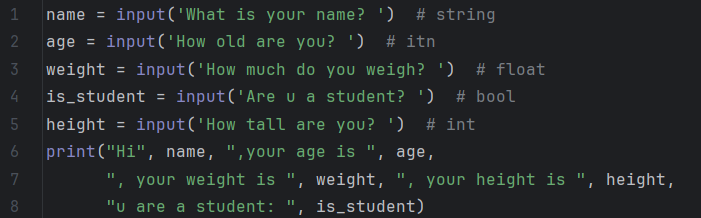
Лабораторная работа №1

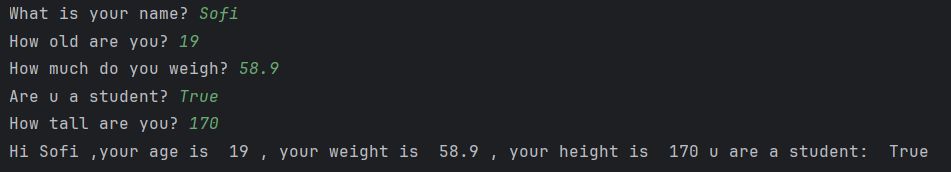
Задание 1

Для ввода информации в переменные используется **input()**, для вывода - **print()**.

Используя эти команды создайте **5 переменных** и введите с клавиатуры в них значения различных типов (целый, вещественный, логический, строка).

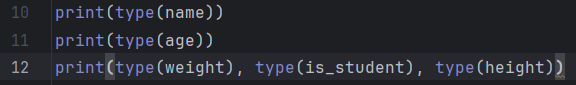
Code:

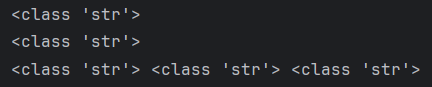


Result:  


При помощи **type()** выведите типы этих переменных на экран.

Code:

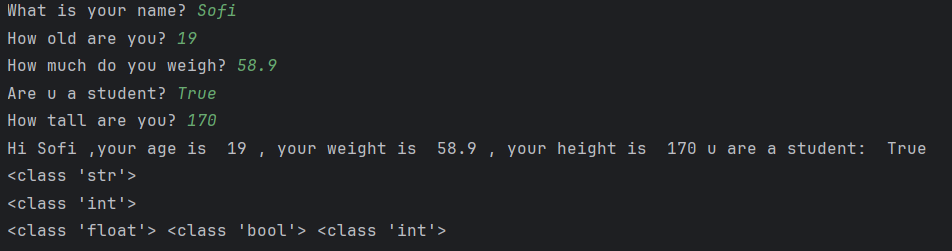


Result:  


Создайте **ещё 5 переменных**, но при их записи используйте команды для преобразования данных в нужный тип. Выведите их на экран вместе с их типом.

Code:



Result:  


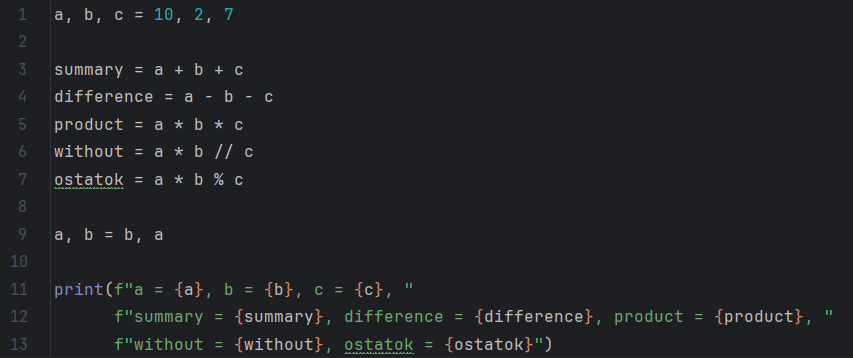
**Задание 2.**

Создайте три числовые переменные, указав их через запятую: **a, b, c** = 10, 2, 7.

Создайте ещё 5 переменных, которые используют переменные **a, b, c** для вычисления суммы, разности, произведения, остатка от деления, деления без остатка.

Поменяйте значения переменных **a, b одной строкой кода.**

Выведите все переменные используя f-строки.

Code:  


Result:

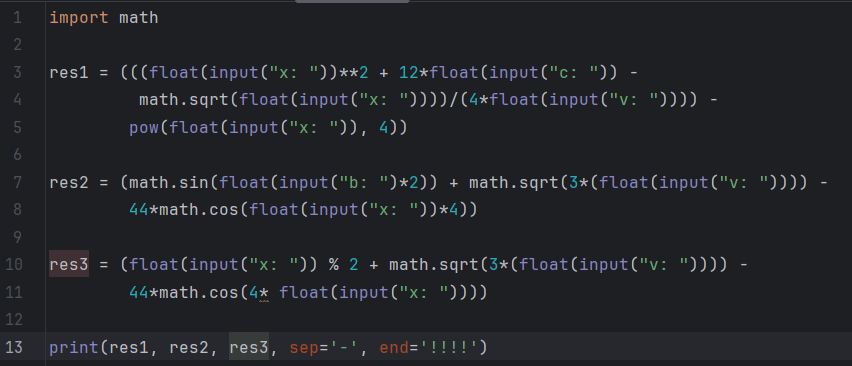


Найдите результат вычислений данных примеров:

* 
* 
* x % 2 + c // 4 - v / 10 + 18b

не создавайте переменные для значений x, c, v, b, а пропишите **input()** прямо в формуле. используйте модуль **Math** для сложных математических вычислений.

Выведете результаты всех 4 формул в одну строчку, через “\*\*-\*\*” и в конце “!!!!” (для этого используйте **sep**, **end** в **print** )

Code:  


Result:



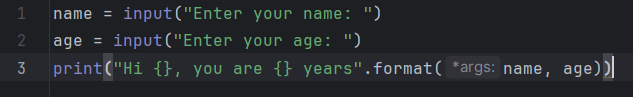
Форматированный вывод.

Форматированный ввод-вывод в Python обеспечивает более удобное и наглядное представление данных при выполнении программ. В Python для форматирования используется метод format() для строк и специальные символы для форматирования чисел.

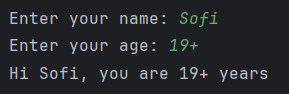
Используя форматированный вывод сделайте следующее:

Напишите программу, которая запрашивает у пользователя его имя и возраст, а затем выводит приветствие в формате "Привет, <имя>! Тебе <возраст> лет."

Code:

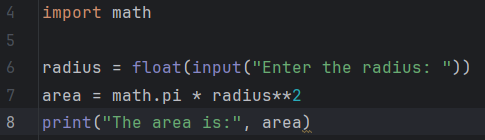


Result:



Напишите программу, которая запрашивает у пользователя радиус окружности и выводит её площадь. Формат вывода: "Площадь окружности радиусом <радиус> равна <площадь>."

Code:

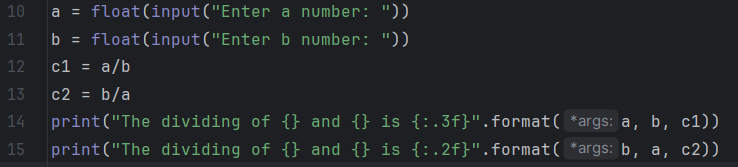


Result:

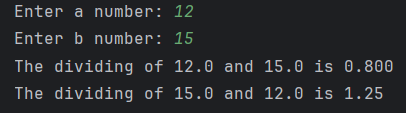


Напишите программу, которая запрашивает у пользователя два числа (a, b) и выводит результаты деления a на b и b на a . " Округлите первое вычисление до 3 знаков после запятой, а второе - до 2.

Code:



Result:

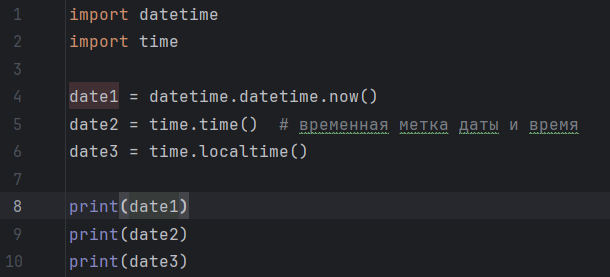


**Задание 5.**

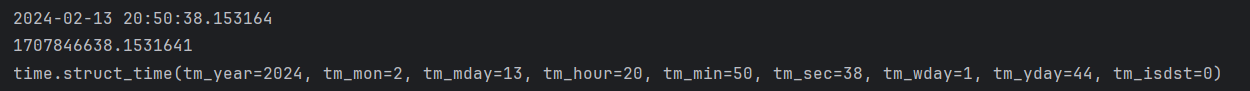
Подключите модуль **time** к файлу.

Попробуйте считать сегодняшнюю дату и время используя **time() и localtime().**

Code:



Result:



Используя форматированный вывод выведите на экран полученные данные в виде:

Текущая дата и время: 2024-01-23 15:30:00

Дата: 23/01/2024

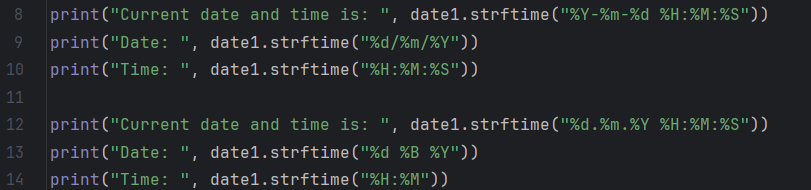
Время: 15:30:00

Текущая дата и время: 23.01.2024 15:30:00

Дата: 23 января 2024

Время: 15:30

Code:



Result:



Лабораторная работа №2 (2ч). Условные конструкции (if и match). Циклы.

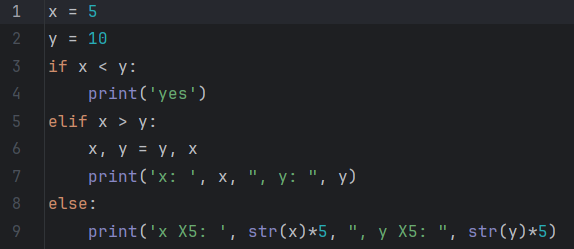
**Задание 1 (используем if).**

Введите в переменные два числа.

Если первое число больше второго, то вывести yes.

Если второе число больше первого поменять значения этих переменных и вывести их на экран.

Если числа равны продублировать их 5 раз в строку (без использования цикла).

Code:  


Result:  



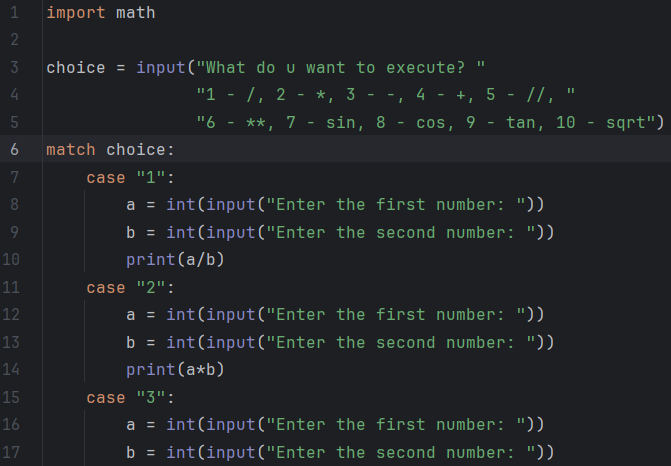


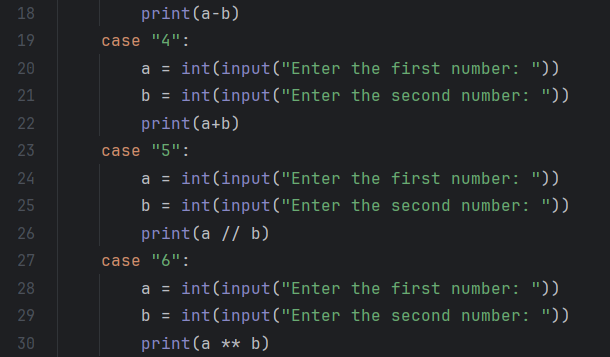

**Задание 2 (используем match).**

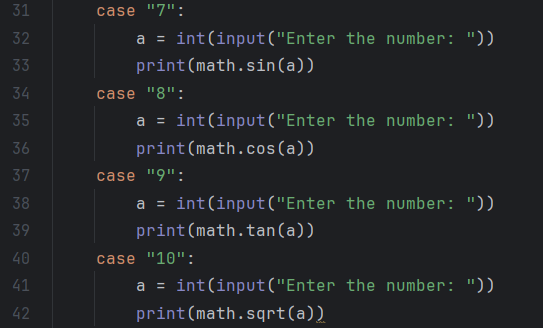
Написать мини-калькулятор.

## Необходимо спросить какую операцию необходимо сделать : /, \*,-,+,//, \*\*, sin, cos, tg, корень. Далее в зависимости от выбранной операции попросить ввести одно или два числа. После этого посчитать результат и вывести его.

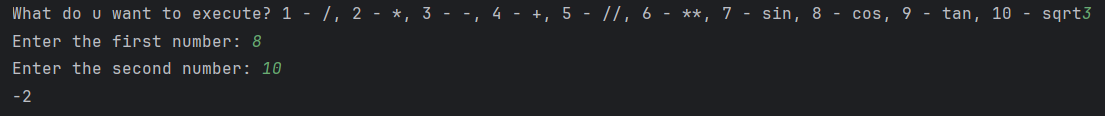
Code:







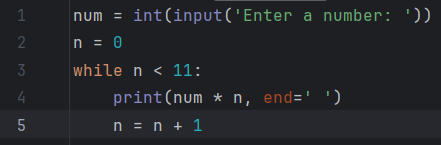
Result:



**Задание 3 (используем while).**

Напишите программу, которая просит ввести с клавиатуры число X с клавиатуры, исходя из введенного число выведите последовательность следующую последовательность (от 0 до 10): 0, 1\*X, 2\*X, …, 10\*X.

Code:



Result:



**Задание 4 (используем for ).**

Используя цикл for выведите на экран все числа (для каждого - отдельный цикл: всего 7):

* от 1 до 15
* от 5 до 20
* от 20 до 2
* от 2 до 16 с шагом 2
* от 4 до 44 с шагом 4
* от 5 до 25 с шагом 5
* от 80 до 0 с шагом 10

**Приведите примеры использования:**

* for \_ in range( V ): pass
* for z, x in range( V ), range( B ): pass
* for index, fruit in enumerate(fruits): pass
* for name, age in zip(names, ages): pass
* for c else

Code:



Result:

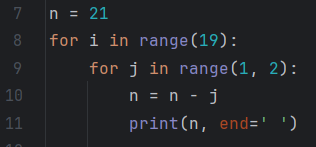


Code:



Result:  

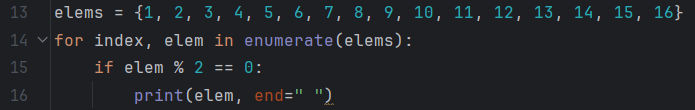

Code:



Result:



Code:



Result:



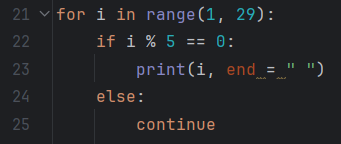
Code:



Result:



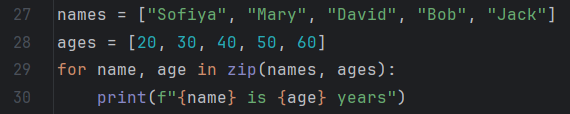
Code:



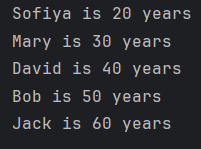
Result:



Code:



Result:



Code:



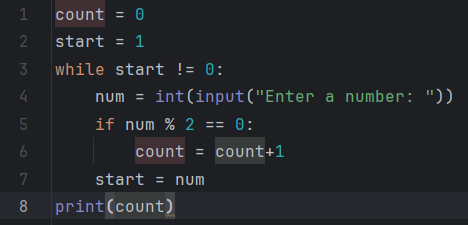
Result:



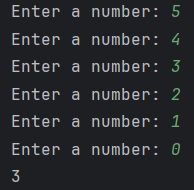
**Задание 5.**

Пользователь вводит ненулевые целые числа до тех пор, пока не введет ноль. Найдите количество четных чисел, которые он ввел.

Code:



Result:



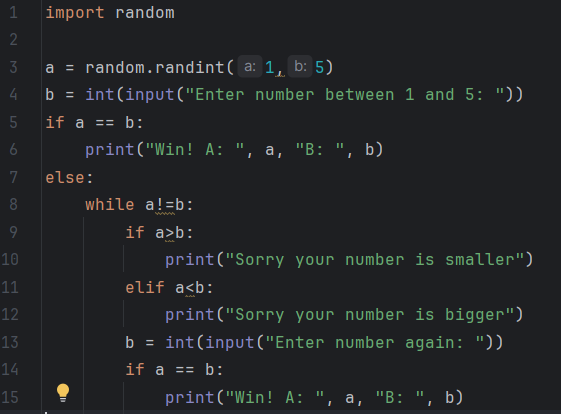
## Задание 6.

## Используйте модуль random. Компьютер загадывает число от А до В (вводятся с клавиатуры).

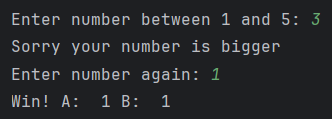
## Напишите угадайку этого числа.

## Программа должна просит игрока угадать число. После того как игрок ввел число, программа должна выводит сообщение «Твоё число больше, чем нужно!» или «Твоё число меньше, чем нужно!». И заново просить угадать число. Когда игрок угадал вывести слово «Победа!»

Code:



Result:

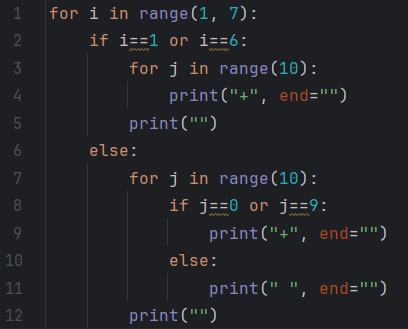


## Задание 7.

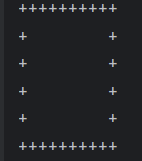
При помощи вложенных циклов постройте фигуры:

|  |  |
| --- | --- |
| №1  + + + + + + + + +  + +  + +  + +  + +  + + + + + + + + + | №2  + + + + + + + + +  + + +  + + + + + + + + +  + + +  + + + + + + + + +  + + +  + + + + + + + + + |

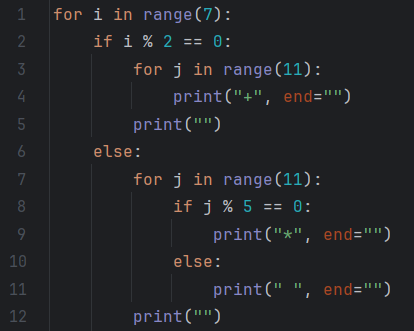
Code:



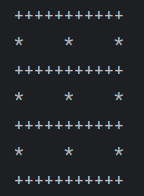
Result:



Code:



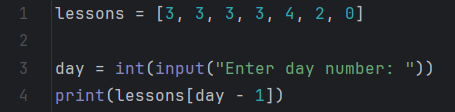
Result:



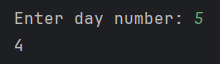
## Задание 8.

## Напишите программу, которая по номеру дня недели — целому числу от 1 до 7 — выдает в качестве результата количество занятий в вашей группе в соответствующий день.

Code:



Result:

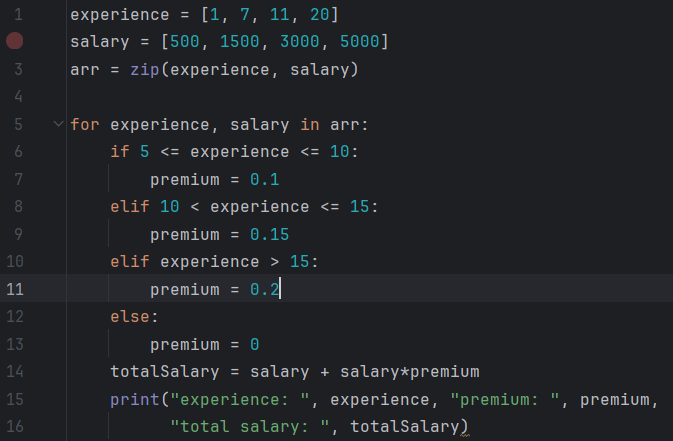


## Задание 9.

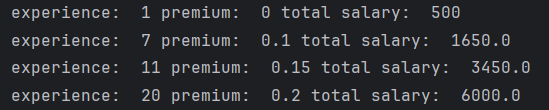
## В зависимости от стажа работы педагогам введена надбавка в размере: для работающих от 5 до 10 лет — 10%; для работающих от 10 до 15 лет — 15%; для работающих свыше 15 лет — 20%.

## Составьте программу, которая по заданному стажу работы и размеру оклада определит размер заработной платы.

Code:



Result:

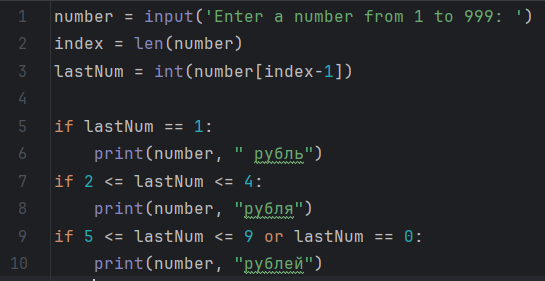


## Задание 10.

## Составить программу, которая после введенного с клавиатуры числа (в диапазоне от 1 до 999), обозначающего денежную величину, дописывает слово «рубль» в правильной форме.

## Например, 5 рублей, 21 рубль, 173 рубля.

Code:



Result:





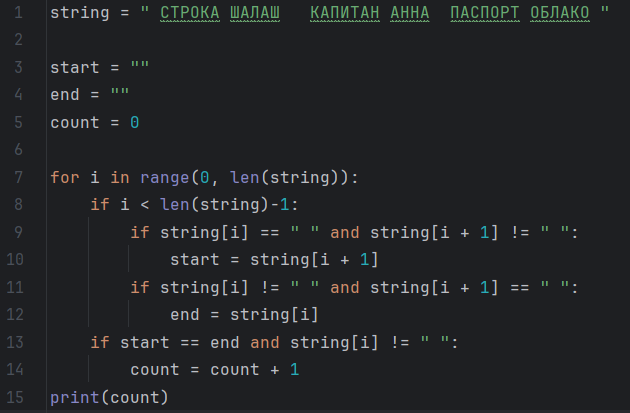


## + Лабораторная работа №3 (4ч). Методы обработки строк и списков. Словари и множества. Регулярные выражения.

**Задание 1 (строки).**

1. Дана строка, состоящая из русских слов, набранных заглавными буквами и разделенных пробелами (одним или несколькими). Найти количество слов, которые начинаются и заканчиваются одной и той же буквой.

Code:



Result:

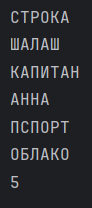


Дана строка, состоящая из русских слов, набранных заглавными буквами и разделенных пробелами (одним или несколькими). Найти количество слов, которые содержат хотя бы одну букву «А».

Code:

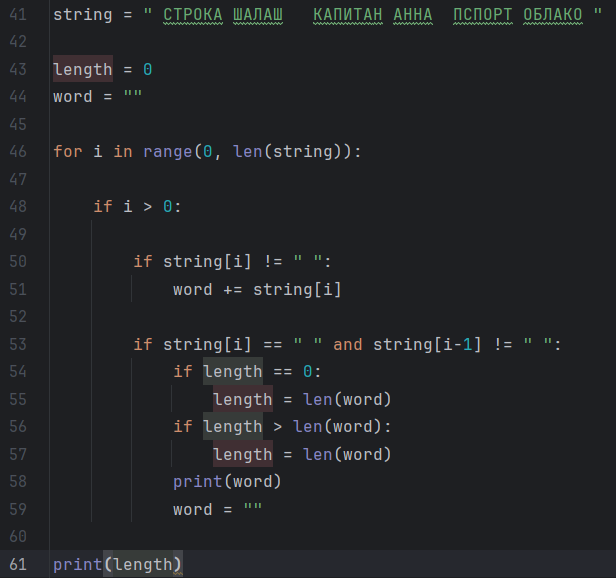


Result:

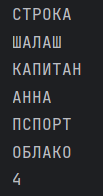


Дана строка, состоящая из русских слов, разделенных пробелами (одним или несколькими). Найти длину самого короткого слова.

Code:

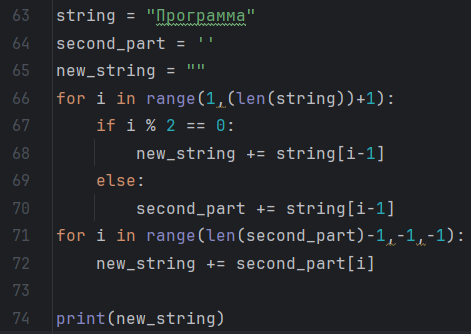


Result:



Дана строка-предложение. Зашифровать ее, поместив вначале все символы, расположенные на четных позициях строки, а затем, в обратном порядке, все символы,расположенные на нечетных позициях (например, строка «Программа» превратит-ся в «ргамамроП»).

Code:

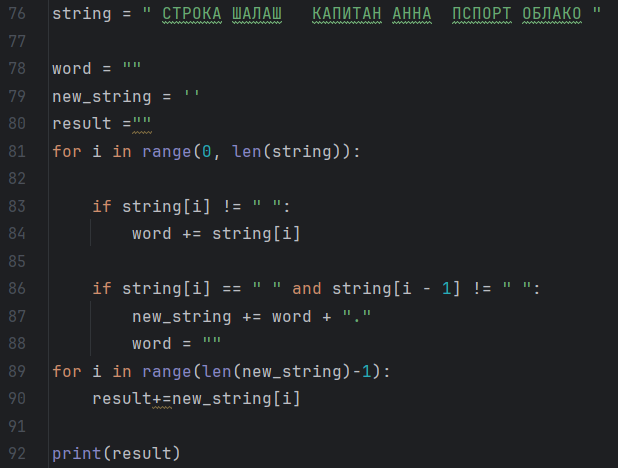


Result:



Дана строка, состоящая из русских слов, разделенных пробелами (одним или несколькими). Вывести строку, содержащую эти же слова, разделенные одним символом «.» (точка). В конце строки точку не ставить.

Result:

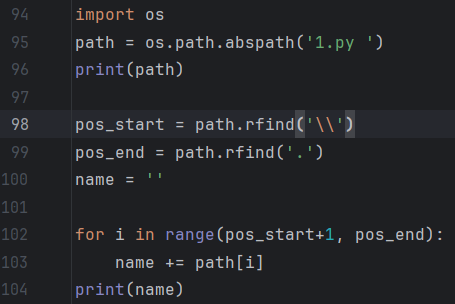


Result:



Дана строка, содержащая полное имя файла, то есть имя диска, список каталогов(путь), собственно имя и расширение (например, d:\ivanov\primer\prog.py). Выделить из этой строки имя файла (без расширения).

Code:

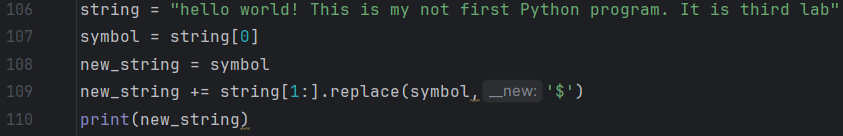


Result:



Получите строку из заданной, в которой все вхождения ее первого символа были изменены на '$', кроме самого первого символа.

Code:

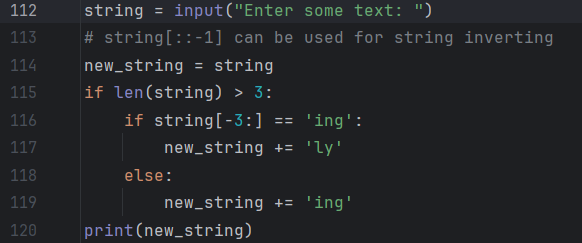


Result:



Добавьте 'ing' в конце заданной строки (длина должна быть не менее 3). Если данная строка уже заканчивается на 'ing', добавьте вместо этого 'ly'. Если длина заданной строки меньше 3, оставьте ее без изменений.

Code:



Result:

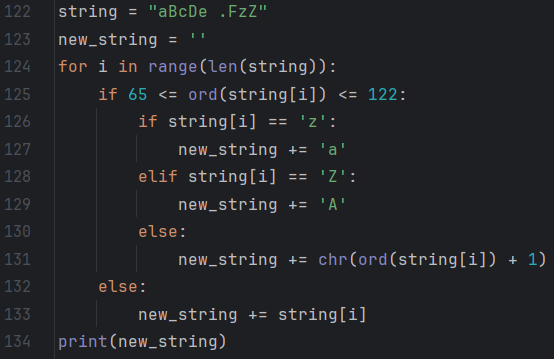






Дана строка-предложение. Зашифровать ее, выполнив циклическую замену каждой буквы на следующую за ней в алфавите и сохранив при этом регистр букв («A» перейдет в «B», «a» — в «b», «B» — в «D», «z» — в «a» и т. д.). Знаки препинания и пробелы не изменять.

Code:



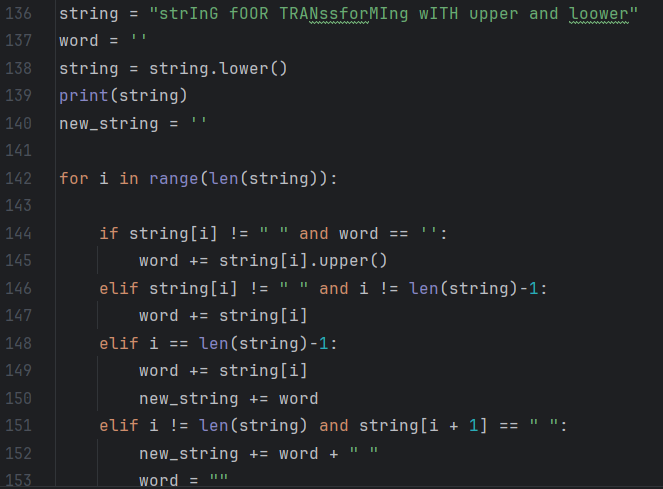
Result:

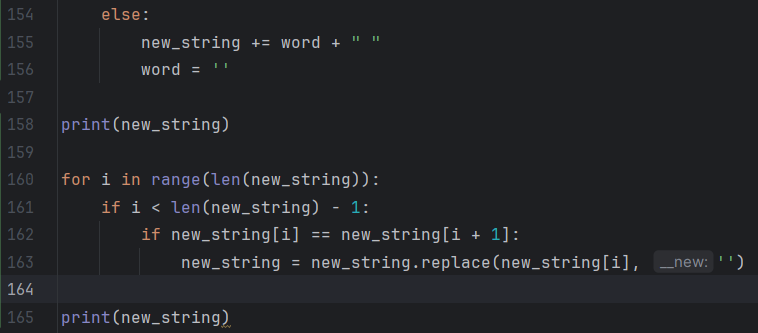


Преобразуйте строку-предложение так, чтобы каждое слово начиналось с большой буквы, а остальные буквы были в нижнем регистре.

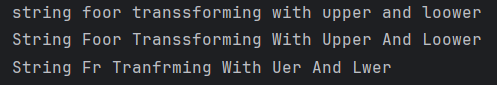
Удалите все дублирующиеся символы из заданной строки.

Code:





Result:



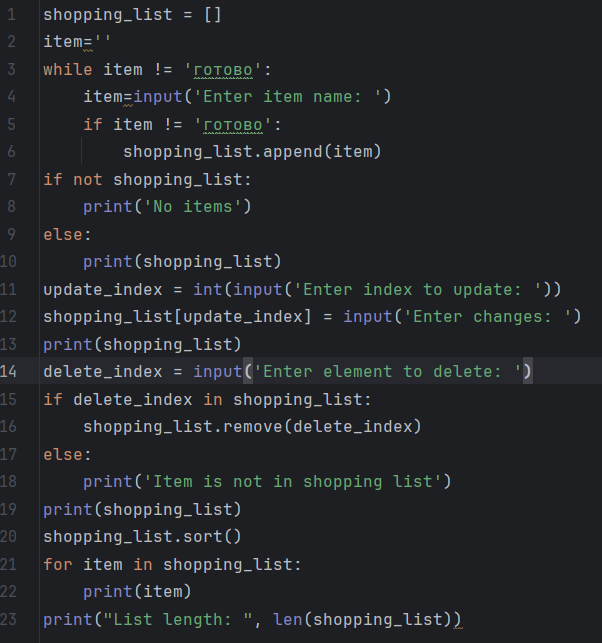
**Задание 2 (списки).**

Управление списком покупок.

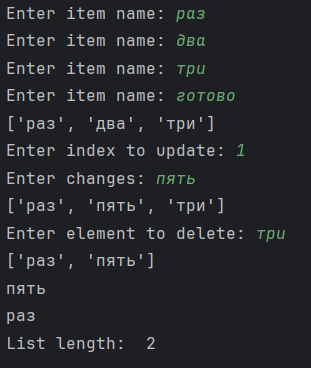
Выполнить ряд задач, используя различные методы, функции и циклы.

1. Создайте пустой список **shopping\_list**.
2. Запросите у пользователя ввод элементов списка покупок до тех пор, пока он не введёт слово "готово".
3. Добавьте каждый введенный элемент в список **shopping\_list**, за исключением слова "готово".
4. Выведите список **shopping\_list** на экран с помощью функции print().
5. Проверьте, пуст ли список. Если список пуст, выведите сообщение "Список покупок пуст!".
6. Предложите пользователю ввести индекс элемента списка, который он хочет обновить.
7. Запросите новое значение для этого элемента.
8. Обновите элемент списка с помощью присваивания.
9. Предложите пользователю ввести элемент списка, который он хочет удалить.
10. Если элемент присутствует в списке, удалите его с помощью метода **remove**(). Если элемент отсутствует, выведите сообщение об ошибке.
11. Отсортируйте список **shopping\_list** в алфавитном порядке с помощью функции **sorted**().
12. Выведите отсортированный список на экран при помощи цикла.
13. Подсчитайте количество элементов в списке **shopping\_list** с помощью функции **len**().

Code:



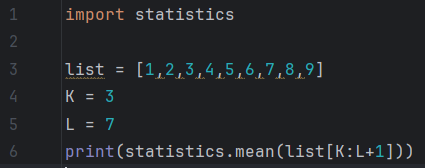
Result:



**Задание 3 (списки).**

1. Дан список, включающий N элементов, и целые числа K и L (1 ≤ K ≤ L ≤ N). Найти среднее арифметическое элементов списка с порядковыми номерами от K до L включительно.

Code:

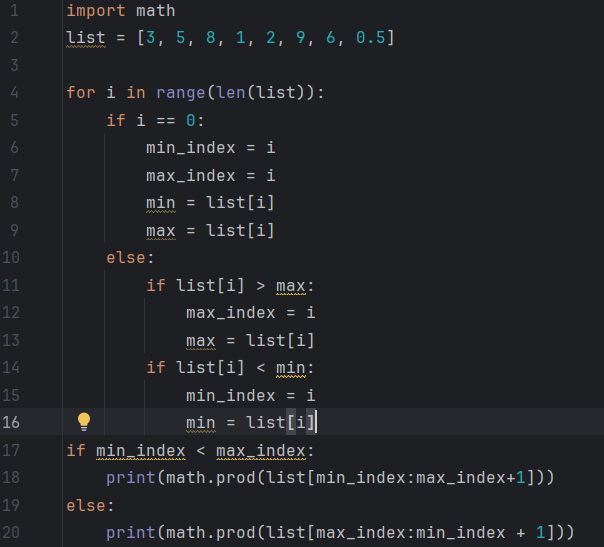


Result:



Дан список. Найти произведение всех элементов списка, расположенных между его минимальным и максимальным элементами, включая минимальный и максимальный элементы.

Code:

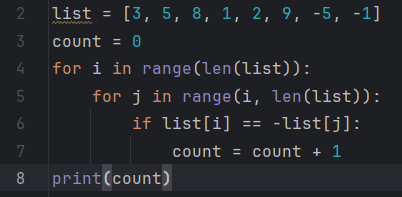


Result:



Дан список. Найти количество пар элементов списка, которые являются взаимно противоположными.

Code:

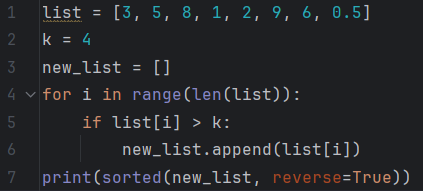


Result:



Дан список. Выбрать из списка все числа больше заданного числа k и упорядочить их по убыванию.

Code:

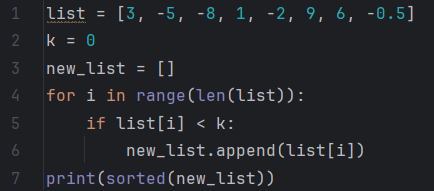


Result:



Дан список. Выбрать из списка все отрицательные числа и упорядочить их по возрастанию.

Code:

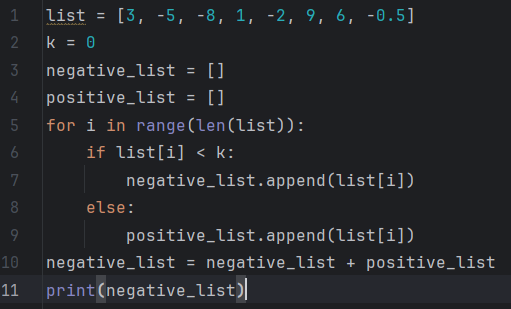


Result:



Дан список чисел. Отсортируйте список, переместив все отрицательные числа в начало списка, а положительные — в конец.

Code:

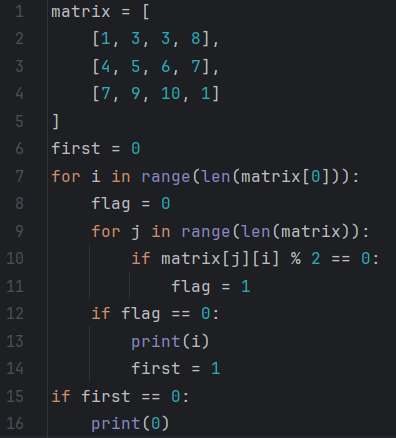


Result:



Дана целочисленная матрица размера M × N. Найти номер первого из ее столбцов,содержащих только нечетные числа. Если таких столбцов нет, то вывести 0.

Code:

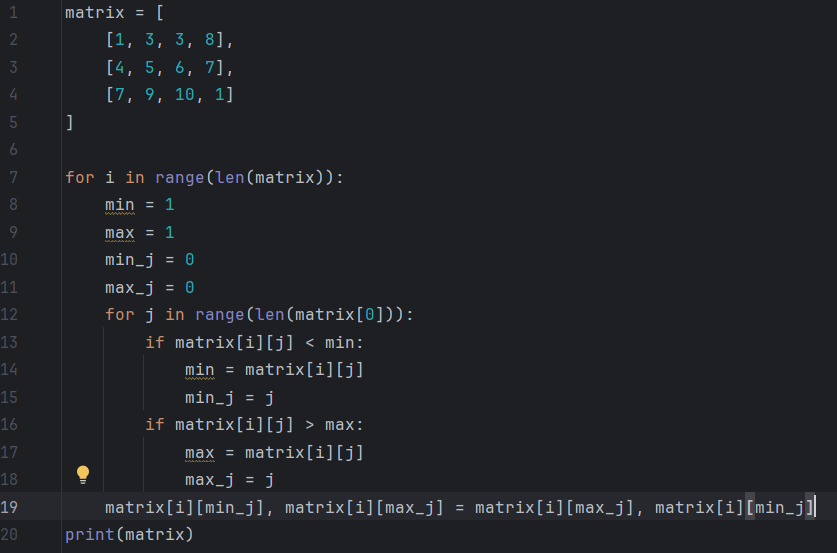


Result:



Дана матрица размера M × N. Преобразовать матрицу, поменяв местами минимальный и максимальный элемент в каждой строке

Code:

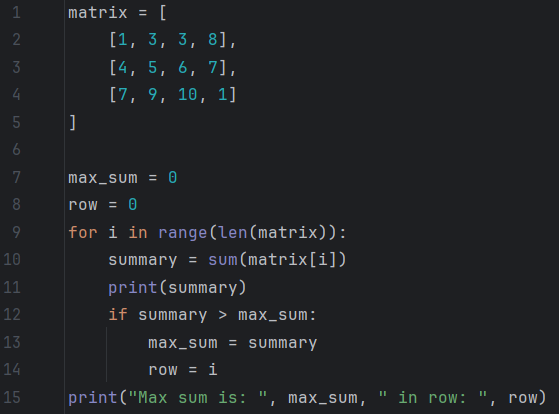


Result:

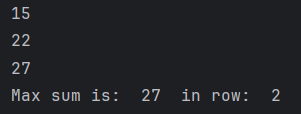


Дана матрица размера M × N. Найти номер ее строки с наибольшей суммой элементов и вывести данный номер, а также значение наибольшей суммы.

Code:



Result:



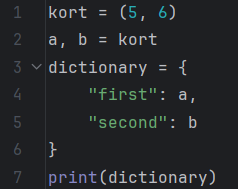
**Задание 4 (кортежи).**

1. Напишите программу:

а) для создания кортежа;

б) для преобразования кортежа в словарь.

Code:



Result:

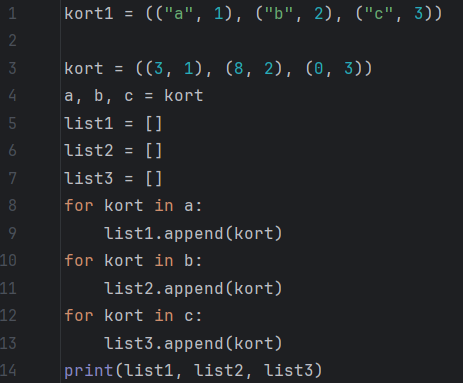


Напишите программу:

* 1. для создания кортежа с различными типами данных;

для распаковки списка кортежей в отдельные списки

code:



Result:

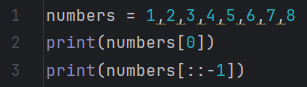


Напишите программу:

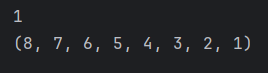
а) для создания кортежа с числами и выведите на экран первый элемент;

б) для того, чтобы перевернуть элементы кортежа в обратном порядке.

Code:



Result:

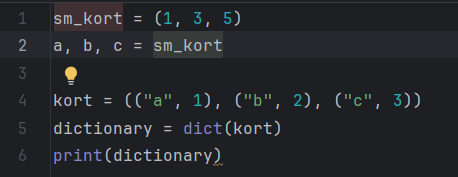


Напишите программу:

а) для распаковки кортежа в несколько переменных;

б) для преобразования списка кортежей в словарь.

Code:



Result:

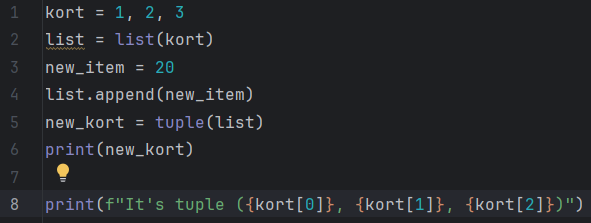


Напишите программу:

а) для добавления элемента в кортеж;

б) для печати кортежа со строковым форматированием. Например, задан кортеж: (100, 200, 300), на экран вывести: это кортеж (100, 200, 300).

Code:



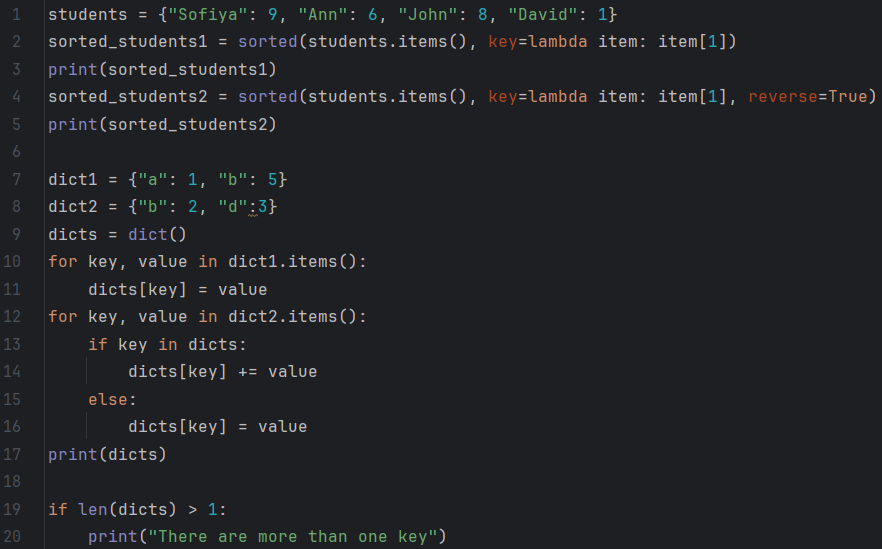
Result:



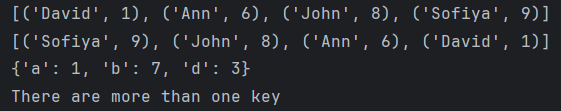
**Задание 5 (словари).**

1. Напишите программу:
   1. для сортировки (по возрастанию и убыванию) словаря по значению;
   2. для объединения двух словарей и суммирующую значения для общих
   3. для проверки наличия нескольких ключей в словаре.

Code:



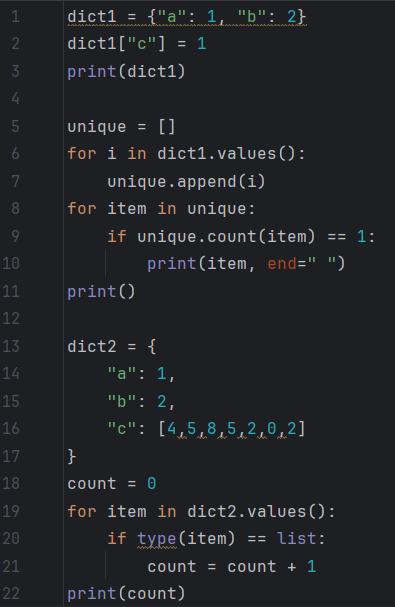
Result:



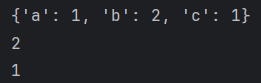
Напишите программу:

* 1. для добавления ключа в словарь;
  2. для печати всех уникальных значений в словаре;
  3. для подсчета количества элементов в значениях словаря, представляющих собой списки.

Code:



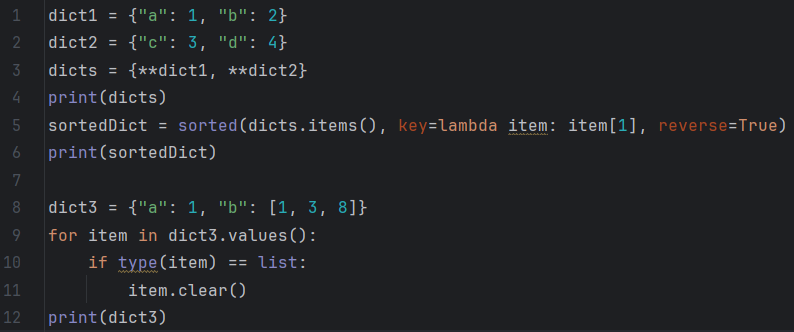
Result:



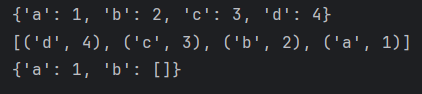
Напишите программу:

* 1. для объединения словарей в один новый словарь;
  2. для сортировки словаря по значению;
  3. чтобы очистить список значений в словаре, если значениями словаря являются списки.

Code:



Result:



Напишите программу:

a. чтобы проверить, существует ли данный ключ в словаре;

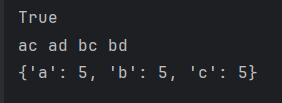
b. для создания и отображения всех комбинаций букв, выбирая каждую букву из другого ключа в словаре, например, задан словарь {'1': ['a', 'b'], '2': ['c', 'd']}, результат: ac, ad, bc, bd;

c. для замены значений словаря их средним значением.

Code:



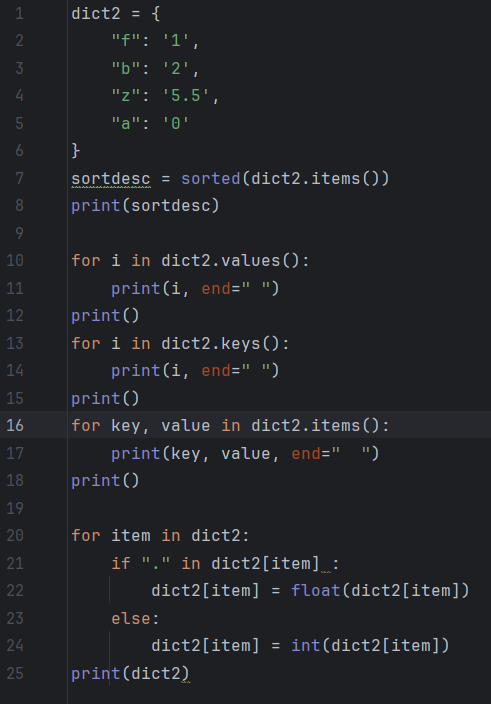
Result:



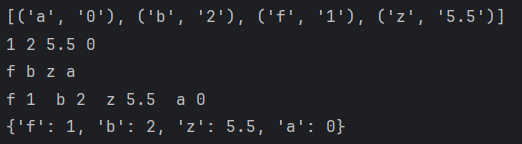
Напишите программу:

* 1. для сортировки заданного словаря по ключу;
  2. чтобы получить ключ, значение и элемент в словаре;
  3. для преобразования строковых значений данного словаря в целочисленные (или числа с плавающей точкой).

Code:



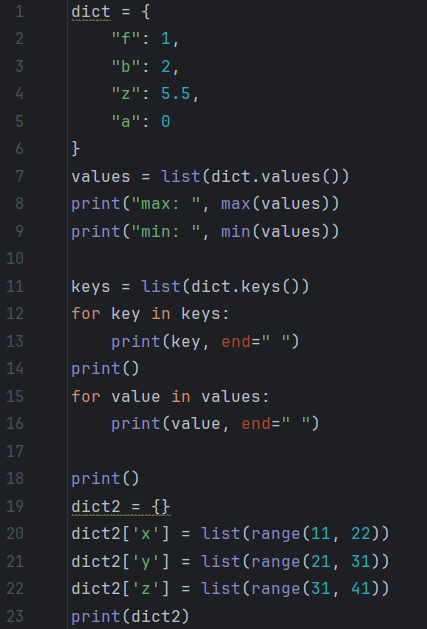
Result:



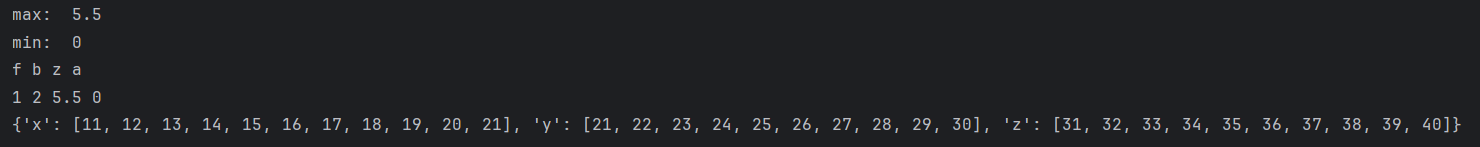
Напишите программу:

* 1. чтобы получить максимальное и минимальное значение в словаре;
  2. для печати в отдельные строки ключей и значений словаря;
  3. для создания словаря ключей x, y и z, где каждый ключ имеет значение списка из 11–20,21–30 и 31–40 соответственно.

Code:

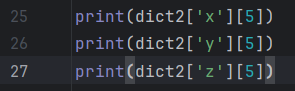


Result:



Получите доступ к пятому значению каждого ключа из словаря.

Code:

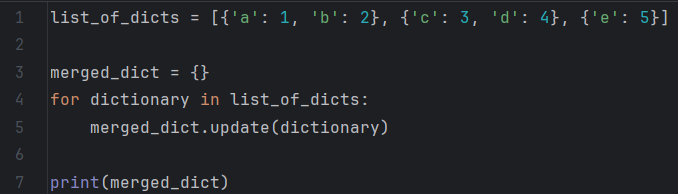


Result:



Напишите программу, которая принимает список словарей и объединяет их в один словарь

Code:

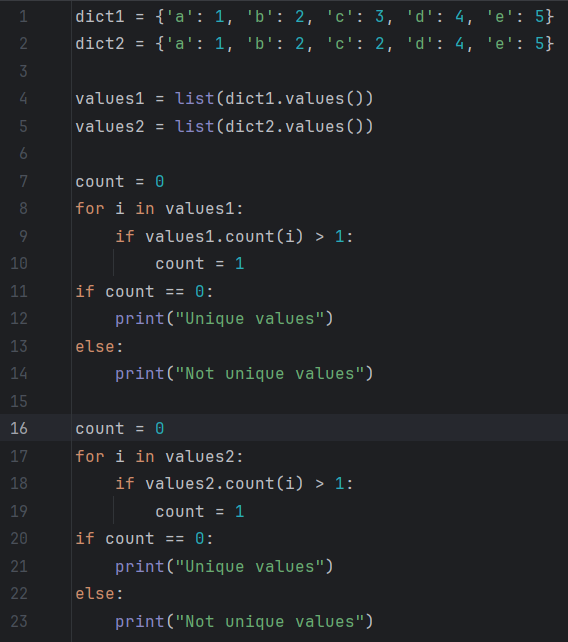


Result:



Напишите программу, которая проверяет, все ли значения в словаре являются уникальными.

Code:



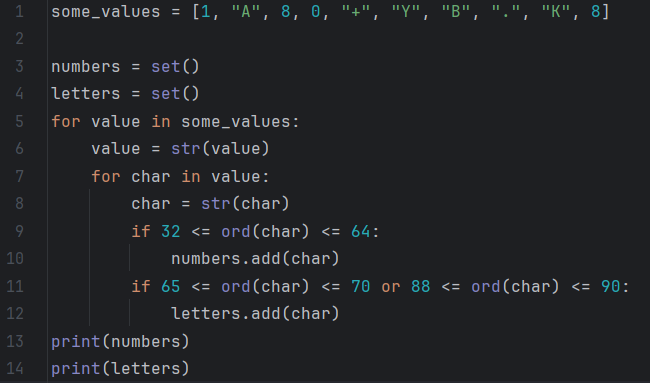
Result:



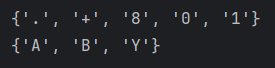
**Задание 6 (множества).**

1. Дана непустая последовательность символов. Построить и напечатать множества,элементами которых являются встречающиеся в последовательности: а) цифры от«0» до «9» и знаки арифметических операций; б) буквы от «A» до «F» и от «X»до «Z».

Code:

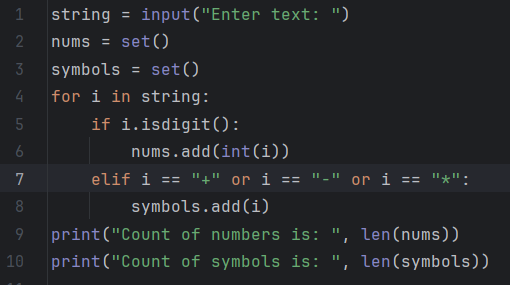


Result:

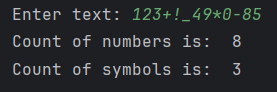


Используя множества, подсчитать общее количество цифр и знаков «+», «–», «\*»в строке, введенной с клавиатуры

Code:



Result:



Используя множества, вывести различные буквы трёх предложений, то есть такие,какие есть только в одном из них.

Code:

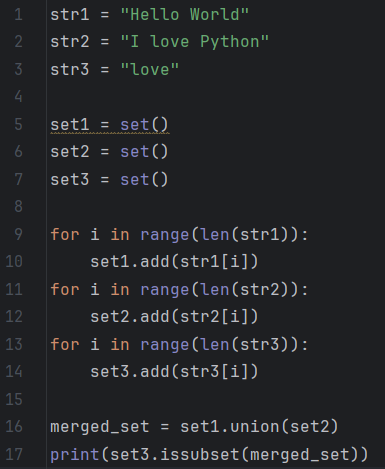


Result:



Даны три строки. Используя множества, определить, можно ли из символов первых двух строк получить третью строку.

Code:

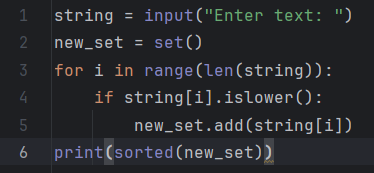


Result:

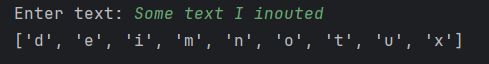


Используя множества, напечатать по одному разу в алфавитном порядке все строчные русские гласные буквы, входящие в заданный текст.

Code:

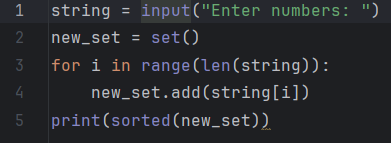


Result:



Используя множества, напечатать в возрастающем порядке все цифры, входящие в десятичную запись данного десятичного числа.

Code:

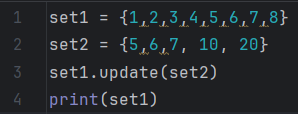


Result:



Даны два множества. Используя множества, найти объединение данных множеств.

Code:



Result:

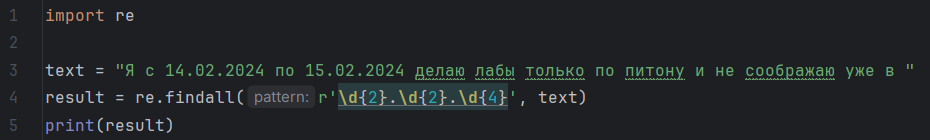


**Задание 7 (регулярные выражения).**

1. **Поиск дат:**

* Создайте строку text, содержащую произвольный текст, в котором присутствуют даты в формате "дд.мм.гггг".
* Используя модуль re, выполните поиск всех дат в тексте с помощью регулярного выражения.
* Выведите на экран все найденные даты.

Code:



Result:



1. **Проверка валидности email-адреса:**

* Предложите пользователю ввести email-адрес.
* Используя модуль re, проверьте введенный адрес на соответствие формату email с помощью регулярного выражения.
* Выведите сообщение о том, является ли введенный адрес валидным или нет.

Code:



Result:

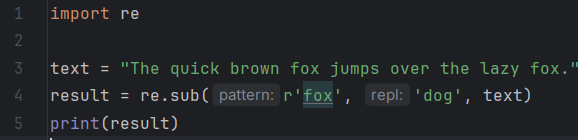




**Замена слов:**

* Создайте строку text, содержащую произвольный текст, в котором присутствуют определенные слова, которые нужно заменить.
* Используя модуль re, выполните замену определенных слов в тексте с помощью регулярного выражения.
* Выведите измененный текст на экран.

Code:



Result:



## Лабораторная работа №4 (4ч). Пользовательские функции. Lambda-выражения. Генераторы.

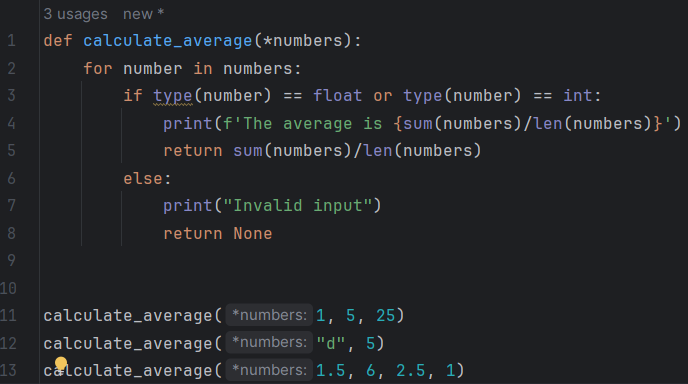
**Задание 1.**

Создайте функцию **calculate\_average**, которая принимает произвольное количество аргументов типа float или int.

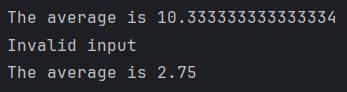
Внутри функции вычислите среднее значение всех переданных аргументов. Верните полученное среднее значение из функции.

Протестируйте функцию, передавая различное количество числовых аргументов, и выведите результат на экран.

Code:



Result:



**Задание 2.**

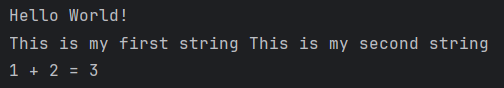
Создайте функцию **merge\_strings**, которая принимает две строки в качестве аргументов.

Внутри функции объедините две строки в одну и верните полученную строку.

Протестируйте функцию, передавая различные строки, и выведите результат на экран.

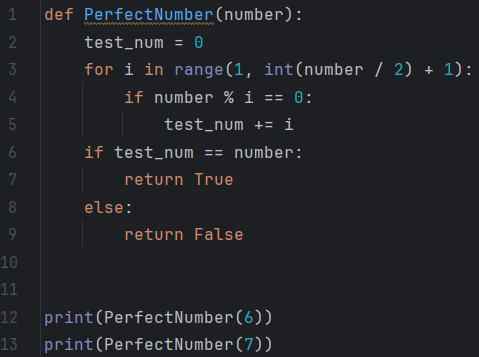
Code:  


Result:



**Задание 3.**

Описать функцию **PerfectNumber**, определяющую, является ли заданное положительное число совершенным числом. Совершенным числом называется число, равное сумме своих делителей (кроме самого числа).

Code:  


Result:

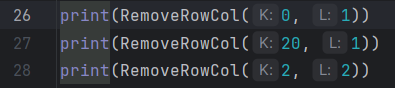


**Задание 4.**

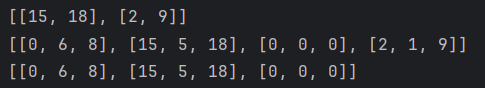
Описать функцию **RemoveRowCol**, удаляющую из матрицы A размера M × N строкуи столбец, которые содержат элемент AK, L (предполагается, что M > 1 и N > 1; еслиK > M или L > N, то матрица не изменяется).

Code:





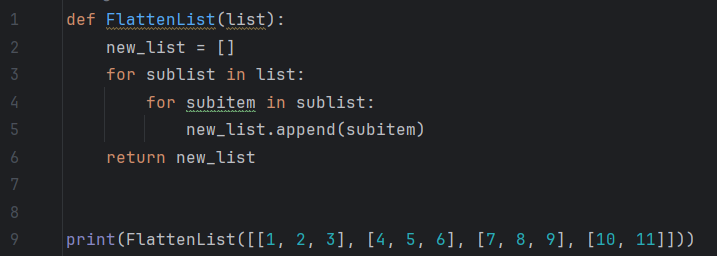
Result:



**Задание 5.**

Описать функцию **FlattenList**, которая принимает вложенный список и возвращает одноуровневый список, содержащий все элементы из вложенных списков.

Code:



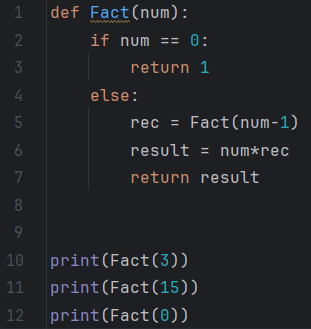
Result:



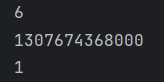
**Задание 6.**

Описать рекурсивные функции **Fact** и **Fact2**, вычисляющие значения факториала N!и двойного факториала N!! соответственно (N > 0 — параметр целого типа).

Code:



Result:



Code:



Result:



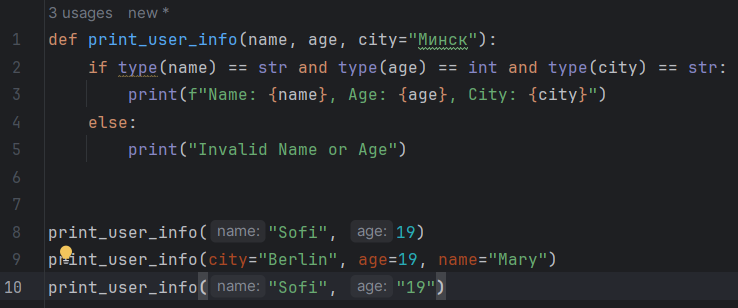
**Задание 7.**

Создайте функцию **print\_user\_info**, которая принимает три аргумента: name типа str, age типа int и city типа str ( по умолчанию - Минск).

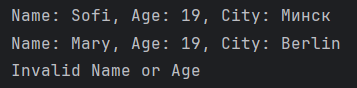
Внутри функции выведите информацию о пользователе в формате "Имя: [name], Возраст: [age], Город: [city]".

Протестируйте функцию, передавая различные значения для аргументов, и выведите результат на экран.

Code:



Result:



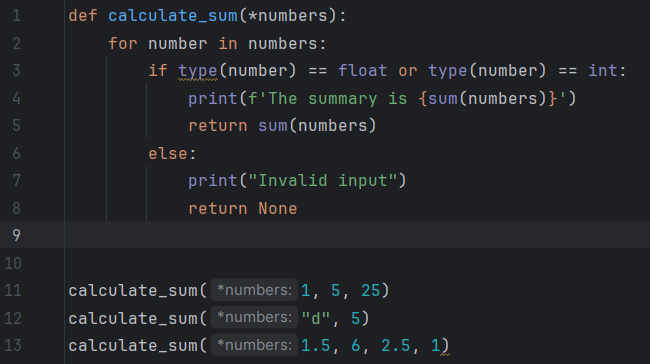
**Задание 8.**

Создайте функцию **calculate\_sum**, которая принимает произвольное количество аргументов типа **int или float**.

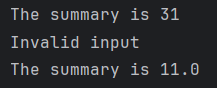
Внутри функции вычислите сумму всех переданных аргументов.

Верните полученную сумму из функции.

Протестируйте функцию, передавая различное количество числовых аргументов, и выведите результат на экран.

Code:  


Resulr:

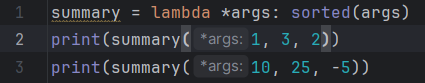


**Задание 9.**

Создайте список **numbers**, содержащий несколько целых чисел. Используя **lambda**-выражение, отсортируйте список в порядке возрастания или убывания.

Выведите отсортированный список на экран.

Code:

D

Result:



**Задание 10.**

Создайте список **names**, содержащий несколько строк (имен).

Используя **lambda**-выражение, отфильтруйте список, оставив только имена, начинающиеся с определенной буквы.

Выведите отфильтрованный список на экран.

Code:



Result:

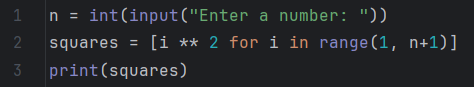


**Задание 11.**

Попросите пользователя ввести целое число n.

Используя генератор списка, создайте список, содержащий квадраты всех чисел от 1 до n.

Выведите полученный список на экран.



Result:

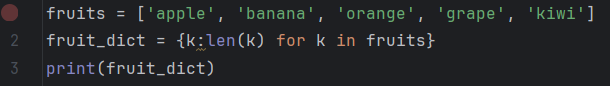


**Задание 12.**

Создайте список **fruits**, содержащий несколько строк (имен фруктов).

Используя генератор словаря, создайте словарь, где ключами будут элементы списка **fruits**, а значениями будут их длины (количество символов).

Выведите полученный словарь на экран.

Code:  


Result:



## + Лабораторная работа №5 (2ч). Подключение и установка модулей. Создание пользовательских модулей.

**Задание 1.**

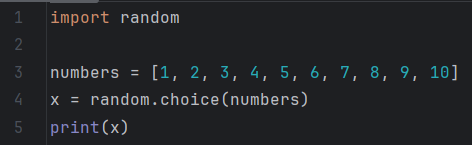
Импортируйте модуль **random** в свою программу.

Используйте функцию **random.choice()** для выбора случайного элемента из списка.

Создайте список с несколькими именами и используйте функцию **random.choice()** для выбора случайного имени из списка.

Выведите выбранное случайное имя на экран.

Code:



Result:



Задание на создание пользовательского модуля:

**Задание 2.**

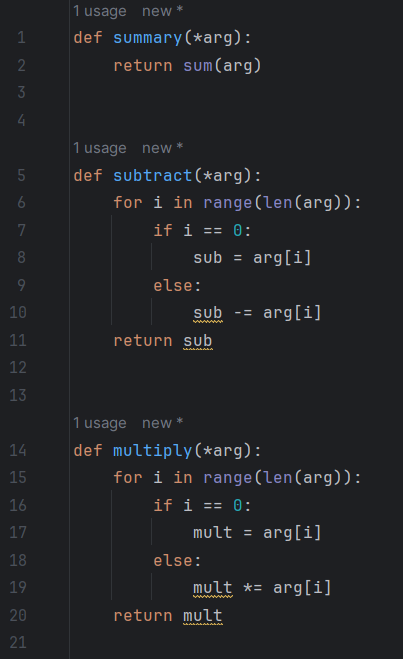
Создайте новый файл с расширением .py и назовите его **calculator.py.**

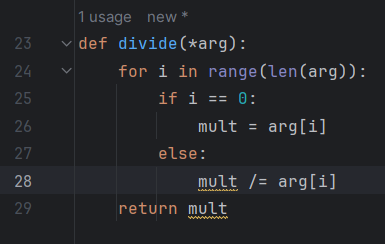
В файле **calculator.py** определите функции для выполнения простых математических операций, таких как сложение, вычитание, умножение и деление.

В основной программе импортируйте модуль **calculator** и используйте его функции для выполнения различных математических операций.

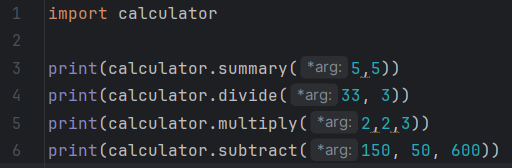
Проверьте работу пользовательского модуля, вызывая функции и выводя результаты на экран.

File calculator.py:

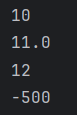




Main code:



Result:



## Задание 3.

Написать модуль **lists**, в котором содержится 5 функций:

1) Функция **randomList**, принимающую 1 аргумент n - длина списка. Которая возвращает список данной длины, заполненной случайными числами от -99 до 99.

2) Функция **randomMatrix**, принимающую 1 аргумент n – длина двумерного списка NxN. Которая возвращает двумерный список данной длины, заполненный случайными числами от 0 до 9.

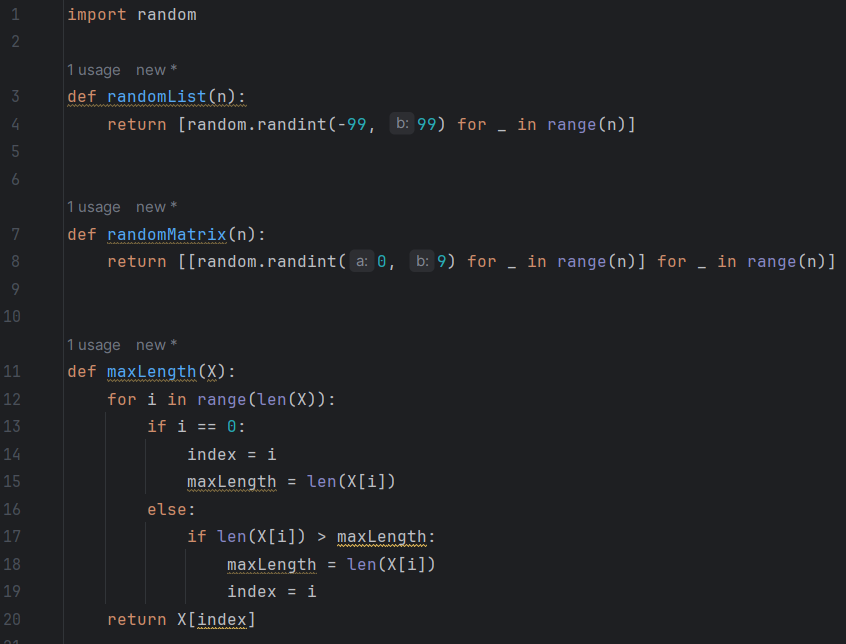
3) Функция **maxLength**, принимает 1 аргумент X – список, состоящий из слов. Функция должна определить самое длинное слова в списке. Функция должна вернуть данное слова.

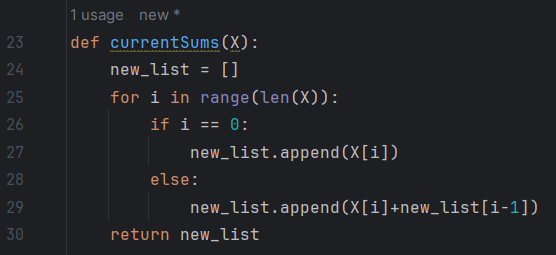
4) Функция **currentSums**, принимающая 1 аргумент X – список чисел. Функция возвращает новый массив из такого же числа элементов, в котором на каждой позиции будет находиться сумма элементов списка X до этой позиции включительно.

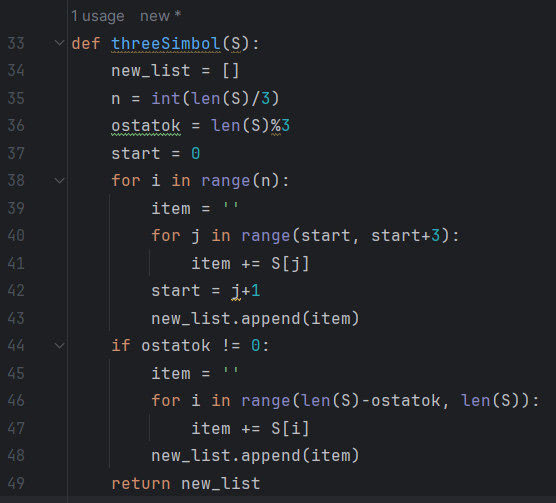
5) Функция **threeSimbol**, принимающая 1 аргумент S – предложение. Функция создает список, элементы которого будут состоять из строк, каждый элемент состоит из 3 последовательный символов предложения.

Подключите модуль с своему файлу и проверьте как работают функции.

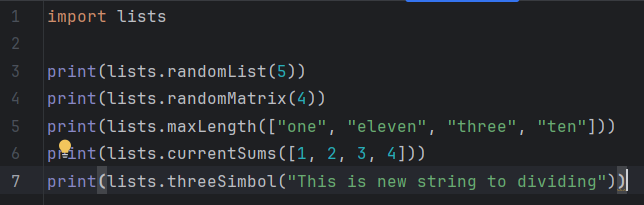
File lists.py:



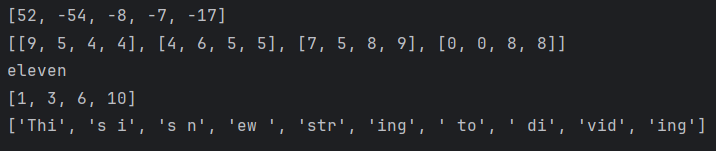




Main code:



Result:



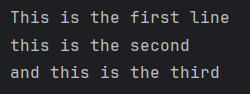
## + Лабораторная работа №6 (4ч). Операции с файловой системой. Практическое применение модулей pickle и shelve.

**Задание 1.**

Создайте текстовый файл **example.txt** и запишите в него несколько строк текста.

Откройте файл **example.txt** в режиме чтения и выведите его содержимое на экран.

Закройте файл.

Code:  
  
result:  


## Задание 2.

Создайте файл с некоторым содержимым, например, data.txt.

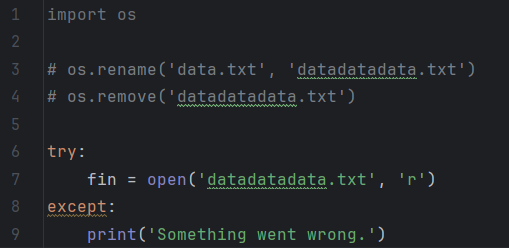
Импортируйте модуль os.

Измените название файла data.txt на **datadatadata.txt**,

Используя удалите файл **datadatadata.txt** с диска.

Проверьте, что файл успешно удален, попытавшись открыть его в режиме чтения.

Code:



Result:



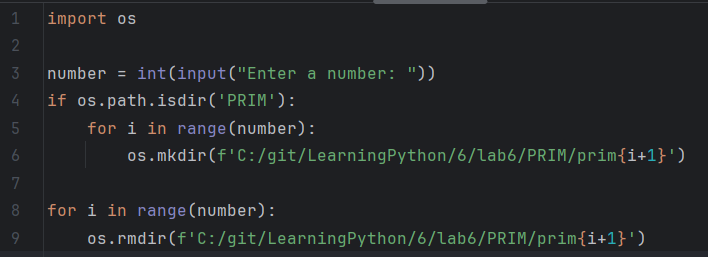
## Задание 3.

Напишите программу, которая спрашивает сколько папок надо создать = N.

После этого создает в папке PRIM нужное количество папок с именами **prim1, prim2, … primN**.

Удалите папки с именами **prim2** и **prim4**.

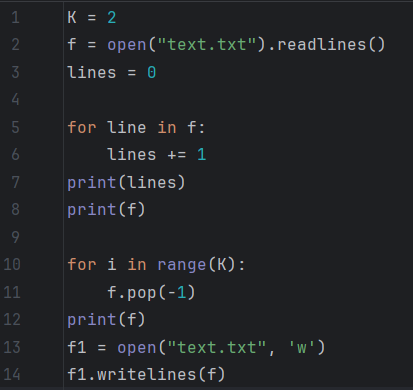
Code:



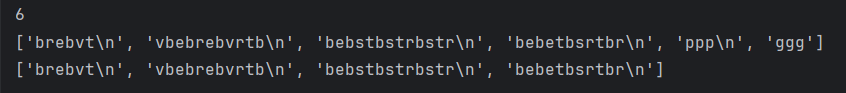
## Задание 4.

1. Дано целое число K (0 < K < 10) и текстовый файл, содержащий более K строк.Удалить из файла последние K строк.

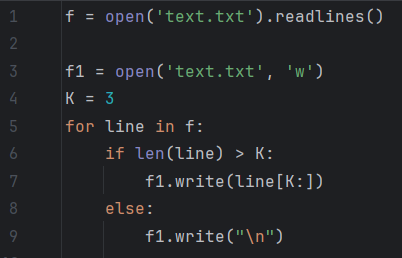
Code:



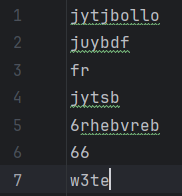
Result:



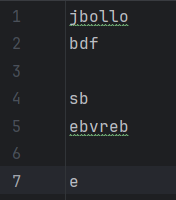
Дано целое число K и текстовый файл. Удалить из каждой строки файла первые K Символов (если длина строки меньше K, то удалить из нее все символы).

Code:  


File before:

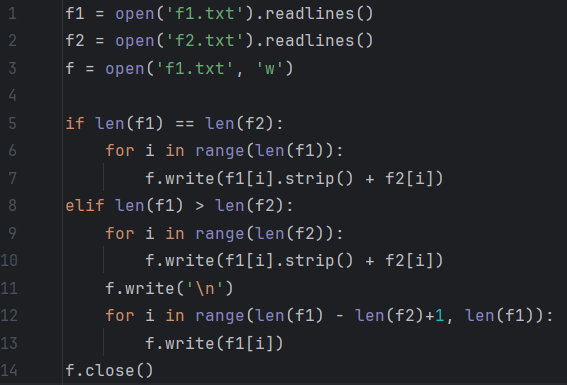


File after:

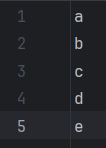


Даны два текстовых файла. Добавить в конец каждой строки первого файла соответствующую строку второго файла. Если второй файл короче первого, то остав-шиеся строки первого файла не изменять

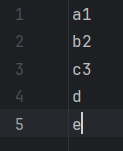
Code:



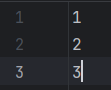
F1.txt before:



F1.txt after:

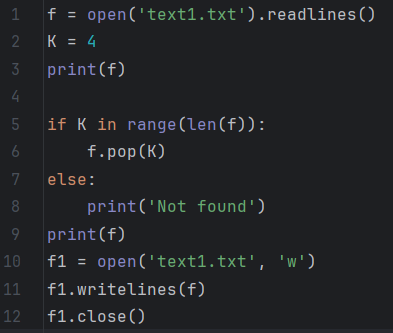


F2.txt:

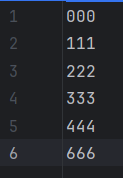


Дано целое число K и текстовый файл. Удалить из файла строку с номером K. Если Строки с таким номером нет, то оставить файл без изменений.

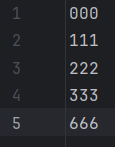
Code:



File before:

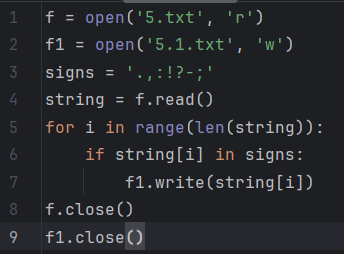


File.after:



Дан текстовый файл. Создать символьный файл, содержащий все знаки препинания, встретившиеся в текстовом файле (в том же порядке).

Code:



File 5.txt:



File 5.1.txt:



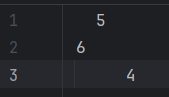
Дан текстовый файл, каждая строка которого изображает целое число, дополненное слева и справа несколькими пробелами. Вывести количество этих чисел и их сум-му.

Code:



Result:  

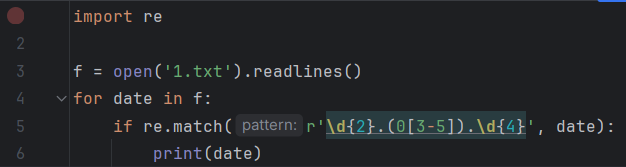

File:



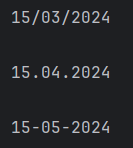
## Задание 5.

1. Дан файл, содержащий различные даты. Каждая дата — это число, месяц и год.Найти все весенние даты.

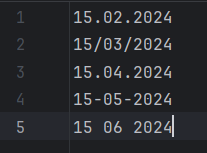
Code:



Result:

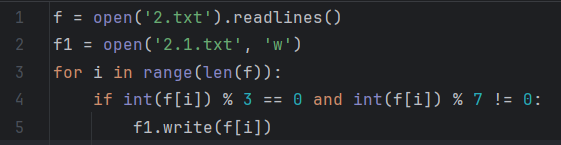


File 1.txt:

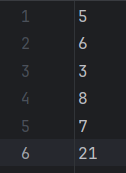


Дан файл f, компоненты которого являются целыми числами. Получить в файле g все компоненты файла f, делящиеся на 3 и не делящиеся на 7.

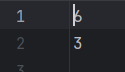
Code:



File 2.txt:

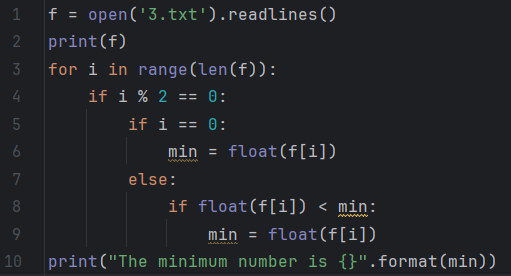


File 2.1.txt:



Дан файл f, компоненты которого являются действительными числами. Найти наименьшее из значений компонент с четными номерами.

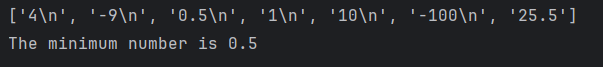
Code:



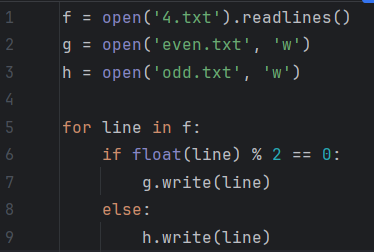
File:



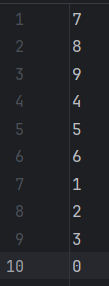
Result:

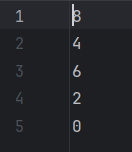


Записать в файл g все четные числа файла f, а в файл h все нечётные. Порядок следования чисел сохраняется.

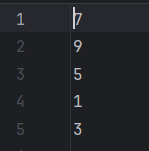
Code:  


File 4.txt:



File even.txt:  


File odd.txt:



**Задание 4.**

Создайте словарь, содержащий некоторые данные, например, информацию о пользователе.

Импортируйте модуль **pickle**.

С помощью функции **pickle.dump()** сериализуйте словарь в файл с расширением .**pickle**.

Затем считайте содержимое файла обратно с помощью функции **pickle.load()** и выведите его на экран.

Code:



Result:



**Задание 5.**

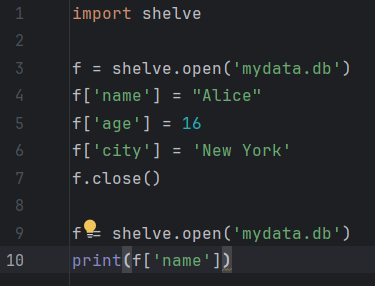
Импортируйте модуль **shelve**.

Создайте файл базы данных **mydata.db** с помощью функции **shelve.open()**.

Добавьте в базу данных несколько записей, используя ключи и значения. Закройте базу данных.

Вновь откройте базу данных **mydata.db** и получите значение по ключу. Выведите полученное значение на экран.

Code:



Result:



## + Лабораторная работа №7 (2ч). База данных SQLite3. Передача и получение данных.

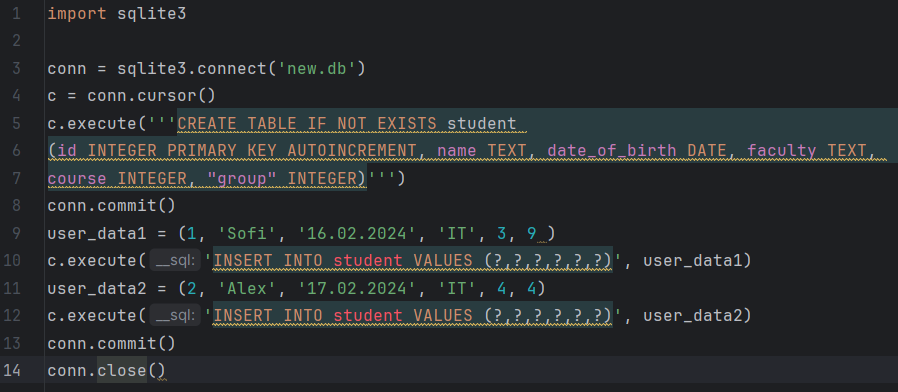
**Задание 1.**

Создайте базу данных SQLite3 с помощью модуля sqlite3.

Создайте таблицу **student** в базе данных с несколькими столбцами|: "id", "name", "date of birth", "faculty", "course", "group ".

Вставьте несколько записей в таблицу, используя оператор INSERT INTO.

Сохраните изменения в базе данных и закройте соединение.

Code:  


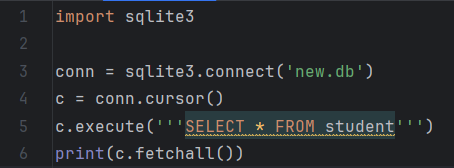
**Задание 2.**

Подключитесь к существующей базе данных SQLite3.

Используя оператор SELECT, выполните запрос к таблице и получите все записи.

Выведите полученные данные на экран.

Code:



Result:



**Задание 3.**

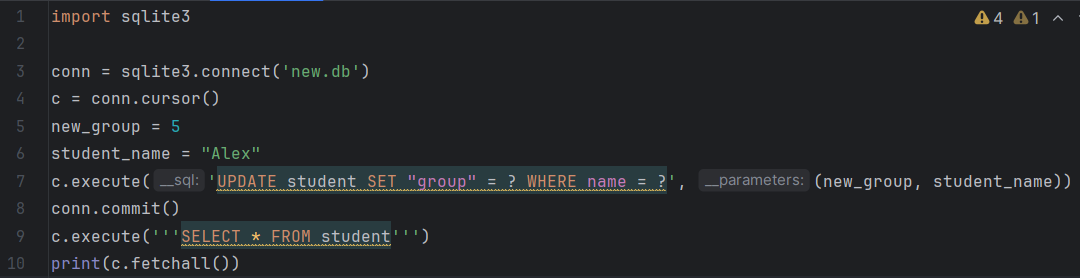
Подключитесь к существующей базе данных SQLite3.

Используя оператор UPDATE, обновите данные в таблице.

Например, можно изменить курс определенной записи или изменить группу.

Сохраните изменения в базе данных и закройте соединение.

Code:



Result:



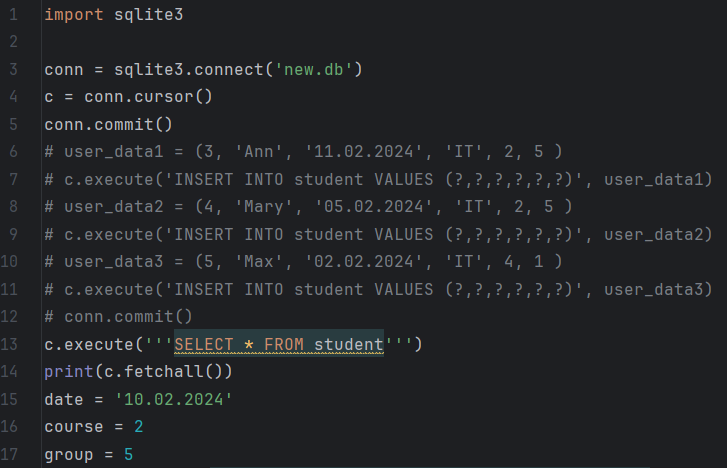
**Задание 4.**

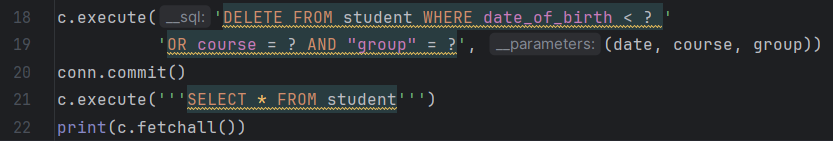
Подключитесь к существующей базе данных SQLite3.

Используя оператор DELETE, удалите определенные записи из таблицы:

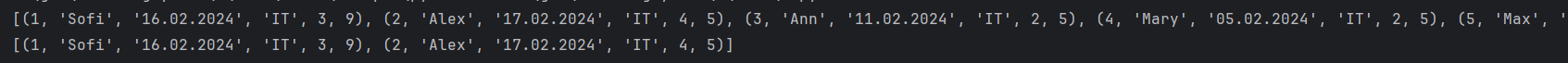
* всех студентом младше 2000 года рождения
* всех студентов 2 курса 5 группы

Сохраните изменения в базе данных и закройте соединение.

Code:  




Result:



## + Лабораторная работа №8 (4ч). Классы и объекты. Практическое применение основных принципов ООП.

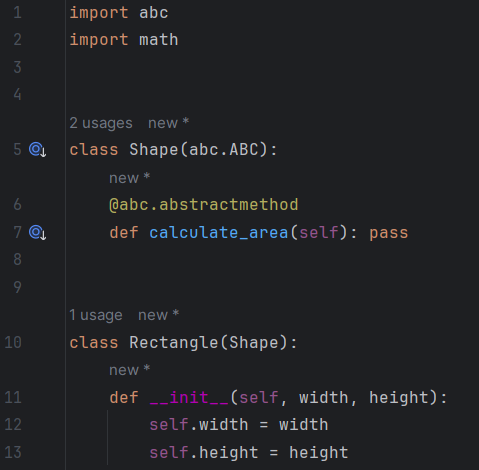
**Задание 1.**

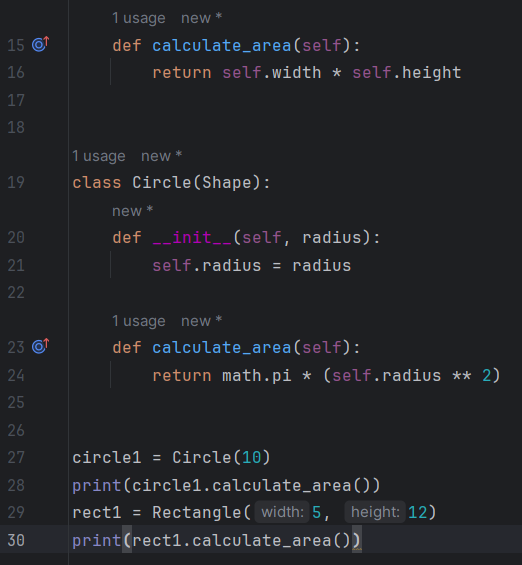
Создайте абстрактный класс **Shape** с методом **calculate\_area**().

Определите два класса, **Circle** и **Rectangle**, которые наследуются от **Shape**.

В каждом из этих классов реализуйте метод **calculate\_area**(), чтобы он возвращал площадь соответствующей фигуры.

Создайте объекты **Circle и Rectangle** и вызовите их методы **calculate\_area**().

Code:  




Result:



**Задание 2 (полиморфизм).**

Создайте класс **Animal** с методом **make\_sound**().

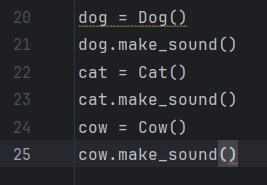
Определите классы **Dog**, **Cat** и **Cow**, которые наследуются от **Animal**.

В каждом из этих классов переопределите метод **make\_sound**(), чтобы он возвращал соответствующий звук животного.

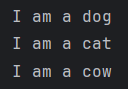
Создайте список объектов разных типов (например, **Dog,** **Cat** и **Cow**) и вызовите для каждого объекта метод **make\_sound**().

Code:





Result:



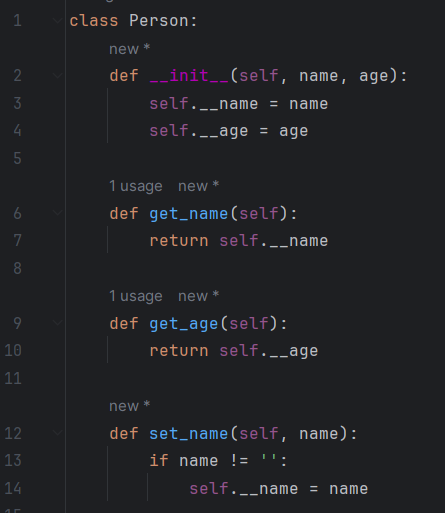
**Задание 3 (инкапсуляция).**

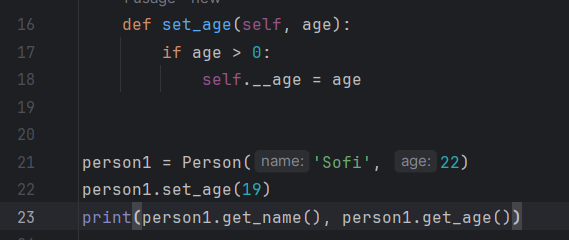
Создайте класс **Person** с закрытыми атрибутами **name** и **age**.

Определите методы **get\_name(), set\_name(), get\_age() и set\_age()**, чтобы получить и установить значения атрибутов.

Создайте объект класса **Person** и используйте методы для получения и установки значений атрибутов.

Code:





Result:



**Задание 4 (наследования).**

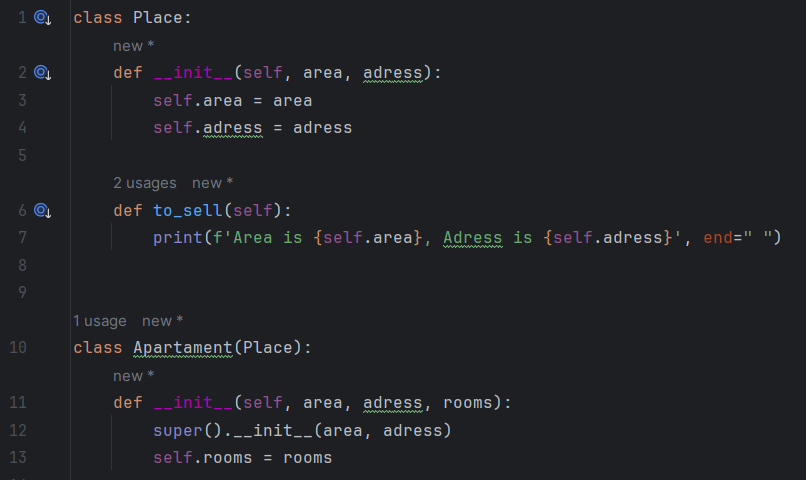
Создайте класс **Place** с атрибутами **area** и **address** и методом **to\_sell(),** который выводит сообщение о продаже данной недвижимости с указанием площади и адреса .

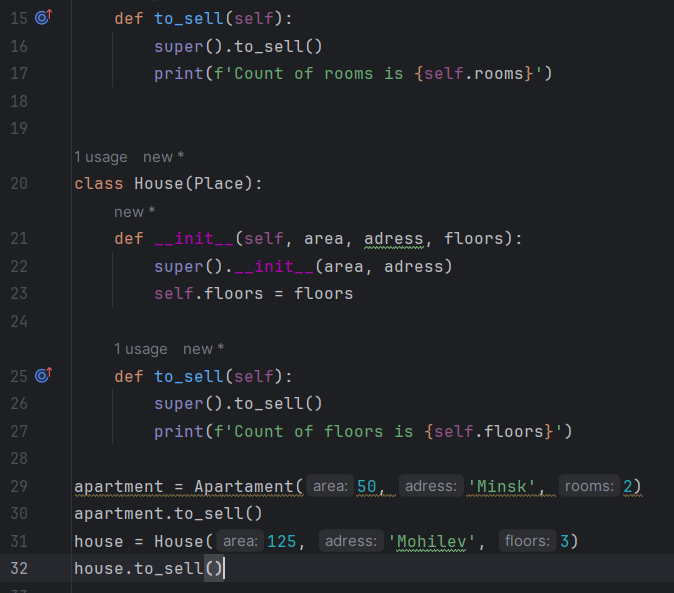
Определите класс **Apartment** , который наследуется от **Place** и добавляет атрибут **rooms**.

Определите класс **House**, который также наследуется от **Place** и добавляет атрибут **floors**.

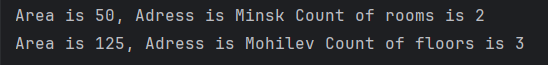
Создайте объекты **Apartment** и **House** и вызовите их методы.

Code:





Result:



**Задание 5 (использование композиции).**

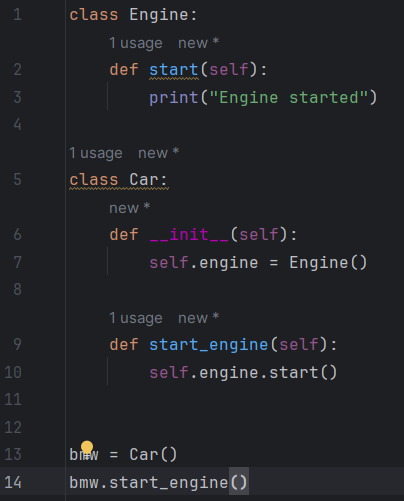
Создайте класс **Engine** с методом **start**(), который выводит сообщение о запуске двигателя.

Создайте класс **Car** с атрибутом **engine**, который будет экземпляром класса **Engine**.

Определите метод **start\_engine**(), который будет вызывать метод **start**() у объекта engine.

Создайте объект класса **Car** и вызовите метод **start\_engine**().

Code:



Result:



**Задание 6 (принцип полного объекта).**

## Создайте класс Animal с атрибутами name и age.

## Определите класс Zoo с атрибутом animals, который будет списком объектов класса Animal.

## Реализуйте методы add\_animal() и remove\_animal() для добавления и удаления животных из списка.

## Создайте объект класса Zoo и добавьте несколько объектов класса Animal. Затем вызовите методы add\_animal() и remove\_animal().

Code:



Result:



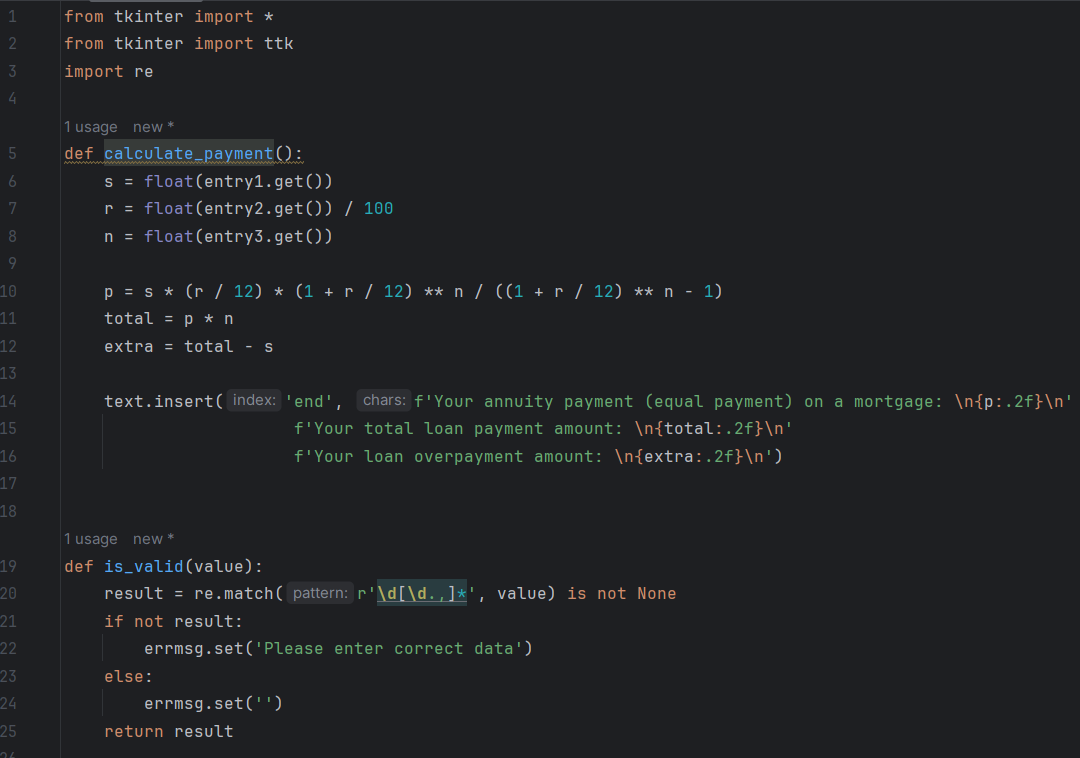
## + Лабораторная работа №9 (2ч). Tkinter. Разработка графического интерфейса приложения.

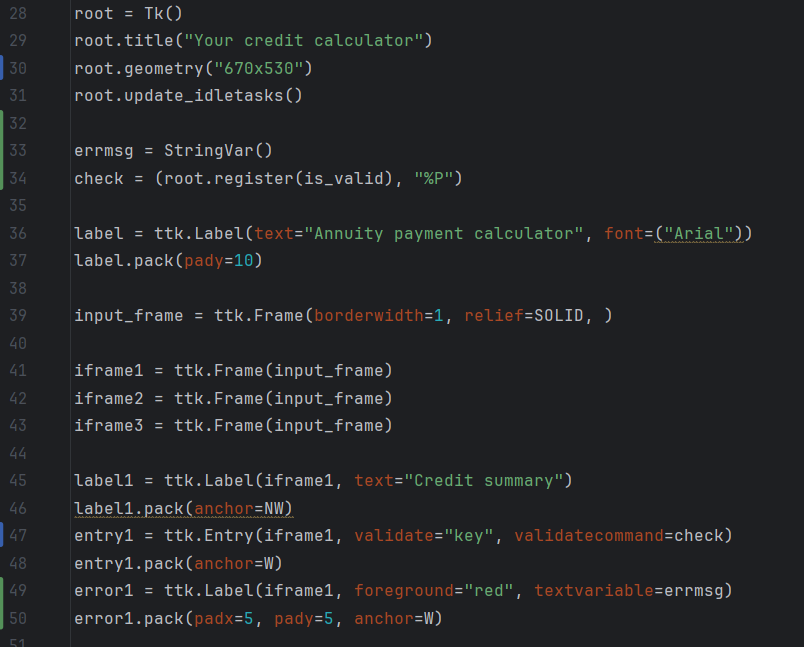
Используя любой способ компоновки элементов в окне создайте **графический интерфейс** любого приложения. Обязательно использовать **минимум 6 виджетов**.

Примеры приложений: тест, опрос, калькулятор, конвертер, построение фигур по выбранным параметрам и др.

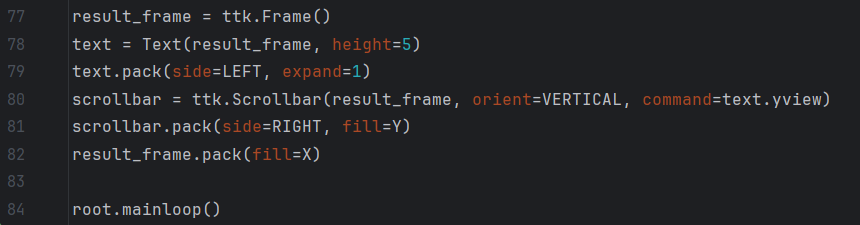
Опишите функционал этого приложения, чтобы оно работало.

Code:

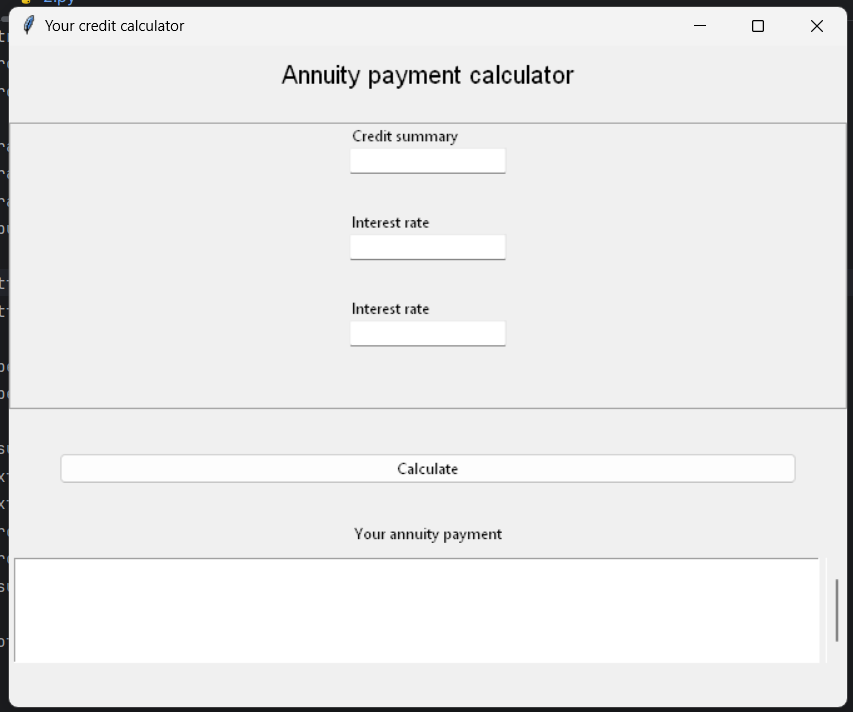


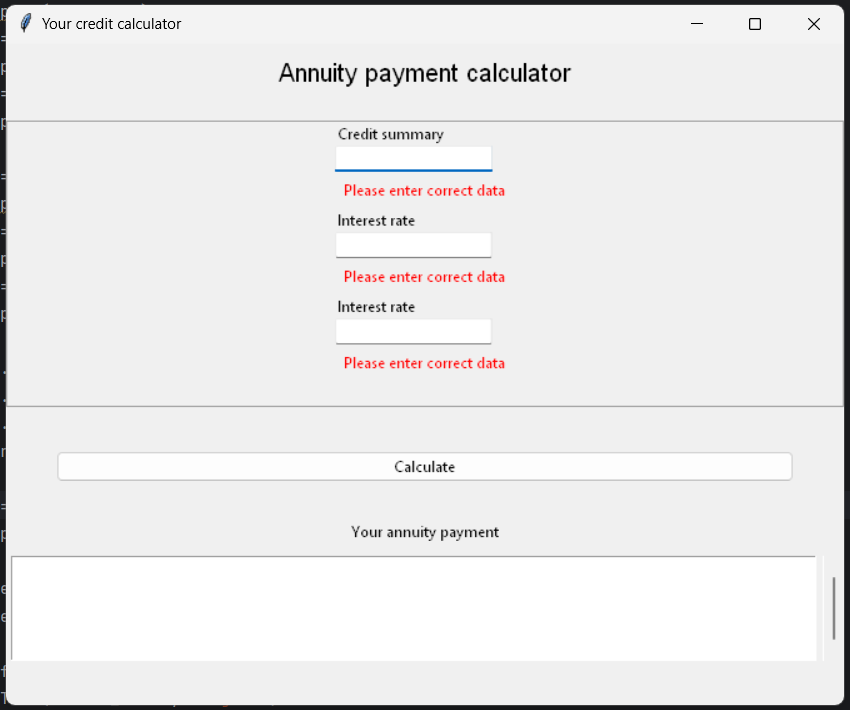


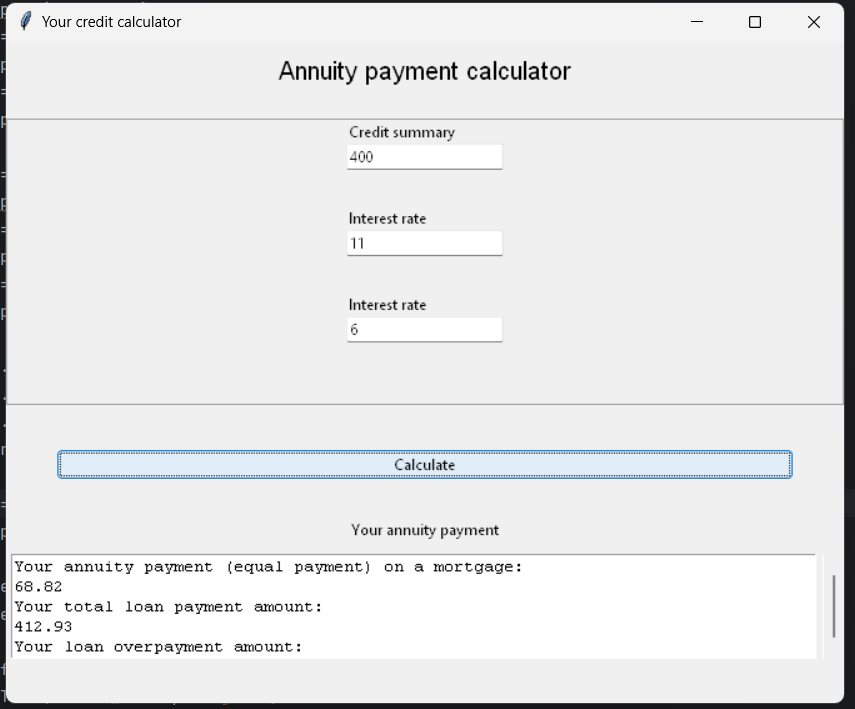




Result:





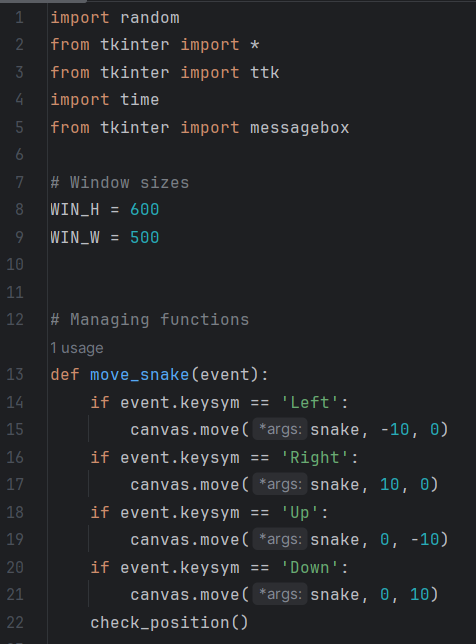


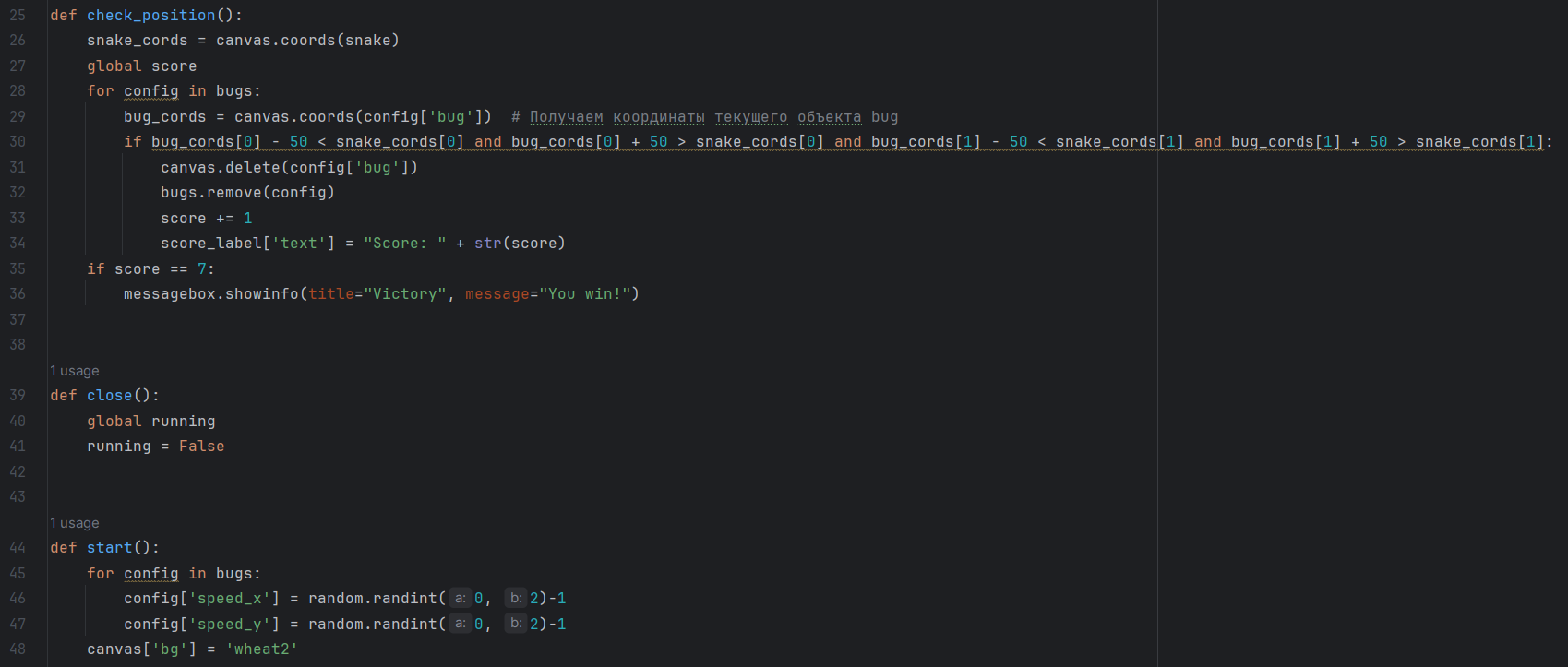
## + Лабораторная работа №10 (4ч). Tkinter. Обработка событий различного типа. Покадровая анимация. Создание простейшей двумерной игры.

**Задание 1.**

Создать графическое приложение с использованием библиотеки Tkinter, которое будет воспроизводить покадровую анимацию и позволит пользователю играть в простейшую двумерную игру.

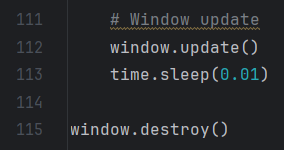
Code:





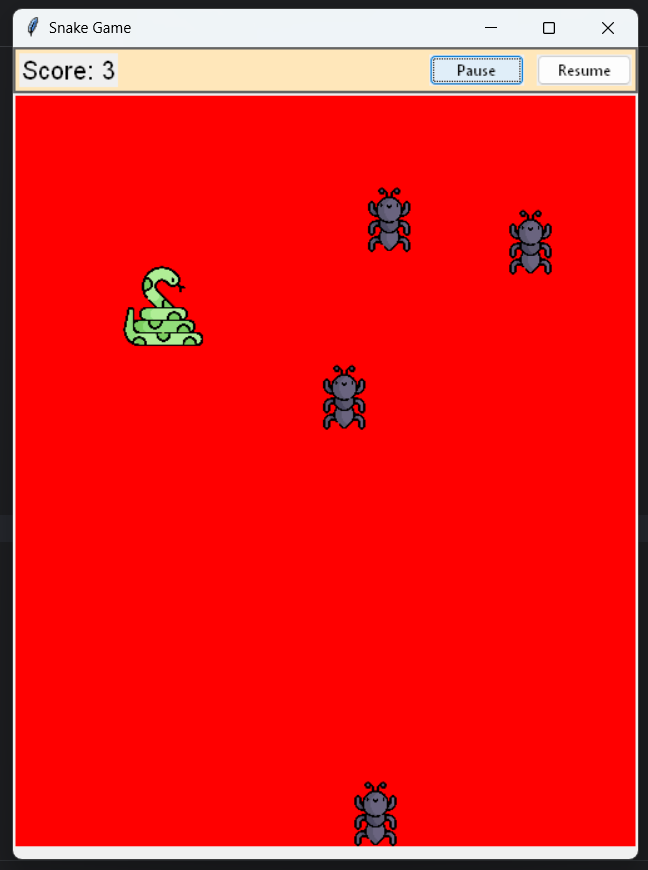
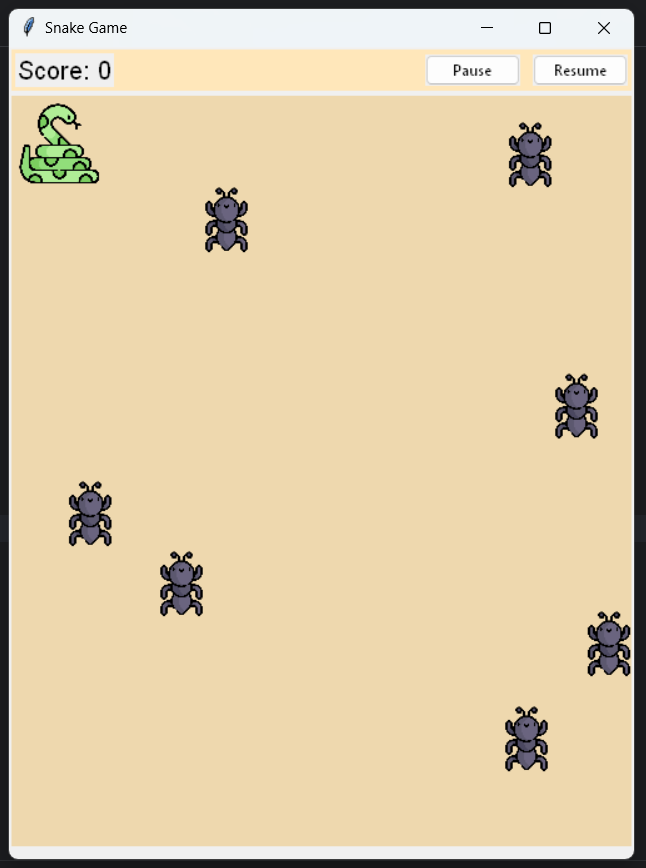


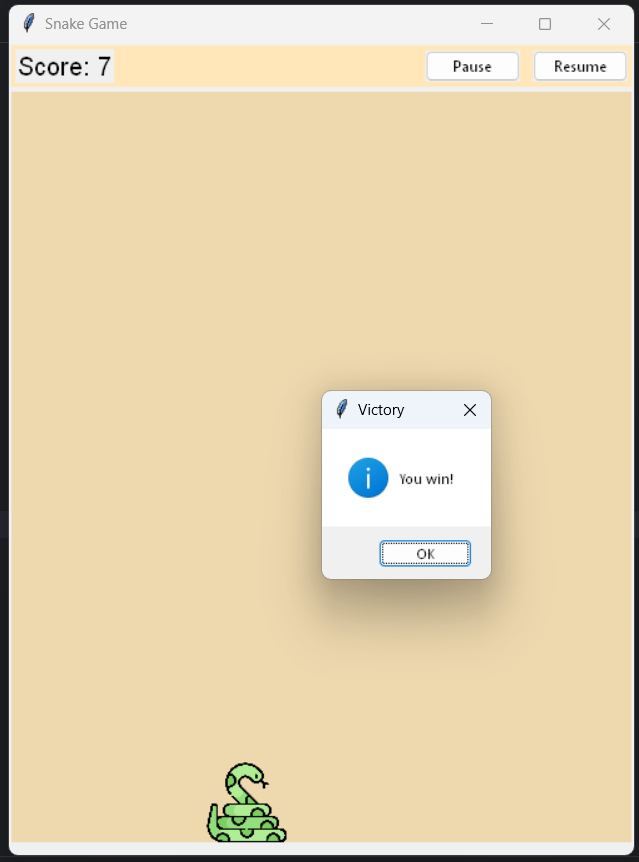




Result:







## Лабораторная работа №11 (2ч). Парсинг. Получение и обработка данных со сторонних ресурсов.

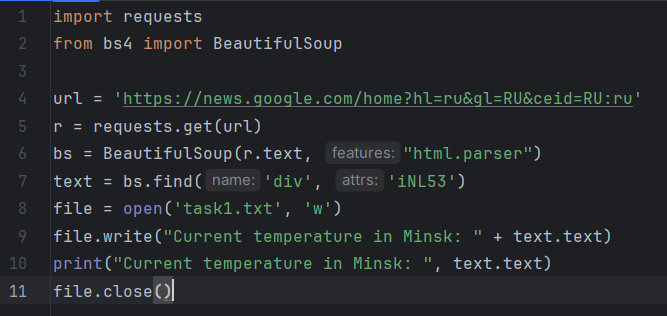
**Задание 1. Парсинг новостей с веб-страницы**

Напишите программу для парсинга новостей с веб-страницы. Программа должна получать данные с выбранного новостного сайта и обрабатывать их для отображения или сохранения в нужном формате.

**Требования**:

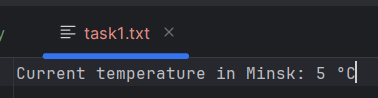
1. Выберите новостной сайт, с которого будете получать данные.
2. Используйте библиотеки для парсинга HTML, например, BeautifulSoup или lxml, для извлечения информации о заголовках новостей, их содержимом и других данных, которые хотите получить.
3. Реализуйте функции для загрузки веб-страницы с новостями, парсинга HTML и извлечения нужных данных.
4. Обработайте полученные данные по вашему усмотрению: отобразите их на экране или сохраните в файл.

Code:

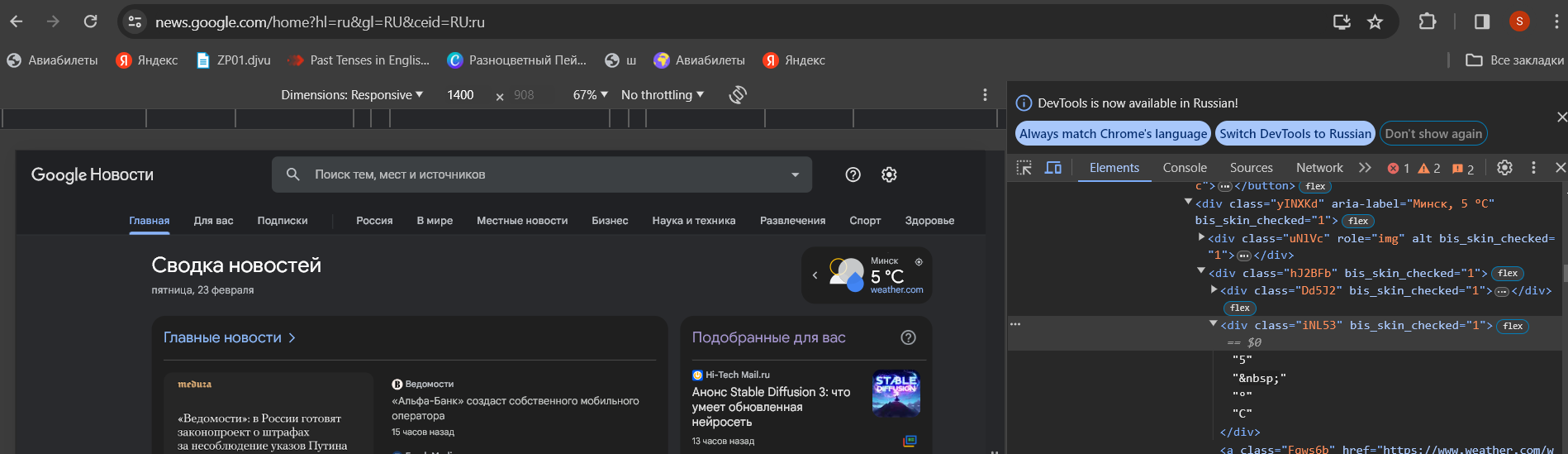


Result:





Web page:



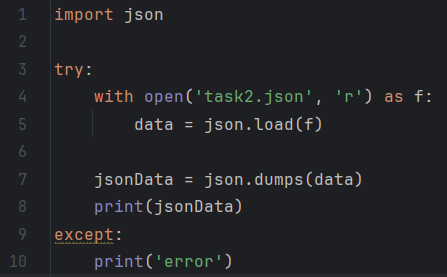
**Задание 2.Парсинг данных из формата JSON**

Напишите программу для парсинга данных из формата JSON. Программа должна загружать JSON-файл, извлекать нужные данные и обрабатывать их в соответствии с вашими потребностями.

**Требования**:

1. Выберите JSON-файл, из которого будете получать данные.
2. Используйте модуль json для загрузки файла JSON и извлечения данных.
3. Реализуйте функции для загрузки файла JSON, чтения данных из него и обработки полученных данных.
4. Обработайте данные по вашему усмотрению: отобразите их на экране или сохраните в другой формат.
5. Обеспечьте обработку возможных ошибок, связанных с загрузкой и обработкой файла JSON, чтобы ваша программа была надежной.

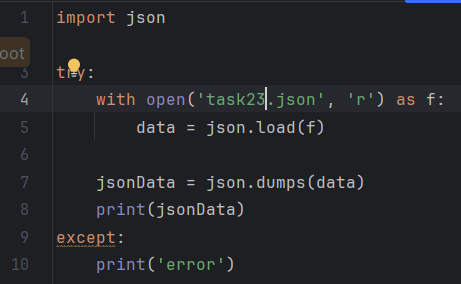
Code:



Result:



Error:





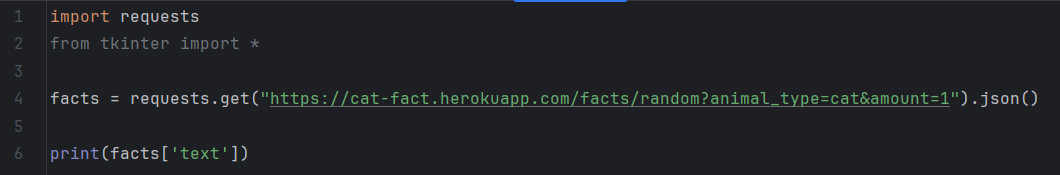
**Задание 3. Парсинг данных с API**

Напишите программу для взаимодействия с API и получения нужных данных. Программа должна отправлять запросы к выбранному API, получать ответы и обрабатывать полученные данные.

**Требования**:

1. Выберите API, с которым будете взаимодействовать, и изучите его документацию для понимания доступных эндпоинтов и параметров запросов.
2. Используйте библиотеку для отправки HTTP-запросов, такую как requests, для отправки запросов к выбранному API.
3. Реализуйте функции для формирования запросов, отправки их и обработки полученных ответов.
4. Извлеките нужные данные из ответов API и обработайте их по вашему усмотрению: отобразите на экране или сохраните в файл.

Code:



Result:

