**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра «Математическая Кибернетика и Информационные технологии»

Лабораторная работа №2

Изучение принципов ООП

Выполнил: Студент группы

БВТ2303

Кунецкий Владислав

Москва

2024

Цели работы:

- познакомиться и изучить основные концепции ООП

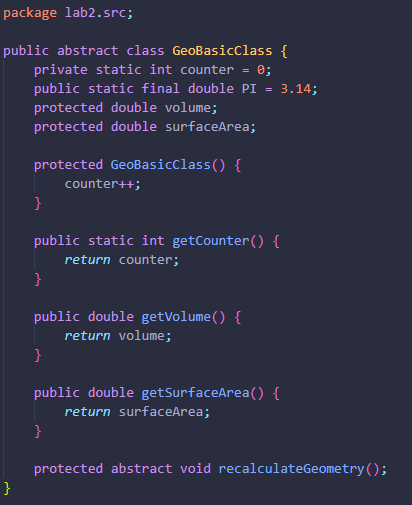
- применить полученные знания на практике, написав свою иерархию классов

Ход работы:

В задании мне попалась следующая иерархия классов:

* Базовый класс: Геометрическая фигура,
* Дочерние классы: Шар, Параллелепипед, Цилиндр

Сначала я создал абстрактный класс, который включал общие методы и атрибуты, присущие любой фигуре (в 3д пространстве).



Скрин 1 – Базовый класс

Я определил следующие методы и атрибуты:

- публичную статическую константу PI. Публичная, потому что в приватности нет необходимости, статическая для того, чтобы она существовала и могла использоваться без создания экзмепляров.

- Обьем и площадь поверхности. Они присуще любой 3д геометрической фигуре. Модификатор protected использован для того, чтобы доступ к этим атрибутам имели другие классы в нашем пакете, описывающие фигуры и при наследование они перенимали их. За пределами пакета доступ к ним запрещен.

- Счетчик определил по заданию. Приватный потому что мы не хотим, чтобы его можно было изменить при прямом обращении к классу и также не хотим, чтобы при наследовании создавался еще один счетчик. Cтатический потому что класс абстрактный, а поле существовать в памяти должно.

-Публичные геттер и сеттеры для объема и площади для получение и изменения их.

-Статический геттер для получения значения счетчика

-Конструктор, который при создании экземпляра класс, унаследовавшего наш абстрактный класс, увеличивает счетчик на 1

-абстрактный метод, который должен переопределить каждый класс, уноследовавший данный.

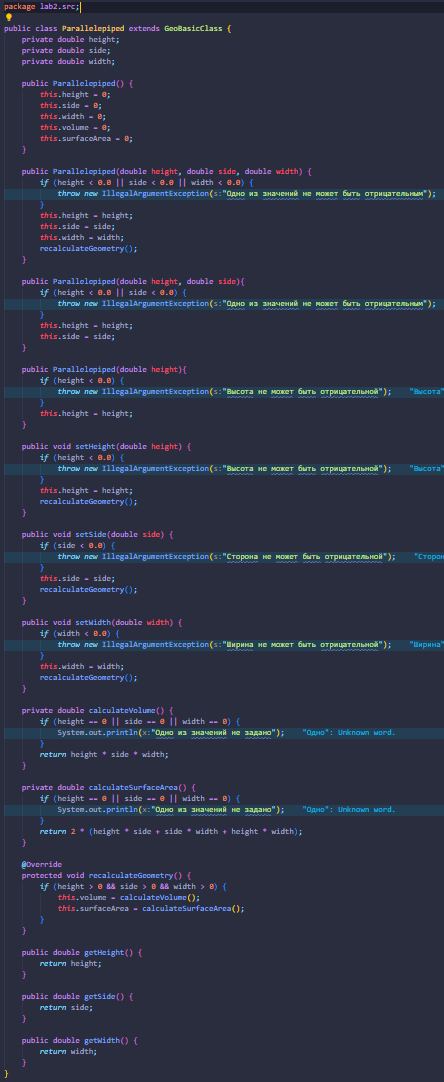
Следует также отменить, что я создал класс с модификатором abstract. Этот модификатор запрещает создание экземпляров и по сути делает класс базовым, для других классов. То, есть он нужен либо для наследования, либо для использования статический методов и переменных, определенных в нем.



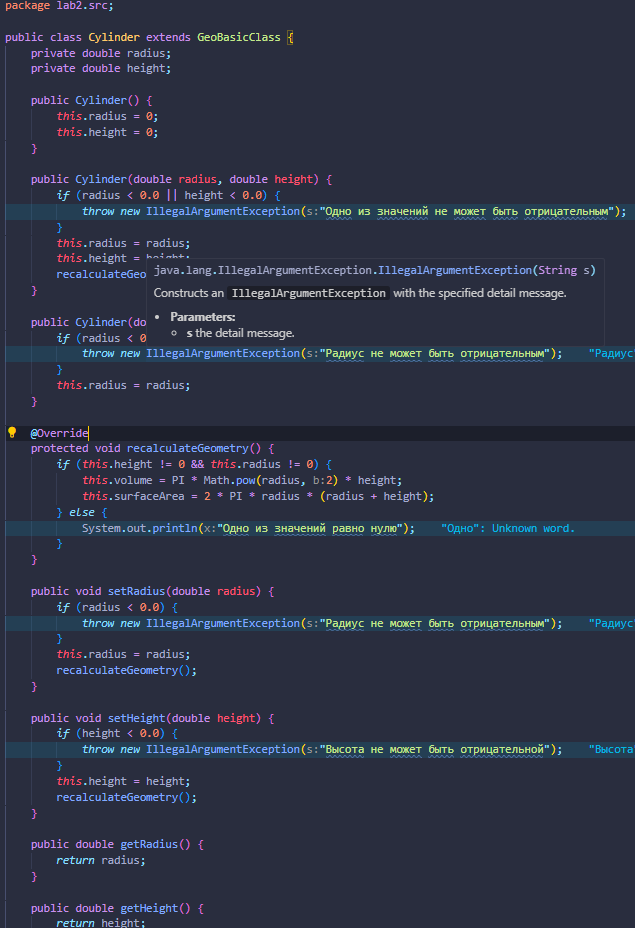
Скрин 2 – Класс, описывающий сферу (шар)

Далее начал создавать классы фигур. С помощью extend наследуем атрибуты и методы нашего базового класса.

В классе я определил атрибут, присущий только шару, и это радиус(приватный, потому что мы не хотим, чтобы его можно было изменить при прямом обращении). Определил 2 конструктора, используя перегрузку. Переопределил абстрактный метод для шарика. Написал сеттер и геттер. Также, с помощью ключевого слово throw определил класс ошибки.



Скрин 3 – Класс параллелепипеда.



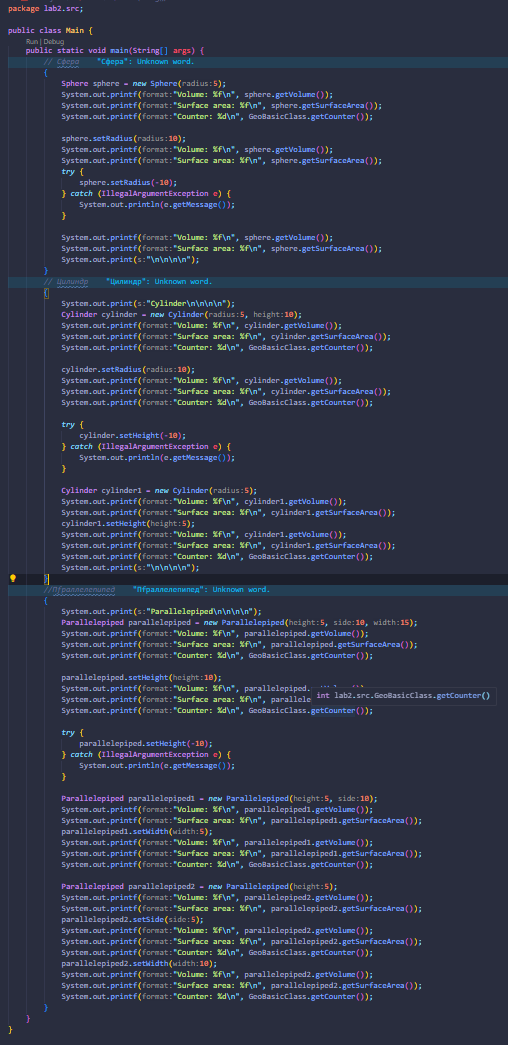
Скрин 4 – Цилиндр.

В классах, описывающих цилиндр и параллелепипед, проводим те же действия.

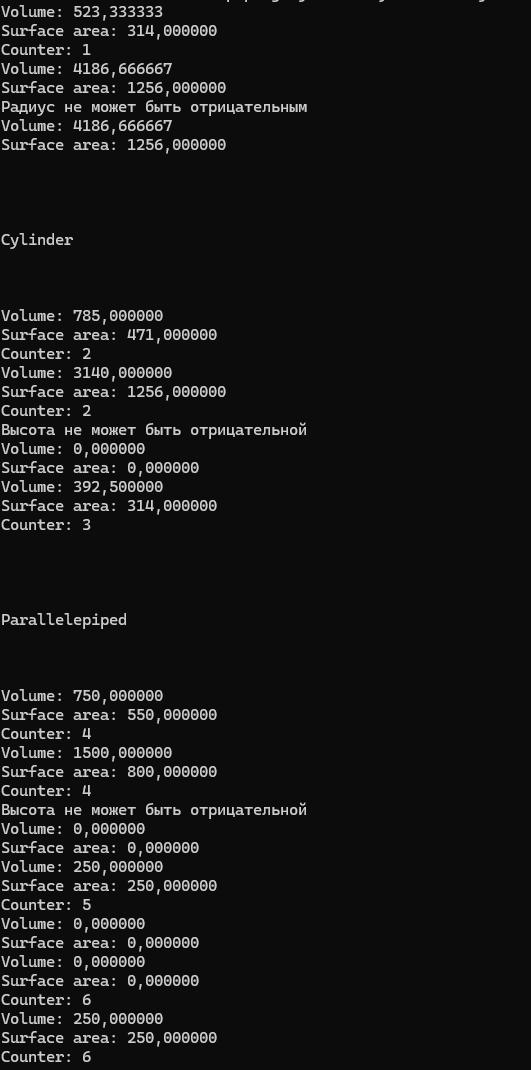
Иерархия классов готова. Теперь протестируем ее. Ниже, на 5 скрине, размещен код, для теста нашей иерархии. Я не смог проверить все возможные случае, но постарался предоставить как можно больше вариантов.

Так как в классах у нас определены исключения, мы можем ловить их с помощью конструкции try catch.

Результаты тестов можно посмотреть в скрине 6



Скрин 5 - Тест



Скрин 6 – Результаты тестов