**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

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

Кафедра «Математическая Кибернетика и Информационные технологии»

Лабораторная работа №6

Работа с классами ч.2

Выполнил: Студент группы

БВТ2402

Деминова Виктория

Москва

2024

**Цель работы:**

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

**Задание:**

Задание 1: Защита данных пользователя

1. Создайте класс **UserAccount**, который представляет аккаунт пользователя с атрибутами: имя пользователя (**username**), электронная почта (**email**) и приватный атрибут пароль (**password**).
2. Используйте конструктор **\_\_init\_\_** для инициализации этих атрибутов.
3. Реализуйте метод **set\_password**(**new\_password**), который позволяет безопасно изменить пароль аккаунта.
4. Реализуйте метод **check\_password**(**password**), который проверяет, соответствует ли введённый пароль текущему паролю аккаунта и возвращает **True** или **False**.
5. Создайте объект класса **UserAccount**, попробуйте изменить пароль и проверить его с помощью методов **set\_password** и **check\_password**.

Задание 2: Полиморфизм и наследование

1. Определите базовый класс **Vehicle** с атрибутами: **make** (марка) и **model** (модель), а также методом **get\_info**(), который возвращает информацию о транспортном средстве.
2. Создайте класс **Car**, наследующий от **Vehicle**, и добавьте в него атрибут **fuel\_type** (тип топлива). Переопределите метод **get\_info**() таким образом, чтобы он включал информацию о типе топлива.

**Ход работы:**

Код программы 1:

class UserAccount:  
 def \_\_init\_\_(self, username, email, password):  
 self.username = username  
 self.email = email  
 self.\_\_password = password  
  
 def set\_password(self, new\_password):  
 self.\_\_password = new\_password  
  
 def check\_password(self, password):  
 return self.\_\_password == password  
  
  
user = UserAccount('Futaba', 'oracle@mail.com', 'qwerty')  
user.set\_password('Wakaba\_Isshiki')  
print(user.check\_password('qwerty'))

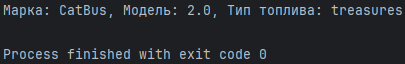
Консоль:



Код программы 2:

class Vehicle:  
 def \_\_init\_\_(self, make, model):  
 self.make = make  
 self.model = model  
  
 def get\_info(self):  
 return f"Марка: {self.make}, Модель: {self.model}"  
  
  
class Car(Vehicle):  
 def \_\_init\_\_(self, make, model, fuel\_type):  
 super().\_\_init\_\_(make, model)  
 self.fuel\_type = fuel\_type  
  
 def get\_info(self):  
 return f"{super().get\_info()}, Тип топлива: {self.fuel\_type}"  
  
  
car = Car('CatBus', 2.0, 'treasures')  
print(car.get\_info())

Консоль:



**Вывод**: Мы получили практический опыт работы с ООП в Python, использования инкапсуляции, наследования.