

Синтаксис Python. Функции, рекурсия.

Справочный материал можно найти в книжке ([ссылка](#), страница 47.)

Тренировочные (необязательные) задачи: А – D.

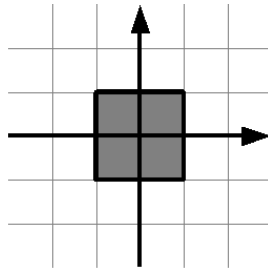
А. Длина отрезка

Даны четыре действительных числа: x_1, y_1, x_2, y_2 . Напишите функцию `distance(x1, y1, x2, y2)`, вычисляющую расстояние между точкой (x_1, y_1) и (x_2, y_2) . Считайте четыре действительных числа и выведите результат работы этой функции.

Input	Output
0 0 1 1	1.4142135623730951

В. Принадлежит ли точка квадрату - I

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.



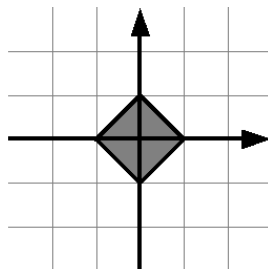
Решение должно содержать функцию `IsPointInSquare(x, y)`, возвращающую `True`, если точка принадлежит квадрату и `False`, если не принадлежит. Основная программа должна считать координаты точки, вызвать функцию `IsPointInSquare` и в зависимости от возвращённого значения вывести на экран необходимое сообщение.

Функция `IsPointInSquare` не должна содержать инструкцию `if`.

Input	Output
0 0	YES
Input	Output
3 -7	NO

С. Принадлежит ли точка квадрату - II

Решите аналогичную задачу для такого квадрата:

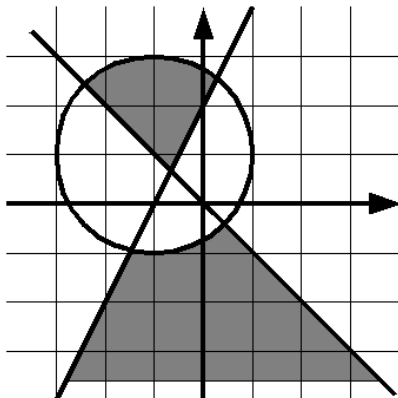


Решение должно соответствовать требованиям для решения задачи В.

Input	Output
1 1	NO

D. *Принадлежит ли точка области*

Проверьте, принадлежит ли точка данной закрашенной области:



Решение оформите в виде функции `IsPointInArea(x, y)`.

Решение должно соответствовать требованиям для решения задачи B.

Input	Output
-1 2	YES

E. *Минимальный делитель числа*

Дано натуральное число $n > 1$. Выведите его наименьший делитель, отличный от 1.

Решение оформите в виде функции `MinDivisor(n)`. Алгоритм должен иметь сложность $O(\sqrt{n})$.

Указание: Если у числа n нет делителя не превосходящего \sqrt{n} , то число n — простое и ответом будет само число n .

Input	Output
4	2

F. *Проверка числа на простоту*

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. Решение должно иметь сложность $O(\sqrt{n})$.

Input	Output
4	NO

G. *Дружественные числа*

Дружественные числа — это два различных натуральных числа, таких, что сумма всех делителей одного числа (меньших самого этого числа) равна другому числу, и наоборот (дружественными являются, например, 220 и 284).

Напишите программу, которая проверяет пару чисел на «дружественность». Используйте функцию, которая вычисляет сумму делителей числа.

Input	Output
220 284	YES

H. *Дружественные числа в диапазоне*

Дружественные числа — это два натуральных числа, таких, что сумма всех делителей одного числа (меньших самого этого числа) равна другому числу, и наоборот (дружественными являются, например, 220 и 284).

Напишите программу, которая находит все пары не равных друг другу дружественных чисел в заданном диапазоне. Используйте функцию, которая вычисляет сумму делителей числа.

В каждой паре сначала выводится меньшее число. Пары чисел должны выводиться в порядке возрастания первого числа из пары.

В случае, если таких пар в указанном диапазоне нет, вывести число 0.

Input	Output
1000 5000	(1184, 1210) (2620, 2924)

I. Сумма цифр не меняется

Напишите программу, которая находит все числа в диапазоне от a до b , сумма цифр которых не меняется при умножении на 2, 3, 4, 5, 6, 7, 8 и 9 (например, число 9). Используйте функцию для вычисления суммы цифр числа.

Выведите числа, соответствующие условию задачи, в возрастающем порядке через пробел. Если на указанном отрезке таких чисел нет — ничего не выводите.

Input	Output
9	9 18 45 90
90	

J. Гиперпростые числа

Простое число называется *гиперпростым*, если любое число, получающееся из него откидыванием нескольких последних цифр, тоже является простым. Например, число 733 — гиперпростое, так как и оно само, и числа 73 и 7 — простые. Напишите программу, которая определяет, верно ли, что переданное ей число N — гиперпростое. Учтите, что число 1 не считается простым.

Input	Output
733	YES

K. Гиперпростые числа в диапазоне

Напишите программу, которая находит все гиперпростые числа в заданном диапазоне. Если в указанном диапазоне гиперпростых чисел нет, вывести число 0.

Input	Output
30	31 37 53 59 71 73 79
100	

■ Все задачи L–Y следует решить при помощи рекурсивных функций.

Для изменения предельной глубины рекурсии используйте функцию `setrecursionlimit` из модуля `sys`.

Глубина рекурсии — максимальное количество вызванных, но незаконченных (отложенных) функций в ходе выполнения программы. По умолчанию глубина рекурсии равна 1000. Пример использования:

```
from sys import setrecursionlimit
...описания функций
setrecursionlimit(200000)
...текст программы
```

L. Сумма чисел

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

Input	Output
1	6
2	
3	
0	

M. Разворот последовательности

Дана последовательность целых чисел, заканчивающаяся числом 0. Выведите эту последовательность в обратном порядке.

Решение этой задачи при помощи рекурсии позволяет обойтись без списков, строк и прочих структур данных для сохранения всех введенных чисел.

Input	Output
1	0
2	3
3	2
0	1

N° Алгоритм Евклида

Даны два неотрицательных целых числа A и B . Требуется найти их наибольший общий делитель при помощи уже известного вам алгоритма Евклида.

Решение будет принято только если рекурсивная функция, вычисляющая НОД, будет иметь ОДИН условный оператор.

Input	Output
12	4
16	

О. *Наименьшее общее кратное*

Наименьшее общее кратное (НОК) двух натуральных чисел — это наименьшее число, которое делится нацело на оба исходных числа. Напишите программу, которая вычисляет НОК двух чисел.

Input	Output
12 16	48

Р° *Быстрое возведение в степень*

Дано вещественное число a и неотрицательное целое n . Вычислите a^n .

Указание: воспользуйтесь тождествами $a^{2n} = (a^2)^n$ и $a^{2n+1} = a^{2n} \cdot a$

Input	Output
1.000000001 1000000000	2.7182820387553908

Q* *Количество вызовов функции (числа Фибоначчи)*

Как известно, очередное число Фибоначчи равно сумме предыдущих двух. Первое и второе число Фибоначчи равны единице.

Программист Вася написал вычисление n -ого числа Фибоначчи с помощью рекурсивной функции. А сколько раз запустится эта функция прежде, чем будет получено значение?

Input	Output
3	3

R. *Сложение без сложения*

Напишите рекурсивную функцию $\text{sum}(a, b)$, возвращающую сумму двух целых неотрицательных чисел. Из всех арифметических операций допускаются только $+1$ и -1 . Циклы использовать нельзя.

Input	Output
2 2	4

S. *Фишки*

Дана полоска из клеток, пронумерованных от 1 до N . На каждом ходе разрешено поставить фишку на клетку (если её там еще нет) или снять фишку с клетки (если она там есть). При этом, можно выбрать не любую клетку, а только клетку под номером 1 или клетку с номером на 1 больше, чем у первой занятой клетки.

Изначально полоска пуста. Требуется занять все клетки.

Программа должна вывести последовательность номеров клеток, с которыми совершается действие. Если фишка снимается, то номер клетки должен выводиться со знаком минус.

Количество действий не должно превышать 10^4 . Если существует несколько возможных решений задачи, то разрешается вывести любое.

Input	Output
3	1 2 -1 3 1

T. *Ханойские башни*

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 (наверху самый маленький диск). 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.

Input	Output
2	1 1 2 2 1 3 1 2 3

U* *Ремонт в Ханое*

Решите задачу T со следующим ограничением: запрещено переключать диски со стержня 1 на стержень 3 и наоборот.

Вам не нужно находить минимальное решение, но количество совершённых перемещений не должно быть больше 200000, при условии, что количество дисков не превосходит 10.

Input	Output
2	1 1 2 1 2 3 2 1 2 1 3 2 1 2 1 2 2 3 1 1 2 1 2 3

V* *Циклические башни*

Решите задачу T со следующим ограничением: диск со стержня 1 можно переключать только на стержень 2, со стержня 2 на 3, а со стержня 3 на 1.

Вам не нужно находить минимальное решение, но количество совершённых перемещений не должно быть больше 200000, при условии, что количество дисков не превосходит 10.

Input	Output
2	1 1 2 1 2 3 2 1 2 1 3 1 2 2 3 1 1 2 1 2 3

W* *Несправедливые башни*

Решите задачу T со следующим ограничением: запрещено класть самый маленький диск (номер 1) на **средний** колышек (номер 2).

Input	Output
2	1 1 3 2 1 2 1 3 1 2 2 3 1 1 3

X* *Сортирующие башни*

Решите задачу T в такой формулировке: первоначально все диски лежат на стержне номер 1. Переместите диски с нечётными номерами на стержень номер 2, а с чётными номерами — на стержень номер 3.

Input	Output
3	1 1 2 2 1 3 1 2 3 3 1 2 1 3 2

Y* *Обменные башни*

Как и в предыдущих задачах, дано три стержня, на первом из которых надето n дисков различного размера. Необходимо их переместить на стержень 3 по следующим правилам: Самый маленький диск (номер 1) можно в любой момент переложить на любой стержень.

Перемещение диска номер 1 со стержня a на стержень b будем обозначать $1\ a\ b$.

Можно поменять два диска, лежащих на вершине двух стержней, если размеры этих дисков отличаются на 1. Например, если на вершине стержня с номером a лежит диск размером 5, а на вершине стержня с номером b лежит диск размером 4, то эти диски можно поменять местами.

Такой обмен двух дисков будем обозначать $0\ a\ b$ (указываются номера стержней, верхние диски которых обмениваются местами).

Для данного числа дисков n , не превосходящего 10, найдите решение головоломки. Вам не нужно находить минимальное решение, но количество совершённых перемещений не должно быть больше 200000.

Input	Output
2	1 1 3 0 1 3 1 1 3