**Переменные. Стандартный ввод/вывод**

Напишите программу:

Тимофей обычно спит ночью X*X* часов и устраивает себе днем тихий час на Y*Y* минут. Определите, сколько всего минут Тимофей спит в сутки.

Внимание, программа принимает значения X и Y из стандартного потока ввода (функция input), результат надо выводить в стандартный поток вывода (функция print). Обратите внимание на то, что приглашение, переданное в качестве аргумента в функцию input, считается выводом вашей программы. Используйте эту функцию без аргументов:

values = input() # без строки приглашения!

Решение: файл **ex00.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Коля каждый день ложится спать ровно в полночь и недавно узнал, что оптимальное время для его сна составляет X минут. Коля хочет поставить себе будильник так, чтобы он прозвенел ровно через X минут после полуночи, однако для этого необходимо указать время сигнала в формате часы, минуты. Помогите Коле определить, на какое время завести будильник.

Часы и минуты в выводе программы должны располагаться на разных строках (см. пример работы программы)

Помните, что для считывания данных нужно вызывать функцию input без аргументов!

Решение: файл **ex01.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Катя узнала, что ей для сна надо X минут. В отличие от Коли, Катя ложится спать после полуночи в Hчасов и Mминут. Помогите Кате определить, на какое время ей поставить будильник, чтобы он прозвенел ровно через Xминут после того, как она ляжет спать.

На стандартный ввод, каждое в своей строке, подаются значения *X*, H и M. Гарантируется, что Катя должна проснуться в тот же день, что и заснуть. Программа должна выводить время, на которое нужно поставить будильник: в первой строке часы, во второй — минуты.

Решение: файл **ex02.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Логические операции, операции сравнения**

Расставьте скобки в выражении

a**and**b**or not**a**and not**b

в соответствии с порядком вычисления выражения (приоритетом операций). Всего потребуется 5 пар скобок (внешние скобки входят в их число).

Решение: файл **ex03.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Выполните код в интерпретаторе Python 3 и вставьте в поле ответа результат вычисления. Постарайтесь разобраться, почему интерпретатор выдал именно такой ответ. Помните, что любые арифметические операции выше по приоритету операций сравнения и логических операторов.

Помните, что в Python логические значения пишутся с большой буквы: True, False.

x = 5

y = 10

y > x \* x or y >= 2 \* x and x < y

Решение: файл **ex04.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Найдите результат выражения для заданных значений a*a* и b*b*. Учитывайте регистр символов при ответе.

a = True

b = False

a and b or not a and not b

Решение: файл **ex05.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Условия: if, else, elif. Блоки, отступы**

Из передачи “Здоровье” Аня узнала, что рекомендуется спать хотя бы A часов в сутки, но пересыпать тоже вредно и не стоит спать более Bчасов. Сейчас Аня спит H часов в сутки. Если режим сна Ани удовлетворяет рекомендациям передачи “Здоровье”, выведите “Это нормально”. Если Аня спит менее A часов, выведите “Недосып”, если же более B часов, то выведите “Пересып”.

Получаемое число A всегда меньше либо равно B.

На вход программе в три строки подаются переменные в следующем порядке: A, B, H.

Решение: файл **ex06.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Требуется определить, является ли данный год високосным.

Напомним, что високосными годами считаются те годы, порядковый номер которых либо кратен 4, но при этом не кратен 100, либо кратен 400 (например, 2000-й год являлся високосным, а 2100-й будет невисокосным годом).

Программа должна корректно работать на числах 1900≤n≤3000.

Выведите "Високосный" в случае, если считанный год является високосным и "Обычный" в обратном случае (не забывайте проверять регистр выводимых программой символов).

Решение: файл **ex07.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Задачи по материалам недели**

В то далёкое время, когда Паша ходил в школу, ему очень не нравилась формула Герона для вычисления площади треугольника, так как казалась слишком сложной. В один прекрасный момент Павел решил избавить всех школьников от страданий и написать и распространить по школам программу, вычисляющую площадь треугольника по трём сторонам.

Одна проблема: так как эта формула не нравилась Павлу, он её не запомнил. Помогите ему завершить доброе дело и напишите программу, вычисляющую площадь треугольника по переданным длинам трёх его сторон по формуле Герона:

S=\sqrt{p(p−a)(p−b)(p−c)}*S*=*p*(*p*−*a*)(*p*−*b*)(*p*−*c*)​где p=\dfrac{a+b+c}2*p*=2*a*+*b*+*c*​ – полупериметр треугольника. На вход программе подаются целые числа, выводом программы должно являться вещественное число, соответствующее площади треугольника.

Решение: файл **ex08.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, принимающую на вход целое число, которая выводит True, если переданное значение попадает в интервал (-15, 12] \cup (14, 17) \cup [19, +\infty)(−15,12]∪(14,17)∪[19,+∞) и False в противном случае (регистр символов имеет значение).

Обратите внимание на разные скобки, используемые для обозначения интервалов. В задании используются полуоткрытые и открытые интервалы. Подробнее про это вы можете прочитать, например, на википедии ([полуинтервал](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D1%83%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D0%B2%D0%B0%D0%BB), [промежуток](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%BC%D0%B5%D0%B6%D1%83%D1%82%D0%BE%D0%BA_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)#.D0.A2.D0.B5.D1.80.D0.BC.D0.B8.D0.BD.D0.BE.D0.BB.D0.BE.D0.B3.D0.B8.D1.8F)).

Решение: файл **ex09.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите простой калькулятор, который считывает с пользовательского ввода три строки: первое число, второе число и операцию, после чего применяет операцию к введённым числам ("первое число" "операция" "второе число") и выводит результат на экран.

Поддерживаемые операции: +, -, /, \*, mod, pow, div, где  
mod — это взятие остатка от деления,  
pow — возведение в степень,  
div — целочисленное деление.

Если выполняется деление и второе число равно 0, необходимо выводить строку "Деление на 0!".

Обратите внимание, что на вход программе приходят вещественные числа.

Решение: файл **ex10.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Жители страны Малевии часто экспериментируют с планировкой комнат. Комнаты бывают треугольные, прямоугольные и круглые. Чтобы быстро вычислять жилплощадь, требуется написать программу, на вход которой подаётся тип фигуры комнаты и соответствующие параметры, которая бы выводила площадь получившейся комнаты.  
Для числа π в стране Малевии используют значение 3.14.

Формат ввода, который используют Малевийцы:

Треугольник a b c

где a, b и c — длины сторон треугольника  
  
прямоугольник a b

где a и b — длины сторон прямоугольника  
  
круг r

где r — радиус окружности

Решение: файл **ex11.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, которая получает на вход три целых числа, по одному числу в строке, и выводит на консоль в три строки сначала максимальное, потом минимальное, после чего оставшееся число.

На ввод могут подаваться и повторяющиеся числа.

Решение: файл **ex12.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

В институте биоинформатики по офису передвигается робот. Недавно студенты из группы программистов написали для него программу, по которой робот, когда заходит в комнату, считает количество программистов в ней и произносит его вслух: "n программистов".

Для того, чтобы это звучало правильно, для каждого n*n* нужно использовать верное окончание слова.

Напишите программу, считывающую с пользовательского ввода целое число n*n* (неотрицательное), выводящее это число в консоль вместе с правильным образом изменённым словом "программист", для того, чтобы робот мог нормально общаться с людьми, например: 1 программист, 2 программиста, 5 программистов.

В комнате может быть очень много программистов. Проверьте, что ваша программа правильно обработает все случаи, как минимум до 1000 человек.

Решение: файл **ex13.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Паша очень любит кататься на общественном транспорте, а получая билет, сразу проверяет, счастливый ли ему попался. Билет считается счастливым, если сумма первых трех цифр совпадает с суммой последних трех цифр номера билета.  
  
Однако Паша очень плохо считает в уме, поэтому попросил вас написать программу, которая проверит равенство сумм и выведет "Счастливый", если суммы совпадают, и "Обычный", если суммы различны.  
  
На вход программе подаётся строка из шести цифр.

Выводить нужно только слово "Счастливый" или "Обычный", с большой буквы.

Решение: файл **ex14.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Циклы. Строки. Списки**

Какое значение будет у переменной i после выполнения фрагмента программы?

i = 0

while i <= 10:

i = i + 1

if i > 7:

i = i + 2

Решение: файл **ex15.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Сколько итераций цикла будет выполнено в этом фрагменте программы?

i = 0

while i <= 10:

i = i + 1

if i > 7:

i = i + 2

Решение: файл **ex16.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Сколько всего знаков \* будет выведено после исполнения фрагмента программы:

i = 0

while i < 5:

print('\*')

if i % 2 == 0:

print('\*\*')

if i > 2:

print('\*\*\*')

i = i + 1

Решение: файл **ex17.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, которая считывает со стандартного ввода целые числа, по одному числу в строке, и после первого введенного нуля выводит сумму полученных на вход чисел.

Решение: файл **ex18.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

В Институте биоинформатики между информатиками и биологами устраивается соревнование. Победителям соревнования достанется большой и вкусный пирог. В команде биологов a a a человек, а в команде информатиков — b b b человек.

Нужно заранее разрезать пирог таким образом, чтобы можно было раздать кусочки пирога любой команде, выигравшей соревнование, при этом каждому участнику этой команды должно достаться одинаковое число кусочков пирога. И так как не хочется резать пирог на слишком мелкие кусочки, нужно найти минимальное подходящее число.

Напишите программу, которая помогает найти это число.   
Программа должна считывать размеры команд (два положительных целых числа a a a и b b b, каждое число вводится на отдельной строке) и выводить наименьшее число d d d, которое делится на оба этих числа без остатка.

Решение: файл **ex19.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Определите, какое значение будет иметь переменная i после выполнения следующего фрагмента программы:

i = 0

s = 0

while i < 10:

i = i + 1

s = s + i

if s > 15:

break

i = i + 1

Решение: файл **ex20.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Определите, какое значение будет иметь переменная i после выполнения следующего фрагмента программы:

i = 0

s = 0

while i < 10:

i = i + 1

s = s + i

if s > 15:

continue

i = i + 1

Решение: файл **ex21.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, которая считывает целые числа с консоли по одному числу в строке.

Для каждого введённого числа проверить:  
если число меньше 10, то пропускаем это число;  
если число больше 100, то прекращаем считывать числа;  
в остальных случаях вывести это число обратно на консоль в отдельной строке.

Решение: файл **ex22.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Когда Павел учился в школе, он запоминал таблицу умножения прямоугольными блоками. Для тренировок ему бы очень пригодилась программа, которая показывала бы блок таблицы умножения.

Напишите программу, на вход которой даются четыре числа a a a, b b b, c c c и d d d, каждое в своей строке. Программа должна вывести фрагмент таблицы умножения для всех чисел отрезка [a;b] [a; b] [a;b] на все числа отрезка [c;d] [c;d] [c;d].

Числа a a a, b b b, c c c и d d d являются натуральными и не превосходят 10, a≤b a \le b a≤b, c≤d c \le d c≤d.

Следуйте формату вывода из примера, для разделения элементов внутри строки используйте '\t' — символ табуляции. Заметьте, что левым столбцом и верхней строкой выводятся **сами числа из заданных отрезков** — заголовочные столбец и строка таблицы.

Решение: файл **ex23.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, которая считывает с клавиатуры два числа a a a и b b b, считает и выводит на консоль среднее арифметическое всех чисел из отрезка [a;b] [a; b] [a;b], которые делятся на 3 3 3.

В приведенном ниже примере среднее арифметическое считается для чисел на отрезке [−5;12] [-5; 12] [−5;12]. Всего чисел, делящихся на 3 3 3, на этом отрезке 6 6 6: −3,0,3,6,9,12 -3, 0, 3, 6, 9, 12 −3,0,3,6,9,12. Их среднее арифметическое равно 4.5 4.5 4.5

На вход программе подаются интервалы, внутри которых всегда есть хотя бы одно число, которое делится на 3 3 3.﻿

Решение: файл **ex24.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

[GC-состав](https://ru.wikipedia.org/wiki/GC-%D1%81%D0%BE%D1%81%D1%82%D0%B0%D0%B2) является важной характеристикой геномных последовательностей и определяется как процентное соотношение суммы всех гуанинов и цитозинов к общему числу нуклеиновых оснований в геномной последовательности.

Напишите программу, которая вычисляет процентное содержание символов G (гуанин) и C (цитозин) в введенной строке (программа не должна зависеть от регистра вводимых символов).

Например, в строке "acggtgttat" процентное содержание символов G и C равно 410⋅100=40.0 \dfrac4{10} \cdot 100 = 40.0 104​⋅100=40.0, где 4 -- это количество символов G и C,  а 10 -- это длина строки.

Решение: файл **ex25.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Есть строка s = "abcdefghijk".  
В поле ответа через пробел запишите строки (без кавычек), полученные в результате следующих операций:

# s = 'abcdefghijk'

s[3:6]

s[:6]

s[3:]

s[::-1]

s[-3:]

s[:-6]

s[-1:-10:-2]

Итоговый формат ответа должен выглядеть следующим образом:

abcd efg hijklmnop qrst uvw xy z

Решение: файл **ex26.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Узнав, что ДНК не является случайной строкой, только что поступившие в Институт биоинформатики студенты группы информатиков предложили использовать алгоритм сжатия, который сжимает повторяющиеся символы в строке.

Кодирование осуществляется следующим образом:  
s = 'aaaabbсaa' преобразуется в 'a4b2с1a2', то есть группы одинаковых символов исходной строки заменяются на этот символ и количество его повторений в этой позиции строки.

Напишите программу, которая считывает строку, кодирует её предложенным алгоритмом и выводит закодированную последовательность на стандартный вывод. Кодирование должно учитывать регистр символов.

Решение: файл **ex27.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Сколько элементов будет содержать список students после следующих операций?

students = ['Ivan', 'Masha', 'Sasha']

students += ['Olga']

students += 'Olga'

Решение: файл **ex28.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Имеется программа, код которой указан ниже.   
Укажите, какие значения будут содержать списки в помеченных участках:

a = [1, 2, 3]

b = a

# значения списка b?

a[1] = 10

# значения списка b?

b[0] = 20

# значения списка a?

a = [5, 6]

# значения списка b?

Решение: файл **ex29.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, на вход которой подается одна строка с целыми числами. Программа должна вывести сумму этих чисел.

Используйте метод split строки.

Решение: файл **ex30.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, на вход которой подаётся список чисел одной строкой. Программа должна для каждого элемента этого списка вывести сумму двух его соседей. Для элементов списка, являющихся крайними, одним из соседей считается элемент, находящий на противоположном конце этого списка. Например, если на вход подаётся список "1 3 5 6 10", то на выход ожидается список "13 6 9 15 7" (без кавычек).

Если на вход пришло только одно число, надо вывести его же.

Вывод должен содержать одну строку с числами нового списка, разделёнными пробелом.

Решение: файл **ex31.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, которая принимает на вход список чисел в одной строке и выводит на экран в одну строку значения, которые повторяются в нём более одного раза.

Для решения задачи может пригодиться метод sort списка.

Выводимые числа не должны повторяться, порядок их вывода может быть произвольным.

Решение: файл **ex32.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Задачи по материалам недели**

Напишите программу, которая считывает с консоли числа (по одному в строке) до тех пор, пока сумма введённых чисел не будет равна 0 и **сразу после этого** выводит сумму квадратов всех считанных чисел.

Гарантируется, что в какой-то момент сумма введённых чисел окажется равной 0, **после этого считывание продолжать не нужно**.

В примере мы считываем числа 1, -3, 5, -6, -10, 13; в этот момент замечаем, что сумма этих чисел равна нулю и выводим сумму их квадратов, не обращая внимания на то, что остались ещё не прочитанные значения.

Решение: файл **ex33.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, которая выводит часть последовательности 1 2 2 3 3 3 4 4 4 4 5 5 5 5 5 ... (число повторяется столько раз, чему равно). На вход программе передаётся неотрицательное целое число n — столько элементов последовательности должна отобразить программа. На выходе ожидается последовательность чисел, записанных через пробел в одну строку.

Например, если n = 7, то программа должна вывести 1 2 2 3 3 3 4.

Решение: файл **ex34.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, которая считывает список чисел lst lst lst из первой строки и число x x x из второй строки, которая выводит все позиции, на которых встречается число x x x в переданном списке lst lst lst.

Позиции нумеруются с нуля, если число x x x не встречается в списке, вывести строку "Отсутствует" (без кавычек, с большой буквы).

Позиции должны быть выведены в одну строку, по возрастанию абсолютного значения.

Решение: файл **ex35.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, на вход которой подаётся прямоугольная матрица в виде последовательности строк, заканчивающихся строкой, содержащей только строку "end" (без кавычек)

Программа должна вывести матрицу того же размера, у которой каждый элемент в позиции i, j равен сумме элементов первой матрицы на позициях (i-1, j), (i+1, j), (i, j-1), (i, j+1). У крайних символов соседний элемент находится с противоположной стороны матрицы.

В случае одной строки/столбца элемент сам себе является соседом по соответствующему направлению.

Решение: файл **ex36.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Выведите таблицу размером n×n n \times n n×n, заполненную числами от 1 1 1 до n2 n^2 n2 по спирали, выходящей из левого верхнего угла и закрученной по часовой стрелке, как показано в примере (здесь n=5 n=5 n=5):

Решение: файл **ex37.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Функции. Словари. Интерпретатор. Модули.**

Есть функция f, которая определена следующим образом:

def f(n):

    return n \* 10 + 5

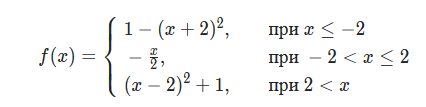
Введите её в интерпретаторе и посчитайте, чему равно значение следующего выражения:

f(f(f(10)))

Решение: файл **ex38.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите функцию f(x), которая возвращает значение следующей функции, определённой на всей числовой прямой:



Решение: файл **ex39.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите функцию modify\_list(l), которая принимает на вход список целых чисел, удаляет из него все нечётные значения, а чётные **нацело** делит на два. Функция не должна ничего возвращать, требуется только изменение переданного списка, например:

lst = [1, 2, 3, 4, 5, 6]

print(modify\_list(lst)) # None

print(lst)   # [1, 2, 3]

modify\_list(lst)

print(lst) # [1]

lst = [10, 5, 8, 3]

modify\_list(lst)

print(lst) # [5, 4]

Функция не должна осуществлять ввод/вывод информации.

Решение: файл **ex40.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите функцию update\_dictionary(d, key, value), которая принимает на вход словарь d d d и два числа: key key key и value value value.

Если ключ key key key есть в словаре d d d, то добавьте значение value value value в список, который хранится по этому ключу.  
Если ключа key key key нет в словаре, то нужно добавить значение в список по ключу 2∗key 2 \* key 2∗key. Если и ключа 2∗key 2 \* key 2∗key нет, то нужно добавить ключ 2∗key 2 \* key 2∗key в словарь и сопоставить ему список из переданного элемента [value] [value] [value].

Требуется реализовать только эту функцию, кода вне неё не должно быть.  
Функция не должна вызывать внутри себя функции input и print.

Решение: файл **ex41.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Когда Антон прочитал «Войну и мир», ему стало интересно, сколько слов и в каком количестве используется в этой книге.

Помогите Антону написать упрощённую версию такой программы, которая сможет подсчитать слова, разделённые пробелом и вывести получившуюся статистику.

Программа должна считывать одну строку со стандартного ввода и выводить для каждого уникального слова в этой строке число его повторений (**без учёта регистра**) в формате "слово количество" (см. пример вывода).   
Порядок вывода слов может быть произвольным, каждое уникальное слово﻿ должно выводиться только один раз.

Решение: файл **ex42.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, на вход которой на первой строке подаётся число nnn, после этого на nnn строках передаются числа xix\_ixi​.

Для каждого числа xix\_ixi​ на отдельной строке выведите значение f(xi)f(x\_i)f(xi​). Функция f(x) уже реализована и доступна для вызова.

Функция вычисляется достаточно долго и зависит только от переданного аргумента x x x. Для того, чтобы уложиться в ограничение по времени, нужно сохранять вычисленные значения.

Решение: файл **ex43.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, которая считывает из файла строку, соответствующую тексту, сжатому с помощью кодирования повторов, и производит обратную операцию, получая исходный текст.

Запишите полученный текст в файл и прикрепите его, как ответ на это задание.

В исходном тексте не встречаются цифры, так что код однозначно интерпретируем.

Примечание. Это первое задание типа Dataset Quiz. В таких заданиях после нажатия "Start Quiz" у вас появляется ссылка "download your dataset". Используйте эту ссылку для того, чтобы загрузить файл со входными данными к себе на компьютер. Запустите вашу программу, используя этот файл в качестве входных данных. Выходной файл, который при этом у вас получится, надо отправить в качестве ответа на эту задачу.

Решение: файл **ex44.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, которая считывает текст из файла (в файле может быть больше одной строки) и выводит самое частое слово в этом тексте и через пробел то, сколько раз оно встретилось. Если таких слов несколько, вывести лексикографически первое (можно использовать оператор < для строк).

В качестве ответа укажите вывод программы, а не саму программу.

Слова, написанные в разных регистрах, считаются одинаковыми.

Решение: файл **ex45.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Имеется файл с данными по успеваемости абитуриентов. Он представляет из себя набор строк, где в каждой строке записана следующая информация:

Фамилия;Оценка\_по\_математике;Оценка\_по\_физике;Оценка\_по\_русскому\_языку

Поля внутри строки разделены точкой с запятой, оценки — целые числа.

Напишите программу, которая считывает файл с подобной структурой и для каждого абитуриента выводит его среднюю оценку по этим трём предметам на отдельной строке, соответствующей этому абитуриенту.

Также в конце файла, на отдельной строке, через пробел запишите средние баллы по математике, физике и русскому языку по всем абитуриентам.

В качестве ответа на задание прикрепите полученный файл со средними оценками.

Примечание. Для разбиения строки на части по символу ';' можно использовать метод split следующим образом:

print('First;Second-1 Second-2;Third'.split(';'))

# ['First', 'Second-1 Second-2', 'Third']

Решение: файл **ex46.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, которая подключает модуль math и, используя значение числа π \pi π из этого модуля, находит для переданного ей на стандартный ввод радиуса круга периметр этого круга и выводит его на стандартный вывод

Решение: файл **ex47.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Напишите программу, которая запускается из консоли и печатает значения всех переданных аргументов на экран (имя скрипта выводить не нужно). Не изменяйте порядок аргументов при выводе.

Для доступа к аргументам командной строки программы подключите модуль sys и используйте переменную argv из этого модуля.

Пример работы программы:

> python3 my\_solution.py arg1 arg2

arg1 arg2

Решение: файл **ex48.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Скачайте файл. В нём указан адрес другого файла, который нужно скачать с использованием модуля requests и посчитать число строк в нём.

Используйте функцию get для получения файла (имеет смысл вызвать метод strip к передаваемому параметру, чтобы убрать пробельные символы по краям).

После получения файла вы можете проверить результат, обратившись к полю text. Если результат работы скрипта не принимается, проверьте поле url на правильность. Для подсчёта количества строк разбейте текст с помощью метода splitlines.

В поле ответа введите одно число или отправьте файл, содержащий одно число.

Решение: файл **ex49.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Имеется набор файлов, каждый из которых, кроме последнего, содержит имя следующего файла.  
Первое слово в тексте последнего файла: "We".

Скачайте предложенный файл. В нём содержится ссылка на первый файл из этого набора.

Все файлы располагаются в каталоге по адресу:  
[https://stepic.org/media/attachments/course67/3.6.3/](https://stepik.org/media/attachments/course67/3.6.3/)

Загрузите содержимое ﻿последнего файла из набора, как ответ на это задание

Решение: файл **ex50.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Задачи по материалам недели**

Напишите программу, которая принимает на стандартный вход список игр футбольных команд с результатом матча и выводит на стандартный вывод сводную таблицу результатов всех матчей.

За победу команде начисляется 3 очка, за поражение — 0, за ничью *—* 1.

Формат ввода следующий:  
В первой строке указано целое число n n n — количество завершенных игр.  
После этого идет n n n строк, в которых записаны результаты игры в следующем формате:  
Первая\_команда;Забито\_первой\_командой;Вторая\_команда;Забито\_второй\_командой

Вывод программы необходимо оформить следующим образом:  
Команда:Всего\_игр Побед Ничьих Поражений Всего\_очков

Конкретный пример ввода-вывода приведён ниже.

Порядок вывода команд произвольный.

Решение: файл **ex51.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

В какой-то момент в Институте биоинформатики биологи перестали понимать, что говорят информатики: они говорили каким-то странным набором звуков.

В какой-то момент один из биологов раскрыл секрет информатиков: они использовали при общении [подстановочный шифр](https://ru.wikipedia.org/wiki/Шифр_подстановки), т.е. заменяли каждый символ исходного сообщения на соответствующий ему другой символ. Биологи раздобыли ключ к шифру и теперь нуждаются в помощи:

Напишите программу, которая умеет шифровать и расшифровывать шифр подстановки. Программа принимает на вход две строки одинаковой длины, на первой строке записаны символы исходного алфавита, на второй строке — символы конечного алфавита, после чего идёт строка, которую нужно зашифровать переданным ключом, и ещё одна строка, которую нужно расшифровать.

Пусть, например, на вход программе передано:  
abcd  
\*d%#  
abacabadaba  
#\*%\*d\*%

Это значит, что символ **a** исходного сообщения заменяется на символ **\*** в шифре, **b** заменяется на **d**, **c** — на **%** и **d** — на **#**.  
Нужно зашифровать строку **abacabadaba** и расшифровать строку **#\*%\*d\*%** с помощью этого шифра. Получаем следующие строки, которые и передаём на вывод программы:  
\*d\*%\*d\*#\*d\*  
dacabac

Решение: файл **ex52.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Простейшая система проверки орфографии может быть основана на использовании списка известных слов.  
Если введённое слово не найдено в этом списке, оно помечается как "ошибка".

Попробуем написать подобную систему.

На вход программе первой строкой передаётся количество d d d известных нам слов, после чего на d d d строках указываются эти слова. Затем передаётся количество l l l строк текста для проверки, после чего l l l строк текста.

Выведите уникальные "ошибки" в произвольном порядке. Работу производите без учёта регистра.

Решение: файл **ex53.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Группа биологов в институте биоинформатики завела себе черепашку.

После дрессировки черепашка научилась понимать и запоминать указания биологов следующего вида:  
север 10  
запад 20  
юг 30  
восток 40  
где первое слово — это направление, в котором должна двигаться черепашка, а число после слова — это положительное расстояние в сантиметрах, которое должна пройти черепашка.

Но команды даются быстро, а черепашка ползёт медленно, и программисты догадались, что можно написать программу, которая определит, куда в итоге биологи приведут черепашку. Для этого программисты просят вас написать программу, которая выведет точку, в которой окажется черепашка после всех команд. Для простоты они решили считать, что движение начинается в точке (0, 0), и движение на восток увеличивает первую координату, а на север — вторую.

Программе подаётся на вход число команд n n n, которые нужно выполнить черепашке, после чего n n n строк с самими командами. Вывести нужно два числа в одну строку: первую и вторую координату конечной точки черепашки. Все координаты целочисленные.

Решение: файл **ex54.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дан файл с таблицей в формате [TSV](https://ru.wikipedia.org/wiki/TSV) с информацией о росте школьников разных классов.

Напишите программу, которая прочитает этот файл и подсчитает для каждого класса средний рост учащегося.

Файл состоит из набора строк, каждая из которых представляет собой три поля:  
Класс Фамилия Рост

Класс обозначается только числом. Буквенные модификаторы не используются. Номер класса может быть от 1 до 11 включительно. В фамилии нет пробелов, а в качестве роста используется натуральное число, но при подсчёте среднего требуется вычислить значение в виде вещественного числа.

Выводить информацию о среднем росте следует в порядке возрастания номера класса (для классов с первого по одиннадцатый). Если про какой-то класс нет информации, необходимо вывести напротив него прочерк, например:

Решение: файл **ex55.py**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_