Классы и перегрузка операторов в языке Python

Лабораторная работа 5

2021

Ход работы

1. Разработать класс Sphere для представления сферы в трехмерном пространстве.
Реализовать следующие методы класса:

- конструктор, принимающий 4 действительных числа: радиус, и 3 координаты центра шара. Если конструктор вызывается без аргументов, создать объект сферы с единичным радиусом и центром в начале координат. Если конструктор вызывается с 1 аргументом, создать объект сферы с соответствующим радиусом и центром в начале координат.
- метод get_volume (), который возвращает действительное число — объем шара, ограниченной текущей сферой.
- метод get_square (), который возвращает действительное число — площадь внешней поверхности сферы.
- метод get_radius (), который возвращает действительное число — радиус сферы.
- метод get_center (), который возвращает тьюпл с 3 действительными числами — координатами центра сферы в том же порядке, в каком они задаются в конструкторе.
- метод set_radius (r), который принимает 1 аргумент — действительное число, и меняет радиус текущей сферы, ничего не возвращая.
- метод set_center (x, y, z), который принимает 3 аргумента — действительных числа, и меняет координаты центра сферы, ничего не возвращая. Координаты задаются в том же порядке, что и в конструкторе.
- метод is_point_inside (x, y, z), который принимает 3 аргумента — действительных числа — координаты некоторой точки в пространстве (в том же порядке, что и в конструкторе), и возвращает логическое значение True или False в зависимости от того, находится эта точка внутри сферы.

2. **Объявите класс Point, который реализует точку на координатной плоскости.**
В классе должны быть реализованы следующие методы:

- метод __init __(), который перегружает конструктор класса. Конструктор класса вызывается при создании экземпляра класса;
- метод getXY(), который возвращает значение координаты точки (x, y) в виде списка;
- метод Show() — выводит координаты точки на экран.

Создайте экземпляр класса Point. Получите у точки значения координат. Выведите координаты на экран.

3. **На основе задания 2 добавьте метод __add __() — который реализует перегрузку оператора сложения +.**
Реализация предусматривает суммирование координат по осям X и Y;
Создайте два объекта и получите третий на основе использования оператора «+».

4. Исправьте класс Point так, что бы была реализованна перегрузка операторов «-», «*», «/», «==».
Все операторы применяются попарно к координатам. Например A/B приведет к C = (A.x/B.x; A.y/B.y);
Проверить использование каждого оператора. Выполнить сравнение двух объектов типа Point.

5. Реализуйте класс Drob, который представляет дроби в виде целых числителя и знаменателя. Выполните следующие условия:

- Конструктор принимает два числа: числитель и знаменатель;
- Метод *normalize* приводит дробь к нормальному виду (Пример: 4/6 к 2/3)
- Метод print при получении объекта такого класса должен вывести дробь в виде a/b;
- Реализуйте перегрузки операторов __add__, __sub__, __mul__, __div__ и проверьте их работу;
- Дробь можно сравнивать с обычными числами. Перегрузите операторы >,>=,<,<=, == и проверьте их работу.

6. Напишите класс Snow по следующему описанию:

- В конструкторе класса инициируется поле, содержащее количество снежинок, выраженное целым числом.

- Класс включает методы перегрузки арифметических операторов: __add__() – сложение, __sub__() – вычитание, __mul__() – умножение, __truediv__() – деление. В классе код этих методов должен выполнять увеличение или уменьшение количества снежинок на число n или в n раз. Метод __truediv__() перегружает обычное (/), а не целочисленное (//) деление. Однако пусть в методе происходит округление значения до целого числа.
- Класс включает метод makeSnow(), который принимает сам объект и число снежинок в ряду, а возвращает строку вида "*****\n*****\n*****…", где количество снежинок между '\n' равно переданному аргументу, а количество рядов вычисляется, исходя из общего количества снежинок.

7. Создайте класс Human, который удовлетворяет следующим условиям:
- Определите для него два статических поля: default_name и default_age.
- Создайте метод __init__(), который помимо self принимает еще два параметра: name и age. Для этих параметров задайте значения по умолчанию, используя свойства default_name и default_age. В методе __init__() определите четыре свойства: Публичные - name и age. Приватные - money и property.
- Реализуйте справочный метод info(), который будет выводить поля name, age, property и money.
- Реализуйте справочный статический метод default_info(), который будет выводить статические поля default_name и default_age.
- Реализуйте приватный метод make_deal(), который будет отвечать за покупки : уменьшать количество денег на счету. В качестве аргументов данный метод принимает строку с описанием покупки property и цену.
- Реализуйте метод earn_money(), увеличивающий значение свойства money.
- Реализуйте метод buy(), который будет проверять, что у человека достаточно денег для покупки, и совершать сделку. Если денег слишком мало - нужно вывести предупреждение в консоль.
- Создайте два таких объекта, проведите покупки, проверьте работоспособность методов.
- Перегрузите нужные операторы так, что бы объект у которого длинна имени + возраст были больше или равны объекта у которого данные показатели меньше.
- Выполните перегрузку операторов «+=», «-=», «*=», «/=» так, что бы они влияли на поле money в рамках одного объекта. Т.е. к примеру для увеличения состояния объекта применить оператор «+=».