МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение

высшего образования

**«Северный (Арктический) федеральный университет имени М.В. Ломоносова»**

##### **технологический колледж императора петра i**

**ОТЧЁТ**

**по лабораторно-практическим работам**

|  |  |  |
| --- | --- | --- |
| По дисциплине/междисциплинарному курсу | |  |
| МДК.01.01 Разработка программных модулей | | |
|  | | |
|  |
|  | | |

|  |  |
| --- | --- |
|  | Выполнил (-а) обучающийся (-аяся):  Антипов Константин Сергеевич |
|  | (Ф.И.О.) |
|  | Специальность:  09.02.07 Информационные системы и программирование |
|  | (код и наименование) |
|  | Курс: 2 |
|  | Группа: ИС-21 184118 |
|  | Преподаватель:  Воронцов Роман Антонович |
|  | (Ф.И.О.) |

Архангельск 2022

**Лабораторная работа №1**

**Работа С Классами**

**Цель работы:** Научиться работать с классами.

**Задание:**1. Создать класс объект кошки Cat, который будет принимать два аргумента, три метода по примеру выше и создать 2-3 экземпляра класса.

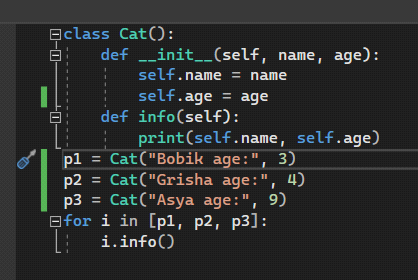


Рисунок 1 – создание класса Cat

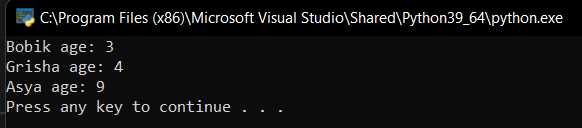


Рисунок 2 – результат программы

2. Решить задачу определения високосный год или нет с применением класса.

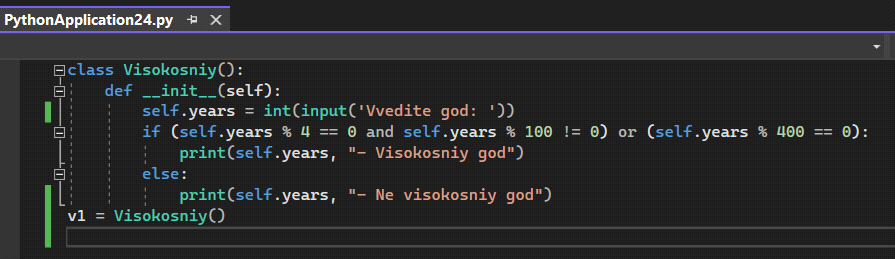


Рисунок 3 – задача с високосным годом

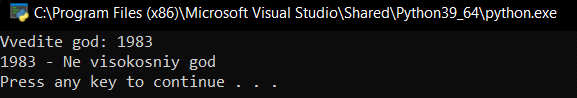


Рисунок 4 – результат программы

Задание по варианту (9 вариант):



Рисунок 1 – листинг кода

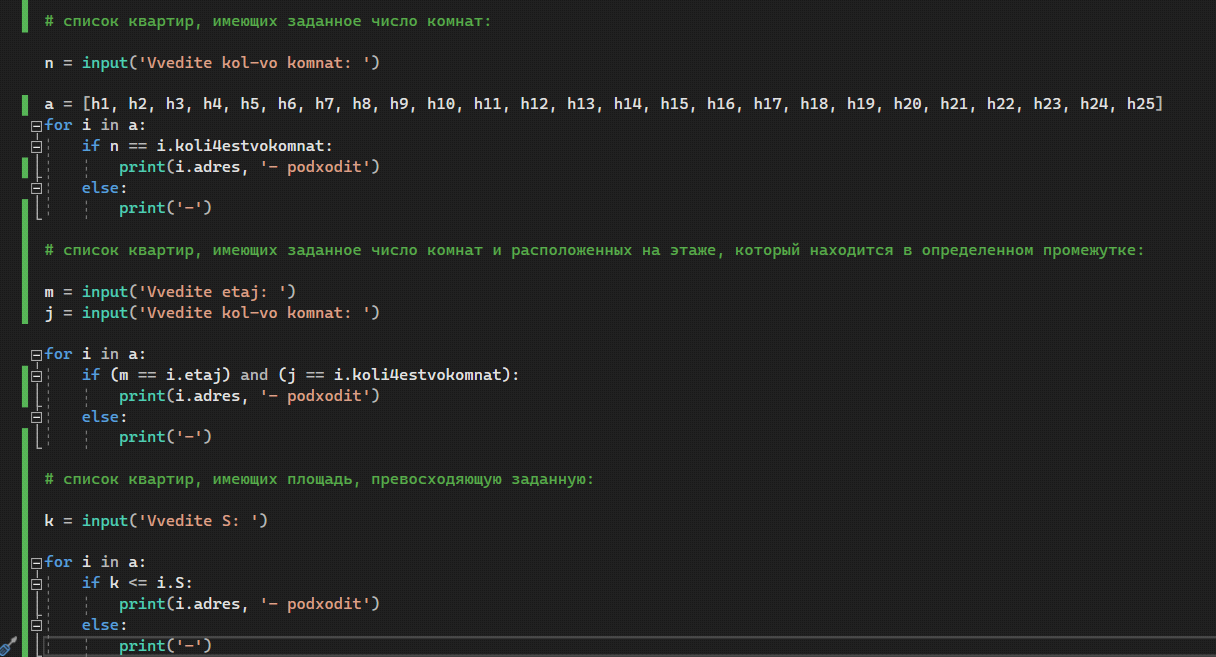


Рисунок 2 – листинг кода

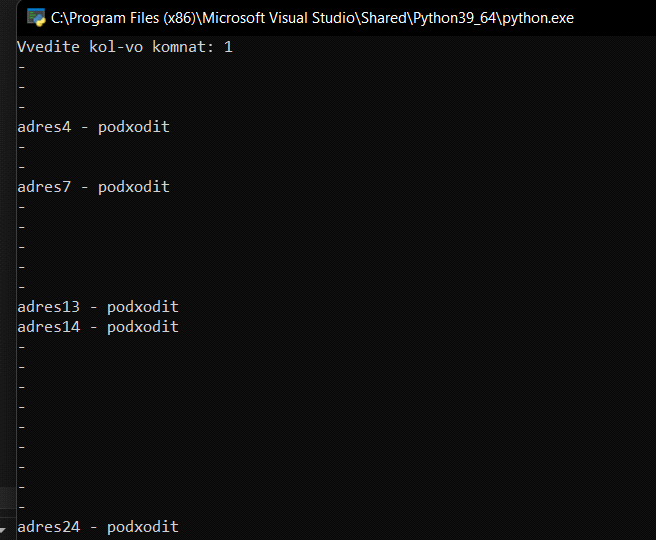


Рисунок 3 – результат программы

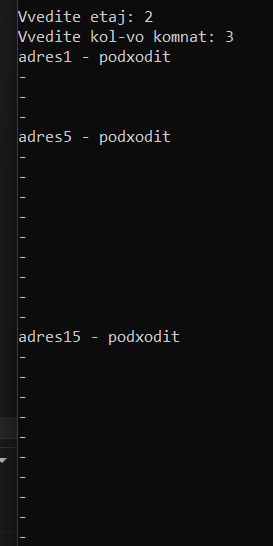


Рисунок 4 – результат программы

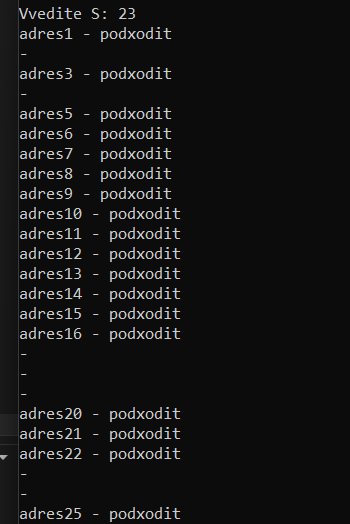


Рисунок 5 – результат программы

**Контрольные вопросы:**

**1. Что представляет собой класс?**

В объектно-ориентированном программировании, представляет собой шаблон для создания объектов, обеспечивающий начальные значения состояний: инициализация полей-переменных и реализация поведения функций или методов.

**2. Как создаются новые классы?**

Классы создаются с помощью инструкции class, за которой следует произвольное имя класса, после которого ставится двоеточие, далее с новой строки и с отступом реализуется тело класса.

**Вывод:**

Я научился работать с классами.

**Лабораторная работа №2**

**Перегрузка Методов**

**Цель работы:** Научиться работать с перегрузкой методов.

**Задание:** Выполните перегрузку метода согласно варианта вашего индивидуального задания.

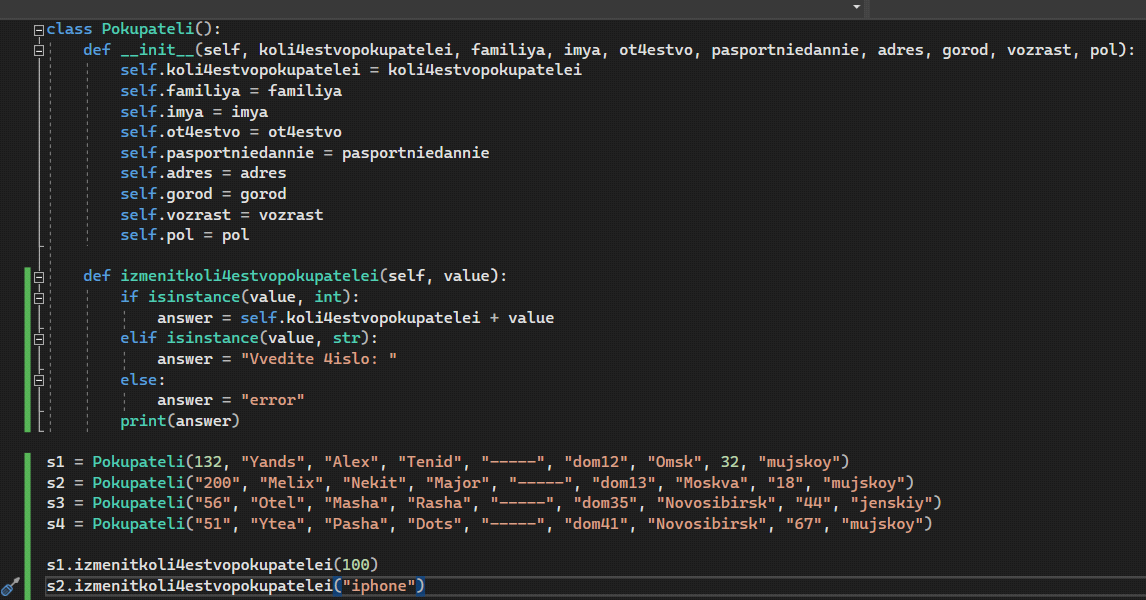


Рисунок 1 – создание перегрузки

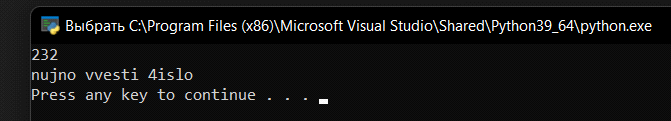


Рисунок 2 – результат программы

**Контрольные вопросы:**

**1. Что такое перегрузка методов?**

Это приём программирования, который позволяет разработчику в одном классе для методов с разными параметрами использовать одно и то же имя.

**2. Что такое перегрузка операторов?**

Это возможность одновременного существования в одной области видимости нескольких различных вариантов применения оператора, имеющих одно и то же имя, но различающихся типами параметров, к которым они применяются.

**Вывод:**

Я научился работать с перегрузкой методов.

**Лабораторная работа №3**

**Определение Операций В Классе**

**Цель работы:** Научиться работать в классе.

**Задание:** Выполнить задания из приложенных файлов:

1. ООП. Задание 6. Перегрузка операций. Документирование.



Рисунок 1 – создание документирования

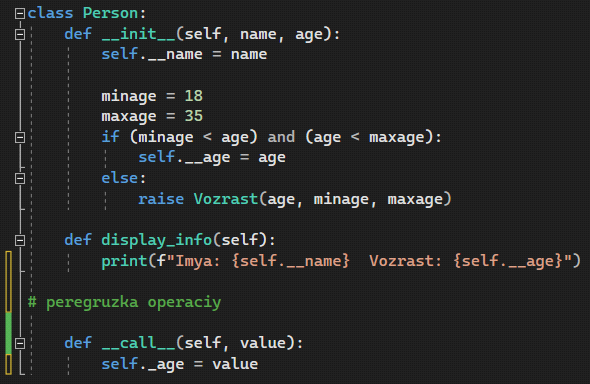


Рисунок 2 – перегрузка операций

2. ООП. Задание 7. Ассоциация. Метапрограммирование.

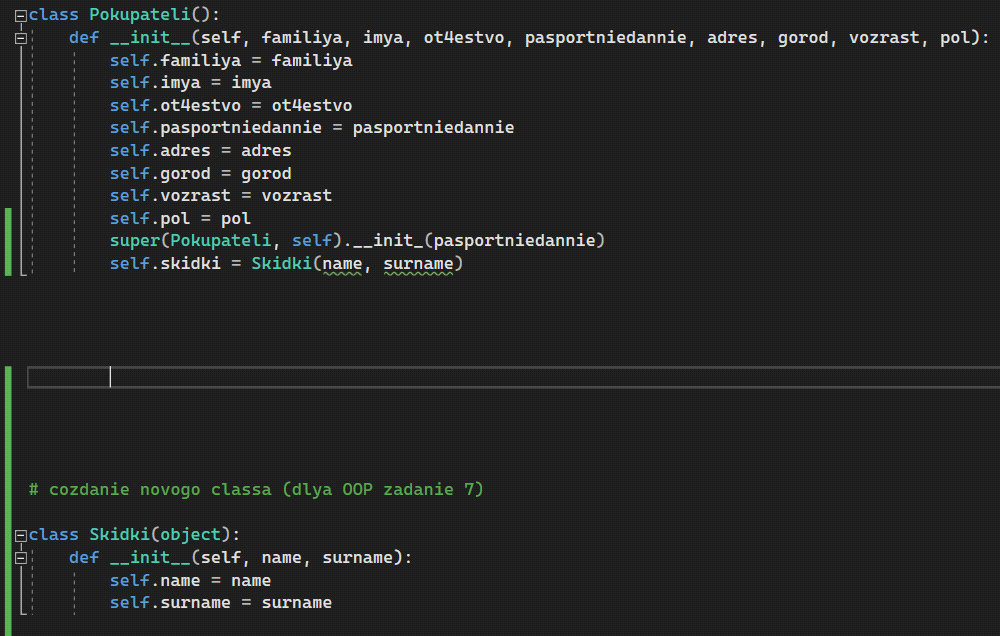


Рисунок 3 – привязка классов

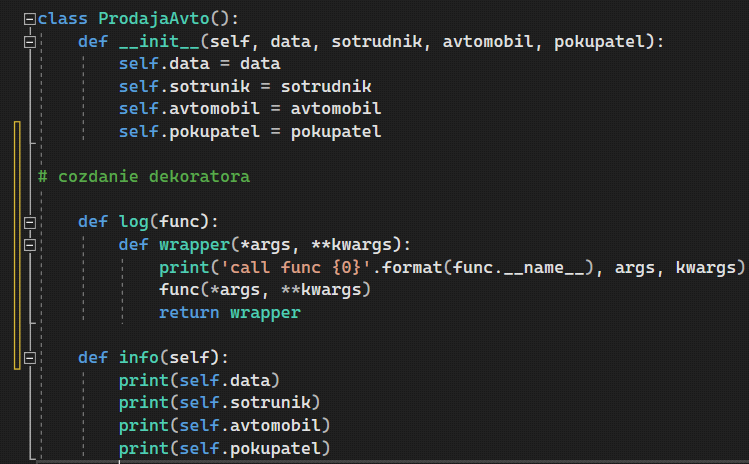


Рисунок 4 – создание декоратора для методов класса

3. ООП. Задание 8. Агрегация. Контейнеры. Итераторы.

**Контрольные вопросы:**

**1. Что такое конструктор?**

Это специальный блок инструкций, вызываемый при создании объекта.

**Вывод:**

Я научился работать в классе.

**Лабораторная работа №4**

**Создание Наследованных Классов.**

**Цель работы:** Научиться работать с наследованными классами.

**Задание:** Выполните задания из файлов:

1. ООП. Задание 2. Конструктор.

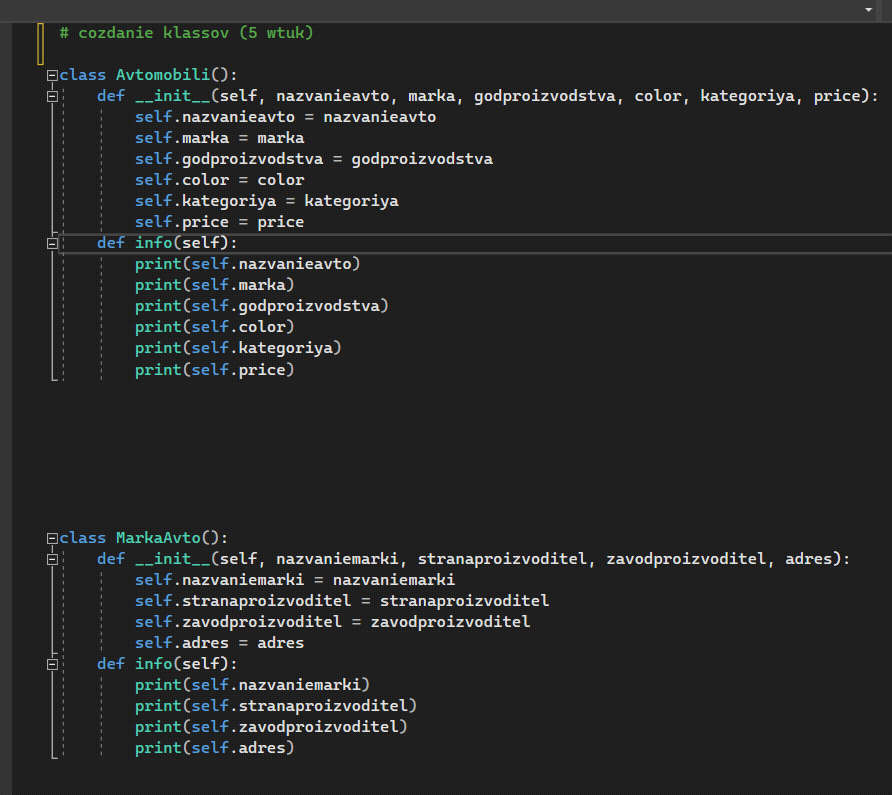


Рисунок 1 – листинг кода (создание классов)

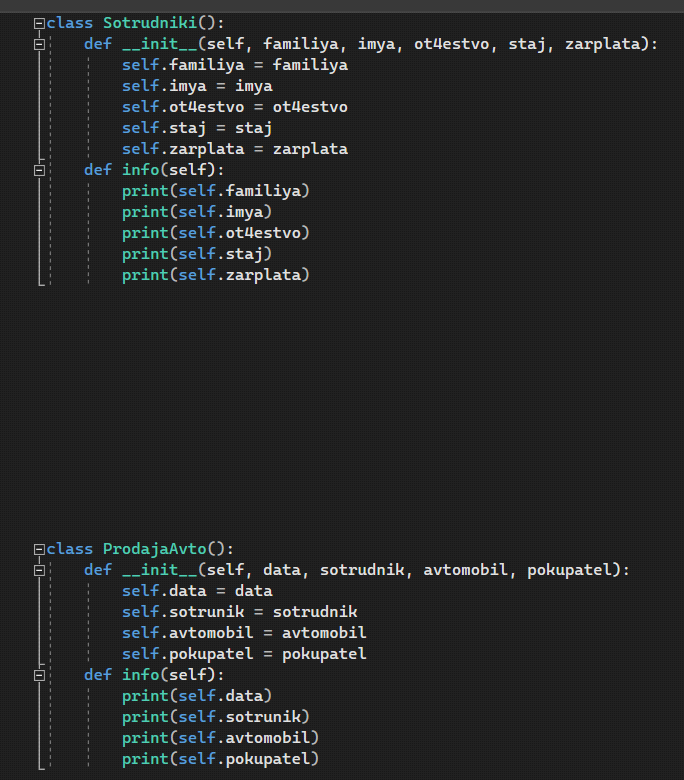


Рисунок 2 – листинг кода (создание классов)

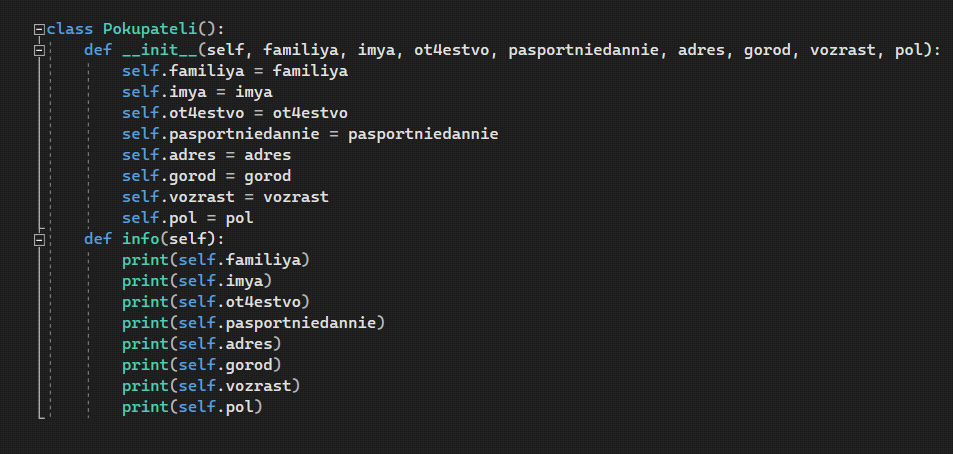


Рисунок 3 – листинг кода (создание классов)

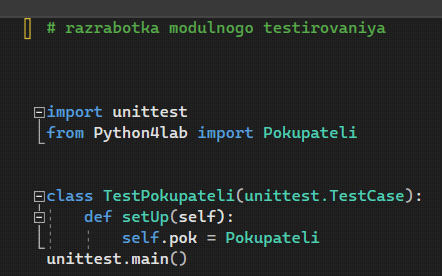


Рисунок 4 – создание модульного тестирования

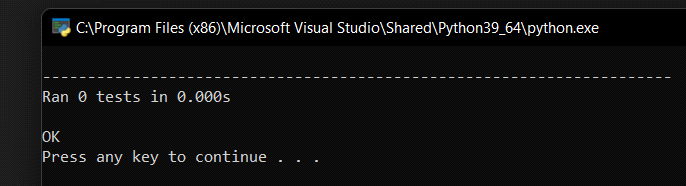


Рисунок 5 – результат тестирования

2. ООП. Задание 3. Деструктор.

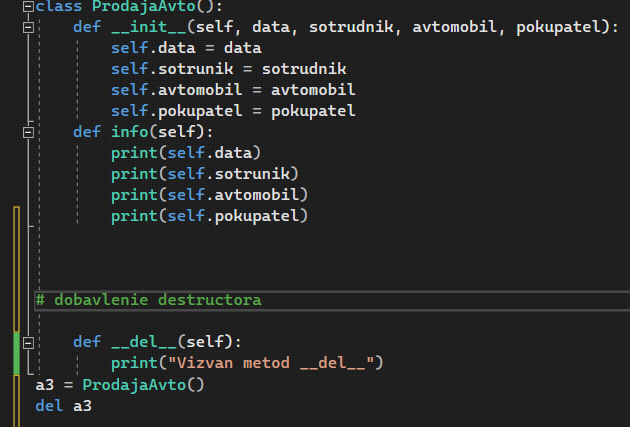


Рисунок 6 – создание деструктора

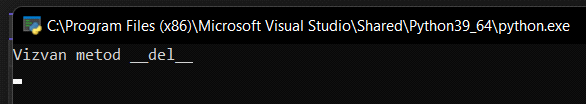


Рисунок 7 – результат деструктора

3. ООП. Задание 4. Наследование. Полиморфизм.

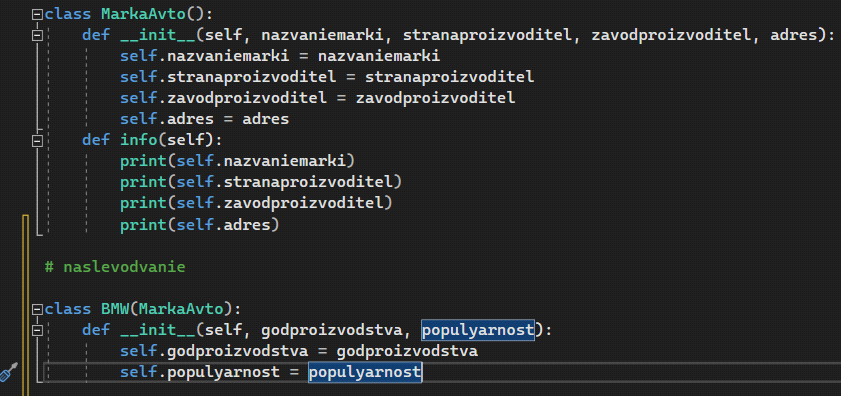


Рисунок 8 – создание наследования класса

**Контрольные вопросы:**

**1. Что такое наследование?**

Наследование — [концепция](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%86%D0%B5%D0%BF%D1%86%D0%B8%D1%8F) [объектно-ориентированного программирования](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), согласно которой [абстрактный тип данных](https://ru.wikipedia.org/wiki/%D0%90%D0%B1%D1%81%D1%82%D1%80%D0%B0%D0%BA%D1%82%D0%BD%D1%8B%D0%B9_%D1%82%D0%B8%D0%BF_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) может наследовать данные и функциональность некоторого существующего типа, способствуя повторному использованию компонентов [программного обеспечения](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5).

**Вывод:**

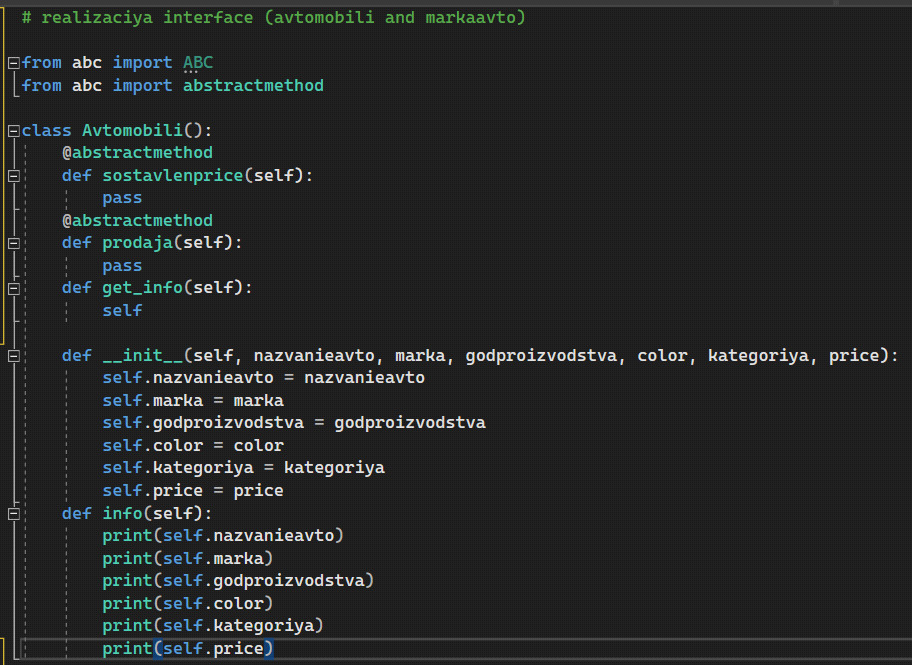
Я научился работать с наследованными классами

**Лабораторная работа №5**

**Работа С Объектами Через Интерфейсы. Использование Стандартных Интерфейсов**

**Цель работы:** Научиться работать с интерфейсами.

**Задание:** 1. Выполнить задание из файла «Интерфейсы Python».



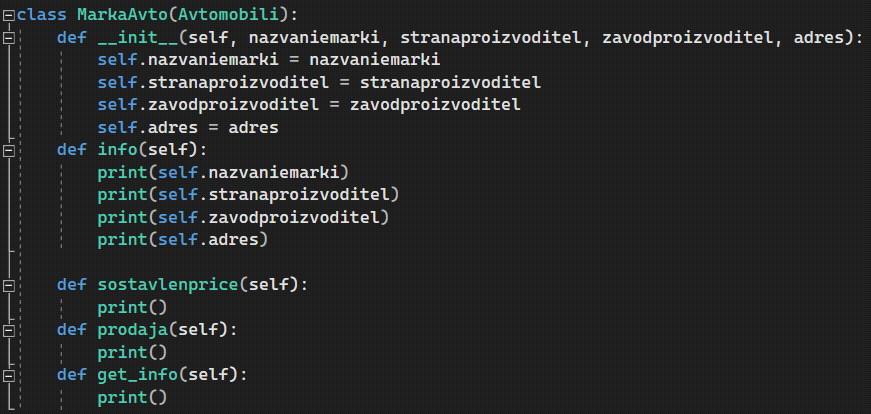


Рисунок 1, 2 – реализация интерфейса

**Контрольные вопросы**:

**1. Что такое интерфейс?**

Интерфейс представляет ссылочный тип, который может определять некоторый функционал - набор методов и свойств без реализации. Затем этот функционал реализуют классы и структуры, которые применяют данные интерфейсы.

**Вывод:**

Я научился работать с интерфейсами.

**Лабораторная работа №6**

**Работа С Типом Данных Структура**

**Цель работы:** Научиться работать с типом данных структуры.

**Задание:** 1. Выполнить задание из приложенного файла «Структуры в Python».

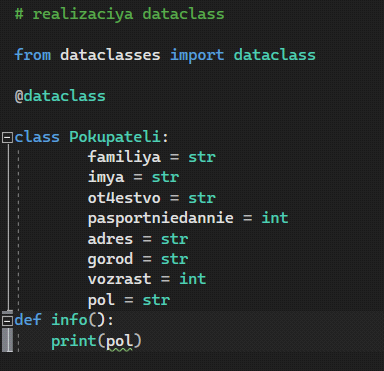


Рисунок 1 – реализация dataclass

**Контрольные вопросы:**

**1. Что такое структура?**

Это совокупность переменных, объединенных одним именем, предоставляющая общепринятый способ совместного хранения информации.

**2. Методы задания структур.**

* графический
* матричный
* аналитический
* множественный

**Вывод:**

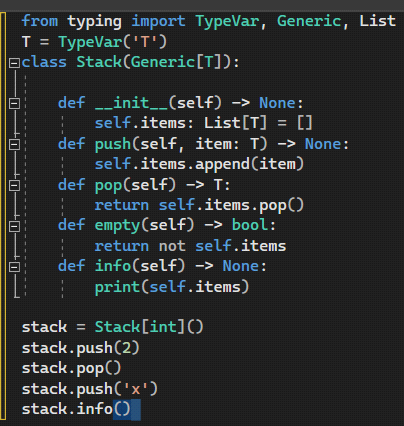
Я научился работать с типом данных структуры.

**Лабораторная работа №7.1**

**Использование Основных Шаблонов**

**Цель работы:** Изучить основные шаблоны.

**Задание:** 1. Выполнить задание из файла «Обобщенные классы в Python»



**Лабораторная работа №7**

**Коллекции. Параметризованные Классы**

**Цель работы:** Научиться работать с коллекциями. Параметризованными классами.

**Задание:**1. Выполнить задание из файла «Коллекции в Python».

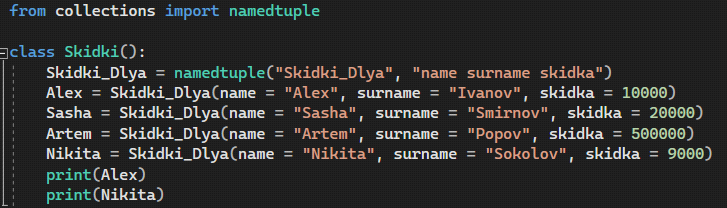


Рисунок 1 – реализация модуля namedtuple

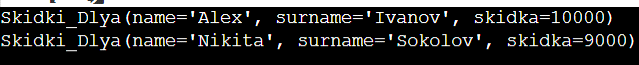


Рисунок 2 – результат программы

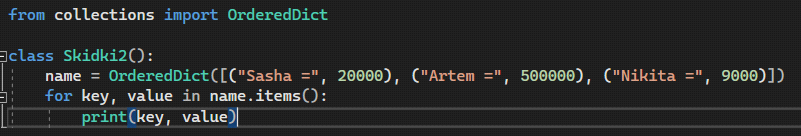


Рисунок 3 – реализация модуля OrderedDict



Рисунок 4 – результат программы

**Контрольные вопросы:**

**1. Основные виды коллекций?**

* defaultdict
* OrderedDict
* counter
* deque
* namedtuple
* enum.Enum (вне модуля)

**2. Применение параметризованных классов.**

Параметризованный класс представляет собой что-то вроде шаблона для построения других классов; шаблон может быть параметризован другими классами, объектами или операциями.

**Вывод:**

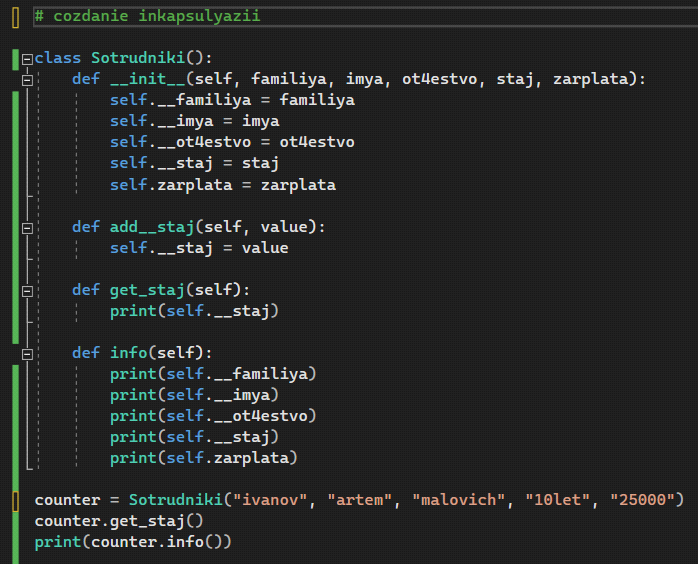
Я научился работать с коллекциями. Параметризованными классами.

**Лабораторная работа №9**

**Инкапсуляция**

**Цель работы:** Научиться работать с механизмом сокрытия компонентов программы.

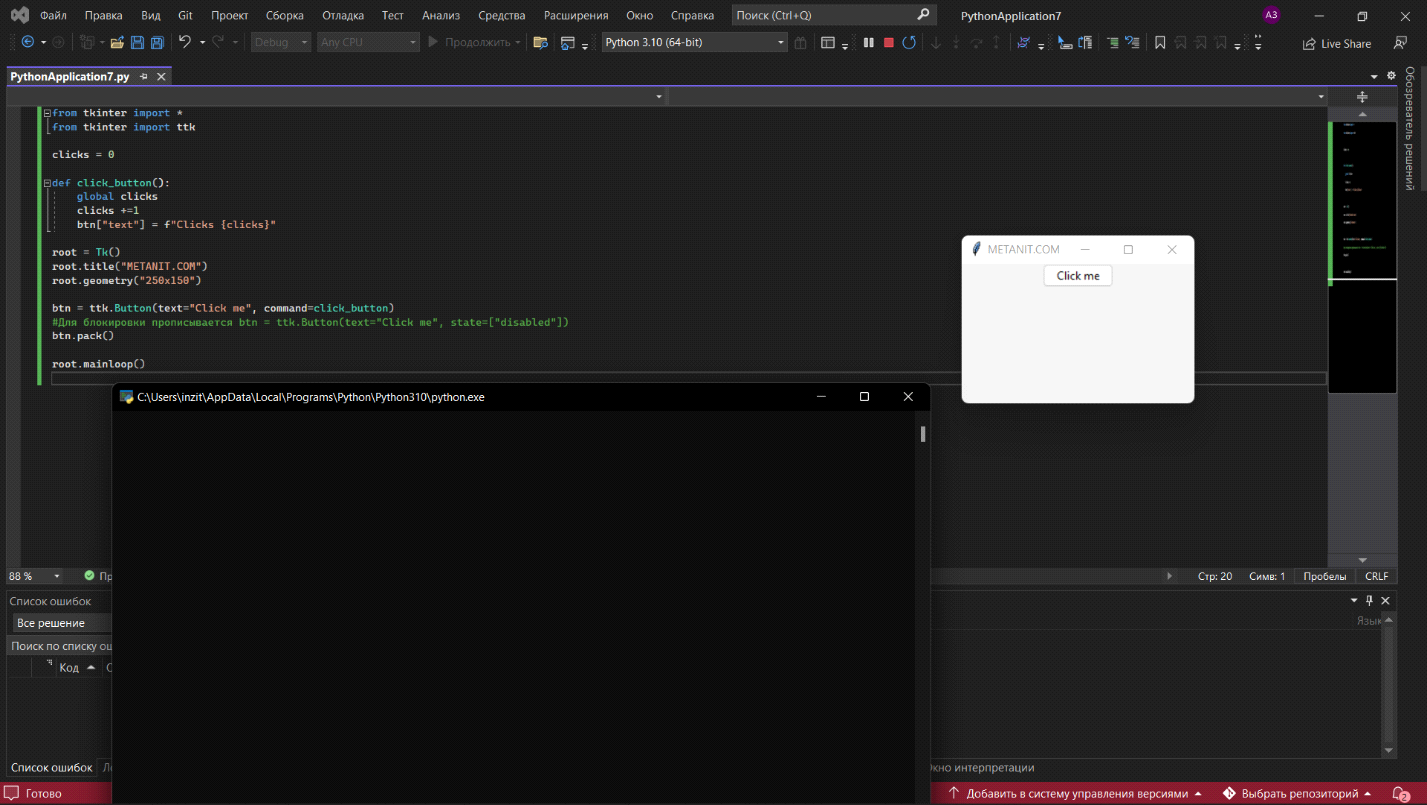
**Задание:** 1. Решите задания из приложенного файла ООП. Задание 5. Инкапсуляция. Исключения.



**Лабораторная работа №13**

**Разработка Приложения С Использованием Текстовых Компонентов**

**Цель работы**: Разработать оконное приложение с использованием текстовых компонентов.



**Контрольные вопросы**

**1.Перечислите основные компоненты интерфейса в tkinter**

* + Button: кнопка
  + Label: текстовая метка
  + Entry: однострочное текстовое поле
  + Text: многострочное текстовое поле
  + Checkbutton: флажок
  + Radiobutton: переключатель или радиокнопка
  + Frame: фрейм, который организует виджеты в группы
  + Listbox: список
  + Combobox: выпадающий список
  + Menu: элемент меню
  + Scrollbar: полоса прокрутки
  + Treeview: позволяет создавать древовидные и табличные элементы
  + Scale: текстовая метка
  + Spinbox: список значений со стрелками для перемещения по элементам
  + Progressbar: текстовая метка
  + Canvas: текстовая метка
* Notebook: панель вкладок

**2.Каким образом можно связать нажатие кнопки с выполнением функции?**

Для обработки нажатия на кнопку необходимо установить в конструкторе параметр command, присвоив ему ссылку на функцию, которая будет срабатывать при нажатии.

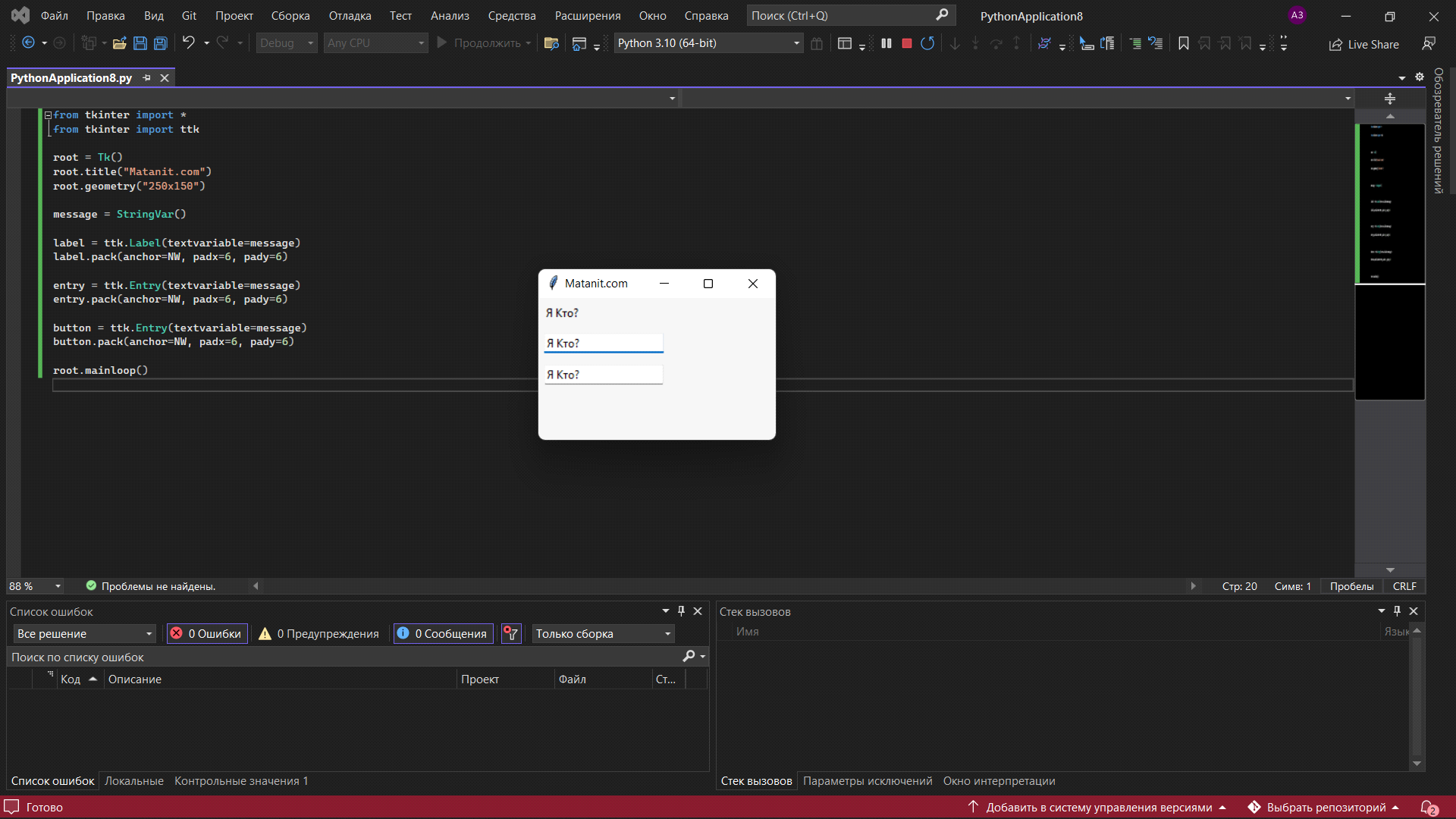
**Вывод**

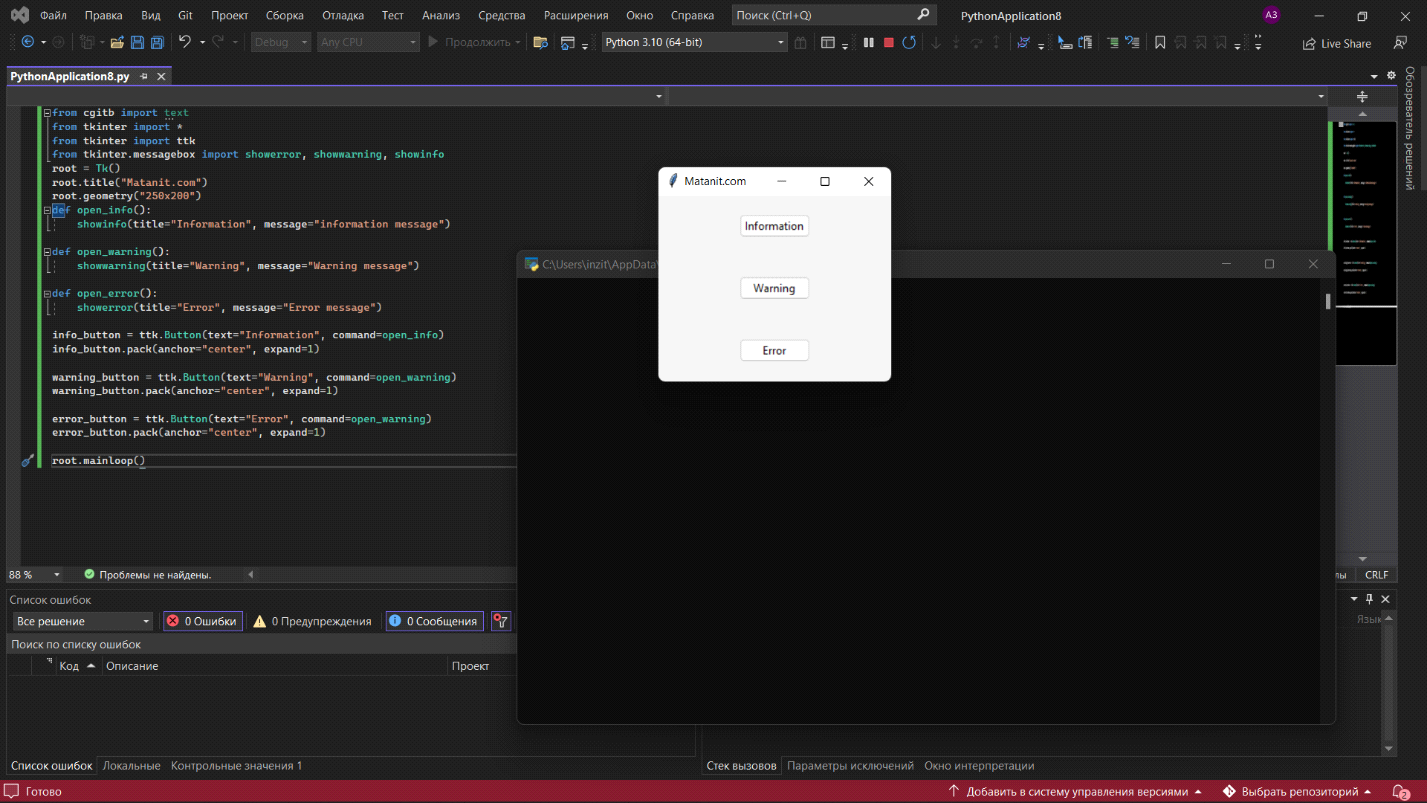
Я разработал оконное приложение с использованием текстовых компонентов.

**Лабораторная работа №14**

**Разработка Приложения С Несколькими Формами**

**Цель работы:** Научиться разрабатывать приложение с несколькими формами.

(Привязка виджетов к переменным)

(MessageBox)

**Контрольные вопросы**

**1. Методы отображения новой формы?**

* anchor: устанавливает позиционирование текста
* background: фоновый цвет
* borderwidth: толщина границы метки
* cursor: курсор указателя мыши при наведении на метку
* font: шрифт текста
* foreground: цвет текста
* height: высота виджета
* image: ссылка на изображение, которое отображается на метке
* justify: устанавливает выравнивание текста. Значение LEFT выравнивает текст по левому краю, CENTER - по центру, RIGHT - по правому краю
* pading: отступы от границ вилжета до его текста
* relief: определяет тип границы, по умолчанию значение FLAT
* text: устанавливает текст метки
* textvariable: устанавливает привязку к элементу StringVar
* underline: указывает на номер символа в тексте кнопки, который подчеркивается. По умолчанию значение -1, то есть никакой символ не подчеркивается
* width: ширина виджета
* wraplength: при положительном значении строки текста будут переносится для вмещения в пространство виджета

**2. С помощью какого метода получается значение поля Entry**

* insert(index, str): вставляет в текстовое поле строку по определенному индексу
* get(): возвращает введенный в текстовое поле текст
* delete(first, last=None): удаляет символ по индексу first. Если указан параметр last, то удаление производится до индекса last. Чтобы удалить до конца, в качестве второго параметра можно использовать значение END.
* focus(): установить фокус на текстовое поле

**Вывод**

Я научился разрабатывать приложение с несколькими формами.

**Практическая работа №1**

**Изучение и настройка системы контроля версий**

**Цель работы:** Научиться работать и настраивать систему контроля версий

**Задание:**

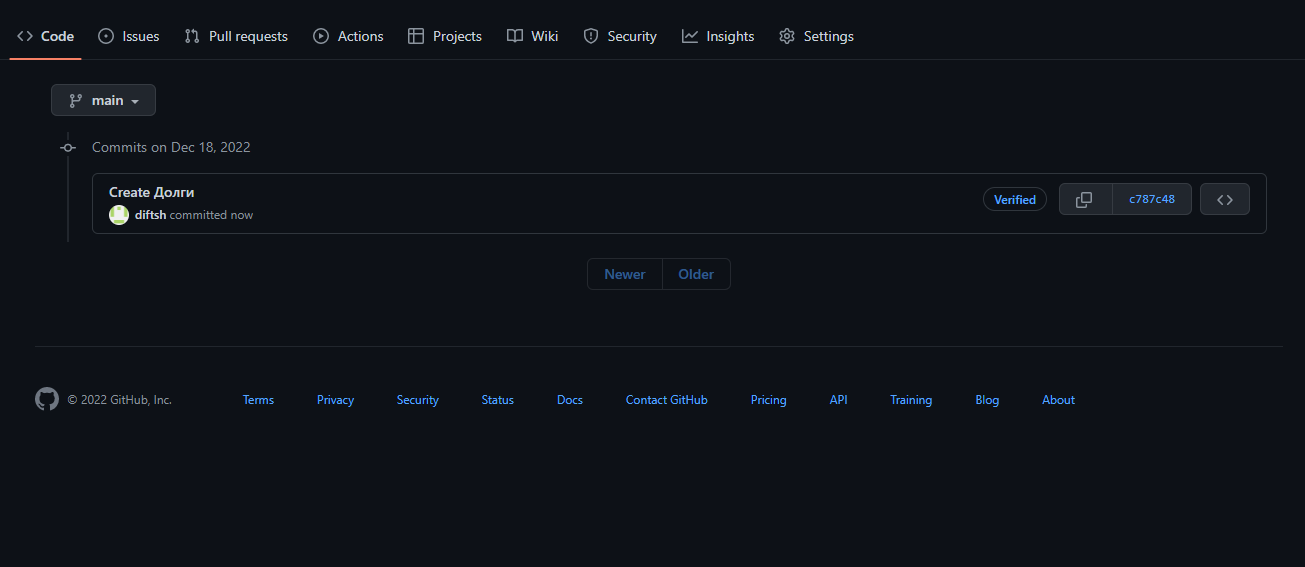


Рисунок 1 – Репозиторий

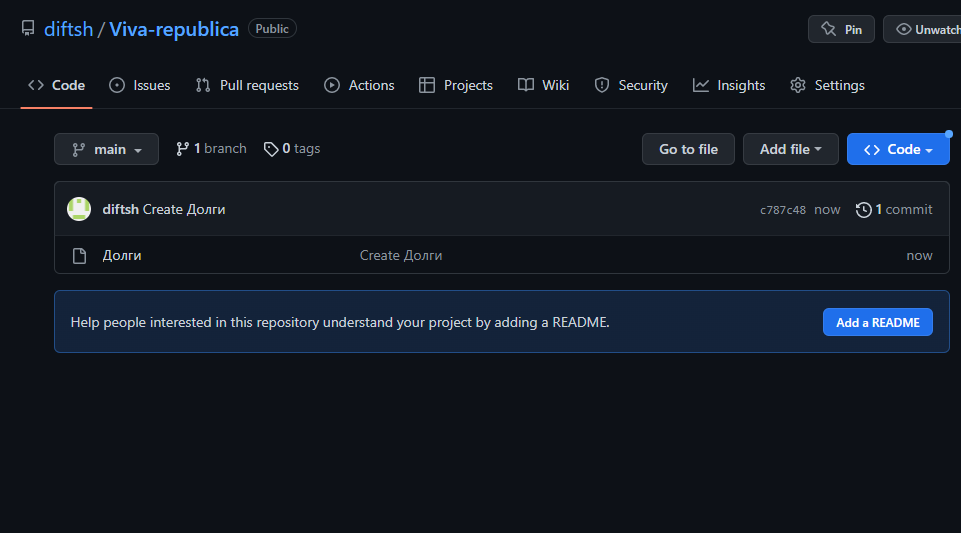


Рисунок 2 – Коммиты

**Контрольные вопросы**

**1.Принцип работы системы контроля версий?**

Система контроля версий страхует от ошибок и возвращает код в то состояние, когда всё работало.

Контрольные точки называются коммитами. Один коммит — это пакет изменений, хранящий информацию с добавленными, отредактированными или удалёнными файлами кода. В один коммит принято добавлять не более десяти изменений — так получается длинная история версий, которая позволяет в случае ошибки откатиться с минимальной потерей работоспособного кода.

**Вывод**

Научился работать и настраивать систему контроля версий

**Практическая работа №3**

**Оценка Сложности Алгоритмов Сортировки. Оценка Сложности Алгоритма Поиска**

**Цель работы:** Научиться разрабатывать алгоритмы сортировки и поиска, а также оценивать их сложность.

**Задание:** 1. Разработать алгоритм сортировки и оценить его сложность.

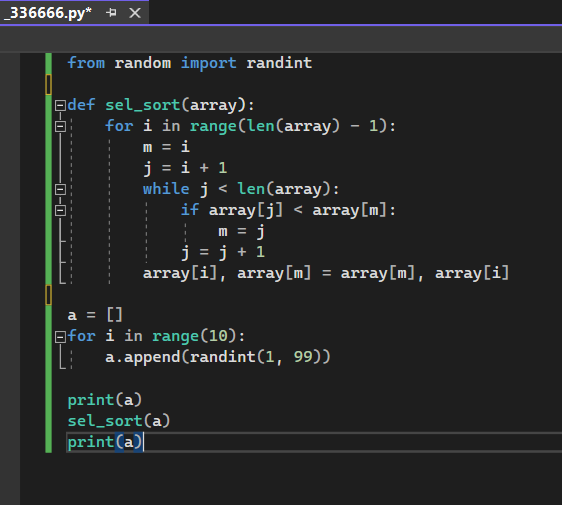


Рисунок 1 – алгоритм сортировки

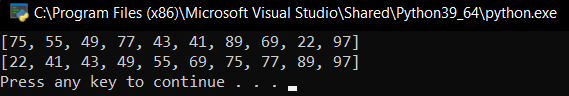


Рисунок 2 – результат алгоритма

Сложность O(n^2)

2. Разработать алгоритм поиска и оценить его сложность.

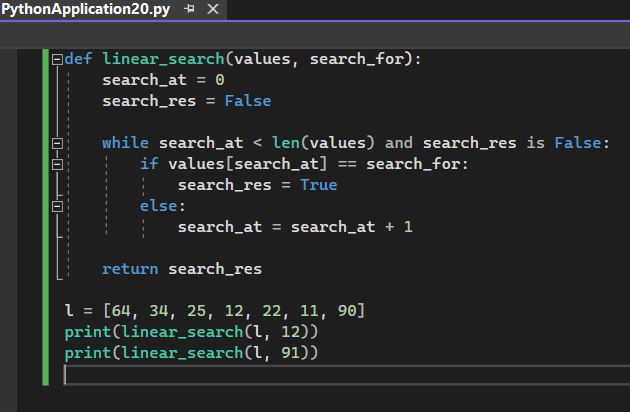


Рисунок 3 – алгоритм поиска

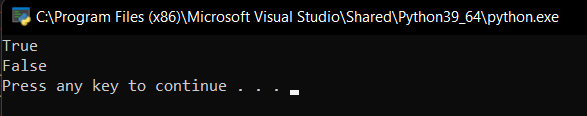


Рисунок 4 – результат алгоритма

Сложность O(n)

**Контрольные вопросы:**

**1. Что такое алгоритм сортировки?**

Это алгоритм для упорядочивания элементов в массиве. В случае, когда элемент в массиве имеет несколько полей, поле, служащее критерием порядка, называется ключом сортировки. На практике в качестве ключа часто выступает число, а в остальных полях хранятся какие-либо данные, никак не влияющие на работу алгоритма.

**2. Что такое алгоритм поиска?**

Это любой алгоритм, который решает задачу поиска, а именно извлекает информацию, хранящуюся в некоторой структуре данных или вычисляемую в пространстве поиска проблемной области, либо с дискретными, либо с непрерывными значениями.

**Вывод:**

В ходе практической работы я научился разрабатывать алгоритмы сортировки и поиска, а также оценивать их сложность.

**Практическая работа №4**

**Оценка Сложности Эвристических Алгоритмов**

**Цель работы:** Научиться оценивать сложность эвристических алгоритмов.

**Задание:** 1. Оценить сложность эвристического алгоритма.

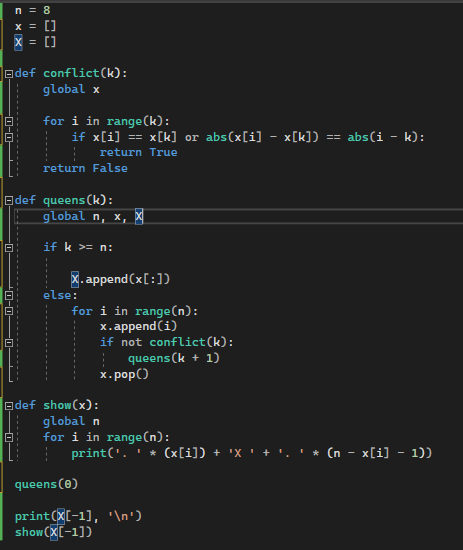


Рисунок 1 – эвристический алгоритм

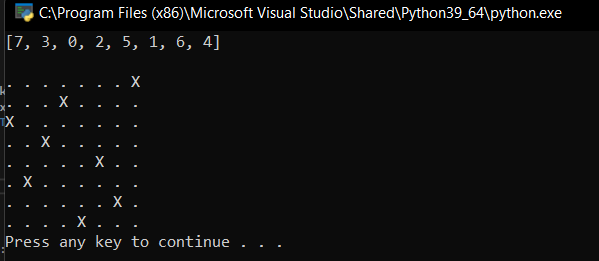


Рисунок 2 – результат кода

Сложность O(n^2)

**Контрольные вопросы:**

**1. Что такое эвристический алгоритм?**

Это алгоритм решения задачи, правильность которого для всех возможных случаев не доказана, но про который известно, что он даёт достаточно хорошее решение в большинстве случаев.

**2. Что такое формулы оценки сложности алгоритмов?**

Они позволяют ориентироваться, какие структуры данных и архитектуры стоит использовать в тех или иных ситуациях.

**Вывод:**

В ходе практической работы я научился оценивать сложность эвристических алгоритмов.

**Практическая работа №5**

**Оценка Сложности Рекурсивных Алгоритмов**

**Цель работы:** Научиться разрабатывать рекурсивные алгоритмы и оценить их сложность.

**Задание:**

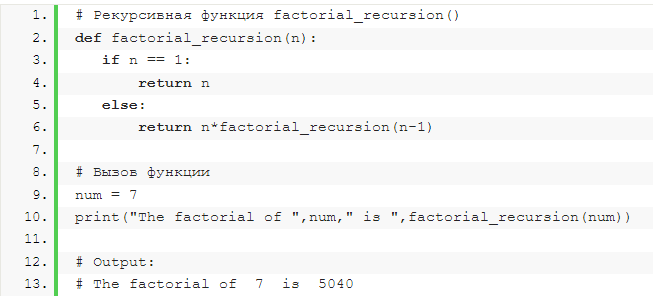


Рисунок 1 – Рекурсивный алгоритм (Сложность O(n))

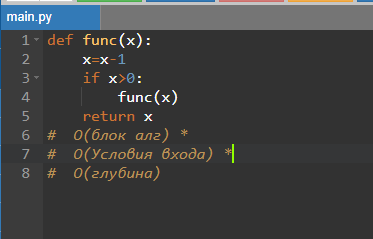


Рисунок 2 – Схема

**Контрольные вопросы:**

**1.Что такое рекурсивный алгоритм?**

Это алгоритм, в описании которого прямо или косвенно содержится обращение к самому себе.

**Практическая работа №6**

**Оформление Документации На Программные Средства**

**Цель работы:** Научиться оформлять документацию на программные средства.

**Задание:** Оформить документацию на программный код в соответствии с основными форматами документирования: Строки документации Google, reStructuredText, Строки документации NumPy/SciPy, Epytext.

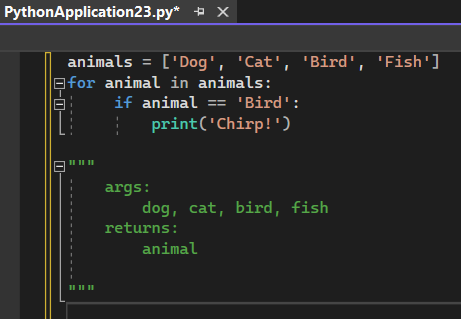


Рисунок 1 – пример GOOGLE DOCSTRINGS

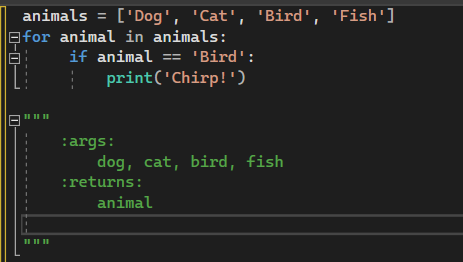


Рисунок 2 – пример RESTRUCTUREDTEXT EXAMPLE



Рисунок 3 – пример NUMPY/SCIPY DOCSTRINGS EXAMPLE

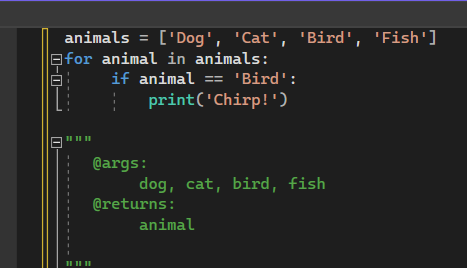


Рисунок 4 – пример EPYTEXT EXAMPLE

**Контрольные вопросы:**

**1. Что такое стандарт PEP8 и PEP257?**

PEP8 – это документ, описывающий соглашение о том, как писать код для языка Python, включая стандартную библиотеку, входящую в состав Python.

PEP 257 – описывает соглашения, связанные со строками документации Python, рассказывает о том, как нужно документировать Python код.

**2. Перечислите основные форматы документирования кода?**

* строки документации Google.
* reStructuredText.
* строки документации NumPy/SciPy.
* epytext.

**Вывод:**

Я научился оформлять документацию на программные средства.