МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное учреждение высшего образования

**«Московский технический университет связи и информатики»**

Лабораторная работа №6

**Работа с классами ч.2**

**по дисциплине**

**«Введение в информационные технологии»**

Выполнил: студент гр. БВТ2403

Косарев С.А.

Проверил:

Москва, 2024 г.

**Цель работы**: Получить практический опыт работы с ООП в Python. использование инкапсуляции, наследования.

**Оборудование:** компьютер с установленным программным обеспечением Python.

**Методы и технологии:**

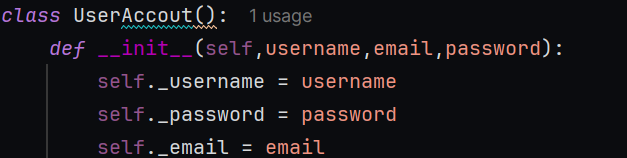
Основные функции, использованные в проекте, включают:

1. Для создания класса был использован метод class. Для инициализации переменных в классе была использована функция \_\_init\_\_.
2. Для забивания переменных в класс был использован self. Для создания функции get\_info() используется def get\_info(). Также была использована f-строка для вывода выражения с использованием переменной. Для вывода значений используется print. Для введения данных был использован input(). Для проверки выполнения/невыполнения условий используется if, else. Для обращения к родительскому классу был использован super.

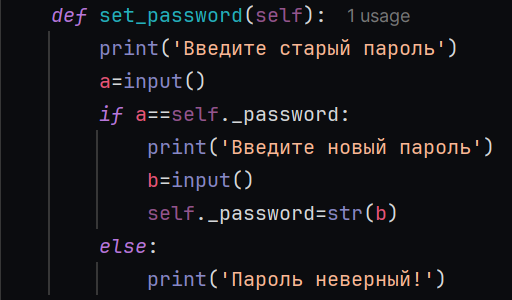
Реализация:

**Задание 1: Защита данных пользователя**

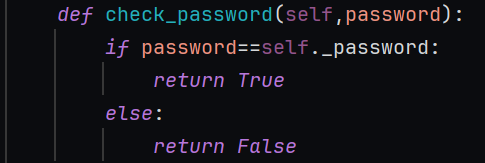
Мы создаем класс UserAccount, в который заводим имя пользователя, почту и пароль с помощью конструктора \_\_init\_\_. Для защищенности данных добавляем в обозначение переменных “\_” после точки.



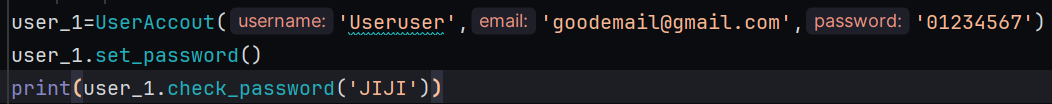
Создаем метод set\_password, в котором проверяем знание старого пароля, для верификации пользователя. Если старый пароль сходится, то даем возможность ввести новый пароль, иначе - нет.



Для проверки соответствия введенного пароля паролю пользователя аккаунта создаем метод check\_password, в который подаем переменную password, если она соответствует принадлежащему паролю, то выводим True, иначе False.

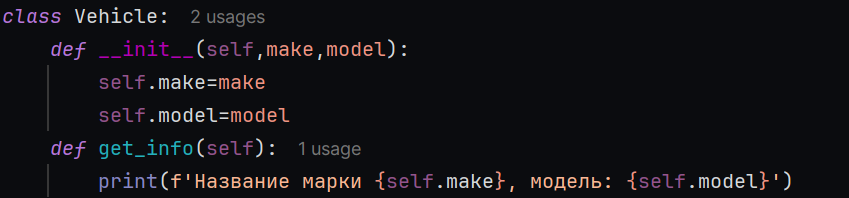


После этого создаем объект user\_1, которому присваиваем имя, почту и пароль. Вызываем функцию set\_password у объекта user\_1. И выводим значение функции check\_password.

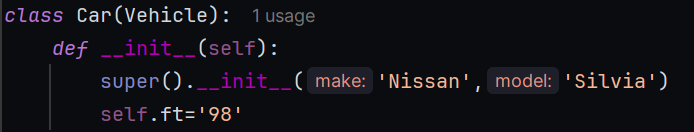


**Задание 2: Полиморфизм и наследование**

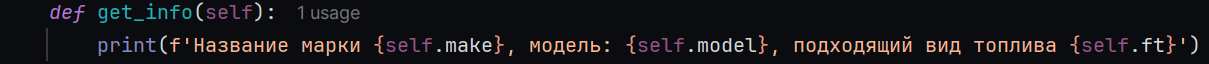
Определяем класс Vehicle с атрибутами: марка и модель, инициализируем их с помощью конструктора \_\_init\_\_. Создаем метод get\_info, который выводит информацию о ТС с помощью f-строки



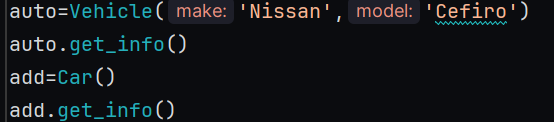
Создаем класс Car, в скобках после “Car” прописываем родительский класс, от которого Car будет наследовать. С помощью super обращаемся к родительскому классу и придаем переменным, которые наследуются от родительского класса значения. Также добавляем ещё одну переменную - значение топлива.



Переопределяем метод get\_info, добавляя в него значение подходящего топлива.



Далее создаем объекты классов Vehicle и Car. В объект классе Vehicle мы вводим значение переменных. Вызываем метод get\_info класса Vehicle, возвращающий марку и модель авто. Также вызываем метод get\_info, но уже класса Car.



Вывод: Мы получили практический опыт работы с ООП в Python. Научились использовать инкапсуляцию и наследование.