МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра “Математическая кибернетика и информационные технологии”

ОТЧЕТ

по лабораторной работе №7

по дисциплине «Введение в информационные технологии»

Тема: «Работа с классами ч.3»

Выполнил: студент группы БВТ2505

Коротков Артём Сергеевич

Проверил: Павликов. А.Е.

Москва, 2025

Цель работы:

Разработать систему управления сотрудниками, демонстрирующую множественное наследование, инкапсуляцию и полиморфизм в Python. Система должна уметь обрабатывать различные типы сотрудников, включая менеджеров и технических специалистов, а также предоставлять возможность для расширения и добавления новых ролей.

Задание:

Задание 1:

1. Создайте класс Employee с общими атрибутами, такими как name (имя), id (идентификационный номер) и методами, например, get_info(), который возвращает базовую информацию о сотруднике.
2. Создайте класс Manager с дополнительными атрибутами, такими как department (отдел) и методами, например, manage_project(), символизирующим управление проектами.
3. Создайте класс Technician с уникальными атрибутами, такими как specialization (специализация), и методами, например, perform_maintenance(), означающим выполнение технического обслуживания.
4. Создайте класс TechManager, который наследует как Manager, так и Technician. Этот класс должен комбинировать управленческие способности и технические навыки, например, иметь методы для управления проектами и выполнения технического обслуживания.
5. Добавьте метод add_employee(), который позволяет TechManager добавлять сотрудников в список подчинённых.
6. Реализуйте метод get_team_info(), который выводит информацию о всех подчинённых сотрудниках.
7. Создайте объекты каждого класса и демонстрируйте их функциональность

Скриншоты выполнения:

Задача 1:

```
Имя сотрудника: Егор, id сотрудника: 001
Имя сотрудника: Костя, id сотрудника: 002
Костя руководит проектами в отделе Продажи
Имя сотрудника: Аля, id сотрудника: 003
Аля выполняет техническое обслуживание по специализации: Сети
Имя сотрудника: Тимур, id сотрудника: 004
Тимур руководит проектами в отделе ИТ
Тимур выполняет техническое обслуживание по специализации: Системы
Информация о команде TechManager's :
Имя сотрудника: Тимур, id сотрудника: 004
Имя сотрудника: Егор, id сотрудника: 001
Имя сотрудника: Костя, id сотрудника: 002
Имя сотрудника: Аля, id сотрудника: 003
```

Исходный код программы:

Задача 1:

```
class Employee:
    def __init__(self, name, id):
        self.name = name
        self.id = id
    def get_info(self):
        return f"Имя сотрудника: {self.name}, id сотрудника: {self.id}"
class Manager(Employee):
    def __init__(self, name, id, department):
        Employee.__init__(self, name, id)
        self.department = department
    def manage_project(self):
        return f"{self.name} руководит проектами в отделе {self.department}"
class Technician(Employee):
    def __init__(self, name, id, specialization):
        Employee.__init__(self, name, id)
        self.specialization = specialization
    def perform_maintenance(self):
        return f"{self.name} выполняет техническое обслуживание по специализации: {self.specialization}"
class TechManager(Manager, Technician):
    def __init__(self, name, id, department, specialization):
        Manager.__init__(self, name, id, department)
        Technician.__init__(self, name, id, specialization)
        self.team = []
    def add_employee(self, employee):
        self.team.append(employee)
    def get_team_info(self):
        return "\n".join(emp.get_info() for emp in self.team)
emp_1 = Employee("Егор", "001")
print(emp_1.get_info())
```

```
emp_2 = Manager("Костя", "002", "Продажи")
print(emp_2.get_info())
print(emp_2.manage_project())
tech_emp_1 = Technician("Аля", "003", "Сети")
print(tech_emp_1.get_info())
print(tech_emp_1.perform_maintenance())
tech_emp_2 = TechManager("Тимур", "004", "ИТ", "Системы")
print(tech_emp_2.get_info())
print(tech_emp_2.manage_project())
print(tech_emp_2.perform_maintenance())
tech_emp_2.add_employee(tech_emp_2)
tech_emp_2.add_employee(emp_1)
tech_emp_2.add_employee(emp_2)
tech_emp_2.add_employee(tech_emp_1)
print("Информация о команде TechManager's :")
print(tech_emp_2.get_team_info())
```

Заключение

В ходе выполнения лабораторной работы были успешно решены следующие задачи:

Разработана система управления сотрудниками, демонстрирующая множественное наследование, инкапсуляцию и полиморфизм в Python.