с данным номером високосным. Если год является високосным, то выведите YES, иначе выведите NO. Напомним, что в соответствии с григорианским календарем, год является високосным, если его номер кратен 4, но не кратен 100, или же если он кратен 400.

**Формат ввода**

Вводится одно натуральное число.

**Формат вывода**

Выведите ответ на задачу.

a = int(input())  
if a % 4 == 0 and a % 100 != 0 or a % 400 == 0:  
 print('YES')  
else:  
 print('NO')

5. Шахматный король ходит по горизонтали, вертикали и диагонали, но только на 1 клетку. Даны две различные клетки шахматной доски, определите, может ли король попасть с первой клетки на вторую одним ходом.

**Формат ввода**

Программа получает на вход четыре числа от 1 до 8 каждое, задающие номер столбца и номер строки сначала для первой клетки, потом для второй клетки.

**Формат вывода**

Программа должна вывести YES, если из первой клетки ходом короля можно попасть во вторую или NO в противном случае.

x = int(input())  
y = int(input())  
x1 = int(input())  
y1 = int(input())  
if -1 <= x1 - x <= 1:  
 if -1 <= y1 - y <= 1:  
 print('YES')  
 else:  
 print('NO')  
else:  
 print('NO')

6. В доме несколько подъездов. В каждом подъезде одинаковое количество квартир. Квартиры нумеруются подряд, начиная с единицы. Может ли в некотором подъезде *первая* квартира иметь номер x, а *последняя* – номер y?

**Формат ввода**

Вводятся два натуральных числа x и y (x ≤ y).

**Формат вывода**

Выведите слово YES (заглавными латинскими буквами), если такое возможно, и NO в противном случае.

x = int(input())  
y = int(input())  
if y % (y - (x - 1)) == 0:  
 print('YES')  
else:  
 print('NO')

7. Заданы две клетки шахматной доски. Если они покрашены в один цвет, то выведите слово YES, а если в разные цвета – то NO.

**Формат ввода**

Вводятся 4 числа - координаты клеток.

**Формат вывода**

Выведите ответ на задачу.

Мой вариант:

x = int(input())  
y = int(input())  
x1 = int(input())  
y1 = int(input())  
if y1 == y or (y1 - y + **2**) % **2** == **0**:  
 if x == x1 or (x1 - x + **2**) % **2** == **0**:  
 print('YES')  
 else:  
 print('NO')  
elif (x1 - x + **2**) % **2** != **0**:  
 print('YES')  
else:  
 print('NO')

Самый красивый вариант:

x= int(input())  
y = int(input())  
x1 = int(input())  
y1 = int(input())  
if (x + y + x1 + y1) % **2** == **0**:  
 print('YES')  
else:  
 print('NO')

8. Шоколадка имеет вид прямоугольника, разделенного на n×m долек. Шоколадку можно один раз разломить по прямой на две части. Определите, можно ли таким образом отломить от шоколадки часть, состоящую ровно из k долек.

**Формат ввода**

Программа получает на вход три числа: n, m, k.

**Формат вывода**

Программа должна вывести одно из двух слов: YES или NO.

n = int(input())  
m = int(input())  
k = int(input())  
if n\*m > k and (k % n == **0** or k % m == **0**):  
 print('YES')  
else:  
 print('NO')

\*\*не сам

9. Для данного числа n<100 закончите фразу “На лугу пасется...” одним из возможных продолжений: “n коров”, “n корова”, “n коровы”, правильно склоняя слово “корова”.

**Формат ввода**

Вводится натуральное число.

**Формат вывода**

Программа должна вывести введенное число n и одно из слов: korov, korova или korovy. Между числом и словом должен стоять ровно один пробел.

x = int(input())  
if x % **10** == **1** and x != **11**:  
 print(x**,** 'korova')  
elif **2** <= x % **10** <= **4** and not **12** <= x <= **14**:  
 print(x**,** 'korovy')  
else:  
 print(x**,** 'korov')

10. В математике функция sign(x) (знак числа) определена так:

sign(x)=1, если x>0,

sign(x)=-1, если x<0,

sign(x)=0, если x=0.

Для данного числа x выведите значение sign(x).

**Формат ввода**

Вводится одно целое число.

**Формат вывода**

Выведите ответ на задачу.

**Примечания**

Эту задачу желательно решить с использованием каскадных инструкций if... elif... else.

n = int(input())  
if n > **0**:  
 print(**1**)  
elif n == **0**:  
 print(**0**)  
else:  
 print(-**1**)

11. Даны координаты двух точек на плоскости, требуется определить, лежат ли они в одной координатной четверти или нет (все координаты отличны от нуля).

**Формат ввода**

Вводятся 4 числа: координаты первой точки (x1,y1) и координаты второй точки (x2,y2).

**Формат вывода**

Программа должна вывести слово YES, если точки находятся в одной координатной четверти, в противном случае вывести слово NO.

x = int(input())  
y = int(input())  
x1 = int(input())  
y1 = int(input())  
if x \* x1 > **0** and y \* y1 > **0**:  
 print('YES')  
else:  
 print('NO')

12. На доске стоит белая шашка. Требуется определить, может ли она попасть в заданную клетку, делая ходы по правилам и не пользуясь ходами дамки (т. е. не используя возможность перемещаться назад после превращения в дамку). Белые шашки могут ходить по клеткам одного цвета по диагонали вверх-влево или вверх-вправо. Ходов может быть несколько!

**Примечания**

Доска имеет размер 8x8, вертикали и горизонтали нумеруются числами от 1 до 8 начиная с левого нижнего угла. Исходная и конечная клетки не совпадают.

**Формат ввода**

Вводится клетка, где стоит шашка, а затем клетка, куда шашка должна попасть.

Каждая клетка описывается номером вертикали, а затем номером горизонтали. Под номером вертикали имеется в виду не номер по вертикали, а номер вертикальной линии считая слева направо. Аналогичная формулировка используется для номера горизонтали: нумерация идет снизу вверх. Например, клетка A2 кодируется как 1 2.

**Формат вывода**

Выведите слово YES (заглавными буквами), если шашка может попасть из начальной клетки в указанную, и NO в противном случае.

x = int(input())  
y = int(input())  
x1 = int(input())  
y1 = int(input())  
# n=abs(x1-x)  
n = x1 - x  
if n < **0**:  
 n = n \* -**1**if (x + y + x1 + y1) % **2** == **0** and n <= (y1 - y):  
 print('YES')  
else:  
 print('NO')

13. Даны три стороны треугольника a,b,c. Определите тип треугольника с заданными сторонами. Выведите одно из четырех слов: rectangular для прямоугольного треугольника, acute для остроугольного треугольника, obtuse для тупоугольного треугольника или impossible, если треугольника с такими сторонами не существует (считаем, что вырожденный треугольник тоже невозможен).

**Формат ввода**

Вводятся три целых числа.

**Формат вывода**

Выведите ответ на задачу.

a = int(input())  
b = int(input())  
c = int(input())  
if (a + b > c) and (a + c > b) and (b + c > a) and a > **0** and b > **0** and c > **0**:  
 if a \*\* **2** == b \*\* **2** + c \*\* **2** or\  
 b \*\* **2** == a \*\* **2** + c \*\* **2** or\  
 c \*\* **2** == a \*\* **2** + b \*\* **2**:  
 print('rectangular')  
 elif a \*\* **2** > b \*\* **2** + c \*\* **2** or\  
 b \*\* **2** > a \*\* **2** + c \*\* **2** or\  
 c \*\* **2** > a \*\* **2** + b \*\* **2**:  
 print('obtuse')  
 elif a \*\* **2** < b \*\* **2** + c \*\* **2** or\  
 b \*\* **2** < a \*\* **2** + c \*\* **2** or\  
 c \*\* **2** < a \*\* **2** + b \*\* **2**:  
 print('acute')  
else:  
 print('impossible')

14. Даны три целых числа A, B, C. Определить, есть ли среди них хотя бы одно четное и хотя бы одно нечетное.

**Формат ввода**

Числа A, B, C, не превышающие по модулю 10000.

**Формат вывода**

Одна строка – "YES" или "NO".

Моё

a = int(input())  
b = int(input())  
c = int(input())  
if (a % **2** == **0** or b % **2** == **0** or c % **2** == **0**) and\  
 (a % **2** != **0** or b % **2** != **0** or c % **2** != **0**):  
 print('YES')  
else:  
 print('NO')

красивей

= int(input())  
b = int(input())  
c = int(input())  
if (a + b) % **2** != **0** or (b + c) % **2** != **0** or (c + a) % **2** != **0**:  
 print('YES')  
else:  
 print('NO')

15. Дано три числа. Упорядочите их в порядке неубывания. Программа должна считывать три числа a,b,c, затем программа должна менять их значения так, чтобы стали выполнены условия a≤b≤c, затем программа выводит тройку a,b,c.

**Формат ввода**

Вводятся три числа.

**Формат вывода**

Выведите ответ на задачу.

**Примечания**

Дополнительные ограничения: нельзя использовать дополнительные переменные, то есть единственной допустимой операцией присваивания является обмен значений двух переменных типа (a, b) = (b, a). Кстати, аналогично можно делать присваивания любого количества переменных. Например, ввод в этой задаче можно оформить так:

a, b, c = int(input()), int(input()), int(input())

a**,** b**,** c = int(input())**,** int(input())**,** int(input())  
if a <= b and b <= c:  
 print(a**,** b**,** c)  
elif a >= b and b <= c:  
 if a <= c:  
 (a**,** b) = (b**,** a)  
 print(a**,** b**,** c)  
 else:  
 (a**,** b**,** c) = (b**,** c**,** a)  
 print(a**,** b**,** c)  
elif a <= b and b >= c:  
 if a >= c:  
 (a**,** b**,** c) = (c**,** a**,** b)  
 print(a**,** b**,** c)  
 else:  
 (b**,** c) = (c**,** b)  
 print(a**,** b**,** c)  
else:  
 (a**,** c) = (c**,** a)  
 print(a**,** b**,** c)

красиво:

a**,** b**,** c = int(input())**,** int(input())**,** int(input())  
if b > c:  
 (b**,** c) = (c**,** b)  
if a > c:  
 (a**,** c) = (c**,** a)  
if a > b:  
 (a**,** b) = (b**,** a)  
print(a**,** b**,** c)

16. Даны три целых числа. Определите, сколько среди них совпадающих. Программа должна вывести одно из чисел: 3 (если все совпадают), 2 (если два совпадает) или 0 (если все числа различны).

**Формат ввода**

Вводятся три целых числа.

**Формат вывода**

Выведите ответ на задачу.

a**,** b**,** c = int(input())**,** int(input())**,** int(input())  
if a == b == c:  
 print(**3**)  
elif a == b or a == c or c == b:  
 print(**2**)  
else:  
 print(**0**)

17. За многие годы заточения узник замка Иф проделал в стене прямоугольное отверстие размером D×E. Замок Иф сложен из кирпичей, размером A×B×C. Определите, сможет ли узник выбрасывать кирпичи в море через это отверстие (очевидно, стороны кирпича должны быть параллельны сторонам отверстия).

**Формат ввода**

Программа получает на вход числа A, B, C, D, E.

**Формат вывода**

Программа должна вывести слово YES или NO.

a**,** b**,** c**,** d**,** e = int(input())**,** int(input())**,** \  
 int(input())**,** int(input())**,** int(input())  
if a <= d and (b <= e or c <= e):  
 print('YES')  
elif b <= d and (a <= e or c <= e):  
 print('YES')  
elif c <= d and (a <= e or b <= e):  
 print('YES')  
else:  
 print('NO')

18. Есть две коробки, первая размером A₁×B₁×C₁, вторая размером A₂×B₂×C₂. Определите, можно ли разместить одну из этих коробок внутри другой, при условии, что поворачивать коробки можно только на 90 градусов вокруг ребер.

**Формат ввода**

Программа получает на вход числа A₁,B₁,C₁,A₂,B₂,C₂.

**Формат вывода**

Программа должна вывести одну из следующих строчек:

Boxes are equal, если коробки одинаковые,

The first box is smaller than the second one, если первая коробка может быть положена во вторую,

The first box is larger than the second one, если вторая коробка может быть положена в первую,

Boxes are incomparable, во всех остальных случаях.

\*\*

a**,** b**,** c**,** a1**,** b1**,** c1 = int(input())**,** int(input())**,** int(input())**,** \  
 int(input())**,** int(input())**,** int(input())  
if b > c:  
 (b**,** c) = (c**,** b)  
if a > c:  
 (a**,** c) = (c**,** a)  
if a > b:  
 (a**,** b) = (b**,** a)  
if b1 > c1:  
 (b1**,** c1) = (c1**,** b1)  
if a1 > c1:  
 (a1**,** c1) = (c1**,** a1)  
if a1 > b1:  
 (a1**,** b1) = (b1**,** a1)  
if a == a1 and b == b1 and c == c1:  
 print('Boxes are equal')  
elif a >= a1 and b >= b1 and c >= c1:  
 print('The first box is larger than the second one')  
elif a <= a1 and b <= b1 and c <= c1:  
 print('The first box is smaller than the second one')  
else:  
 print('Boxes are incomparable')

19. На склад, который имеет форму прямоугольного параллелепипеда, привезли ноутбуки, упакованные в коробки. Каждая коробка также имеет форму прямоугольного параллелепипеда. По правилам хранения коробки с ноутбуками должны быть размещены на складе с выполнением следующих двух условий:

* Стороны коробок должны быть параллельны сторонам склада.
* Коробку при помещении на склад разрешается расположить где угодно (с выполнением предыдущего условия), в том числе на другой коробке, но все коробки должны быть ориентированы одинаково (т.е. нельзя одну коробку расположить “стоя”, а другую —“лежа”)

Напишите программу, которая по размерам склада и размерам коробки с ноутбуком определит максимальное количество ноутбуков, которое может быть размещено на складе.

**Формат ввода**

Программа получает на вход шесть натуральных чисел. Первые три задают длину, высоту и ширину склада. Следующие три задают соответственно длину, высоту и ширину коробки с ноутбуком.

**Формат вывода**

Программа должна вывести одно число — максимальное количество ноутбуков, которое может быть размещено на складе.

\*\*

a**,** b**,** c**,** a1**,** b1**,** c1 = int(input())**,** int(input())**,** int(input())**,** \  
 int(input())**,** int(input())**,** int(input())  
d = (a // a1) \* (b // b1) \* (c // c1)  
e = (a // a1) \* (b // c1) \* (c // b1)  
f = (a // b1) \* (b // c1) \* (c // a1)  
g = (a // b1) \* (b // a1) \* (c // c1)  
h = (a // c1) \* (b // b1) \* (c // a1)  
i = (a // c1) \* (b // a1) \* (c // b1)  
if d >= e:  
 e = d  
if f >= e:  
 e = f  
if g >= e:  
 e = g  
if h >= e:  
 e = h  
if i >= e:  
 print(i)  
else:  
 print(e)

20. В кафе мороженое продают по три шарика и по пять шариков. Можно ли купить ровно k шариков мороженого?

**Формат ввода**

Вводится число k (целое,положительное)

**Формат вывода**

Программа должна вывести слово YES, если при таких условиях можно набрать ровно k шариков (не больше и не меньше), в противном случае - вывести NO.

\*\* матиндукция

k = int(input())  
if k < **3** or k == **4** or k == **7**:  
 print('NO')  
else:  
 print('YES')

21. Решить в целых числах уравнение: (ax+b) / (cx+d) =0

**Формат ввода**

Вводятся 4 числа: a,b,c,d; c и d не равны нулю одновременно.

**Формат вывода**

Необходимо вывести все решения, если их число конечно, “NO” (без кавычек), если решений нет, и “INF” (без кавычек), если решений бесконечно много.

a**,** b**,** c**,** d**,** = int(input())**,** int(input())**,** int(input())**,** int(input())  
no = 'NO'  
inf = 'INF'  
x = int()  
if c == **0** and d == **0**:  
 print(no)  
elif a == **0** and b == **0**:  
 print(inf)  
elif a != **0**:  
 x = (-b // a)  
 if c \* x + d == **0** or (-b % a) != **0**:  
 print(no)  
 else:  
 print((-b // a))  
else:  
 print(no)

22. На сковородку одновременно можно положить k котлет. Каждую котлету нужно с каждой стороны обжаривать m минут непрерывно. За какое наименьшее время удастся поджарить с обеих сторон n котлет?

**Формат ввода**

Программа получает на вход три числа: k,m,n.

**Формат вывода**

Программа должна вывести одно число: наименьшее количество минут.

k**,** m**,** n = int(input())**,** int(input())**,** int(input())  
if n \* **2** <= k \* **2**:  
 time = m \* **2**elif **2** \* n % k == **0**:  
 time = m \* (**2** \* n // k)  
else:  
 time = m \* ((**2** \* n // k) + **1**)  
print(time)

\*\* жарим не котлеты, а стороны котлет!!!

23. Вдоль прямой выложены три спички. Необходимо переложить одну из них так, чтобы при поджигании любой спички сгорали все три. Для того чтобы огонь переходил с одной спички на другую, необходимо чтобы эти спички соприкасались (хотя бы концами).

Требуется написать программу, определяющую, какую из трех спичек необходимо переместить.

**Формат ввода**

Вводятся шесть целых чисел : l₁, r₁, l₂, r₂, l₃, r₃ – координаты первой, второй и третьей спичек соответственно (0 ≤ lᵢ < rᵢ ≤ 100). Каждая спичка описывается координатами левого и правого концов по горизонтальной оси OX.

**Формат вывода**

Выведите номер искомой спички. Если возможных ответов несколько, то выведите наименьший из них (наименьший по номеру спички). В случае, когда нет необходимости перемещать какую-либо спичку, выведите 0. Если же требуемого результата достигнуть невозможно, то выведите -1.

\*\*

l1**,** r1**,** l2**,** r2**,** l3**,** r3 = int(input())**,** int(input())**,** int(input())**,** \  
 int(input())**,** int(input())**,** int(input())  
n1 = **1**n2 = **2**n3 = **3**if l2 < l1:  
 (l1**,** l2) = (l2**,** l1)  
 (r1**,** r2) = (r2**,** r1)  
 (n1**,** n2) = (n2**,** n1)  
if l3 < l2:  
 (l2**,** l3) = (l3**,** l2)  
 (r2**,** r3) = (r3**,** r2)  
 (n2**,** n3) = (n3**,** n2)  
if l2 < l1:  
 (l1**,** l2) = (l2**,** l1)  
 (r1**,** r2) = (r2**,** r1)  
 (n1**,** n2) = (n2**,** n1)  
long1 = r1 - l1  
long2 = r2 - l2  
long3 = r3 - l3  
c1 = l2 - r1  
c2 = l3 - r2  
if (l2 <= r1 and l3 <= r2) or r1 >= l3:  
 print(**0**)  
elif c1 > long3 and c2 > long1:  
 print(-**1**)  
elif (n3 > n1 and **0** < c2 <= long1) or (n3 > n1 and c2 <= **0** and c1 > **0**):  
 print(n1)  
elif long3 >= c1 > **0**:  
 print(n3)  
elif (n3 < n1 and **0** < c1 <= long3) or (n3 < n1 and c1 <= **0** and c2 > **0**):  
 print(n3)  
elif (long1 >= c2 > **0**) or (c2 <= **0** and long1 >= c1 > **0**):  
 print(n1)

24. В одну транспортную компанию поступил заказ на перевозку двух ящиков из одного города в другой. Для перевозки ящики решено было упаковать в специальный контейнер.

Ящики и контейнер имеют вид прямоугольных параллелепипедов. Длина, ширина и высота первого ящика — l₁,w₁ и h₁, соответствующие размеры второго ящика – l₂,w₂ и h₂. Контейнер имеет длину, ширину и высоту lc,wc и hc.

Поскольку ящики содержат хрупкое оборудование, после упаковки в контейнер каждый из них должен остаться в строго вертикальном положении. Таким образом, ящики можно разместить рядом или один на другом. Для надежного закрепления в контейнере стороны ящиков должны быть параллельны его сторонам. Иначе говоря, если исходно ящики были расположены так, что все их стороны параллельны соответствующим сторонам контейнера, то каждый из них разрешается перемещать и поворачивать относительно вертикальной оси на угол, кратный 90 градусам (относительно горизонтальной оси ни контейнер, ни ящики поворачивать нельзя).

Разумеется, после упаковки оба ящика должны полностью находиться внутри контейнера и не должны пересекаться.

Выясните, можно ли поместить ящики в контейнер с соблюдением указанных условий.

**Формат ввода**

Во входных данных записаны числа l₁, w₁, h₁, l₂,w₂, h₂, lc, wc и hc. Все размеры — целые положительные числа, не превышающие 1000. Числа в строках разделены пробелами.

25. По данному целому числу N распечатайте все квадраты натуральных чисел,не превосходящие N, в порядке возрастания.

**Формат ввода**

Вводится натуральное число.

**Формат вывода**

Выведите ответ на задачу.

k = int(input())  
n = **1**while n \*\* **2** <= k:  
 print(n \*\* **2,** end=' ')  
 n += **1**

26. Дано целое число, не меньшее 2. Выведите его наименьший натуральный делитель, отличный от 1.

**Формат ввода**

Вводится целое положительное число.

**Формат вывода**

Выведите ответ на задачу.

k = int(input())  
n = k + **1**while n > **1**:  
 if k % n == **0**:  
 m = n  
 n -= **1**print(m)

27. По данному числу N распечатайте все целые степени двойки, не превосходящие N, в порядке возрастания.Операцией возведения в степень пользоваться нельзя!

**Формат ввода**

Вводится натуральное число.

**Формат вывода**

Выведите ответ на задачу.

x = int(input())  
m = **1**while m <= x:  
 print(m**,** end=' ')  
 m \*= **2**

28. Дано натуральное число N. Выведите слово YES, если число N является точной степенью двойки, или слово NO в противном случае. Операцией возведения в степень пользоваться нельзя!

**Формат ввода**

Вводится натуральное число.

**Формат вывода**

Выведите ответ на задачу.

x = int(input())  
m = **1**while m <= x:  
 if m == x:  
 print('YES')  
 break  
 m \*= **2** if m > x:  
 print('NO')  
 break

или

N = int(input())  
i = **1**while i < N:  
 i \*= **2**if i == N:  
 print("YES")  
else:  
 print("NO")

29. По данному натуральному числу N выведите такое наименьшее целое число k, что 2ᵏ≥N.

Операцией возведения в степень пользоваться нельзя!

**Формат ввода**

Вводится натуральное число.

**Формат вывода**

Выведите ответ на задачу.

n = int(input())  
k = **1**m = **0**while k < n:  
 k \*= **2** m += **1**print(m)

30. В первый день спортсмен пробежал X километров, а затем он каждый день увеличивал пробег на 10% от предыдущего значения (для решения задачи разрешается использовать числа с запятой, которые в Питоне пишутся через точку).

По данному числу X определите номер дня, на который пробег спортсмена составит не менее Y километров.

**Формат ввода**

Программа получает на вход числа X и Y.

**Формат вывода**

Программа должна вывести одно натуральное число.

x**,** y = int(input())**,** int(input())  
n = **1**while x < y:  
 x = x \* **1.1** n += **1**print(n)

31. Последовательность состоит из целых чисел и завершается числом 0. Определите значение наибольшего элемента последовательности.

**Формат ввода**

Вводится последовательность целых чисел, оканчивающаяся числом 0 (само число 0 в последовательность не входит, а служит как признак ее окончания).

**Формат вывода**

Выведите ответ на задачу.

x = int(input())  
maxx = **0**while x != **0**:  
 if x >= maxx:  
 maxx = x  
 x = int(input())  
print(maxx)

32. По данному натуральному n вычислите сумму 1²+2²+3²+...+n².

**Формат ввода**

Вводится натуральное число.

**Формат вывода**

Выведите ответ на задачу.

x = int(input())  
n = **1**ssq = **0**while n <= x:  
 ssq += n \*\* **2** n += **1**print(ssq)

33. Программа получает на вход последовательность целых неотрицательных чисел, каждое число записано в отдельной строке. Последовательность завершается числом 0, при считывании которого программа должна закончить свою работу и вывести количество членов последовательности (не считая завершающего числа 0).

Числа, следующие за числом 0, считывать не нужно.

**Формат ввода**

Вводится последовательность целых чисел, заканчивающаяся числом 0.

**Формат вывода**

Выведите ответ на задачу.

**Примечание**

По этой задаче возникает непонимание. Например, как ввести первый тест, если после ввода нуля программа завершает работу и не дает ввести оставшееся число 5?

Можно ввести несколько строчек за раз (в том числе и полный тест), скопировав и вставив их.

a = int(input())  
n = **0**while a != **0**:  
 a = int(input())  
 n += **1**print(n)

34. Определите сумму всех элементов последовательности, завершающейся числом 0.

**Формат ввода**

Вводится последовательность целых чисел, оканчивающаяся числом 0 (само число 0 в последовательность не входит, а служит как признак ее окончания).

**Формат вывода**

Выведите ответ на задачу.

n = int(input())  
seqSum = **0**while n != **0**:  
 seqSum += n  
 n = int(input())  
print(seqSum)

35. Определите среднее значение всех элементов последовательности, завершающейся числом 0. Использовать массивы в данной задаче нельзя.

**Формат ввода**

Вводится последовательность целых чисел, оканчивающаяся числом 0 (само число 0 в последовательность не входит, а служит как признак ее окончания).

**Формат вывода**

Выведите ответ на задачу.

a = s = int(input())  
n = **0**while a != **0**:  
 a = int(input())  
 s += a  
 n += **1**print(s / n)

36. Определите количество четных элементов в последовательности, завершающейся числом 0.

**Формат ввода**

Вводится последовательность целых чисел, оканчивающаяся числом 0 (само число 0 в последовательность не входит, а служит как признак ее окончания).

**Формат вывода**

Выведите ответ на задачу.

a = int(input())  
n = **0**while a != **0**:  
 if a % **2** == **0**:  
 n += **1** a = int(input())  
print(n)

37. Последовательность состоит из натуральных чисел и завершается числом 0. Определите, сколько элементов этой последовательности больше предыдущего элемента.

**Формат ввода**

Вводится последовательность целых чисел, оканчивающаяся числом 0 (само число 0 в последовательность не входит, а служит как признак ее окончания).

**Формат вывода**

Выведите ответ на задачу.

a = s = int(input())  
n = **0**while a != **0**:  
 a = int(input())  
 if a > s:  
 n += **1** s = a  
print(n)

38. Последовательность состоит из натуральных чисел и завершается числом 0. Определите значение второго по величине элемента в этой последовательности, то есть элемента, который будет наибольшим, если из последовательности удалить одно вхождение наибольшего элемента.

**Формат ввода**

Вводится последовательность натуральных чисел, оканчивающаяся числом 0 (само число 0 в последовательность не входит, а служит как признак ее окончания).

x = int(input())  
maxx = x  
secmaxx = **0**while x != **0**:  
 x = int(input())  
 if x >= maxx:  
 secmaxx = maxx  
 maxx = x  
 if maxx > x >= secmaxx:  
 secmaxx = x  
print(secmaxx)

\*\* не работает на отрицательных!!!

39. Последовательность состоит из натуральных чисел и завершается числом 0. Определите количество элементов этой последовательности, которые равны ее наибольшему элементу.

**Формат ввода**

Вводится последовательность целых чисел, оканчивающаяся числом 0 (само число 0 в последовательность не входит, а служит как признак ее окончания).

**Формат вывода**

Выведите ответ на задачу.

x = int(input())  
maxx = x  
n = **1**while x != **0**:  
 x = int(input())  
 if x > maxx:  
 maxx = x  
 n = **1** elif x == maxx:  
 maxx = x  
 n += **1**print(n)

40. Последовательность Фибоначчи определяется так:

F[0] = 0, F[1] = 1, ..., F[n] = F[n-1] + F[n-2].

По данному числу n определите n-е число Фибоначчи F[n].

**Формат ввода**

Вводится натуральное число n.

**Формат вывода**

Выведите ответ на задачу.

a = int(input())  
n = f = s = **1**t = **0**while n < a:  
 f = s + t  
 (t**,** s) = (s**,** f)  
 n += **1**print(f)

41. Последовательность Фибоначчи определяется так:

F[0]=0, F[1]=1, ..., F[n]=F[n-1]+F[n-2].

Дано натуральное число A. Определите, каким по счету числом Фибоначчи оно является, то есть выведите такое число n, что F[n]=A.

Если А не является числом Фибоначчи, выведите число -1.

**Формат ввода**

Вводится натуральное число A.

**Формат вывода**

Выведите ответ на задачу.

a = int(input())  
n = f = s = **1**t = **0**while a > f:  
 f = s + t  
 (t**,** s) = (s**,** f)  
 n += **1**if a == f:  
 print(n)  
else:  
 print(-**1**)

42. Исполнитель “Раздвоитель” преобразует натуральные числа. У него есть две команды: “Вычесть 1” и “Разделить на 2”, первая команда уменьшает число на 1, вторая команда уменьшает число в два раза, если оно чётное,иначе происходит ошибка. Дано два натуральных числа A и B (A>B). Напишите алгоритм для Раздвоителя, который преобразует число A в число B и при этом содержит минимальное число команд. Команды алгоритма нужно выводить по одной в строке, первая команда обозначается, как -1, вторая команда как :2.

**Примечание**

В этой задаче следует переходить из 2 в 1 командой :2. Если выполнять переход командой -1, то тестирующая система выдаст вердикт WA.

**Формат ввода**

Вводятся два натуральных числа A и B.

**Формат вывода**

Выведите ответ на задачу.

a**,** b = int(input())**,** int(input())  
while a > b:  
 if a % **2** == **1**:  
 a -= **1** print('-1')  
 else:  
 a = a // **2** if a >= b:  
 print(':2')  
 else:  
 a = a \* **2** - **1** print('-1')

better

a**,** b = int(input())**,** int(input())  
while a != b:

if a % **2** == **1** or a // **2** < b:  
 a -= **1** print('-1')  
 else:  
 a = a // **2** print(':2')

43.Переставьте цифры числа в обратном порядке .

**Формат ввода**

Задано единственное число N

**Формат вывода**

Необходимо вывести цифры данного числа в обратном порядке.

Первое решение выводит числовое значение с обратным порядком цифр.

a = int(input())  
b = str(a % **10**)  
a = a // **10**while a != **0**:  
 d = str(a % **10**)  
 d = b + d  
 b = d  
 a = (a - a % **10**) // **10**print(int(b))

это просто печатает цифры в обратном порядке

n = int(input())  
m = **1**while m <= n:  
 print((n % (m \* **10**)) // m**,** sep=''**,** end='')  
 m = m \* **10**

44. Назовем число палиндромом, если оно не меняется при перестановке его цифр в обратном порядке. Напишите программу, которая по заданному числу K выводит количество натуральных палиндромов, не превосходящих K.

**Формат ввода**

Задано единственное число K (1≤K≤100000).

**Формат вывода**

Необходимо вывести количество натуральных палиндромов, не превосходящих K.

s = int(input())  
p = **1**count = n = **0**while p <= s:  
 result = str()  
 m = **1** n += **1** while m <= n:  
 a = (n % (m \* **10**)) // m  
 result += str(a)  
 m \*= **10** if int(result) == n:  
 count += **1** p += **1**print(count)

45.Дана последовательность натуральных чисел, завершающаяся числом 0. Определите, какое наибольшее число подряд идущих элементов этой последовательности равны друг другу.

**Формат ввода**

Вводится последовательность натуральных чисел, оканчивающаяся числом 0 (само число 0 в последовательность не входит, а служит как признак ее окончания).

**Формат вывода**

Выведите ответ на задачу.

x = int(input())  
x1 = x  
m = **1**n = **1**while x != **0**:  
 x = int(input())  
 if x == x1:  
 m += **1** if m >= n:  
 n = m  
 else:  
 x1 = x  
 m = **1**print(n)

46. Дана последовательность натуральных чисел, завершающаяся числом 0. Определите наибольшую длину монотонного фрагмента последовательности (то есть такого фрагмента, где все элементы либо больше предыдущего, либо меньше).

**Формат ввода**

Вводится последовательность целых чисел, оканчивающаяся числом 0 (само число 0 в последовательность не входит, а служит как признак ее окончания).

**Формат вывода**

Выведите ответ на задачу.

a = a1 = int(input())  
count\_up = count\_up1 = count\_down = count\_down1 = **1**while a != **0**:  
 if a == a1:  
 (count\_up**,** count\_down**,** a1) = (**1, 1,** a)  
 elif a < a1:  
 (count\_up**,** a1) = (**1,** a)  
 count\_down += **1** if count\_down > count\_down1:  
 count\_down1 = count\_down  
 elif a > a1:  
 (count\_down**,** a1) = (**1,** a)  
 count\_up += **1** if count\_up > count\_up1:  
 count\_up1 = count\_up  
 a = int(input())  
if count\_up1 >= count\_down1:  
 print(count\_up1)  
else:  
 print(count\_down1)

47. Определите наименьшее расстояние между двумя локальными максимумами последовательности натуральных чисел, завершающейся числом 0. Локальным максимумом называется такое число в последовательности, которое больше своих соседей. Если в последовательности нет двух локальных максимумов, выведите число 0. Начальное и конечное значение при этом локальными максимумами не считаются.

**Формат ввода**

Вводится последовательность целых чисел, оканчивающаяся числом 0 (само число 0 в последовательность не входит, а служит как признак ее окончания).

**Формат вывода**

Выведите ответ на задачу.

**Примечания**

Пояснение к тестам:

В первом тесте локальными максимумами являются все двойки (они больше соседей). Между последними - расстояние наименьшее.

Во втором тесте нет локального максимума.

**Примеры**

Тест 1

Входные данные:

1

2

1

1

2

1

2

1

0

Вывод программы:

2

Тест 2

Входные данные:

1

2

3

0

Вывод программы:

0

### Вещественные числа

1. Даны длины сторон треугольника. Вычислите площадь треугольника.

**Формат ввода**

Вводятся три положительных **действительных**числа.

**Формат вывода**

Выведите ответ на задачу.

a**,** b**,** c = float(input())**,** float(input())**,** float(input())  
p = (a + b + c) / **2** # площадь треугольника  
s = (p \* (p - a) \* (p - b) \* (p - c)) \*\* **0.5** # по формуле Герона  
print(s)

2. По данному числу n вычислите сумму (1 / 1²)+(1 / 2²)+(1 / 3²)+...+(1 / n²).

**Формат ввода**

Вводится целое положительное число.

**Формат вывода**

Выведите ответ на задачу.

n = int(input())  
m = **0**s = float()  
while m != n:  
 m += **1** s += **1** / m \*\* **2**print(s)

3. Дано положительное действительное число X. Выведите его дробную часть.

**Формат ввода**

Вводится положительное действительное число.

**Формат вывода**

Выведите ответ на задачу.

n = float(input())  
print(n % **1**)

4. Цена товара обозначена в рублях с точностью до копеек, то есть действительным числом с двумя цифрами после десятичной точки. Запишите в две целочисленные переменные стоимость товара в виде целого числа рублей и целого числа копеек и выведите их на экран. При решении этой задачи нельзя пользоваться условными инструкциями и циклами.

**Формат ввода**

Вводится неотрицательное действительное число.

**Формат вывода**

Выведите ответ на задачу.

n = float(input())  
print(round(n // **1**)**,** round((n % **1**) \* **100**))

5. По российский правилам числа округляются до ближайшего целого числа,а если дробная часть числа равна 0.5, то число округляется вверх. Дано неотрицательное число x, округлите его по этим правилам. Обратите внимание, что функция round не годится для этой задачи!

**Формат ввода**

Вводится неотрицательное число.

**Формат вывода**

Выведите ответ на задачу.

n = float(input())  
if n % **1** < **0.5**:  
 print(int(n // **1**))  
else:  
 print(int(n // **1**) + **1**)

6. Процентная ставка по вкладу составляет P процентов годовых, которые прибавляются к сумме вклада. Вклад составляет X рублей Y копеек. Определите размер вклада через год. При решении этой задачи нельзя пользоваться условными инструкциями и циклами.

**Формат ввода**

Программа получает на вход целые числа P, X, Y.

**Формат вывода**

Программа должна вывести два числа: величину вклада через год в рублях и копейках. Дробная часть копеек отбрасывается.

p**,** x**,** y = int(input())**,** int(input())**,** int(input())  
s = (**100** \* x + y)  
ns = s + s \* p / **100**print(int(ns) // **100,** int(ns) % **100**)

7. Процентная ставка по вкладу составляет P процентов годовых, которые прибавляются к сумме вклада через год. Вклад составляет X рублей Y копеек. Определите размер вклада через K лет.

**Формат ввода**

Программа получает на вход целые числа P, X, Y, K.

**Формат вывода**

Программа должна вывести два числа: величину вклада через K лет в рублях и копейках. Дробное число копеек по истечение года отбрасывается. Перерасчет суммы вклада (с отбрасыванием дробных частей копеек) происходит ежегодно.

p**,** x**,** y**,** k = int(input())**,** int(input())**,** int(input())**,** int(input())  
s = (**100** \* x + y)  
n = **0**while n < k:  
 n += **1** ns = int(s + s \* p / **100**)  
 s = ns  
print(int(ns / **100**)**,** int(ns % **100**))

8. Дан многочлен P(x) = a[n] xⁿ+a[n-1] xⁿ⁻¹+...+a[1] x+a[0] и число x. Вычислите значение этого многочлена, воспользовавшись схемой Горнера:

P(x) = ( ... ( ( ( a[n] x + a[n-1] ) x + a[n-2] ) x + a[n-3] ) ... ) x + a[0]

**Формат ввода**

Сначала программе подается на вход целое неотрицательное число n ≤ 20, затем действительное число x, затем следует n+1 вещественных чисел — коэффициенты многочлена от старшего к младшему.

**Формат вывода**

Программа должна вывести значение многочлена.

**Примечания**

При решении этой задачи нельзя использовать массивы и операцию возведения в степень. Программа должна иметь сложность O(n), то есть при увеличении количества входных данных в k раз время выполнения программы должно вырастать примерно в k раз.

n**,** x**,** = int(input())**,** float(input())  
count = **0**p = pit = **0**while count < n + **1**:  
 count += **1** p = pit  
 a = float(input())  
 p += a  
 pit = p \* x  
print(p)

or

n**,** x**,** = int(input())**,** float(input())  
count = **0**p = **0**while count < n + **1**:  
 count += **1** a = float(input())  
 p \*= x  
 p += a  
print(p)

9. Дана последовательность натуральных чисел x₁, x₂ ..., xn. Стандартным отклонением называется величина

σ=sqrt(((x₁-s)²+(x₂-s)²+…+(xn-s)²) / (n-1))

где s = ((x₁+x₂+…+xn) / n) — среднее арифметическое последовательности, а sqrt - квадратный корень. Определите стандартное отклонение для данной последовательности натуральных чисел, завершающейся числом 0.

**Формат ввода**

Вводится последовательность целых чисел, оканчивающаяся числом 0 (само число 0 в последовательность не входит, а служит как признак ее окончания).

**Формат вывода**

Выведите ответ на задачу.

from math import sqrt  
  
a = float(input())  
x = a  
sumsqr = a \*\* **2**n = **1**while a != **0**:  
 a = float(input())  
 if a != **0**:  
 sumsqr += a \*\* **2** x += a  
 n += **1**print(sqrt((sumsqr - x \*\* **2** / n) / (n - **1**)))

10. Даны действительные коэффициенты a, b, c, при этом a != 0. Решите квадратное уравнение ax²+bx+c=0 и выведите все его корни.

**Формат ввода**

Вводятся три действительных числа.

**Формат вывода**

Если уравнение имеет два корня, выведите два корня в порядке возрастания, если один корень — выведите одно число, если нет корней — не выводите ничего.

from math import sqrt  
  
a**,** b**,** c = float(input())**,** float(input())**,** float(input())  
d = b \*\* **2** - **4** \* a \* c  
if d < **0**:  
 print(' ')  
elif d == **0**:  
 print(-b / (**2** \* a))  
else:  
 x1 = (- b + sqrt(d)) / (**2** \* a)  
 x2 = (-b - sqrt(d)) / (**2** \* a)  
 if x1 > x2:  
 (x1**,** x2) = (x2**,** x1)  
 print(x1**,** x2)

11. Даны произвольные действительные коэффициенты a, b, c. Решите уравнение ax²+bx+c=0.

**Формат ввода**

Вводятся три действительных числа.

**Формат вывода**

Если данное уравнение не имеет корней, выведите число 0. Если уравнение имеет один корень, выведите число 1, а затем этот корень. Если уравнение имеет два корня, выведите число 2, а затем два корня в порядке возрастания. Если уравнение имеет бесконечно много корней, выведите число 3.

from math import sqrt  
  
a**,** b**,** c = float(input())**,** float(input())**,** float(input())  
d = b \*\* **2** - **4** \* a \* c  
if a == b == c == **0**:  
 print(**3**)  
if a == **0**:  
 if b == **0** and c != **0**:  
 print(**0**)  
 elif b != **0** and c == **0**:  
 print(**1, 0**)  
 elif b != **0** and c != **0**:  
 print(**1,** -c / b)  
if a != **0**:  
 if b != **0** and c == **0**:  
 x1 = **0** x2 = (-b / a)  
 if x1 > x2:  
 (x1**,** x2) = (x2**,** x1)  
 print(**2,** x1**,** x2)  
 elif b == **0** and c != **0**:  
 if -c / a < **0**:  
 print(**0**)  
 else:  
 x1 = - sqrt(-c / a)  
 x2 = sqrt(-c / a)  
 if x1 > x2:  
 (x1**,** x2) = (x2**,** x1)  
 print(**2,** x1**,** x2)  
 if b == **0** and c == **0**:  
 print(**1, 0**)  
if a != **0** and b != **0** and c != **0**:  
 if d < **0**:  
 print(**0**)  
 elif d == **0**:  
 print(**1,** -b / (**2** \* a))  
 else:  
 x1 = (- b + sqrt(d)) / (**2** \* a)  
 x2 = (-b - sqrt(d)) / (**2** \* a)  
 if x1 > x2:  
 (x1**,** x2) = (x2**,** x1)  
 print(**2,** x1**,** x2)

12. Даны вещественные числа a, b, c, d, e, f. Известно, что система линейных уравнений:

ax + by = e

cx + dy = f

имеет ровно одно решение. Выведите два числа x и y, являющиеся решением этой системы.

**Формат ввода**

Вводятся шесть чисел a, b, c, d, e, f

- коэффициенты уравнений системы.

**Формат вывода**

Выведите ответ на задачу.

a**,** b**,** c = float(input())**,** float(input())**,** float(input())  
d**,** e**,** f = float(input())**,** float(input())**,** float(input())  
x = (d \* e - b \* f) / (a \* d - b \* c)  
y = (a \* f - c \* e) / (a \* d - b \* c)  
print(x**,** y)

13. Даны числа a, b, c, d, e, f. Решите систему линейных уравнений

ax + by = e

cx + dy = f

**Формат ввода**

Вводятся 6 чисел a, b, c, d, e, f — коэффициенты уравнений.

**Формат вывода**

Вывод программы зависит от вида решения этой системы.

Если система не имеет решений, то программа должна вывести единственное число 0.

Если система имеет бесконечно много решений, каждое из которых имеет вид y=px+q, то программа должна вывести число 1, а затем значения p и q.

Если система имеет единственное решение (x₀,y₀), то программа должна вывести число 2, а затем значения x₀ и y₀.

Если система имеет бесконечно много решений вида x=x₀, y — любое, то программа должна вывести число 3, а затем значение x₀.

Если система имеет бесконечно много решений вида y=y₀, x — любое, то программа должна вывести число 4, а затем значение y₀.

Если любая пара чисел (x,y) является решением, то программа должна вывести число 5.

14. **Формат ввода**

Дана строка.

**Формат вывода**

Сначала выведите третий символ этой строки.

Во второй строке выведите предпоследний символ этой строки.

В третьей строке выведите первые пять символов этой строки.

В четвертой строке выведите всю строку, кроме последних двух символов.

В пятой строке выведите все символы с четными индексами (считая, что индексация начинается с 0, поэтому символы выводятся начиная с первого).

В шестой строке выведите все символы с нечетными индексами, то есть начиная со второго символа строки.

В седьмой строке выведите все символы в обратном порядке.

В восьмой строке выведите все символы строки через один в обратном порядке, начиная с последнего.

В девятой строке выведите длину данной строки.

**Примечание**

Данная строка имеет достаточную длину для выполнения всех предложенных срезов.

s = input()  
print(s[**2**]**,** s[-**2**]**,** s[:**5**]**,** s[:-**2**]**,** s[::**2**]**,** s[**1**::**2**]**,** sep=('\n'))  
print(s[-**1**::-**1**]**,** s[-**1**::-**2**]**,** len(s)**,** sep=('\n'))

15. Дана строка. Если в этой строке буква f встречается только один раз, выведите её индекс. Если она встречается два и более раз, выведите индекс её первого и последнего появления. Если буква f в данной строке не встречается, ничего не выводите. При решении этой задачи нельзя использовать метод count и циклы.

**Формат ввода**

Вводится строка.

**Формат вывода**

Выведите ответ на задачу.

s = input()  
pos = s.find('f')  
if pos == -**1**:  
 print(' ')  
else:  
 sub = s[-**1**:pos:-**1**]  
 if sub.find('f') != -**1**:  
 pos1 = len(s) - **1** - sub.find('f')  
 print(pos**,** pos1)  
 else:  
 print(pos)

16. Дана строка, в которой буква h встречается минимум два раза.Удалите из этой строки первое и последнее вхождение буквы h,а также все символы, находящиеся между ними.

**Формат ввода**

Вводится строка.

**Формат вывода**

Выведите ответ на задачу.

s = input()  
pos = s.find('h')  
if pos == -**1**:  
 print(' ')  
else:  
 sub = s[-**1**:pos:-**1**]  
 if sub.find('h') != -**1**:  
 pos1 = len(s) - **1** - sub.find('h')  
 print(s[:pos] + s[pos1 + **1**:])

17. Дана строка, в которой буква h встречается как минимум два раза. Выведите измененную строку: повторите последовательность символов, заключенную между первым и последним появлением буквы h два раза (сами буквы h не входят в повторяемый фрагмент, т. е. их повторять не надо).

**Формат ввода**

Вводится строка.

**Формат вывода**

Выведите ответ на задачу.

s = input()  
pos = s.find('h')  
if pos == -**1**:  
 print(' ')  
else:  
 sub = s[-**1**:pos:-**1**]  
 if sub.find('h') != -**1**:  
 pos1 = len(s) - **1** - sub.find('h')  
 print(s[:pos1] + s[pos + **1**:pos1] + s[pos1:])

18. Дана строка. Найдите в этой строке второе вхождение буквы f и выведите индекс этого вхождения. Если буква f в данной строке встречается только один раз, выведите число -1, а если не встречается ни разу, выведите число -2. При решении этой задачи нельзя использовать метод count.

**Формат ввода**

Вводится строка.

**Формат вывода**

Выведите ответ на задачу.

s = input()  
pos = s.find('f')  
if pos != -**1**:  
 sub = s[pos + **1**:]  
 if sub.find('f') == -**1**:  
 print(-**1**)  
 else:  
 pos1 = sub.find('f') + **1** + pos  
 print(pos1)  
else:  
 print(-**2**)

19. Дана строка, состоящая ровно из двух слов, разделенных пробелом. Переставьте эти слова местами. Результат запишите в строку и выведите получившуюся строку. При решении этой задачи нельзя пользоваться циклами и инструкцией if.

**Формат ввода**

Вводится строка.

**Формат вывода**

Выведите ответ на задачу.

s = input()  
print(s[s.find(' ') + **1**:]**,** s[:s.find(' ')])

20. Дана строка, состоящая из слов, разделенных пробелами. Определите, сколько в ней слов.

**Формат ввода**

Вводится строка.

**Формат вывода**

Выведите ответ на задачу.

s = input()  
print(s.count(' ') + **1**)

21. Дана строка. Замените в этой строке все цифры 1 на слово one.

**Формат ввода**

Вводится строка.

**Формат вывода**

Выведите ответ на задачу.

s = input()  
print(s.replace('1'**,** 'one'))

22. Дана строка. Удалите из этой строки все символы @.

**Формат ввода**

Вводится строка.

**Формат вывода**

Выведите ответ на задачу.

s = input()  
n = **0**m = s.count('@')  
while n != m:  
 news = s[:s.find('@')] + s[s.find('@') + **1**:]  
 s = news  
 n += **1**print(s)

21. Дана строка. Замените в этой строке все появления буквы h на букву H, кроме первого и последнего вхождения.

**Формат ввода**

Вводится строка.

**Формат вывода**

Выведите ответ на задачу.

s = input()  
print(s[:s.find('h') + **1**] + s[s.find('h') +\  
**1**:s.rfind('h')].replace('h'**,** 'H') + s[s.rfind('h'):])

что равнозначно

s = input()  
l = s.find('h')  
r = s.rfind('h')  
s\_part = s[l + **1**:r].replace('h'**,** 'H')  
print(s[:l + **1**] + s[l + **1**:r].replace('h'**,** 'H') + s[r:])

22. Дана строка. Получите новую строку, вставив между каждыми двумя символами исходной строки символ \*. Выведите полученную строку.

**Формат ввода**

Вводится строка.

**Формат вывода**

Выведите ответ на задачу.

s = input()  
n = **1**while n < len(s):  
 s\_new = s[:n] + '\*' + s[n:]  
 s = s\_new  
 n += **2**print(s)

23. Дана строка. Удалите из нее все символы, чьи индексы делятся на 3.Символы строки нумеруются, начиная с нуля.

**Формат ввода**

Вводится строка.

**Формат вывода**

Выведите ответ на задачу.

**Примечания**

Ввод и вывод осуществлять с помощью файлов

s = input()  
n = **2**s = s[**1**:]  
while n < len(s):  
 s\_new = s[:n] + s[n + **1**:]  
 s = s\_new  
 n += **2**print(s)

### Функции и рекурсия

1. Напишите функцию min4(a, b, c, d), вычисляющую минимум четырех чисел, которая не содержит инструкции if, а использует стандартную функцию min от двух чисел. Считайте четыре целых числа и выведите их минимум.

**Формат ввода**

Вводятся четыре целых числа.

**Формат вывода**

Выведите ответ на задачу.

def min4(a**,** b**,** c**,** d):  
 e = min(a**,** b)  
 d = min(c**,** d)  
 f = min(e**,** d)  
 return f  
  
  
a**,** b**,** c**,** d = int(input())**,** int(input())**,** int(input())**,** int(input())  
print(min4(a**,** b**,** c**,** d))

2. Даны четыре действительных числа: x₁, y₁, x₂, y₂. Напишите функцию distance(x1, y1, x2, y2), вычисляющую расстояние между точками (x₁,y₁) и (x₂,y₂). Считайте четыре действительных числа и выведите результат работы этой функции.

**Формат ввода**

Вводятся четыре действительных числа.

**Формат вывода**

Выведите ответ на задачу.

def dist(x1**,** y1**,** x2**,** y2):  
 dist = ((x1 - x2) \*\* **2** + (y1 - y2) \*\* **2**)\*\***0.5** return dist  
  
  
x1**,** y1**,** x2**,** y2 = float(input())**,** float(input())**,** float(input())**,** float(input())  
print(dist(x1**,** y1**,** x2**,** y2))

3. Напишите функцию, вычисляющую длину отрезка по координатам его концов. С помощью этой функции напишите программу, вычисляющую периметр треугольника по координатам трех его вершин.

**Формат ввода**

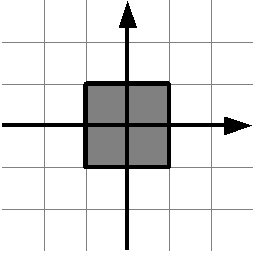
На вход программе подается 6 целых чисел — координат x₁, y₁, x₂, y₂, x₃, y₃ вершин треугольника. Все числа по модулю не превосходят 30 000.

**Формат вывода**

Выведите значение периметра этого треугольника с точностью до 6 знаков после десятичной точки.

def dist(x1**,** y1**,** x2**,** y2):  
 dist = ((x1 - x2) \*\* **2** + (y1 - y2) \*\* **2**) \*\* **0.5** return dist  
  
  
x1**,** y1 = float(input())**,** float(input())  
x2**,** y2 = float(input())**,** float(input())  
x3**,** y3 = float(input())**,** float(input())  
a = dist(x1**,** y1**,** x2**,** y2)  
b = dist(x1**,** y1**,** x3**,** y3)  
c = dist(x2**,** y2**,** x3**,** y3)  
print('{0:.12}'.format(a + b + c))

4. Даны два действительных числа x и y. Проверьте, принадлежит ли точка с координатами (x,y) заштрихованному квадрату (включая его границу). Если точка принадлежит квадрату, выведите слово YES, иначе выведите слово NO. На рисунке сетка проведена с шагом 1.



Решение должно содержать функцию IsPointInSquare(x, y), возвращающую True, если точка принадлежит квадрату и False, если не принадлежит. Основная программа должна считать координаты точки, вызвать функцию IsPointInSquare и в зависимости от возвращенного значения вывести на экран необходимое сообщение. Функция IsPointInSquare не должна содержать инструкцию if.

**Формат ввода**

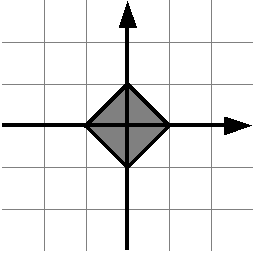
Вводятся два действительных числа.

**Формат вывода**

Выведите ответ на задачу.

def isPointInSquare(x**,** y):  
 return -**1** <= x <= **1** and -**1** <= y <= **1**x**,** y = float(input())**,** float(input())  
if isPointInSquare(x**,** y):  
 print('YES')  
else:  
 print('NO')

5. Даны два действительных числа x и y. Проверьте, принадлежит ли точка с координатами(x,y) заштрихованному квадрату (включая его границу). Если точка принадлежит квадрату, выведите слово YES,иначе выведите слово NO. На рисунке сетка проведена с шагом 1.



Решение должно содержать функцию IsPointInSquare(x, y), возвращающую True, если точка принадлежит квадрату и False, если не принадлежит. Основная программа должна считать координаты точки, вызвать функцию IsPointInSquareи в зависимости от возвращенного значения вывести на экран необходимое сообщение.

Функция IsPointInSquare не должна содержать инструкцию if.

**Формат ввода**

Вводятся два действительных числа.

**Формат вывода**

Выведите ответ на задачу.

def isPointInSquare1(x**,** y):  
 return -**1** <= x + y <= **1** and -**1** <= x - y <= **1**x**,** y = float(input())**,** float(input())  
if isPointInSquare1(x**,** y):  
 print('YES')  
else:  
 print('NO')

6. Даны пять действительных чисел: x, y, xc, yc, r.

Проверьте, принадлежит ли точка (x,y) кругу с центром (xc, yc) и радиусом r.

Если точка принадлежит кругу, выведите слово YES, иначе выведите слово NO.

Решение должно содержать функцию IsPointInCircle(x, y, xc, yc, r), возвращающую True, если точка принадлежит кругу и False, если не принадлежит.

Основная программа должна считать координаты точки, вызвать функцию IsPointInCircle и в зависимости от возвращенного значения вывести на экран необходимое сообщение. Функция IsPointInCircle не должна содержать инструкцию if.

**Формат ввода**

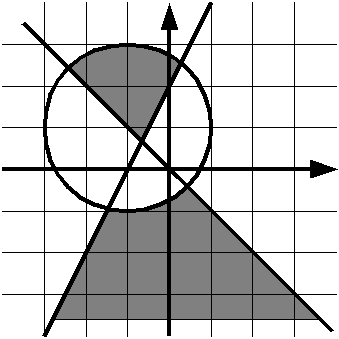
Вводится пять действительных чисел.

**Формат вывода**

Выведите ответ на задачу.

def dist(x1**,** y1**,** x2**,** y2):  
 dist = ((x1 - x2) \*\* **2** + (y1 - y2) \*\* **2**) \*\* **0.5** return dist  
  
  
def isPointInCircle(x**,** y**,** xc**,** yc**,** r):  
 return dist(x**,** y**,** xc**,** yc) <= r  
  
  
x**,** y = float(input())**,** float(input())  
xc**,** yc**,** r = float(input())**,** float(input())**,** float(input())  
if isPointInCircle(x**,** y**,** xc**,** yc**,** r):  
 print('YES')  
else:  
 print('NO')

7. Проверьте, принадлежит ли точка данной закрашенной области:



Если точка принадлежит области (область включает границы), выведите слово YES, иначе выведите слово NO. Решение должно содержать функцию IsPointInArea(x, y), возвращающую True, если точка принадлежит области и False, если не принадлежит. Основная программа должна считать координаты точки, вызвать функцию IsPointInArea и в зависимости от возвращенного значения вывести на экран необходимое сообщение. Функция IsPointInArea не должна содержать инструкцию if.

**Формат ввода**

Вводится два действительных числа.

**Формат вывода**

Выведите ответ на задачу.

**Замечание**

В задаче подразумевается, что нижняя область продолжается вниз бесконечно (картинка может ввести в заблуждение, как будто область заканчивается на y = -3.5). Т.е. например для ввода

0

-5

ответ должен быть YES.

def dist1(x1**,** y1):  
 dist = ((x1 + **1**) \*\* **2** + (y1 - **1**) \*\* **2**) \*\* **0.5** return dist  
  
  
def isPointInCircle1(x**,** y):  
 return dist1(x**,** y) <= **2**def isPointInArea(x**,** y):  
 return x + y >= **0** and y - **2** \* x >= **2** and isPointInCircle1(x**,** y) or \  
 x + y <= **0** and y - **2** \* x <= **2** and dist1(x**,** y) >= **2**x**,** y = float(input())**,** float(input())  
if isPointInArea(x**,** y):  
 print('YES')  
else:  
 print('NO')

8. Напишите функцию xor(x, y)

реализующую функцию "Исключающее ИЛИ" двух логических переменных x и y.

Функция xor должна возвращать True, если ровно один из ее аргументов x или y, но не оба одновременно равны True.

**Формат ввода**

Вводится 2 числа - x и y (x и y равны 0 или 1, 0 соответствует значению False, 1 соответствует значению True).

**Формат вывода**

Необходимо вывести 0 или 1 - значение функции от x и y.

def xor(x**,** y):  
 if (x == **1** or y == **1**) and x != y:  
 return print(**1**)  
 return print(**0**)  
  
  
x**,** y = float(input())**,** float(input())  
xor(x**,** y)

или

def xor(x**,** y):  
 if x + y == **1**:  
 return (**1**)  
 return (**0**)  
  
  
x**,** y = float(input())**,** float(input())  
print(xor(x**,** y))

9. Дано натуральное число n>1. Выведите его наименьший делитель, отличный от 1. Решение оформите в виде функции MinDivisor(n). Алгоритм должен иметь сложность порядка корня квадратного из n.

**Указание.** Если у числа n нет делителя не превосходящего корня из n, то число n — простое и ответом будет само число n. А у всех составных чисел обязательно есть делители, отличные от единицы и не превосходящие корня из n.

**Формат ввода**

Вводится натуральное число.

**Формат вывода**

Выведите ответ на задачу.

def minDivisor(num):  
 n = **2** while num % n != **0**:  
 n += **1** if n > num \*\* **0.5**:  
 return num  
 return n  
  
  
k = int(input())

print(minDivisor(k))

10.Дано натуральное число n>1. Проверьте, является ли оно простым. Программа должна вывести слово YES, если число простое и NO, если число составное. Решение оформите в виде функции IsPrime(n), которая возвращает True для простых чисел и False для составных чисел. Программа должна иметь сложность O(корень из n): количество действий в программе должно быть пропорционально квадратному корню из n (иначе говоря, при увеличении входного числа в k раз, время выполнения программы должно увеличиваться примерно в корень из k раз).

**Формат ввода**

Вводится натуральное число.

**Формат вывода**

Выведите ответ на задачу.

\*\*

def isPrime(num):  
 a = **2** while num \*\* **0.5** >= a and num % a != **0**:  
 a += **1** return num \*\* **0.5** < a  
  
  
k = int(input())  
if isPrime(k):  
 print('YES')  
else:  
 print('NO')

11. Дано действительное положительное число a и целое неотрицательное число n. Вычислите aⁿ, не используя циклы и стандартную функцию pow, но используя рекуррентное соотношение aⁿ=a⋅aⁿ⁻¹.

Решение оформите в виде функции power(a, n) (которая возвращает aⁿ).

**Формат ввода**

Вводятся действительное положительное число a и целое неотрицательное число n.

**Формат вывода**

Выведите ответ на задачу: print(power(a, n)).

def power(a**,** n):  
 if n == **0**:  
 return **1** if n == **1**:  
 return a  
 return a \* power(a**,** n - **1**)  
  
  
a = float(input())  
n = float(input())  
print(power(a**,** n))

\*\*ппц

12. Дано действительное положительное число a и целоe число n. Вычислите aⁿ. Решение оформите в виде функции power(a, n). Стандартной функцией возведения в степерь пользоваться нельзя.

**Формат ввода**

Вводится действительное положительное число a и целоe число n.

**Формат вывода**

Выведите ответ на задачу.

**Примечания**

Здесь не нужна рекурсия.

def power(a**,** n):  
 count = **0** b = **1** if n < **0**:  
 while count != n:  
 b \*= a  
 count -= **1** return **1** / b  
 if n == **0**:  
 return **1** else:  
 while count != n:  
 b \*= a  
 count += **1** return b  
  
  
a = float(input())  
n = float(input())  
print(power(a**,** n))

13. Напишите рекурсивную функцию sum(a, b), возвращающую сумму двух целых неотрицательных чисел. Из всех арифметических операций допускаются только +1 и -1. Также нельзя использовать циклы.

**Формат ввода**

Вводятся два удовлетворяющих условию задачи числа. Числа не превышают 900.

**Формат вывода**

Выведите ответ на задачу.

def sum(a**,** b):  
 if b != **0**:  
 return **1** + sum(a**,** b - **1**)  
 else:  
 return a  
  
  
a = int(input())  
b = int(input())  
print(sum(a**,** b))

14. Возводить в степень можно гораздо быстрее, чем за n умножений! Для этого нужно воспользоваться следующими рекуррентными соотношениями: aⁿ = (a²)ⁿ/² при четном n, aⁿ=a⋅aⁿ⁻¹ при нечетном n. Реализуйте алгоритм быстрого возведения в степень. Если вы все сделаете правильно,то сложность вашего алгоритма будет O(logn).

**Формат ввода**

Вводится действительное число a и целое неотрицательное число n.

**Формат вывода**

Выведите ответ на задачу.

def power(a**,** n):  
 if n == **0**:  
 return **1** # if n == 1:  
 # return a  
 if n % **2** != **0**:  
 return a \* power(a**,** n - **1**)  
 if n % **2** == **0**:  
 return power(a \* a**,** n / **2**)  
  
  
a = float(input())  
n = int(input())  
print(power(a**,** n))

15. Для быстрого вычисления наибольшего общего делителя двух чисел используют алгоритм Евклида. Он построен на следующем соотношении: НОД(a,b)=НОД(b,a % b).

Реализуйте рекурсивный алгоритм Евклида в виде функции gcd(a, b).

**Формат ввода**

Вводится два целых положительных числа.

**Формат вывода**

Выведите ответ на задачу.

def gcd(a**,** b):  
 return a if b == **0** else gcd(b**,** a % b)  
  
  
a**,** b = int(input())**,** int(input())  
print(gcd(a**,** b))

16. Даны два натуральных числа n и m.

Сократите дробь (n / m), то есть выведите два других числа p и q таких, что (n / m) = (p / q) и дробь (p / q) — несократимая.

Решение оформите в виде функции ReduceFraction(n, m), получающая значения n и m и возвращающей кортеж из двух чисел: return p, q.

Тогда вывод можно будет оформить как print(\*ReduceFraction(n, m)).

**Формат ввода**

Вводятся два натуральных числа.

**Формат вывода**

Выведите ответ на задачу.

def gcd(a**,** b):  
 return a if b == **0** else gcd(b**,** a % b)  
  
  
def reduceFraction(n**,** m):  
 return n // gcd(n**,** m)**,** m // gcd(n**,** m)  
  
  
n**,** m = int(input())**,** int(input())  
print(\*reduceFraction(n**,** m))

17. Напишите функцию phib(n), которая по данному целому неотрицательному n возвращает n-e число Фибоначчи. В этой задаче нельзя использовать циклы - используйте рекурсию.

**Формат ввода**

Вводится удовлетворяющее условию задачи число.

**Формат вывода**

Выведите ответ на задачу.

**Примечание**

Обратите внимание на нумерацию чисел, показанную в примерах.

def phib(n):  
 if n < **2**:  
 return n  
 return phib(n - **1**) + phib(n - **2**)  
  
  
n = int(input())  
print(phib(n))

18. По данным целым числам n и k (0≤k≤n) вычислите C из n по k.

Решение оформите в виде функции C(n, k).

Для решения используйте рекуррентное соотношение:



И равенства:

С(n, 1)=n

C(n, n)=1

**Формат ввода**

Вводятся удовлетворяющие условию задачи числа n и k.

**Формат вывода**

Выведите ответ на задачу.

def czn(n**,** k):  
 if k == **0**:  
 return **1** elif k == n:  
 return **1** else:  
 return czn(n - **1,** k) + czn(n - **1,** k - **1**)  
  
  
n**,** k = int(input())**,** int(input())  
print(czn(n**,** k))

19. Дана последовательность чисел, завершающаяся числом 0. Найдите сумму всех этих чисел, не используя цикл.

**Формат ввода**

Вводится последовательность целых чисел, оканчивающаяся числом 0 (само число 0 в последовательность не входит, а служит как признак ее окончания).

**Формат вывода**

Выведите ответ на задачу.

def sumit():  
 n = int(input())  
 if n == **0**:  
 return n  
 return n + sumit()  
  
  
print(sumit())

20. Дана последовательность целых чисел, заканчивающаяся числом 0 (включён в последовательность). Выведите эту последовательность в обратном порядке. При решении этой задачи нельзя пользоваться массивами и прочими динамическими структурами данных.Рекурсия вам поможет.

**Формат ввода**

Вводится последовательность целых чисел, оканчивающаяся числом 0.

**Формат вывода**

Выведите ответ на задачу.

def printit():  
 n = int(input())  
 if n != **0**:  
 printit()  
 print(n)  
  
  
printit()

21. Головоломка “Ханойские башни” состоит из трех стержней, пронумерованных числами 1, 2, 3. На стержень 1 надета пирамидка из n дисков различного диаметра в порядке убывания диаметра (снизу находится самый большой диск, а сверху — самый маленький). Диски можно перекладывать с одного стержня на другой по одному, при этом диск нельзя класть на диск меньшего диаметра. Необходимо переложить всю пирамидку со стержня 1 на стержень 3 за минимальное число перекладываний.

Напишите программу, которая решает головоломку; для данного числа дисков n печатает последовательность перекладываний в формате a b c, где a — номер перекладываемого диска, b — номер стержня с которого снимается данный диск, c — номер стержня на который надевается данный диск.

Например, строка 1 2 3 означает перемещение диска номер 1 со стержня 2 на стержень 3. В одной строке печатается одна команда. Диски пронумерованы числами от 1 до n в порядке возрастания диаметров.

Программа должна вывести минимальный (по количеству произведенных операций) способ перекладывания пирамидки из данного числа дисков.

Указание: подумайте, как переложить пирамидку из одного диска? Из двух дисков? Из трех дисков? Из четырех дисков? Пусть мы научились перекладывать пирамидку из n дисков с произвольного стержня на любой другой, как переложить пирамидку из n+1 диска, если можно пользоваться решением для n дисков.

Напишите функцию move (n, x, y), которая печатает последовательность перекладываний дисков для перемещения пирамидки высоты n со стержня номер x на стержень номер y.

**Формат ввода**

Вводится натуральное число — количество дисков.

**Формат вывода**

Выведите ответ на задачу.

def hanoi(n**,** i**,** k):  
 if n == **1**:  
 print(n**,** i**,** k)  
 else:  
 tmp = **6** - i - k  
 hanoi(n - **1,** i**,** tmp)  
 print(n**,** i**,** k)  
 hanoi(n - **1,** tmp**,** k)  
  
  
n = int(input())  
hanoi(n**, 1, 3**)

22. Теорема Лагранжа утверждает, что любое натуральное число можно представить в виде суммы четырех точных квадратов. По данному числу n найдите такое представление: напечатайте от 1 до 4 натуральных чисел, квадраты которых дают в сумме данное число.

**Формат ввода**

Программа получает на вход одно натуральное число n < 10000.

**Формат вывода**

Программа должна вывести от 1 до 4 натуральных чисел, квадраты которых дают в сумме данное число.

### Кортежи, цикл for, списки

1. Даны два целых числа A и B (при этом A≤B). Выведите все числа от A до B включительно.

**Формат ввода**

Вводятся два целых числа.

**Формат вывода**

Выведите ответ на задачу.

a**,** b = int(input())**,** int(input())  
numbers = range(a**,** b + **1**)  
for number in numbers:  
 print(number**,** end=' ')

2. Даны два целых числа A и В. Выведите все числа от A до B включительно, в порядке возрастания,если A < B, или в порядке убывания в противном случае.

**Формат ввода**

Вводятся два целых числа.

**Формат вывода**

Выведите ответ на задачу.

a**,** b = int(input())**,** int(input())  
if a < b:  
 numbers = range(a**,** b + **1**)  
 for number in numbers:  
 print(number**,** end=' ')  
else:  
 numbers = range(a**,** b - **1,** -**1**)  
 for number in numbers:  
 print(number**,** end=' ')

3. Дано натуральное число n. Напечатайте все n-значные нечетные натуральные числа в порядке убывания.

**Формат ввода**

Вводится натуральное число.

**Формат вывода**

Выведите ответ на задачу.

n = int(input())  
numbers = range(**10** \*\* n - **1,** (**10** \*\* (n - **1**)) - **1,** -**2**)  
for number in numbers:  
 print(number**,** end=' ')

4. По данному натуральном n вычислите сумму 1²+2²+3²+...+n².

**Формат ввода**

Вводится натуральное число.

**Формат вывода**

Выведите ответ на задачу.

n = int(input())  
sumsqr = **0**numbers = range(**1,** n + **1**)  
for number in numbers:  
 sumsqr += number \*\* **2**print(sumsqr)

Напишите программу, которая по данному числу n от 1 до 9 выводит на экран n флагов. Изображение одного флага имеет размер 4×4 символов, между двумя соседними флагами также имеется пустой (из пробелов) столбец. Разрешается вывести пустой столбец после последнего флага. Внутри каждого флага должен быть записан его номер — число от 1 до n.

**Формат ввода**

Вводится натуральное число.

**Формат вывода**

Выведите ответ на задачу.

+\_\_\_

|1 /

|\_\_\

|

n = int(input())  
numbers = range(**1,** n + **1**)  
a = '+\_\_\_ '  
c = '|\_\_\\ '  
d = '| '  
print(a \* n**,** sep='')  
for number in numbers:  
 if number == n:  
 print(f'|{number} / '**,** end='\n')  
 else:  
 print(f'|{number} / '**,** end='')  
print(c \* n)  
print(d \* n)

5. Дано несколько чисел. Подсчитайте, сколько из них равны нулю, и выведите это количество.

**Формат ввода**

Cначала вводится число N, затем вводится ровно N целых чисел.

**Формат вывода**

Выведите ответ на задачу.

m = int(input())  
numbers = range(**1,** m + **1**)  
count = **0**for n in numbers:  
 a = int(input())  
 if a == **0**:  
 count += **1**print(count)

6. По данному натуральному n≤9 выведите лесенку из n ступенек, i-я ступенька состоит из чисел от 1 до i без пробелов.

**Формат ввода**

Вводится натуральное число.

**Формат вывода**

Выведите ответ на задачу.

m = int(input())  
numbers = tuple(range(**1,** m + **1**))  
a = str()  
for n in numbers:  
 a += str(numbers[n-**1**])  
 print(a)

чужое(обратить внимание на \*)

a = int(input())  
for n in range(**1,** a + **1**):  
 print(\*range(**1,** n + **1**)**,** sep='')

7. Найдите и выведите все двузначные числа, которые равны удвоенному произведению своих цифр.

**Формат ввода**

Программа не требует ввода данных с клавиатуры, просто выводит список искомых чисел.

**Формат вывода**

Выведите ответ на задачу.

myrange = range(**10, 100**)  
for n in myrange:  
 if **2** \* (n // **10**) \* (n % **10**) == n:  
 print(n**,** end='')

8. Даны числа a, b, c, d, e. Подсчитайте количество таких целых чисел от 0 до 1000 (включительно), которые являются корнями уравнения (ax³+bx²+cx+d)/(x-e)=0, и выведите их количество.

**Формат ввода**

Вводятся целые числа a, b, c, d и e.

**Формат вывода**

Выведите ответ на задачу.

a**,** b**,** c**,** d**,** e = int(input())**,** int(input())**,** int(input())**,** \  
 int(input())**,** int(input())  
myrange = range(**1001**)  
count = **0**for n in myrange:  
 numerator = (a \* n \*\* **3** + b \* n \*\* **2** + c \* n + d)  
 denominator = (n - e)  
 if numerator == **0** and denominator != **0**:  
 count += **1**print(count)

9. По данному натуральному n вычислите сумму 1!+2!+3!+...+n!. В решении этой задачи можно использовать только один цикл.

**Формат ввода**

Вводится натуральное число n.

**Формат вывода**

Выведите ответ на задачу.

def f(n):  
 assert n >= **0** if n == **0**:  
 return **1** return f(n - **1**) \* n  
  
  
m**,** n = **0,** int(input())  
for n in range(**1,** n + **1**):  
 m += f(n)  
print(m)

10. Для настольной игры используются карточки с номерами от 1 до N.Одна карточка потерялась. Найдите ее, зная номера оставшихся карточек.

**Формат ввода**

Дано число N, далее N-1 номер оставшихся карточек (различные числа от 1 до N).

**Формат вывода**

Программа должна вывести номер потерянной карточки.

**Примечания**

Для самых умных – массивами и аналогичными структурами данныхпользоваться нельзя.

cardquol**,** n = int(input())**, 0**firstlist = list(range(**1,** cardquol + **1**))  
k = len(firstlist)  
while n < k - **1**:  
 n += **1** a = int(input())  
 if a in firstlist:  
 firstlist.remove(a)  
print(\*firstlist)

11. Даны два четырёхзначных числа A и B. Выведите все четырёхзначные числа на отрезке от A до B, запись которых является палиндромом.

**Формат ввода**

Вводятся два целых числа A и B

**Формат вывода**

Выведите ответ на задачу.

a**,** b = int(input())**,** int(input())  
for n in range(a**,** b + **1**):  
 c = str(n)  
 if int(c) == int(c[-**1**::-**1**]):  
 print(c)

12.Выведите все элементы списка с четными индексами (то есть A[0], A[2], A[4], ...). Программа должна быть эффективной и не выполнять лишних действий!

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

mynumlist = list(map(int**,** input().split()))  
i = **0**while i < len(mynumlist):  
 print(mynumlist[i]**,** end=' ')  
 i += **2**

или

mynumlist = list(map(int**,** input().split()))  
print(\*mynumlist[::**2**])

13. Выведите все четные элементы списка.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

mynumlist = list(map(int**,** input().split()))  
for n in mynumlist:  
 if n % **2** == **0**:  
 print(n**,** end=' ')

14. Найдите количество положительных элементов в данном списке.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

mynumlist = list(map(int**,** input().split()))  
c = **0**for n in mynumlist:  
 if n > **0**:  
 c += **1**print(c)

15. Найдите наибольшее значение в списке и индекс последнего элемента, который имеет данное значение за один проход по списку, не модифицируя этот список и не используя дополнительного списка.

Выведите два значения.

mynumlist = list(map(int**,** input().split()))  
i = **1**indmax = **0**maxinlist = mynumlist[**0**]  
while i < len(mynumlist):  
 if mynumlist[i] >= maxinlist:  
 maxinlist = mynumlist[i]  
 indmax = i  
 i += **1**print(maxinlist**,** indmax)

16. Дан список чисел. Выведите все элементы списка, которые больше предыдущего элемента.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

mynumlist = list(map(int**,** input().split()))  
i = **1**while i < len(mynumlist):  
 if mynumlist[i] > mynumlist[i - **1**]:  
 print(mynumlist[i]**,** end=' ')  
 i += **1**

лучше

a = list(map(int**,** input().split()))  
print(\*[a[i] for i in range(**1,** len(a)) if a[i] > a[i-**1**]])

17. Дан список. Определите, является ли он монотонно возрастающим(то есть верно ли, что каждый элемент этого списка больше предыдущего).Выведите YES, если массив монотонно возрастает и NO в противном случае.Решение оформите в виде функции IsAscending(A).В данной функции должен быть один цикл while, не содержащий вложенных условий и циклов — используйте схему линейного поиска.

def is\_ascending(a):  
 i = **1** while i < len(a) and a[i] > a[i - **1**]:  
 i += **1** return print(int(i == len(a)) \* 'YES'**,** int(len(a) != i) \* 'NO'**,** sep='')  
  
  
a = list(map(int**,** input().split()))  
is\_ascending(a)

18. Дан список чисел. Если в нем есть два соседних элемента одного знака, выведите эти числа. Если соседних элементов одного знака нет - не выводите ничего. Если таких пар соседей несколько - выведите первую пару.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

a = list(map(int**,** input().split()))  
i = **1**while i < len(a):  
 if a[i] // a[i - **1**] < **0**:  
 i += **1** else:  
 print(a[i - **1**]**,** a[i]**,** sep=' ')  
 i = len(a)

19. Дан список чисел. Определите, сколько в этом списке элементов, которые больше двух своих соседей и выведите количество таких элементов.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

a = list(map(int**,** input().split()))  
n = **0**for i in range(**1,** len(a) - **1**):  
 if a[i - **1**] < a[i] > a[i + **1**]:  
 n += **1**print(n)

20. Дан список чисел. Выведите значение наибольшего элемента в списке, а затем индекс этого элемента в списке. Если наибольших элементов несколько, выведите их значение и индекс первого из них.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

a = list(map(int**,** input().split()))  
i = imax = **0**amax = a[i]  
for i in range(**1,** len(a)):  
 if a[i] > amax:  
 amax = a[i]  
 imax = i  
 elif a[i] == amax:  
 amax = a[i]  
print(amax**,** imax**,** sep=' ')

21. Выведите значение наименьшего из всех положительных элементов в списке. Известно, что в списке есть хотя бы один положительный элемент, а значения всех элементов списка по модулю не превосходят 1000.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

a = list(map(int**,** input().split()))  
i = **0**amin = **1000**for i in range(**1,** len(a)):  
 if **0** < a[i] < amin:  
 amin = a[i]  
 i += **1**print(amin)

22. Выведите значение наименьшего нечетного элемента списка, гарантируется, что хотя бы один нечётный элемент в списке есть.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

a = list(map(int**,** input().split()))  
amin = a[**0**]  
for i in range(**1,** len(a)):  
 if a[i] % **2** == **1**:  
 if amin % **2** == **0**:  
 amin = a[i]  
 elif a[i] <= amin:  
 amin = a[i]  
print(amin)

23. Выведите элементы данного списка в обратном порядке, не изменяя сам список.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

a = list(map(int**,** input().split()))  
print(\*a[-**1**::-**1**]**,** sep=' ')

24. Переставьте элементы данного списка в обратном порядке, затем выведите элементы полученного списка. Эта задача отличается от предыдущей тем, что вам нужно изменить значения элементов самого списка, поменяв местами A[0] c A[n-1], A[1] с A[n-2], а затем вывести элементы списка подряд.

Предлагается в учебных целях проделать это вручную, например, не используя срезов и стандартных функций.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

a = list(map(int**,** input().split()))  
c = len(a)  
for i in range(c // **2**):  
 b = a[i]  
 a[i] = a[c - **1** - i]  
 a[c - **1** - i] = b  
print(\*a**,** sep=' ')

25. Дан список из чисел и индекс элемента в списке k. Удалите из списка элемент с индексом k, сдвинув влево все элементы, стоящие правее элемента с индексом k.

Программа получает на вход список, затем число k. Программа сдвигает все элементы,а после этого удаляет последний элемент списка при помощи метода pop().

Программа должна осуществлять сдвиг непосредственно в списке, а не делать это при выводе элементов. Также нельзя использовать дополнительный список.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке. В следующей строке вводится одно целое число.

**Формат вывода**

Выведите ответ на задачу.

**Примечание**

Метод pop(index) у списка позволяет решить эту задачу в одну команду. Но в этой задаче мы хотим, чтобы метод pop(index) не использовался: реализуйте удаление элемента самостоятельно.

a = input().split()  
k = int(input())  
for i in range(k**,** len(a) - **1**):  
 a[i] = a[i + **1**]  
a.pop()  
print(\*a**,** sep=' ')

26. Дан список целых чисел, число k и значение C. Необходимо вставить в список на позицию с индексом k элемент, равный C, сдвинув все элементы, имевшие индекс не менее k, вправо.

Поскольку при этом количество элементов в списке увеличивается, после считывания списка в его конец нужно будет добавить новый элемент, используя метод append.

Вставку необходимо осуществлять уже в считанном списке, не делая этого при выводе и не создавая дополнительного списка.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке. В следующей строке вводятся два целых числа.

**Формат вывода**

Выведите ответ на задачу.

**Примечание**

Метод insert у списка позволяет решить эту задачу в одну команду. Но в этой задаче мы хотим, чтобы метод insert не использовался: реализуйте вставку элемента самостоятельно.

a = input().split()  
k**,** c = map(int**,** input().split())  
last = a[len(a) - **1**]  
if len(a) > k:  
 for i in range(len(a) - **1,** k**,** -**1**):  
 a[i] = a[i - **1**]  
 a[k] = c  
 a.append(last)  
else:  
 a.append(c)  
print(\*a**,** sep=' ')

\*\*

27. Напишите программу, которая находит в массиве элемент, самый близкий по величине к данному числу.

**Формат ввода**

В первой строке задается одно натуральное число N, не превосходящее 1000 – размер массива. Во второй строке содержатся N чисел – элементы массива (целые числа, не превосходящие по модулю 1000). В третьей строке вводится одно целое число x, не превосходящее по модулю 1000.

**Формат вывода**

Вывести значение элемента массива, ближайшее к x. Если таких чисел несколько, выведите любое из них.

n = int(input())  
a = list(map(int**,** input().split()))[:n]  
x = int(input())  
newlist = []  
for i in range(len(a)):  
 b = abs(a[i] - x)  
 newlist.append(b)  
imin = **0**mindif = newlist[**0**]  
for i in range(**1,** len(newlist)):  
 if newlist[i] <= mindif:  
 mindif = newlist[i]  
 imin = i  
print(a[imin])

28. Петя перешёл в другую школу. На уроке физкультуры ему понадобилось определить своё место в строю.Помогите ему это сделать.

**Формат ввода**

Программа получает на вход невозрастающую последовательность натуральных чисел,означающих рост каждого человека в строю. После этого вводится число X – рост Пети.Все числа во входных данных натуральные и не превышают 200.

**Формат вывода**

Выведите номер, под которым Петя должен встать в строй. Если в строю есть люди с одинаковым ростом,таким же, как у Пети, то он должен встать после них.

a = list(map(int**,** input().split()))  
x = int(input())  
petind = len(a)  
for i in range(len(a) - **1,** -**1,** -**1**):  
 if a[i] < x:  
 petind = i  
 elif a[i] == [x]:  
 print(i)  
 break  
print(petind + **1**)

без break

a = list(map(int**,** input().split()))  
x = int(input())  
count = **1**for i in range(len(a)):  
 if a[i] >= x:  
 count += **1**print(count)

29.Дан список, упорядоченный по неубыванию элементов в нем.Определите, сколько в нем различных элементов.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

a = list(map(int**,** input().split()))  
count = **1**for i in range(len(a) - **1**):  
 if a[i + **1**] > a[i]:  
 count += **1**print(count)

30. Переставьте соседние элементы списка (A[0] c A[1],A[2] c A[3] и т.д.).Если элементов нечетное число, то последний элемент остается на своем месте.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

a = list(map(int**,** input().split()))  
for i in range(**0,** len(a) - **1, 2**):  
 a[i]**,** a[i + **1**] = a[i + **1**]**,** a[i]  
print(\*a)

31. Циклически сдвиньте элементы списка вправо(A[0] переходит на место A[1], A[1] на место A[2], ..., последний элемент переходит на место A[0]).

**Примечания**

Используйте минимально возможное количество операций присваивания.

Не используйте срезы.

Измените начальный список и вывдите ответ с помощью print(\*listName).

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

a = list(map(int**,** input().split()))  
a.append(a[len(a) - **1**])  
for i in range(len(a) - **1, 0,** -**1**):  
 a[i]**,** a[i - **1**] = a[i - **1**]**,** a[i]  
a.pop()  
print(\*a)

32. В списке все элементы попарно различны. Поменяйте местами минимальный и максимальный элемент этого списка.

**Формат ввода**

Вводится список целых чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

a = list(map(int**,** input().split()))  
amax = amin = a[**0**]  
imin = imax = **0**for i in range(len(a)):  
 if a[i] > amax:  
 amax = a[i]  
 imax = i  
 elif a[i] < amin:  
 amin = a[i]  
 imin = i  
a[imax]**,** a[imin] = a[imin]**,** a[imax]  
print(\*a)

33.Дан список, заполненный произвольными целыми числами. Найдите в этом списке два числа, произведение которых максимально. Выведите эти числа в порядке неубывания.

Решение должно иметь сложность O(n), где n - размер списка. То есть сортировку использовать нельзя.

a = list(map(int**,** input().split()))  
b = []  
c = []  
count = **0**while count != **2**:  
 b.append(min(a))  
 a.remove(min(a))  
 c.append(max(a))  
 a.remove(max(a))  
 count += **1**if b[**0**] \* b[**1**] >= c[**0**] \* c[**1**]:  
 print(\*b)  
else:  
 print(c[**1**]**,** c[**0**])

34. В данном списке из n≤10⁵ целых чисел найдите три числа,произведение которых максимально.

Решение должно иметь сложность O(n), где n - размер списка. То есть сортировку использовать нельзя.

Выведите три искомых числа в любом порядке.

a = list(map(int**,** input().split()))  
d = a[:]  
b = []  
c = []  
count = **0**maxa = max(a)  
a.remove(maxa)  
while count != **2**:  
 b.append(min(d))  
 d.remove(min(d))  
 c.append(max(a))  
 a.remove(max(a))  
 count += **1**if abs(b[**0**] \* b[**1**]) \* maxa >= (c[**0**] \* c[**1**]) \* maxa:  
 print(\*b**,** maxa)  
else:  
 print(c[**1**]**,** c[**0**]**,** maxa)

35. Дан список чисел. Посчитайте, сколько в нем пар элементов, равных друг другу. Считается, что любые два элемента, равные друг другу образуют одну пару, которую необходимо посчитать.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

a = list(map(int**,** input().split()))  
quolduble = **0**for i in range(len(a)):  
 b = a[i]  
 for i in range(i + **1,** len(a)):  
 if a[i] - b == **0**:  
 quolduble += **1**print(quolduble)

и вот так(не моё)

s = list(map(int**,** input().split()))  
print(sum(s[i+**1**:].count(s[i]) for i in range(len(s))))

36. Дан список. Выведите те его элементы, которые встречаются в списке только один раз. Элементы нужно выводить в том порядке, в котором они встречаются в списке.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

a = list(map(int**,** input().split()))  
for i in range(len(a)):  
 if a.count(a[i]) == **1**:  
 print(a[i]**,** end=' ')

или

a = list(map(int**,** input().split()))  
print(\*[a[i] for i in range(len(a)) if a.count(a[i]) == **1**])

37. N кеглей выставили в один ряд, занумеровав их слева направо числами от 1 до N. Затем по этому ряду бросили K шаров, при этом i-й шар сбил все кегли с номерами от lᵢ до rᵢ включительно. Определите, какие кегли остались стоять на месте.

**Формат ввода**

Программа получает на вход количество кеглей N и количество бросков K. Далее идет K пар чисел lᵢ, rᵢ, при этом 1 ≤ lᵢ ≤ rᵢ ≤ N ≤ 100.

**Формат вывода**

Программа должна вывести последовательность из N символов, где j-й символесть “I”, если j-я кегля осталась стоять, или “.”, если j-я кегля была сбита.

n**,** k = map(int**,** input().split())  
count = **0**mylist = []  
myfinal = []  
for b in range(n):  
 myfinal.append('I')  
while count != k:  
 l**,** i = map(int**,** input().split())  
 myrange = range(l**,** i + **1**)  
 count += **1** for number in myrange:  
 if mylist.count(number) == **0**:  
 mylist.append(number)  
for m in range(len(mylist)):  
 myfinal[mylist[m] - **1**] = '.'  
print(\*myfinal**,** sep='')

38.

39.

40.

Сортировка

1. Даны два целочисленных списка A и B, упорядоченных по неубыванию. Объедините их в один упорядоченный список С (то есть он должен содержать len(A)+len(B) элементов). Решение оформите в виде функции merge(A, B), возвращающей новый список. Алгоритм должен иметь сложность O(len(A)+len(B)). Модифицировать исходные списки запрещается. Использовать функцию sorted и метод sort запрещается.

**Формат ввода**

Программа получает на вход два неубывающих списка, каждый в отдельной строке.

**Формат вывода**

Программа должна вывести последовательность неубывающих чисел, полученных объединением двух данных списков.

def merge(a**,** b):  
 i = j = **0** sumlist = []  
 while i <= len(a) - **1**:  
 if j <= len(b) - **1**:  
 if a[i] >= b[j]:  
 sumlist.append(b[j])  
 j += **1** else:  
 sumlist.append(a[i])  
 i += **1** else:  
 for l in range(i**,** len(a)):  
 sumlist.append(a[l])  
 i += **1** if j < len(b):  
 for l in range(j**,** len(b)):  
 sumlist.append(b[l])  
 return sumlist  
  
  
a = list(map(int**,** input().split()))  
b = list(map(int**,** input().split()))  
print(\*merge(a**,** b))

2. Даны два списка, упорядоченных по возрастанию (каждый список состоит из различных элементов).

Найдите пересечение множеств элементов этих списков, то есть те числа, которые являются элементами обоих списков. Алгоритм должен иметь сложность O(len(A)+len(B)).

Решение оформите в виде функции Intersection(A, B). Функция должна возвращать список пересечения данных списков в порядке возрастания элементов. Модифицировать исходные списки запрещается.

**Формат ввода**

Программа получает на вход два возрастающих списка, каждый в отдельной строке.

**Формат вывода**

Программа должна вывести последовательность возрастающих чисел, являющихся элементами обоих списков.

3. Отсортируйте данный массив, используя встроенную сортировку.

**Формат ввода**

Первая строка входных данных содержит количество элементов в массиве N, N ≤ 10⁵. Далее идет N целых чисел, не превосходящих по абсолютной величине 10⁹.

**Формат вывода**

Выведите эти числа в порядке неубывания.

n = int(input())  
a = list(map(int**,** input().split()[:n]))  
a.sort()  
print(\*a)

4. В обувном магазине продается обувь разного размера. Известно, что одну пару обуви можно надеть на другую, если она хотя бы на три размера больше. В магазин пришел покупатель.Требуется определить, какое наибольшее количество пар обуви сможет предложить ему продавец так, чтобы он смог надеть их все одновременно.

**Формат ввода**

Сначала вводится размер ноги покупателя (обувь меньшего размера он надеть не сможет), в следующей строке — размеры каждой пары обуви в магазине через пробел. Размер — натуральное число, не превосходящее 100.

**Формат вывода**

Выведите единственное число — максимальное количество пар обуви, которое сможет надеть покупатель.

size = int(input())  
count = **0**a = list(map(int**,** input().split()))  
a.sort()  
for i in a:  
 if i >= size:  
 count += **1** size = i + **3**print(count)

5. Системный администратор вспомнил, что давно не делал архива пользовательских файлов. Однако, объем диска, куда он может поместить архив, может быть меньше чем суммарный объем архивируемых файлов.

Известно, какой объем занимают файлы каждого пользователя.

Напишите программу, которая по заданной информации о пользователях и свободному объему на архивном диске определит максимальное число пользователей, чьи данные можно поместить в архив.

**Формат ввода**

Программа получает на вход в одной строке число S – размер свободного места на диске (натуральное, не превышает 10000), и число N – количество пользователей (натуральное, не превышает 100), после этого идет N чисел - объем данных каждого пользователя (натуральное, не превышает 1000), записанных каждое в отдельной строке.

**Формат вывода**

Выведите наибольшее количество пользователей, чьи данные могут быть помешены в архив.

**Примеры**

s**,** n = map(int**,** input().split())  
a = []  
i = count = **0**while i <= n - **1**:  
 a.append(int(input()))  
 i += **1**a.sort()  
for space in a:  
 if s - space >= **0**:  
 s = s - space  
 count += **1**print(count)

или

s**,** n = map(int**,** input().split())  
a = sorted(int(input()) for \_ in range(n))  
count = **0**for space in a:  
 if s - space >= **0**:  
 s = s - space  
 count += **1**print(count)

6. Штаб гражданской обороны Тридесятой области решил обновить план спасения на случай ядерной атаки. Известно, что все n селений Тридесятой области находятся вдоль одной прямой дороги. Вдоль дороги также расположены m бомбоубежищ, в которых жители селений могут укрыться на случай ядерной атаки.

Чтобы спасение в случае ядерной тревоги проходило как можно эффективнее, необходимо для каждого селения определить ближайшее к нему бомбоубежище.

**Формат ввода**

В первой строке вводится число n - количество селений (1 <= n <= 100000). Вторая строка содержит n различных целых чисел, i-е из этих чисел задает расстояние от начала дороги до i-го селения. В третьей строке входных данных задается число m - количество бомбоубежищ (1 <= m <= 100000). Четвертая строка содержит m различных целых чисел, i-е из этих чисел задает расстояние от начала дороги до i-го бомбоубежища. Все расстояния положительны и не превышают 10⁹. Селение и убежище могут располагаться в одной точке.

**Формат вывода**

Выведите n чисел - для каждого селения выведите номер ближайшего к нему бомбоубежища. Бомбоубежища пронумерованы от 1 до m в том порядке, в котором они заданы во входных данных.

**Указание**

Создайте список кортежей из пар (позиция селения, его номер в исходном списке), а также аналогичный список для бомбоубежищ. Отсортируйте эти списки.

Перебирайте селения в порядке возрастания.

Для селения ближайшими могут быть два соседних бомбоубежища, среди них надо выбрать ближайшее. При переходе к следующему селению не обязательно искать ближайшее бомбоубежище с самого начала. Его можно искать начиная с позиции, найденной для предыдущего города. Аналогично, не нужно искать подходящее бомбоубежище до конца списка бомбоубежищ: достаточно найти самое близкое. Если Вы неэффективно реализуете эту часть, то решение тесты не пройдет.

Для хранения ответа используйте список, где индекс будет номером селения, а по этому индексу будет запоминаться номер бомбоубежища.

Не проходит по времени.

n = int(input())  
village = list(map(int**,** input().split()[:n]))  
m = int(input())  
save = list(map(int**,** input().split()[:m]))  
village\_list = []  
save\_list = []  
village\_index = list(range(n))  
save\_index = list(range(m))  
final\_list = []  
for i in range(n):  
 c = (village[i]**,** village\_index[i] + **1**)  
 village\_list.append(c)  
village\_list.sort()  
for j in range(m):  
 c = (save[j]**,** save\_index[j] + **1**)  
 save\_list.append(c)  
save\_list.sort()  
j = **0**i = **0**while j != len(save\_list) - **1** and i != len(village\_list):  
 if abs(village\_list[i][**0**] - save\_list[j][**0**]) \  
 <= abs(village\_list[i][**0**] - save\_list[j + **1**][**0**]):  
 d = (village\_list[i][**1**]**,** save\_list[j][**1**])  
 final\_list.append(d)  
 i += **1** else:  
 j += **1**else:  
 if j >= len(save\_list) - **1**:  
 while i < len(village\_list):  
 d = (village\_list[i][**1**]**,** save\_list[j][**1**])  
 final\_list.append(d)  
 i += **1**final\_list.sort()  
for i in range(len(final\_list)):  
 print(final\_list[i][**1**]**,** end=' ')

\*\*

n = int(input())  
sn = list(map(int**,** input().split()))  
m = int(input())  
sm = list(map(int**,** input().split()))  
a = []  
b = []  
c = sn  
for i in range(n):  
 sn1 = (sn[i]**,** i + **1**)  
 a.append(sn1)  
for i in range(m):  
 sm1 = (sm[i]**,** i + **1**)  
 b.append(sm1)  
a.sort()  
b.sort()  
i = **0**j = **0**while i < n and j < m - **1**:  
 if abs(a[i][**0**] - b[j][**0**]) <= abs(a[i][**0**] - b[j + **1**][**0**]):  
 c[a[i][**1**] - **1**] = (b[j][**1**])  
 i += **1** else:  
 j += **1** if j == m - **1**:  
 while i < n:  
 c[a[i][**1**] - **1**] = (b[j][**1**])  
 i += **1** else:  
 if abs(a[i][**0**] - b[j][**0**]) <= abs(a[i][**0**] - b[j + **1**][**0**]):  
 c[a[i][**1**] - **1**] = (b[j][**1**])  
 i += **1**if m == **1**:  
 for i in range(n):  
 c[i] = **1**print(\*c)

7. В олимпиаде по информатике принимало участие несколько человек.

Определите и выведите средние баллы участников олимпиады в 9 классе, в 10 классе, в 11 классе.

**Входные данные**

Информация о результатах олимпиады записана в файле, каждая строка которого имеет вид:

фамилия имя класс балл.

Фамилия и имя — текстовые строки, не содержащие пробелов. Класс - одно из трех чисел 9, 10, 11. Балл - целое число от 0 до 100.

В этой задаче файл необходимо считывать построчно, не сохраняя содержимое файла в памяти целиком.

**Выходные данные**

Выведите три числа: средние баллы по 9 классу, по 10 классу, по 11 классу. Входной файл в кодировке utf-8 (используйте open('input.txt', 'r', encoding='utf-8')).

fin = open('input.txt'**,** 'r'**,** encoding='utf8')  
i = k = m = **0**n = t = e = **0**for line in fin:  
 l = list(line.split())  
 if int(l[**2**]) == **9**:  
 i += **1** n += int(l[**3**])  
 if int(l[**2**]) == **10**:  
 k += **1** t += int(l[**3**])  
 if int(l[**2**]) == **11**:  
 m += **1** e += int(l[**3**])  
print(n / i**,** t / k**,** e / m)

8.Известно, что фамилии всех участников — различны. Сохраните в массивах список всех участников и выведите его, отсортировав по фамилии в лексикографическом порядке. При выводе указываете фамилию, имя участника и его балл.

Используйте для ввода и вывода файлы input.txt и output.txt с указанием кодировки utf8. Например, для чтения откройте файл с помощью open('input.txt', 'r', encoding='utf8').

**Входные данные**

Строки вида "Фамилия Имя НомерШколы Балл".

**Выходные данные**

Строки вида "Фамилия Имя Балл", отсортированные по фамилии.

**Примечание**

Если у Вас Wrong Answer на первом тесте и в вердикте в качестве правильного ответа показываются знаки вопросов, это не значит, что их действительно нужно выводить. Это баг Курсеры – в вердикте кириллица не поддерживается. Курсера знает о проблеме с 25.10.2018 и возможно починит.

В итоге, при WA на первом тесте не стоит смотреть на вердикт, нужно искать ошибку в решении.

fin = open('input.txt'**,** 'r'**,** encoding='utf8')  
outFile = open('output.txt'**,** 'w'**,** encoding='utf8')  
scoolist = []  
for line in fin:  
 scool = [line.split()[**0**]**,** line.split()[**1**]**,** line.split()[**3**]]  
 scoolist.append(scool)  
 scoolist.sort()  
for n in range(len(scoolist)):  
 print(\*scoolist[n]**,** file=outFile)  
fin.close()  
outFile.close()

9.Дан список из N (N≤2\*10⁵) элементов, которые принимают целые значения от 0 до 100 (100 включая).

Отсортируйте этот список в порядке неубывания элементов. Выведите полученный список.

Решение оформите в виде функции CountSort(A), которая модифицирует передаваемый ей список. Использовать встроенные функции сортировки нельзя.

def countsort():  
 marks = list(map(int**,** input().split()))  
 cntMarks = [**0**] \* (max(marks) + **1**)  
 for mark in marks:  
 cntMarks[mark] += **1** for nowMark in range(max(marks) + **1**):  
 print((str(nowMark) + ' ') \* cntMarks[nowMark]**,** end='')  
  
  
countsort()

или лучше

marks = []  
  
  
def countsort(marks):  
 cntMarks = [**0**] \* (max(marks) + **1**)  
 for mark in marks:  
 cntMarks[mark] += **1** for nowMark in range(max(marks) + **1**):  
 print((str(nowMark) + ' ') \* cntMarks[nowMark]**,** end='')  
  
  
m = list(map(int**,** input().split()))  
countsort(m)

10. На региональном этапе Всероссийской олимпиады школьников по информатике в 2009 году предлагалась следующая задача.

Всем известно, что со временем клавиатура изнашивается, и клавиши на ней начинают залипать. Конечно, некоторое время такую клавиатуру еще можно использовать, но для нажатий клавиш приходиться использовать большую силу.

При изготовлении клавиатуры изначально для каждой клавиши задается количество нажатий, которое она должна выдерживать. Если знать эти величины для используемой клавиатуры, то для определенной последовательности нажатых клавиш можно определить, какие клавиши в процессе их использования сломаются, а какие — нет.

Требуется написать программу, определяющую, какие клавиши сломаются в процессе заданного варианта эксплуатации клавиатуры.

**Формат ввода**

Первая строка входных данных содержит целое число n (1≤n≤1000) — количество клавиш на клавиатуре. Вторая строка содержит n целых чисел —с₁, с₂, … , сn, где сᵢ (1≤cᵢ≤100000) — количество нажатий,выдерживаемых i-ой клавишей. Третья строка содержит целое число k (1≤k≤100000) — общее количество нажатий клавиш, и последняя строка содержит k целых чисел pj (1≤pj≤n) — последовательность нажатых клавиш.

**Формат вывода**

Программа должна вывести n строк, содержащих информацию об исправности клавиш. Если i-я клавиша сломалась, то i-ая строка должна содержать слово YES, если же клавиша работоспособна — слово NO.

11. В олимпиаде по информатике принимало участие несколько человек. Победителем олимпиады становится человек, набравший больше всех баллов. Победители определяются независимо по каждому классу. Определите количество баллов, которое набрал победитель в каждом классе. Гарантируется, что в каждом классе был хотя бы один участник.

**Формат ввода**

Информация о результатах олимпиады записана в файле, каждая строка которого имеет вид:фамилия имя класс балл.

Фамилия и имя — текстовые строки, не содержащие пробелов. Класс - одно из трех чисел 9, 10, 11. Балл - целое число от 0 до 100.

В этой задаче файл необходимо считывать построчно, не сохраняя содержимое файла в памяти целиком.

**Формат вывода**

Выведите три числа: баллы победителя олимпиады по 9 классу, по 10 классу, по 11 классу. Входной файл в кодировке utf-8 (В Python используйте open('input.txt', 'r', encoding='utf-8')).

12. В олимпиаде участвовало N человек. Каждый получил определенное количество баллов, при этом оказалось, что у всех участников разное число баллов. Упорядочите список участников олимпиады в порядке убывания набранных баллов.

**Формат ввода**

Программа получает на вход число участников олимпиады N. Далее идет N строк, в каждой строке записана фамилия участника, затем, через пробел, набранное им количество баллов.

**Формат вывода**

Выведите список участников (только фамилии) в порядке убывания набранных баллов.

n = int(input())  
myList = []  
for i in range(n):  
 scoolMan = input().split()  
 myList.append([int(scoolMan[**1**])**,** scoolMan[**0**]])  
myList.sort(reverse=True)  
for j in myList:  
 print(j[**1**])

13. В условиях предыдущей задачи определите количество школьников, ставших победителями в каждом классе. Победителями объявляются все, кто набрал наибольшее число баллов по данному классу. Гарантируется, что в каждом классе был хотя бы один участник.

**Формат вывода**

Выведите три числа: количество победителей олимпиады по 9 классу, по 10 классу, по 11 классу.

14. Для поступления в вуз абитуриент должен предъявить результаты трех экзаменов в виде ЕГЭ, каждый из них оценивается целым числом от 0 до 100 баллов. При этом абитуриенты, набравшие менее 40 баллов (неудовлетворительную оценку) по любому экзамену из конкурса выбывают. Остальные абитуриенты участвуют в конкурсе по сумме баллов за три экзамена.

В конкурсе участвует N человек, при этом количество мест равно K. Определите проходной балл, то есть такое количество баллов, что количество участников, набравших столько или больше баллов не превосходит K, а при добавлении к ним абитуриентов, набравших наибольшее количество баллов среди непринятых абитуриентов, общее число принятых абитуриентов станет больше K.

**Формат ввода**

Программа получает на вход количество мест K. Далее идут строки с информацией об абитуриентах, каждая из которых состоит из имени (текстовая строка содержащая произвольное число пробелов) и трех чисел от 0 до 100, разделенных пробелами.

Используйте для ввода файл input.txt с указанием кодировки utf8 (для создания такого файла на своем компьютере в программе Notepad++ следует использовать кодировку UTF-8 без BOM).

**Формат вывода**

Программа должна вывести проходной балл в конкурсе. Выведенное значение должно быть минимальным баллом, который набрал абитуриент, прошедший по конкурсу.

Также возможны две ситуации, когда проходной балл не определен.

Если будут зачислены все абитуриенты, не имеющие неудовлетворительных оценок, программа должна вывести число 0.

Если количество имеющих равный максимальный балл абитуриентов больше чем K, программа должна вывести число 1.

Используйте для вывода файл output.txt с указанием кодировки utf8.

**Предупреждение**

Пожалуйста, тестируйте файловый ввод и вывод на своем компьютере. В этой задаче слушатели часто получают ошибки вроде RE на первом тесте, протестировав у себя с помощью консоли и просто заменив input() на чтение из файла перед сдачей. К сожалению, такую замену не всегда удается сделать без ошибок, и решение слушателей действительно перестает правильно работать даже на первом тесте.

inFile = open('input.txt'**,** 'r'**,** encoding='utf8')  
outFile = open('output.txt'**,** 'w'**,** encoding='utf8')  
k = int(inFile.readline())  
myList = []  
for line in inFile:  
 if int(line.split()[-**1**]) >= **40** and int(line.split()[-**2**]) >= **40** and \  
 int(line.split()[-**3**]) >= **40**:  
 points = (int(line.split()[-**1**]) +  
 int(line.split()[-**2**]) +  
 int(line.split()[-**3**])  
 )  
 myList.append(points)  
myList.sort(reverse=True)  
if myList and len(myList) > k:  
 while len(myList) > k and myList[**0**] != myList[-**1**]:  
 myList = myList[:myList.index(min(myList))]  
 if len(myList) > k and myList[**0**] == myList[-**1**]:  
 print(**1,** file=outFile)  
 else:  
 print(min(myList)**,** file=outFile)  
else:  
 print(**0,** file=outFile)  
inFile.close()  
outFile.close()

15. В олимпиаде по информатике принимало участие N человек. Определите школы, из которых в олимпиаде принимало участие больше всего участников. В этой задаче необходимо считывать данные **построчно**, не сохраняя в памяти данные обо всех участниках, а только подсчитывая число участников для каждой школы.

**Формат ввода**

Информация о результатах олимпиады записана в файле, каждая из строк которого имеет вид:

фамилия имя школа балл

Фамилия и имя — текстовые строки, не содержащие пробелов. Школа — целое число от 1 до 99. Балл — целое число от 0 до 100.

**Формат вывода**

Выведите номера этих школ в порядке возрастания.

inFile = open('input.txt'**,** 'r'**,** encoding='utf8')  
scoolNum = []  
for line in inFile:  
 scoolNum.append(int(line.split()[-**2**]))  
scoolVol = [**0**] \* **100**for num in scoolNum:  
 scoolVol[num] += **1**for i in range(scoolVol.count(max(scoolVol))):  
 b = scoolVol.index(max(scoolVol))  
 print(b**,** end=' ')  
 scoolVol[b] = **0**

16. Зачет проводится отдельно в каждом классе. Победителями олимпиады становятся школьники, которые набрали наибольший балл среди всех участников в данном классе.

Для каждого класса определите максимальный балл, который набрал школьник, не ставший победителем в данном классе.

**Формат вывода**

Выведите три целых числа.

17. После затянувшегося совещания директор фирмы решил заказать такси, чтобы развезти сотрудников по домам. Он заказал N машин — ровно столько, сколько у него сотрудников. Однако когда они подъехали, оказалось, что у каждого водителя такси свой тариф за 1 километр.

Директор знает, какому сотруднику сколько километров от работы до дома (к сожалению, все сотрудники живут в разных направлениях, поэтому нельзя отправить двух сотрудников на одной машине). Теперь директор хочет определить, сколько придется заплатить за перевозку всех сотрудников. Естественно, директор хочет заплатить как можно меньшую сумму.

**Формат ввода**

В первой строке записаны N чисел через пробел, задающих расстояния в километрах от работы до домов сотрудников компании. Во второй строке записаны N чисел — тарифы за проезд одного километра в такси.

**Формат вывода**

Выведите одно целое число — наименьшую сумму, которую придется заплатить за доставку всех сотрудников.

18. В Государственную Думу Федерального Собрания Российской Федерации выборы производятся по партийным спискам. Каждый избиратель указывает одну партию, за которую он отдает свой голос. В Государственную Думу попадают партии, которые набрали не менее 7% от числа голосов избирателей.

Дан список партий и список голосов избирателей. Выведите список партий, которые попадут в Государственную Думу.

**Формат ввода**

В первой строке входного файла написано слово PARTIES:. Далее идет список партий, участвующих в выборах.

Затем идет строка, содержащая слово VOTES:. За ним идут названия партий, за которые проголосовали избиратели, по одному названию в строке. Названия могут быть только строками из первого списка.

**Формат вывода**

Программа должна вывести названия партий, получивших не менее 7% от числа голосов в том порядке, в котором они следуют в первом списке.

19. Формат входных данных аналогичен предыдущей задаче. Выведите список всех партий, участвовавших в выборах, отсортировав его в порядке убывания количества голосов избирателей, а при равном количестве голосов - в лексикографическом порядке.

### Множества и словари

1. Дан список чисел, который может содержать до 100000 чисел.Определите, сколько в нем встречается различных чисел.

**Формат ввода**

Вводится список целых чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

myList = list(map(int**,** input().split()))  
mySet = set(myList)  
print(len(mySet))

print(len(set(list(map(int**,** input().split())))))

2. Даны два списка чисел, которые могут содержать до 100000 чисел каждый. Посчитайте, сколько чисел содержится одновременно как в первом списке, так и во втором.

**Примечание**. Эту задачу на Питоне можно решить в одну строчку.

**Формат ввода**

Вводятся два списка чисел. Все числа каждого списка находятся на отдельной строке.

**Формат вывода**

Выведите ответ на задачу.

print(len(set(map(int**,** input().split())) & set(map(int**,** input().split()))))

3. Даны два списка чисел, которые могут содержать до 10000 чисел каждый. Выведите все числа, которые входят как в первый, так и во второй список, в порядке возрастания.

**Примечание**

Эту задачу на Питоне можно решить в одну строчку.

**Формат ввода**

Вводятся два списка целых чисел. Все числа каждого списка находятся на отдельной строке.

**Формат вывода**

Выведите ответ на задачу.

print(\*sorted(map(int**,** set(input().split()) & set(input().split()))))

4. Во входной строке записана последовательность чисел через пробел. Для каждого числа выведите слово YES (в отдельной строке), если это число ранее встречалось в последовательности или NO, если не встречалось.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

myList = set()  
for elm in list(map(int**,** input().split())):  
 if elm in myList:  
 print('YES')  
 myList.add(elm)  
 else:  
 print('NO')  
 myList.add(elm)

если обрабатывать список а не множество-по времени не проходит

5. Аня и Боря любят играть в разноцветные кубики, причем у каждого из них свой набор и в каждом наборе все кубики различны по цвету. Однажды дети заинтересовались, сколько существуют цветов таких, что кубики каждого цвета присутствуют в обоих наборах. Для этого они занумеровали все цвета случайными числами. На этом их энтузиазм иссяк, поэтому вам предлагается помочь им в оставшейся части. Номер любого цвета — это целое число в пределах от 0 до 10⁹.

**Формат ввода**

В первой строке входного файла записаны числа N и M — количество кубиков у Ани и Бори соответственно. В следующих N строках заданы номера цветов кубиков Ани. В последних M строках номера цветов кубиков Бори.

**Формат вывода**

Выведите сначала количество, а затем отсортированные по возрастанию номера цветов таких, что кубики каждого цвета есть в обоих наборах, затем количество и отсортированные по возрастанию номера остальных цветов у Ани, потом количество и отсортированные по возрастанию номера остальных цветов у Бори.

**Примечание**

Для ускорения ввода попробуйте использовать чтение входных данных из файла.

6. Во входном файле (вы можете читать данные из sys.stdin, подключив библиотеку sys) записан текст. Словом считается последовательность непробельных символов идущих подряд, слова разделены одним или большим числом пробелов или символами конца строки. Определите, сколько различных слов содержится в этом тексте.

**Формат ввода**

Вводится текст.

**Формат вывода**

Выведите ответ на задачу.

inFile = open('input.txt'**,** 'r'**,** encoding='utf8')  
mySet = []  
b = []  
for line in inFile:  
 b = line.split()  
 for i in range(len(b)):  
 mySet.append(b[i])  
print(len(set(mySet)))

7. Август и Беатриса играют в игру. Август загадал натуральное число от 1 до n. Беатриса пытается угадать это число, для этого она называет некоторые множества натуральных чисел. Август отвечает Беатрисе YES, если среди названных ею чисел есть задуманное или NO в противном случае. После нескольких заданных вопросов Беатриса запуталась в том, какие вопросы она задавала и какие ответы получила и просит вас помочь ей определить, какие числа мог задумать Август.

**Формат ввода**

Первая строка входных данных содержит число n — наибольшее число, которое мог загадать Август. Далее идут строки, содержащие вопросы Беатрисы. Каждая строка представляет собой набор чисел, разделенных пробелами. После каждой строки с вопросом идет ответ Августа: YES или NO. Наконец, последняя строка входных данных содержит одно слово HELP.

**Формат вывода**

Вы должны вывести (через пробел, в порядке возрастания) все числа, которые мог задумать Август.

inFile = open('input.txt'**,** 'r'**,** encoding='utf8')  
c = []  
n = int(inFile.readline())  
b = set(range(**1,** n + **1**))  
for line in inFile:  
 c.append(line[:-**1**])  
for i in range(len(c)):  
 if c[i] == 'YES':  
 b &= set(map(int**,** c[i - **1**].split()))  
 elif c[i] == 'NO':  
 b -= set(map(int**,** c[i - **1**].split()))  
 elif c[i] == 'HELP':  
 break  
myList = sorted(list(b))  
print(\*myList)

8. Август и Беатриса продолжают играть в игру, но Август начал жульничать. На каждый из вопросов Беатрисы он выбирает такой вариант ответа YES или NO, чтобы множество возможных задуманных чисел оставалось как можно больше. Например, если Август задумал число от 1 до 5, а Беатриса спросила про числа 1 и 2, то Август ответит NO, а если Беатриса спросит про 1, 2, 3, то Август ответит YES. Если же Бетриса в своем вопросе перечисляет ровно половину из задуманных чисел, то Август из вредности всегда отвечает NO. Наконец, Август при ответе учитывает все предыдущие вопросы Беатрисы и свои ответы на них, то есть множество возможных задуманных чисел уменьшается.

**Формат ввода**

Вам дана последовательность вопросов Беатрисы. Приведите ответы Августа на них. Первая строка входных данных содержит число n — наибольшее число, которое мог загадать Август. Далее идут строки, содержащие вопросы Беатрисы. Каждая строка представляет собой набор чисел, разделенных пробелами. Последняя строка входных данных содержит одно слово HELP.

**Формат вывода**

Для каждого вопроса Беатрисы выведите ответ Августа на этот вопрос. После этого выведите (через пробел, в порядке возрастания) все числа, которые мог загадать Август после ответа на всевопросы Беатрисы.

9. Каждый из N школьников некоторой школы знает Mᵢ языков. Определите, какие языки знают все школьники и языки, которые знает хотя бы один из школьников.

**Формат ввода**

Первая строка входных данных содержит количество школьников N. Далее идет N чисел Mᵢ, после каждого из чисел идет Mᵢ строк, содержащих названия языков, которые знает i-й школьник. Длина названий языков не превышает 1000 символов, количество различных языков не более 1000. 1≤N≤1000, 1≤Mᵢ≤500.

**Формат вывода**

В первой строке выведите количество языков, которые знают все школьники. Начиная со второй строки - список таких языков. Затем - количество языков, которые знает хотя бы один школьник, на следующих строках - список таких языков.

sumLang = []  
minLang = []  
volScool = int(input())  
for i in range(volScool):  
 m = int(input())  
 for j in range(m):  
 minLang.append(input())  
 sumLang.append(set(minLang))  
 minLang.clear()  
itog1 = set(sumLang[**0**])  
itog2 = itog1.copy()  
for lang in sumLang:  
 itog1 |= set(lang)  
 itog2 &= set(lang)  
print(len(itog2))  
for n in itog2:  
 print(n)  
print(len(itog1))  
for k in itog1:  
 print(k)

10. На Новом проспекте для разгрузки было решено пустить два новых автобусных маршрута на разных участках проспекта. Известны конечные остановки каждого из автобусов. Определите количество остановок, на которых можно пересесть с одного автобуса на другой.

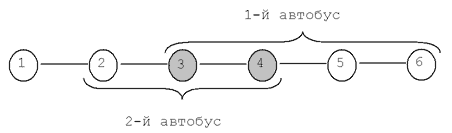
**Формат ввода**

Вводятся четыре числа, не превосходящие 100, задающие номера конечных остановок. Сначала для первого, потом второго автобуса (см. примеры и рисунок).

**Формат вывода**

Ваша программа должна выводить одно число – искомое количество остановок.

**Примечания**



Пояснения Первый пример (см. рисунок): первый автобус ходит с 3-й остановки по 6-ю и обратно, а второй с 2-й по 4-ю и обратно. Пересесть с одного автобуса на другой можно на 3-й и 4-й остановках. Их две. Второй пример: автобусы не имеют общих остановок.

11. Политическая жизнь одной страны очень оживленная. В стране действует K политических партий, каждая из которых регулярно объявляет национальную забастовку. Дни, когда хотя бы одна из партий объявляет забастовку, при условии, что это не суббота или воскресенье (когда и так никто не работает), наносят большой ущерб экономике страны. i-я партия объявляет забастовки строго каждые bᵢ дней, начиная с дня с номером aᵢ. То есть i-я партия объявляет забастовки в дни aᵢ, aᵢ+bᵢ, aᵢ+2bᵢ и т.д. Если в какой-то день несколько партий объявляет забастовку, то это считается одной общенациональной забастовкой. В календаре страны N дней, пронумерованных от 1 до N. Первый день года является понедельником, шестой и седьмой дни года — выходные, неделя состоит из семи дней.

**Формат ввода**

Программа получает на вход число дней в году N (1≤N≤10⁶) и число политических партий K (1≤K≤100). Далее идет K строк, описывающие графики проведения забастовок.i-я строка содержит числа aᵢ и bᵢ (1≤aᵢ,bᵢ≤N).

**Формат вывода**

Выведите единственное число: количество забастовок, произошедших в течение года.

**Примечания**

Первая партия объявляет забастовки в дни 2, 5, 8, 11, 14, 17. Вторая партия объявляет забастовки в дни 3, 8, 13, 18. Третья партия — в дни 9 и 17. Дни номер 6, 7, 13, 14 являются выходными. Таким образом, забастовки пройдут в дни 2, 3, 5, 8, 9, 11, 17, 18.

12. Телефонные номера в адресной книге мобильного телефона имеют один из следующих форматов:

* +7<код><номер>
* 8<код><номер>
* <номер>

где <номер> — это семь цифр, а <код> — это три цифры или три цифры в круглых скобках. Если код не указан, то считается, что он равен 495. Кроме того, в записи телефонного номера может стоять знак “-” между любыми двумя цифрами (см. пример). На данный момент в адресной книге телефона Васи записано всего три телефонных номера, и он хочет записать туда еще один. Но он не может понять, не записан ли уже такой номер в телефонной книге. Помогите ему! Два телефонных номера совпадают, если у них равны коды и равны номера. Например, +7(916)0123456 и 89160123456 — это один и тот же номер.

**Формат ввода**

В первой строке входных данных записан номер телефона, который Вася хочет добавить в адресную книгу своего телефона. В следующих трех строках записаны три номера телефонов, которые уже находятся в адресной книге телефона Васи. Гарантируется, что каждая из записей соответствует одному и13.з трех приведенных в условии форматов.

**Формат вывода**

Для каждого телефонного номера в адресной книге выведите YES, если он совпадает с тем телефонным номером,который Вася хочет добавить в адресную книгу или NO в противном случае.

13. Дан список стран и городов каждой страны. Затем даны названия городов. Для каждого города укажите, в какой стране он находится.

**Формат ввода**

Программа получает на вход количество стран N. Далее идет N строк, каждая строка начинается с названия страны, затем идут названия городов этой страны. Название каждого город состоит из одного слова. В следующей строке записано число M, далее идут M запросов — названия каких-то M городов, перечисленных выше.

**Формат вывода**

Для каждого из запроса выведите название страны, в котором находится данный город.

14. Во входном файле (вы можете читать данные из файла input.txt) записан текст. Словом считается последовательность непробельных подряд идущих символов. Слова разделены одним или большим числом пробелов или символами конца строки. Для каждого слова из этого текста подсчитайте, сколько раз оно встречалось в этом тексте ранее.

**Формат ввода**

Вводится текст.

**Формат вывода**

Выведите ответ на задачу.

inFile = open('input.txt'**,** 'r'**,** encoding='utf8')  
countDict = {}  
for line in inFile:  
 for word in line.split():  
 a = countDict.get(word**, 0**)  
 print(a**,** end=' ')  
 countDict[word] = a + **1**

15. Вам дан словарь, состоящий из пар слов. Каждое слово является синонимом к парному ему слову. Все слова в словаре различны. Для одного данного слова определите его синоним.

**Формат ввода**

Программа получает на вход количество пар синонимов N. Далее следует N строк, каждая строка содержит ровно два слова-синонима. После этого следует одно слово.

**Формат вывода**

Программа должна вывести синоним к данному слову.

**Примечания**

Эту задачу можно решить и без словарей (сохранив все входные данные в списке), но решение со словарем будет более простым.

n = int(input())  
sinDict = {}  
for i in range(n):  
 key**,** name = input().split()  
 sinDict[key] = name  
word = input()  
for sinon in sinDict:  
 if sinon == word:  
 print(sinDict[sinon])  
 elif sinDict[sinon] == word:  
 print(sinon)

16. Как известно, в США президент выбирается не прямым голосованием, а путем двухуровневого голосования. Сначала проводятся выборы в каждом штате и определяется победитель выборов в данном штате. Затем проводятся государственные выборы: на этих выборах каждый штат имеет определенное число голосов — число выборщиков от этого штата. На практике, все выборщики от штата голосуют в соответствии с результатами голосования внутри штата, то есть на заключительной стадии выборов в голосовании участвуют штаты, имеющие различное число голосов. Вам известно за кого проголосовал каждый штат и сколько голосов было отдано данным штатом. Подведите итоги выборов: для каждого из участника голосования определите число отданных за него голосов.

**Формат ввода**

Каждая строка входного файла содержит фамилию кандидата, за которого отдают голоса выборщики этого штата, затем через пробел идет количество выборщиков,отдавших голоса за этого кандидата.

**Формат вывода**

Выведите фамилии всех кандидатов в лексикографическом порядке,затем, через пробел, количество отданных за них голосов.

17. Дан текст. Выведите слово, которое в этом тексте встречается чаще всего. Если таких слов несколько, выведите то, которое меньше в лексикографическом порядке.

**Формат ввода**

Вводится текст.

**Формат вывода**

Выведите ответ на задачу.

import sys  
  
t = sys.stdin.read()  
tT = tuple(sorted([a for b in t.split('\n') for a in b.split()]**,** reverse=True))  
maxCount = **0**oftenWord = str()  
for word in tT:  
 if tT.count(word) >= maxCount:  
 maxCount = tT.count(word)  
 oftenWord = word  
print(oftenWord)

import sys  
  
text = sys.stdin.read()  
txList = sorted([a for b in text.split('\n') for a in b.split()]**,** reverse=True)  
wordDict = {}  
for word in txList:  
 wordDict[txList.count(word)] = word  
print(wordDict[len(wordDict)])

infile = open('input.txt'**,** encoding='utf8')  
txList = []  
for line in infile:  
 txList.extend(line.split())  
wordDict = {}  
for word in sorted(txList**,** reverse=True):  
 wordDict[txList.count(word)] = word  
print(wordDict[len(wordDict)])

все дают рантайм еррор

\*\*

infile = open('input.txt'**,** encoding='utf8')  
txList = []  
for line in infile:  
 txList.extend(line.split())  
wordDict = {}  
for word in txList:  
 wordDict[word] = wordDict.get(word**, 0**) + **1**a = max(sorted([i for i**,** k in wordDict.items()])**,** key=lambda x: wordDict[x])  
print(a)

18. Дан текст. Выведите все слова, встречающиеся в тексте, по одному на каждую строку. Слова должны быть отсортированы по убыванию их количества появления в тексте, а при одинаковой частоте появления — в лексикографическом порядке.

**Указание.**

После того, как вы создадите словарь всех слов, вам захочется отсортировать его по частоте встречаемости слова. Желаемого можно добиться, если создать список, элементами которого будут кортежи из двух элементов: частота встречаемости слова и само слово. Например, [(2, 'hi'), (1, 'what'), (3, 'is')]. Тогда стандартная сортировка будет сортировать список кортежей, при этом кортежи сравниваются по первому элементу, а если они равны —то по второму. Это почти то, что требуется в задаче.

**Формат ввода**

Вводится текст.

**Формат вывода**

Выведите ответ на задачу.

import sys  
  
text = sys.stdin.read()  
txList = [a for b in text.split('\n') for a in b.split()]  
wordDict = {}  
newList = []  
for word in txList:  
 wordDict[word] = wordDict.get(word**, 0**) + **1**for key**,** item in wordDict.items():  
 newList.append([item**,** key])  
wordDict.clear()  
for i in range(len(newList)):  
 if newList[i][**0**] not in wordDict:  
 wordDict[newList[i][**0**]] = []  
 wordDict[newList[i][**0**]].append(newList[i][**1**])  
for word in sorted(wordDict**,** reverse=True):  
 print(\*sorted(wordDict[word])**,** sep='\n')

19. В выборах Президента Российской Федерации побеждает кандидат, набравший свыше половины числа голосов избирателей. Если такого кандидата нет, то во второй тур выборов выходят два кандидата, набравших наибольшее число голосов.

**Формат ввода**

Каждая строка входного файла содержит имя кандидата, за которого отдал голос один избиратель. Известно, что общее число кандидатов не превосходит 20, но в отличии от предыдущих задач список кандидатов явно не задан. Читайте данные из файла input.txt с указанием кодировки utf8.

**Формат вывода**

Если есть кандидат, набравший более 50% голосов, программа должна вывести его имя. Если такого кандидата нет, программа должна вывести имя кандидата, занявшего первое место, затем имя кандидата, занявшего второе место. Выводите данные в файл output.txt с указанием кодировки utf8.

inFile = open('input.txt'**,** encoding='utf8')  
outFile = open('output.txt'**,** 'w'**,** encoding='utf8')  
voteList = newList = []  
presDict = {}  
count = **0**for line in inFile:  
 presDict[line.strip()] = presDict.get(line.strip()**, 0**) + **1** count += **1**for key**,** item in presDict.items():  
 voteList.append([item**,** key])  
newList = sorted(voteList**,** reverse=True)  
print(newList[**0**][**1**]**,** file=outFile)  
if newList[**0**][**0**] <= count / **2**:  
 print(newList[**1**][**1**]**,** file=outFile)

для себя-сортировка словаря по значению по убыванию(по возрастанию (х[1]): res = sorted(nowDict.items(), key=lambda x: (-x[1]))

20. Статья 83 закона “О выборах депутатов Государственной Думы Федерального Собрания Российской Федерации” определяет следующий алгоритм пропорционального распределения мест в парламенте.

Необходимо распределить 450 мест между партиями, участвовавших в выборах. Сначала подсчитывается сумма голосов избирателей, поданных за каждую партию и подсчитывается сумма голосов, поданных за все партии. Эта сумма делится на 450, получается величина, называемая “первое избирательное частное” (смысл первого избирательного частного - это количество голосов избирателей, которое необходимо набрать для получения одного места в парламенте). Далее каждая партия получает столько мест в парламенте, чему равна целая часть от деления числа голосов за данную партию на первое избирательное частное.Если после первого раунда распределения мест сумма количества мест, отданных партиям, меньше 450, то оставшиеся места передаются по одному партиям, в порядке убывания дробной части частного от деления числа голосов за данную партию на первое избирательное частное. Если же для двух партий эти дробные части равны, то преимущество отдается той партии, которая получила большее число голосов.

**Формат ввода**

На вход программе подается список партий, участвовавших в выборах. Каждая строка входного файла содержит название партии (строка, возможно, содержащая пробелы), затем, через пробел, количество голосов, полученных данной партией – число, не превосходящее 10⁸.

**Формат вывода**

Программа должна вывести названия всех партий и количество голосов в парламенте, полученных данной партией. Названия необходимо выводить в том же порядке, в котором они шли во входных данных.

21. Дана база данных о продажах некоторого интернет-магазина. Каждая строка входного файла представляет собой запись вида

Покупатель товар количество, где

Покупатель — имя покупателя (строка без пробелов),

товар — название товара (строка без пробелов),

количество — количество приобретенных единиц товара.

Создайте список всех покупателей, а для каждого покупателя подсчитайте количество приобретенных им единиц каждого вида товаров.

**Формат ввода**

Вводятся сведения о покупках в указанном формате.

**Формат вывода**

Выведите список всех покупателей в лексикографическом порядке,после имени каждого покупателя выведите двоеточие, затем выведите список названий всех приобретенных данным покупателем товаров в лексикографическом порядке, после названия каждого товара выведите количество единиц товара, приобретенных данным покупателем.Информация о каждом товаре выводится в отдельной строке.

inFile = open('input.txt'**,** encoding='utf8')  
preDict = {}  
count = **0**for line in inFile:  
 name**,** prod**,** vol = line.split()  
 if name not in preDict:  
 preDict[name] = {}  
 if prod not in preDict[name]:  
 preDict[name][prod] = **0** preDict[name][prod] += int(vol)  
for client in sorted(preDict):  
 print(client + ':')  
 for product in sorted(preDict[client]):  
 print(product**,** preDict[client][product])

22. Некоторый банк хочет внедрить систему управления счетами клиентов, поддерживающую следующие операции:

Пополнение счета клиента.

Снятие денег со счета.

Запрос остатка средств на счете.

Перевод денег между счетами клиентов.

Начисление процентов всем клиентам.

Вам необходимо реализовать такую систему. Клиенты банка идентифицируются именами (уникальная строка, не содержащая пробелов). Первоначально у банка нет ни одного клиента. Как только для клиента проводится операция пополнения, снятия или перевода денег, ему заводится счет с нулевым балансом. Все дальнейшие операции проводятся только с этим счетом. Сумма на счету может быть как положительной, так и отрицательной, при этом всегда является целым числом.

**Формат ввода**

Входной файл содержит последовательность операций. Возможны следующие операции:

DEPOSIT name sum - зачислить сумму sum на счет клиента name. Если у клиента нет счета, то счет создается.

WITHDRAW name sum - снять сумму sum со счета клиента name. Если у клиента нет счета, то счет создается.

BALANCE name - узнать остаток средств на счету клиента name.

TRANSFER name1 name2 sum - перевести сумму sum со счета клиента name1 на счет клиента name2. Если у какого-либо клиента нет счета, то ему создается счет.

INCOME p - начислить всем клиентам, у которых открыты счета, p% от суммы счета. Проценты начисляются только клиентам с положительным остатком на счету, если у клиента остаток отрицательный, то его счет не меняется. После начисления процентов сумма на счету остается целой, то есть начисляется только целое число денежных единиц. Дробная часть начисленных процентов отбрасывается.

**Формат вывода**

Для каждого запроса BALANCE программа должна вывести остаток на счету данного клиента. Если же у клиента с запрашиваемым именем не открыт счет в банке, выведитеERROR.

23. Учительница задала Пете домашнее задание — в заданном тексте расставить ударения в словах, после чего поручила Васе проверить это домашнее задание. Вася очень плохо знаком с данной темой,поэтому он нашел словарь, в котором указано, как ставятся ударения в словах. К сожалению, в этом словаре присутствуют не все слова. Вася решил, что в словах, которых нет в словаре, он будет считать, что Петя поставил ударения правильно, если в этом слове Петей поставлено ровно одно ударение.

Оказалось, что в некоторых словах ударение может быть поставлено больше, чем одним способом. Вася решил, что в этом случае если то, как Петя поставил ударение, соответствует одному из приведенных в словаре вариантов, он будет засчитывать это как правильную расстановку ударения, а если не соответствует, то как ошибку. Вам дан словарь, которым пользовался Вася и домашнее задание, сданное Петей. Ваша задача — определить количество ошибок,которое в этом задании насчитает Вася.

**Формат ввода**

Вводится сначала число N — количество слов в словаре (0≤N≤20000). Далее идет N строк со словами из словаря. Каждое слово состоит не более чем из 30 символов. Все слова состоят из маленьких и заглавных латинских букв. В каждом слове заглавная ровно одна буква — та, на которую попадает ударение. Слова в словаре расположены в алфавитном порядке. Если есть несколько возможностей расстановки ударения в одном и том же слове, то эти варианты в словаре идут в произвольном порядке.

Далее идет упражнение, выполненное Петей. Упражнение представляет собой строку текста, суммарным объемом не более 300000 символов. Строка состоит из слов, которые разделяются между собой ровно одним пробелом. Длина каждого слова не превышает 30 символов. Все слова состоят из маленьких и заглавных латинских букв (заглавными обозначены те буквы, над которыми Петя поставил ударение). Петя мог по ошибке в каком-то слове поставить более одного ударения или не поставить ударения вовсе.

**Формат вывода**

Выведите количество ошибок в Петином тексте, которые найдет Вася.

**Примечания**

Примечания к примерам тестов

1. В слове cannot, согласно словарю возможно два варианта расстановки ударения. Эти варианты в словаре могут быть перечислены в любом порядке (т.е. как сначала cAnnot, а потом cannOt, так и наоборот). Две ошибки, совершенные Петей — это слова be (ударение вообще не поставлено) и fouNd (ударение поставлено неверно). Слово thE отсутствует в словаре, но поскольку в нем Петя поставил ровно одно ударение, признается верным.

2. Неверно расставлены ударения во всех словах, кроме The (оно отсутствует в словаре, в нем поставлено ровно одно ударение). В остальных словах либо ударные все буквы (в слове PAGE), либо не поставлено ни одного ударения.

24. В генеалогическом древе у каждого человека, кроме родоначальника, есть ровно один родитель. Каждом элементу дерева сопоставляется целое неотрицательное число, называемое высотой. У родоначальника высота равна 0, у любого другого элемента высота на 1 больше, чем у его родителя.Вам дано генеалогическое древо, определите высоту всех его элементов.

**Формат ввода**

Программа получает на вход число элементов в генеалогическом древе N. Далее следует N-1 строка, задающие родителя для каждого элемента древа, кроме родоначальника.Каждая строка имеет вид имя\_потомка имя\_родителя.

**Формат вывода**

Программа должна вывести список всех элементов древа в лексикографическом порядке.После вывода имени каждого элемента необходимо вывести его высоту.

Эта задача имеет решение сложности O(n), но вам достаточнонаписать решение сложности O(n²) (не считая сложности обращенияк элементам словаря).Пример ниже соответствует приведенному древу рода Романовых

### Функциональное программирование

1. Дан список чисел, который может содержать до 100000 чисел.Определите, сколько в нем встречается различных чисел.

**Формат ввода**

Вводится список целых чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

print(  
 len(  
 set(  
 map(  
 int**,** input().split()  
 )  
 )  
 )  
)

2. Во входном файле (вы можете читать данные из sys.stdin, подключив библиотеку sys) записан текст. Словом считается последовательность непробельных символов идущих подряд, слова разделены одним или большим числом пробелов или символами конца строки. Определите, сколько различных слов содержится в этом тексте.

**Формат ввода**

Вводится текст.

**Формат вывода**

Выведите ответ на задачу.

from sys import stdin  
  
print(  
 len(  
 set(  
 stdin.read().strip().split()  
 )  
 )  
)

3. Выведите значение наименьшего нечетного элемента списка, гарантируется, что хотя бы один нечётный элемент в списке есть.

**Формат ввода**

Вводится список чисел. Все числа списка находятся на одной строке.

**Формат вывода**

Выведите ответ на задачу.

print(  
 min(  
 filter(  
 lambda x: x % **2** == **1,** map(  
 int**,** input().split()  
 )  
 )  
 )  
)

4. Проверьте, есть ли среди данных N чисел нули.

**Формат ввода**

Вводится число N, а затем N чисел.

**Формат вывода**

Выведите True, если среди введенных чисел есть хотя бы один нуль, или False в противном случае.

print(  
 any(  
 map(  
 lambda x: x == **0,** map(  
 lambda x: int(  
 input())**,** range(  
 int(  
 input()  
 )  
 )  
 )  
 )  
 )  
)

5. На вход подаётся последовательность натуральных чисел длины n≤1000.Посчитайте произведение пятых степеней чисел в последовательности.

**Формат ввода**

Вводится последовательность чисел

**Формат вывода**

Выведите ответ на задачу

**Примечания**

Для решения задачи используйте функцию reduce из модуля functools

from functools import reduce  
  
print(  
 (  
 reduce(  
 lambda x**,** y: (x \* y)**,** map(  
 int**,** input().split()  
 )  
 )  
 )  
 \*\* **5**)

6. Булева функция XOR (сложение по модулю два) задаётся следующей таблицей истинности:

xor(0,0)=0

xor(0,1)=1

xor(1,0)=1

xor(1,1)=0

На вход подаются две последовательности (a₁,…,an) и (b₁,…,bn) из 0 и 1.

Вычислите последовательность из (c₁,…,cn), где каждая cᵢ=xor(aᵢ,bᵢ).

**Формат ввода**

На вход подаются две строки из 0 и 1, разделённых пробелами.

Первая строка — это последовательность (a₁,…,an).

Вторая строка — это последовательность (b₁,…,bn).

**Формат вывода**

Выведите последовательность (c₁,…,cn), разделяя каждый элемент пробелом

**Примечания**

Для решения задачи можете использовать функцию zip.

print(  
 \*map(  
 lambda x: x[**0**] ^ x[**1**]**,** zip(  
 map(  
 int**,** input().split()  
 )**,** map(  
 int**,** input().split()  
 )  
 )  
 )  
)

7. По заданной последовательности:

(a₁,…,an)

посчитайте последовательность частичных сумм:

(S₁,…,Sn),

где Sk=a₁+a₂+…+ak.

**Формат ввода**

Вводится последовательность чисел (a₁,…,an), разделённая пробелами.

**Формат вывода**

Выведите последовательность (S₁,…,Sn), разделяя числа пробелами.

**Примечания**

Для решения задачи можно воспользоваться функцией accumulate из модуля itertools.

8. По заданному на входе числу 0≤n≤2000 выведите последовательность факториалов:

0!,1!,2!,…,n!

**Формат ввода**

Вводится число n.

**Формат вывода**

Выведите последовательность факториалов, разделяя числа пробелами

9. По данному числу N выведите все перестановки чисел от 1 до N в лексикографическом порядке.

**Формат ввода**

Задано 1 число: N (0<N<10).

**Формат вывода**

Необходимо вывести все перестановки чисел от 1 до N в лексикографическом порядке. Перестановки выводятся по одной в строке, числа в перестановке выводятся без пробелов.

10. XOR для произвольного числа аргументов определяется следующим образом:

xor(a₁,a₂,…,an)= xor(a₁, xor(a₂, xor(a₃,… xor(an))…)

XOR от n последовательностей A₁,…,An (Aᵢ=Aᵢ₁,…,Aᵢk) равных длин k — это последовательность C=xor(A₁,…,An),такая, что:

cᵢ=xor(A₁ᵢ,…Anᵢ)

Посчитайте XOR от n последовательностей равной длины k.

**Формат ввода**

На первой строке записано число 2≤n≤1000 — количество последовательностей.

На последующих n строках записаны последовательности A₁,…,An из 0 и 1, разделённых пробелами равной длины 1≤k≤1000.

**Формат вывода**

Выведите последовательность C=xor(A₁,…,An), разделяя числа последовательности пробелами.

11. Выведите все простые на отрезке [2;n].

**Формат ввода**

Вводится число 2≤n≤100000.

**Формат вывода**

Выведите все простые числа из отрезка [2,n] в порядке возрастания

**Примечания**

Напомним, что проверить число на то, простое ли оно можно за количество операций порядка √(N). Также напомним, что функция math.sqrt работает значительно быстрее, чем (x \*\* 1/2).

12. Перед началом тараканьих бегов всем болельщикам было предложено сделать по две ставки на результаты бегов. Каждая ставка имеет вид "Таракан №A придет раньше, чем таракан №B". Организаторы бегов решили выяснить, могут ли тараканы прийти в таком порядке, чтобы у каждого болельщика сыграла ровно одна ставка из двух (то есть чтобы ровно одно из двух утверждений каждого болельщика оказалось верным). Считается, что никакие два таракана не могут прийти к финишу одновременно.

**Формат ввода**

В первой строке входных данных содержатся два разделенных пробелом натуральных числа: число K, не превосходящее 10, - количество тараканов и число N, не превосходящее 100, - количество болельщиков. Все тараканы пронумерованы числами от 1 до K. Каждая из следующих N строк содержит 4 натуральных числа A, B, C, D, не превосходящих K, разделенных пробелами. Они соответствуют ставкам болельщика "Таракан №A придет раньше, чем таракан №B" и "Таракан №C придет раньше, чем таракан №D".

**Формат вывода**

Если завершить бега так, чтобы у каждого из болельщиков сыграла ровно одна из двух ставок, можно, то следует вывести номера тараканов в том порядке, в котором они окажутся в итоговой таблице результатов (сначала номер таракана, пришедшего первым, затем номер таракана, пришедшего вторым и т. д.) в одну строку через пробел. Если таких вариантов несколько, выведите любой из них. Если требуемого результата добиться нельзя, выведите одно число 0.

### Классы

1. Реализуйте класс *Matrix*. Он должен содержать:

* Конструктор от списка списков. Гарантируется, что списки состоят из чисел, не пусты и все имеют одинаковый размер. Конструктор должен **копировать** содержимое списка списков, т. е. при изменении списков, от которых была сконструирована матрица, содержимое матрицы изменяться не должно.
* Метод *\_\_str\_\_*, переводящий матрицу в строку. При этом элементы внутри одной строки должны быть разделены знаками табуляции, а строки — переносами строк. После каждой строки не должно быть символа табуляции и в конце не должно быть переноса строки.
* Метод *size* без аргументов, возвращающий кортеж вида *(число строк, число столбцов)*. Пример теста с участием этого метода есть в следующей задаче этой недели.

**Как сдавать задачи этой недели?**

На проверку вы должны сдать только файл, содержащий описание класса и одну строку вне класса (в качестве основной программы):



1

exec(stdin.read())

И еще одну строку в начале файла:



1

from sys import stdin

Для тестирования класса вы можете вместо строки *exec(stdin.read())* вставлять код из примеров или писать свой код.

**Формат ввода**

Вводится исходный код тестирующей программы на языке Python.

**Формат вывода**

Выведите результат её работы в текущем окружении при помощи *exec*, как это указано в условии.

**Примеры**

Тест 1

Входные данные:

# Task 1 check 1

m = Matrix([[1, 0], [0, 1]])

print(m)

m = Matrix([[2, 0, 0], [0, 1, 10000]])

print(m)

m = Matrix([[-10, 20, 50, 2443], [-5235, 12, 4324, 4234]])

print(m)

Вывод программы:

1 0

0 1

2 0 0

0 1 10000

-10 20 50 2443

-5235 12 4324 4234

Тест 2

Входные данные:

# Task 1 check 2

m1 = Matrix([[1, 0, 0], [1, 1, 1], [0, 0, 0]])

m2 = Matrix([[1, 0, 0], [1, 1, 1], [0, 0, 0]])

print(str(m1) == str(m2))

Вывод программы:

True

Тест 3

Входные данные:

# Task 1 check 3

m = Matrix([[1, 1, 1], [0, 100, 10]])

print(str(m) == '1\t1\t1\n0\t100\t10')

Вывод программы:

True

from copy import deepcopy  
from sys import stdin  
  
  
class Matrix:  
 def \_\_init\_\_(self**,** matrix):  
 self.matrix = deepcopy(matrix)  
  
 def \_\_str\_\_(self):  
 return '\n'.join(  
 [''.join(['\t' f'{i}' for i in row]).lstrip('\t') for row in  
 self.matrix])  
  
 def size(self):  
  
 rows = len(self.matrix)  
 cols = **0** for row in self.matrix:  
 if len(row) > cols:  
 cols = len(row)  
 return (rows**,** cols)  
  
  
exec(stdin.read())

2. Добавьте в предыдущий класс следующие методы:

* *\_\_add\_\_*, принимающий вторую матрицу того же размера и возвращающий сумму матриц.
* *\_\_mul\_\_*, принимающий число типа *int* или *float* и возвращающий матрицу, умноженную на скаляр.
* *\_\_rmul\_\_*, делающий то же самое, что и *\_\_mul\_\_*. Этот метод будет вызван в том случае, аргумент находится справа. Для реализации этого метода в коде класса достаточно написать *\_\_rmul\_\_ = \_\_mul\_\_*.

**Иллюстрация:**

* В следующем случае вызовется *\_\_mul\_\_*: *Matrix([[0, 1], [1, 0]) \* 10*.
* В следующем случае вызовется *\_\_rmul\_\_* (так как у *int* не определен *\_\_mul\_\_* для матрицы справа): *10 \* Matrix([[0, 1], [1, 0])*.

Разумеется, данные методы не должны менять содержимое матрицы.

**Формат ввода**

Как в предыдущей задаче.

**Формат вывода**

Как в предыдущей задаче.

Тест 1

Входные данные:

# Task 2 check 1

m = Matrix([[10, 10], [0, 0], [1, 1]])

print(m.size())

Вывод программы:

(3, 2)

Тест 2

Входные данные:

# Task 2 check 2

m1 = Matrix([[1, 0, 0], [0, 1, 0], [0, 0, 1]])

m2 = Matrix([[0, 1, 0], [20, 0, -1], [-1, -2, 0]])

print(m1 + m2)

Вывод программы:

1 1 0

20 1 -1

-1 -2 1

Тест 3

Входные данные:

# Task 2 check 3

m = Matrix([[1, 1, 0], [0, 2, 10], [10, 15, 30]])

alpha = 15

print(m \* alpha)

print(alpha \* m)

Вывод программы:

15 15 0

0 30 150

150 225 450

15 15 0

0 30 150

150 225 450

from copy import deepcopy  
from sys import stdin  
  
  
class Matrix:  
 def \_\_init\_\_(self**,** matrix):  
 self.matrix = deepcopy(matrix)  
  
 def \_\_str\_\_(self):  
 return '\n'.join(  
 [''.join(['\t' f'{i}' for i in row]).lstrip('\t') for row in  
 self.matrix])  
  
 def size(self):  
  
 rows = len(self.matrix)  
 cols = **0** for row in self.matrix:  
 if len(row) > cols:  
 cols = len(row)  
 return (rows**,** cols)  
  
 def \_\_add\_\_(self**,** other):  
 result = Matrix  
 result.matrix = []  
 for i in range(len(self.matrix)):  
 temp = []  
 for j in range(len(self.matrix[i])):  
 temp.append(self.matrix[i][j] + other.matrix[i][j])  
 result.matrix.append(temp)  
 return Matrix(result.matrix)  
  
 def \_\_mul\_\_(self**,** other):  
 if isinstance(other**,** int) or isinstance(other**,** float):  
 result = Matrix  
 result.matrix = []  
 for i in range(len(self.matrix)):  
 temp = []  
 for j in range(len(self.matrix[i])):  
 temp.append(self.matrix[i][j] \* other)  
 result.matrix.append(temp)  
 return Matrix(result.matrix)  
  
 \_\_rmul\_\_ = \_\_mul\_\_  
  
  
exec(stdin.read())

3. Добавьте в программу из предыдущей задачи класс *MatrixError*, содержащий внутри *self* поля *matrix1* и *matrix2* — ссылки на матрицы.

В класс *Matrix* внесите следующие изменения:

* Добавьте в метод *\_\_add\_\_* проверку на ошибки в размере входных данных, чтобы при попытке сложить матрицы разных размеров было выброшено исключение *MatrixError* таким образом, чтобы *matrix1* поле *MatrixError* стало первым аргументом *\_\_add\_\_* (просто *self*), а *matrix2* — вторым (второй операнд для сложения).
* Реализуйте метод *transpose*, транспонирующий матрицу и возвращающую результат (данный метод модифицирует экземпляр класса *Matrix*)
* Реализуйте статический метод *transposed*, принимающий *Matrix* и возвращающий транспонированную матрицу. [Пример](https://ru.wikibooks.org/wiki/Python/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BD%D0%B0_Python#%D0%A1%D1%82%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4) статического метода.

**Формат ввода**

Как в предыдущей задаче.

**Формат вывода**

Тест 1

Входные данные:

# Task 3 check 1

# Check exception to add method

m1 = Matrix([[1, 0, 0], [0, 1, 0], [0, 0, 1]])

m2 = Matrix([[0, 1, 0], [20, 0, -1], [-1, -2, 0]])

print(m1 + m2)

m2 = Matrix([[0, 1, 0], [20, 0, -1]])

try:

m = m1 + m2

print('WA\n' + str(m))

except MatrixError as e:

print(e.matrix1)

print(e.matrix2)

Вывод программы:

1 1 0

20 1 -1

-1 -2 1

1 0 0

0 1 0

0 0 1

0 1 0

20 0 -1

Тест 2

Входные данные:

# Task 3 check 2

m = Matrix([[10, 10], [0, 0], [1, 1]])

print(m)

m1 = m.transpose()

print(m)

print(m1)

Вывод программы:

10 10

0 0

1 1

10 0 1

10 0 1

10 0 1

10 0 1

Тест 3

Входные данные:

# Task 3 check 3

m = Matrix([[10, 10], [0, 0], [1, 1]])

print(m)

print(Matrix.transposed(m))

print(m)

Вывод программы:

10 10

0 0

1 1

10 0 1

10 0 1

10 10

0 0

1 1

from copy import deepcopy  
from sys import stdin  
  
  
class MatrixError(BaseException):  
 def \_\_init\_\_(self**,** Matrix**,** other):  
 self.matrix1 = Matrix  
 self.matrix2 = other  
  
  
class Matrix:  
 def \_\_init\_\_(self**,** matrix):  
 self.matrix = deepcopy(matrix)  
  
 def \_\_str\_\_(self):  
 return '\n'.join(  
 [''.join(['\t' f'{i}' for i in row]).lstrip('\t') for row in  
 self.matrix])  
  
 def size(self):  
 rows = len(self.matrix)  
 cols = **0** for row in self.matrix:  
 if len(row) > cols:  
 cols = len(row)  
 return (rows**,** cols)  
  
 def \_\_add\_\_(self**,** other):  
 result = Matrix  
 result.matrix = []  
 if self.size() == other.size():  
 for i in range(len(self.matrix)):  
 temp = []  
 for j in range(len(self.matrix[i])):  
 temp.append(self.matrix[i][j] + other.matrix[i][j])  
 result.matrix.append(temp)  
 return Matrix(result.matrix)  
 else:  
 raise MatrixError(self**,** other)  
  
 def \_\_mul\_\_(self**,** other):  
 if isinstance(other**,** int) or isinstance(other**,** float):  
 result = Matrix  
 result.matrix = []  
 for i in range(len(self.matrix)):  
 temp = []  
 for j in range(len(self.matrix[i])):  
 temp.append(self.matrix[i][j] \* other)  
 result.matrix.append(temp)  
 return Matrix(result.matrix)  
  
 \_\_rmul\_\_ = \_\_mul\_\_  
  
 def transpose(self):  
 self.matrix = list(zip(\*self.matrix))  
 return Matrix(self.matrix)  
  
 @staticmethod  
 def transposed(self):  
 other = list(zip(\*self.matrix))  
 return Matrix(other)  
  
  
exec(stdin.read())

4. Внесите следующие изменение в предыдущую программу:

* Измените метод *\_\_mul\_\_* таким образом, чтобы матрицы можно было умножать как на скаляры, так и на другие матрицы. В случае, если две матрицы перемножить невозможно, то тогда выбросьте ошибку *MatrixError*. Первая матрице в ошибке — это *self*, вторая — это второй операнд умножения.

**Формат ввода**

Как в предыдущей задаче.

**Формат вывода**

Как в предыдущей задаче.

**Примеры**

Тест 1

Входные данные:

# Task 4 check 1

mid = Matrix([[1, 0, 0],[0, 1, 0],[0, 0, 1]])

m1 = Matrix([[3, 2], [-10, 0], [14, 5]])

m2 = Matrix([[5, 2, 10], [-0.5, -0.25, 18], [-22, -2.5, -0.125]])

print(mid \* m1)

print(mid \* m2)

print(m2 \* m1)

try:

m = m1 \* m2

print("WA It should be error")

except MatrixError as e:

print(e.matrix1)

print(e.matrix2)

Вывод программы:

3 2

-10 0

14 5

5.0 2.0 10.0

-0.5 -0.25 18.0

-22.0 -2.5 -0.125

135 60

253.0 89.0

-42.75 -44.625

3 2

-10 0

14 5

5 2 10

-0.5 -0.25 18

-22 -2.5 -0.125

Тест 2

Входные данные:

# Task 4 check 2

mid = Matrix([[1, 0, 0],[0, 1, 0],[0, 0, 1]])

m1 = Matrix([[3, 2], [-10, 0], [14, 5]])

m2 = Matrix([[5, 2, 10], [-0.5, -0.25, 18], [-22, -2.5, -0.125]])

print(0.5 \* m2)

print(m2 \* (0.5 \* mid \* m1))

Вывод программы:

2.5 1.0 5.0

-0.25 -0.125 9.0

-11.0 -1.25 -0.0625

67.5 30.0

126.5 44.5

-21.375 -22.3125

Тест 3

Входные данные:

# Task 4 check 3

mid = Matrix([[1, 0, 0],[0, 1, 0],[0, 0, 1]])

m1 = Matrix([[3, 2], [-10, 0], [14, 5]])

m2 = Matrix([[5, 2, 10], [-0.5, -0.25, 18], [-22, -2.5, -0.125]])

print(5 \* m2)

print(m2 \* (120 \* mid \* m1))

Вывод программы:

25 10 50

-2.5 -1.25 90

-110 -12.5 -0.625

16200 7200

30360.0 10680.0

-5130.0 -5355.0

5. Пусть экземпляр класса *Matrix* задаёт систему линейных алгебраических уравнений.

Тогда добавьте в класс метод *solve*, принимающий вектор-строку свободных членов и возвращающий строку-список, состоящую из *float* — решение системы, если оно единственно. Если решений нет или оно не единственно — выдайте какую-нибудь ошибку.

**Формат ввода**

Как в предыдущей задаче.

**Формат вывода**

Как в предыдущей задаче.

**Примеры**

Тест 1

Входные данные:

# Task 5 check 1

m = Matrix([[1, 0, 0], [0, 1, 0], [0, 0, 1]])

print(m.solve([1,1,1]))

Вывод программы:

[1.0, 1.0, 1.0]

Тест 2

Входные данные:

# Task 5 check 2

m = Matrix([[1, 1, 1], [0, 2, 0], [0, 0, 4]])

print(m.solve([1,1,1]))

Вывод программы:

[0.25, 0.5, 0.25]

Тест 3

Входные данные:

# Task 5 check 3

m = Matrix([[1, 1, 1], [0, 1, 2], [0.5, 1, 1.5]])

try:

s = m.solve([1,1,1])

print('WA No solution')

except Exception as e:

print('OK')

Вывод программы:

OK

6. К программе в предыдущей задаче добавьте класс *SquareMatrix* — наследник *Matrix* с операцией возведения в степень *\_\_pow\_\_*, принимающей натуральную степень (включая ноль), в которую нужно возвести матрицу. Используйте быстрое возведение в степень.

**Формат ввода**

Как в предыдущей задаче.

**Формат вывода**

Как в предыдущей задаче.

**Примеры**

Тест 1

Входные данные:

# Task 6 check 1

m = SquareMatrix([[1, 0], [0, 1]])

print(isinstance(m, Matrix))

Вывод программы:

True

Тест 2

Входные данные:

# Task 6 check 2

m = SquareMatrix([[1, 0], [0, 1]])

print(m \*\* 0)

Вывод программы:

1 0

0 1

Тест 3

Входные данные:

# Task 6 check 3

m = SquareMatrix([[1, 1, 0, 0, 0, 0],

[0, 1, 1, 0, 0, 0],

[0, 0, 1, 1, 0, 0],

[0, 0, 0, 1, 1, 0],

[0, 0, 0, 0, 1, 1],

[0, 0, 0, 0, 0, 1]]

)

print(m)

print('----------')

print(m \*\* 1)

print('----------')

print(m \*\* 2)

print('----------')

print(m \*\* 3)

print('----------')

print(m \*\* 4)

print('----------')

print(m \*\* 5)

Вывод программы:

1 1 0 0 0 0

0 1 1 0 0 0

0 0 1 1 0 0

0 0 0 1 1 0

0 0 0 0 1 1

0 0 0 0 0 1

----------

1 1 0 0 0 0

0 1 1 0 0 0

0 0 1 1 0 0

0 0 0 1 1 0

0 0 0 0 1 1

0 0 0 0 0 1

----------

1 2 1 0 0 0

0 1 2 1 0 0

0 0 1 2 1 0

0 0 0 1 2 1

0 0 0 0 1 2

0 0 0 0 0 1

----------

1 3 3 1 0 0

0 1 3 3 1 0

0 0 1 3 3 1

0 0 0 1 3 3

0 0 0 0 1 3

0 0 0 0 0 1

----------

1 4 6 4 1 0

0 1 4 6 4 1

0 0 1 4 6 4

0 0 0 1 4 6

0 0 0 0 1 4

0 0 0 0 0 1

----------

1 5 10 10 5 1

0 1 5 10 10 5

0 0 1 5 10 10

0 0 0 1 5 10

0 0 0 0 1 5

0 0 0 0 0 1