РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

«Наследование и полиморфизм в языке Python»

Отчет по лабораторной работе № 4.3

по дисциплине «Объектно-ориентированное программирование»

Выполнил студент группы ИВТ-б-о-21-1

Стригалов Дмитрий.

« » _____ 20__г.

Подпись студента_____

Работа защищена « »_____20__г.

Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2023

Цель работы: приобретение навыков по созданию иерархии классов при написании программ с помощью языка программирования Python версии 3.x.

Порядок выполнения работы:

Задание 1.

Разработайте программу по следующему описанию. В некой игрестратегии есть солдаты и герои. У всех есть свойство, содержащее уникальный номер объекта, и свойство, в котором хранится принадлежность команде. У солдат есть метод "иду за героем", который в качестве аргумента принимает объект типа "герой". У героев есть метод увеличения собственного уровня. В основной ветке программы создается по одному герою для каждой команды. В цикле генерируются объекты-солдаты. Их принадлежность команде определяется случайно. Солдаты разных команд добавляются в разные списки. Измеряется длина списков солдат противоборствующих команд и выводится на экран. У героя, принадлежащего команде с более длинным списком, увеличивается уровень. Отправьте одного из солдат первого героя следовать за ним. Выведите на экран идентификационные номера этих двух юнитов.

Код программы:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import random


class Soldier:
    def __init__(self, number, team):
        self.number = number
        self.team = team

    def go_to_hero(self, hero):
        print(f"Солдат {self.number} идет за героем {hero.number}")


class Hero:
    def __init__(self, number):
        self.number = number
        self.level = 1

    def increase_level(self):
        self.level += 1


if __name__ == "__main__":
    hero1 = Hero(1)
    hero2 = Hero(2)

    soldiers_team1 = []
    soldiers_team2 = []
```

```
    for _ in range(10):
        number = random.randint(1, 100)
        team = random.choice([1, 2])
        soldier = Soldier(number, team)

        if soldier.team == 1:
            soldiers_team1.append(soldier)
        else:
            soldiers_team2.append(soldier)

    if len(soldiers_team1) > len(soldiers_team2):
        hero1.increase_level()
    else:
        hero2.increase_level()

    soldier_to_follow = random.choice(soldiers_team1)
    soldier_to_follow.go_to_hero(hero1)

    print(f"Идентификационный номер солдата: {soldier_to_follow.number}")
    print(f"Идентификационный номер героя: {hero1.number}")
```

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import random


class Soldier:
    def __init__(self, number, team):
        self.number = number
        self.team = team

    def go_to_hero(self, hero):
        print(f"Солдат {self.number} идет за героем {hero.number}")


class Hero:
    def __init__(self, number):
        self.number = number
        self.level = 1

    def increase_level(self):
        self.level += 1


if __name__ == "__main__":
    hero1 = Hero(1)
    hero2 = Hero(2)

    soldiers_team1 = []
    soldiers_team2 = []

    for _ in range(10):
        number = random.randint(1, 100)
        team = random.choice([1, 2])
        soldier = Soldier(number, team)

        if soldier.team == 1:
            soldiers_team1.append(soldier)
        else:
            soldiers_team2.append(soldier)
```

```
if len(soldiers_team1) > len(soldiers_team2):
    hero1.increase_level()
else:
    hero2.increase_level()

soldier_to_follow = random.choice(soldiers_team1)
soldier_to_follow.go_to_hero(hero1)

print(f"Идентификационный номер солдата: {soldier_to_follow.number}")
print(f"Идентификационный номер героя: {hero1.number}")
```

Рисунок 1 - Результат выполнения задания 1

Индивидуальное задание 1. Вариант 16

Создать класс Triad (тройка чисел); определить методы увеличения полей на 1. Определить класс-наследник Time с полями: час, минута, секунда. Пере определить методы увеличения полей на 1 и определить методы увеличения на n секунд и минут.

```
Тройка чисел: 1, 2, 3
Время: 10 часов, 25 минут, 40 секунд
Тройка чисел: 2, 3, 4
Время: 11 часов, 26 минут, 41 секунд
Время: 11 часов, 27 минут, 51 секунд
```

Рисунок 2- Результат выполнения индивидуального задания 1

Индивидуальное задание 2.

Создать абстрактный базовый класс Array с виртуальными методами сложения и

поэлементной обработки массива foreach( ). Разработать производные классы SortArray и XorArray. В первом классе операция сложения реализуется как объединение множеств, а поэлементная обработка — сортировка. Во втором классе операция сложения реализуется как исключающее ИЛИ, а поэлементная обработка — вычисление корня.

```
Тройка чисел: 1, 2, 3
Время: 10 часов, 25 минут, 40 секунд
Тройка чисел: 2, 3, 4
Время: 11 часов, 26 минут, 41 секунд
Время: 11 часов, 27 минут, 51 секунд
```

Рисунок 3- Результат выполнения индивидуального задания 2

Контрольные вопросы:

1. Что такое наследование и как оно реализовано в языке Python?

Наследование — это когда один класс (подкласс) получает свойства и методы другого класса (суперкласса). Подкласс может наследовать все публичные атрибуты и методы своего суперкласса и добавлять свои собственные. В языке Python наследование реализуется с помощью ключевого слова class. Для создания подкласса нужно указать имя суперкласса в скобках после имени подкласса. Подкласс получает все атрибуты и методы суперкласса, их можно использовать напрямую или переопределить.

2. Что такое полиморфизм и как он реализован в языке Python?

Полиморфизм — это возможность объектов разных классов иметь одно и то же имя метода, но каждый класс может предоставить свою собственную реализацию этого метода. Это позволяет использовать одинаковое имя метода для объектов различных классов, что упрощает программирование и повышает гибкость кода. В языке Python полиморфизм реализуется через наследование и переопределение методов. Если в подклассе метод с тем же именем переопределяется, то при вызове этого метода на объекте подкласса будет использоваться его реализация, а не реализация суперкласса. Это позволяет использовать одинаковые методы с разным поведением для разных классов.

3. Что такое «утиная» типизация в языке Python?

«Утиная» типизация (англ. duck typing) — это концепция в языке программирования Python, основанная на философии «если она выглядит как утка, плавает как утка и крякает как утка, то это, вероятно, и есть утка». В контексте Python утиная типизация означает, что тип объекта определяется по его возможностям и методам, а не по его явно заданному типу. Иными словами, если объект обладает определенными методами, то мы можем использовать его

как экземпляр нужного типа, не задумываясь о его фактическом классе или интерфейсе.

4. Каково назначение модуля abc языка Python?

Модуль abc (аббревиатура от "Abstract Base Classes") является частью стандартной библиотеки языка Python и предоставляет средства для определения абстрактных базовых классов.

5. Как сделать некоторый метод класса абстрактным?

Необходимо декорировать его методы как абстрактные, а реализацию выносить в классы-наследники.

6. Как сделать некоторое свойство класса абстрактным?

Можно потребовать атрибут в конкретных классах, определив их с помощью @abstractproperty.

7. Каково назначение функции isinstance?

Функция isinstance() проверяет, является ли объект экземпляром указанного класса или его подкласса.

Вывод: в ходе работы были приобретены навыки по созданию иерархии классов при написании программ с использованием языка программирования Python версии 3.10.