1. Класс "Точка на плоскости"

- **Инвариант**: Координаты точки должны быть действительными числами.
- **Наследование**: Создайте класс Point3D, который наследует от Point2D и добавляет третью координату.
- **LSP**: Убедитесь, что все методы Point2D (например, вычисление расстояния до другой точки) работают корректно для Point3D, игнорируя третью координату.
- **Методы**:
  - Расстояние между двумя точками.
  - Проверка, лежит ли точка на заданной прямой.

---

2. Класс "Вектор"

- **Инвариант**: Компоненты вектора должны быть действительными числами.
- **Наследование**: Создайте класс Vector3D, который наследует от Vector2D и добавляет третью компоненту.
- **LSP**: Убедитесь, что методы Vector2D (например, скалярное произведение) работают корректно для Vector3D, игнорируя третью компоненту.
- **Методы**:
  - Скалярное произведение.
  - Векторное произведение (для 3D).
  - Длина вектора.

---

3. Класс "Матрица"

- **Инвариант**: Элементы матрицы должны быть действительными числами, а размерность матрицы должна быть положительной.
- **Наследование**: Создайте класс SquareMatrix, который наследует от Matrix и добавляет методы, специфичные для квадратных матриц.
- **LSP**: Убедитесь, что методы Matrix (например, умножение матриц) работают корректно для SquareMatrix.
- **Методы**:
  - Умножение матриц.
  - Нахождение определителя (для квадратных матриц).
  - Транспонирование матрицы.

---

4. Класс "Многоугольник"

- **Инвариант**: Количество вершин многоугольника должно быть не менее 3. Никакие три точки не должны лежать на одной прямой.
- **Наследование**: Создайте класс RegularPolygon, который наследует от Polygon и добавляет свойство равносторонности.
- **LSP**: Убедитесь, что методы Polygon (например, вычисление площади) работают корректно для RegularPolygon.
- **Методы**:
  - Вычисление площади.
  - Проверка, является ли многоугольник выпуклым.

---

5. Класс "Окружность"

- **Инвариант**: Радиус окружности должен быть положительным числом. Расстояние от центра окружности до любой точки окружности – величина постоянная.
- **Наследование**: Создайте класс Ellipse, который наследует от Circle и добавляет второй радиус.
- **LSP**: Убедитесь, что методы Circle (например, вычисление длины окружности) работают корректно для Ellipse, если оба радиуса равны.
- **Методы**:
  - Длина окружности.
  - Площадь.
  - Проверка, пересекаются ли две окружности.

---

6. Класс "Прямоугольник"

- **Инвариант**: Длины сторон должны быть положительными числами. Углы между сторонами равны.
- **Наследование**: Создайте класс Square, который наследует от Rectangle и добавляет ограничение на равенство сторон.
- **LSP**: Убедитесь, что методы Rectangle (например, вычисление площади) работают корректно для Square.
- **Методы**:
  - Вычисление диагонали.
  - Площадь.

---

7. Класс "Треугольник"

- **Инвариант**: Сумма длин любых двух сторон должна быть больше третьей. Вершина не лежат на одной прямой.
- **Наследование**: Создайте класс RightTriangle, который наследует от Triangle и добавляет свойство прямоугольности.
- **LSP**: Убедитесь, что методы Triangle (например, вычисление площади) работают корректно для RightTriangle.
- **Методы**:
  - Вычисление площади.
  - Проверка, является ли треугольник прямоугольным.

---

8. Класс "Парабола"

- **Инвариант**: Коэффициенты параболы должны быть действительными числами, а коэффициент при x^2 не должен быть равен нулю.
- **Наследование**: Создайте класс QuadraticFunction, который наследует от Parabola и добавляет методы для работы с квадратичными функциями.
- **LSP**: Убедитесь, что методы Parabola (например, нахождение вершины) работают корректно для QuadraticFunction.
- **Методы**:
  - Нахождение вершины.
  - Вычисление дискриминанта.

---

9. Класс "Комплексное число"

- **Инвариант**: Действительная и мнимая части должны быть действительными числами.
- **Наследование**: Создайте класс Quaternion, который наследует от ComplexNumber и добавляет две дополнительные компоненты.
- **LSP**: Убедитесь, что методы ComplexNumber (например, сложение) работают корректно для Quaternion, если дополнительные компоненты равны нулю.
- **Методы**:
  - Сложение и умножение.
  - Нахождение модуля.

---

10. Класс "Линейное уравнение"

- **Инвариант**: Коэффициенты уравнения должны быть действительными числами. Коэффициент при неизвестной не должен быть равен 0.
- **Наследование**: Создайте класс SystemOfLinearEquations, который наследует от LinearEquation и добавляет методы для работы с системами уравнений.
- **LSP**: Убедитесь, что методы LinearEquation (например, решение уравнения) работают корректно для SystemOfLinearEquations, если система состоит из одного уравнения.
- **Методы**:
  - Решение уравнения.
  - Проверка, имеет ли система уравнений решение.

---

11. Класс "Эллипс"

- **Инвариант**: Оба радиуса должны быть положительными числами.
- **Наследование**: Создайте класс Circle, который наследует от Ellipse и добавляет ограничение на равенство радиусов.
- **LSP**: Убедитесь, что методы Ellipse (например, вычисление площади) работают корректно для Circle.
- **Методы**:
  - Вычисление площади.
  - Проверка, является ли эллипс окружностью.

---

12. Класс "Гипербола"

- **Инвариант**: Коэффициенты гиперболы должны быть действительными числами, а коэффициенты при x^2 и y^2 должны иметь разные знаки.
- **Наследование**: Создайте класс ConicSection, который наследует от Hyperbola и добавляет методы для работы с коническими сечениями.
- **LSP**: Убедитесь, что методы Hyperbola (например, нахождение асимптот) работают корректно для ConicSection, если объект является гиперболой.
- **Методы**:
  - Нахождение асимптот.
  - Проверка, является ли гипербола равнобочной.