Министерство цифрового развития и массовых коммуникаций
Российской Федерации

Ордена Трудового Красного Знамени федеральное государственное
бюджетное образовательное учреждение
высшего образования
«Московский технический университет связи и информатики»
(МТУСИ)

Кафедра Информационных технологий

Отчет по лабораторной работе № 7

по дисциплине «Введение в информационные технологии»

на тему: «Работа с классами»

Выполнил: студент группы БВТ2402
Левая Валерия Валерьевна

Москва 2024

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.

Ход работы:

Задание:

1. Создадим класс Employee с общими атрибутами, такими как **name** (имя), **id** (идентификационный номер) и методами, например, **get_info**(), который возвращает базовую информацию о сотруднике.
2. Создадим класс **Manager** с дополнительными атрибутами, такими как **department** (отдел) и методами, например, **manage_project**(), символизирующим управление проектами.
3. Создадим класс **Technician** с уникальными атрибутами, такими как **specialization** (специализация), и методами, например, **perform_maintenance**(), означающим выполнение технического обслуживания.
4. Создадим 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. Добавим метод **add_employee**(), который позволяет **TechManager** добавлять сотрудников в список подчинённых.
6. Реализуем метод **get_team_info**(), который выводит информацию о всех подчинённых сотрудниках.
7. Создадим объекты каждого класса и продемонстрируем их функциональность.

Код:

done.py

done.py > Manager > manage_project > project_name

```python
class Employee:
    def __init__(self, name, id):
        self.name = name
        self.id = id

    def get_info(self):
        return f'Имя сотрудника: {self.name}, его идентификационный номер: {self.id}'


class Manager(Employee):
    def __init__(self, name, id, departament):
        Employee.__init__(self, name, id)
        self.departament = departament

    def manage_project(self, project_name):
        return f'Сотрудник {self.name} управляет проектом {project_name} в {self.departament} в отделе)'


class Technician(Employee):
    def __init__(self, name, id, specialization):
        Employee.__init__(self, name, id)
        self.specialization = specialization

    def perform_maintenance(self, equipment):
        return f'Сотрудник {self.name} со специализацией {self.specialization} выполняет техническое обслуживание {equipment}'

class TechManager(Manager, Technician):
    def __init__(self, name, id, departament, specialization):
        Manager.__init__(self, name, id, departament)
        Technician.__init__(self, name, id, specialization)
        self.team = []

    def add_employee(self, employee):
        self.team.append(employee)

    def get_team_info(self):
        if not self.team:
            return f'У {self.name} еще нет подчиненных'
        team_info = [employee.get_info() for employee in self.team]
        return f'Команда {self.name}:\n' + '\n'.join(team_info)


if __name__ == '__main__':
    employee = Employee('Икром', 1)
    manager = Manager('Егор', 2, 'Акции')
    technician = Technician('Матвей', 3, 'Электрика')
    tech_manager = TechManager('Владос', 4, 'Продажи', 'Механик')

    print(employee.get_info())
    print(manager.get_info())
    print(technician.get_info())
    print(tech_manager.get_info())

    print(manager.manage_project('Увеличение продаж'))
    print(technician.perform_maintenance('Компьютера'))

    tech_manager.add_employee(employee)
    tech_manager.add_employee(manager)
    tech_manager.add_employee(technician)

    print(tech_manager.get_team_info())
```

Итог:

```
& C:/Users/levaa/AppData/Local/Programs/Python/Python312/python.exe "c:/Users/levaa/OneDrive/Рабочий стол/ввикт/l7/done.py"
Имя сотрудника: Икром, его идентификационный номер: 1
Имя сотрудника: Егор, его идентификационный номер: 2
Имя сотрудника: Матвей, его идентификационный номер: 3
Имя сотрудника: Владос, его идентификационный номер: 4
Сотрудник Егор управляет проектом Увеличение продаж в Акции в отделе)
Сотрудник Матвей со специализацией Электрика выполняет техническое обслуживание Компьютера
Команда Владос:
Имя сотрудника: Икром, его идентификационный номер: 1
Имя сотрудника: Егор, его идентификационный номер: 2
Имя сотрудника: Матвей, его идентификационный номер: 3
PS C:\Users\levaa\OneDrive\Рабочий стол\ввикт\l7>
```

Вывод: Я разработала систему управления сотрудниками, демонстрирующую множественное наследование, инкапсуляцию и полиморфизм в Python.