

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №3
дисциплины «Объектно-ориентированное программирование»
Вариант 3

Выполнил:
Говоров Егор Юрьевич
3 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем»,
очная форма обучения

(подпись)

Проверил:
Воронкин Роман Александрович,
доцент департамента цифровых,
робототехнических систем и
электроники

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2024 г.

Тема: Наследование и полиморфизм в языке Python

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

Ссылка на репозиторий: https://github.com/Artorias1469/object-oriented-programming_3.git

Ход работы:

1. Проработал примеры:

```
D:\Users\Admin\anaconda3\envs\object-oriented-programming_3\python.exe
3/4
Введите обыкновенную дробь: 5/8
5/8
11/8
1/8
15/32
5/6
```

Рис 1. Работа программы из примера 1

```
D:\Users\Admin\anaconda3\envs\object-oriented-programming_3\python.exe
I have 3 sides
I have 4 sides
I have 5 sides
I have 6 sides
```

Рис 2. Работа программы из примера 2

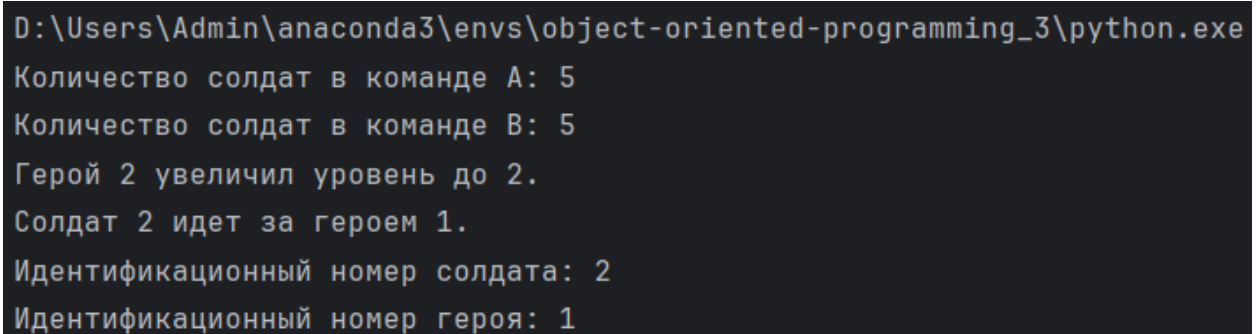
```
D:\Users\Admin\anaconda3\envs\object-oriented-programming_3\python.exe
I can walk and run
I can crawl
I can bark
I can roar
```

Рис 3. Работа программы из примера 3

2. Выполнение задачи про солдатиков и героев:

В некой игре-стратегии 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.



```
D:\Users\Admin\anaconda3\envs\object-oriented-programming_3\python.exe
Количество солдат в команде A: 5
Количество солдат в команде B: 5
Герой 2 увеличил уровень до 2.
Солдат 2 идет за героем 1.
Идентификационный номер солдата: 2
Идентификационный номер героя: 1
```

Рис 4. Работа программы hero-soldier

3. Выполнение индивидуальных заданий

Составить программу с использованием иерархии классов. Номер варианта необходимо получить у преподавателя. В раздел программы, начинающийся после инструкции `if __name__ == '__main__':` добавить код, демонстрирующий возможности разработанных классов.

Создать класс `Liquid` (жидкость), имеющий поля названия и плотности. Определить методы переназначения и изменения плотности. Создать производный класс `Alcohol` (спирт), имеющий крепость. Определить методы переназначения и изменения крепости.

```
D:\Users\Admin\anaconda3\envs\object-oriented-programming_3\python.exe
Жидкость: название=Вода, плотность=997 кг/м³
Жидкость: название=Дистиллированная вода, плотность=998 кг/м³
Спирт: название=Этанол, плотность=789 кг/м³, крепость=95%
Спирт: название=Этанол, плотность=789 кг/м³, крепость=70%
```

Рис 5. Выполнение индивидуальной программы 1

Требуется реализовать абстрактный базовый класс, определив в нем абстрактные методы и свойства. Эти методы определяются в производных классах. В базовых классах должны быть объявлены абстрактные методы ввода/вывода, которые реализуются в производных классах.

Вызывающая программа должна продемонстрировать все варианты вызова переопределенных абстрактных методов. Написать функцию вывода, получающую параметры базового класса по ссылке и демонстрирующую виртуальный вызов.

Создать абстрактный базовый класс Body (тело) с абстрактными функциями вычисления площади поверхности и объема. Создать производные классы: Parallelepiped (параллелепипед) и Ball (шар) со своими функциями площади поверхности и объема.

```
D:\Users\Admin\anaconda3\envs\object-oriented-programming_3\python.exe "D:\Users\
Параллелепипед:
Параллелепипед: длина=3, ширина=4, высота=5
Площадь поверхности: 94 кв.ед., Объем: 60 куб.ед.

Шар:
Шар: радиус=7
Площадь поверхности: 615.7521601035994 кв.ед., Объем: 1436.7550402417319 куб.ед.
```

Рис 6. Выполнение индивидуальной программы 2

Ответы на вопросы:

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

Наследование — это механизм, позволяющий создать новый класс на основе существующего, унаследовав его атрибуты и методы. В Python наследование реализуется путем указания родительского класса в скобках после имени нового класса: `class ChildClass(ParentClass)`. Python поддерживает как одиночное, так и множественное наследование.

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

Полиморфизм позволяет методам объектов разных классов обрабатывать вызовы методов с одинаковыми именами по-разному. В Python полиморфизм достигается за счет переопределения методов в дочерних классах и использования базовых классов с виртуальными методами.

3. Что такое "утиная" типизация в Python?

Утиная типизация — это принцип, когда тип объекта определяется не его принадлежностью к классу, а наличием нужных методов и атрибутов. В Python этот принцип выражается в том, что методы могут работать с объектами разных классов, если у них есть требуемые методы или атрибуты.

4. Каково назначение модуля abc в Python?

Модуль abc предоставляет инструменты для создания абстрактных базовых классов, которые содержат объявления методов, не имеющие реализации. Этот модуль позволяет определять методы, которые обязаны реализовать дочерние классы.

5. Как сделать метод класса абстрактным?

Чтобы сделать метод абстрактным, необходимо импортировать декоратор `@abstractmethod` из модуля abc и применить его к методу. Например:

python

Копировать код

```
from abc import ABC, abstractmethod
class MyAbstractClass(ABC):
    @abstractmethod
    def my_method(self):
        pass
```

6. Как сделать свойство класса абстрактным?

Абстрактное свойство создается путем использования декоратора

`@abstractmethod` с `@property`. Пример:

python

Копировать код

```
from abc import ABC, abstractmethod
```

```
class MyAbstractClass(ABC):
```

```
    @property
```

```
    @abstractmethod
```

```
    def my_property(self):
```

```
        pass
```

7. Каково назначение функции `isinstance`?

`isinstance` проверяет, является ли объект экземпляром указанного класса или его подкласса.