Give Class diagram for a Modified Shape Class Hierarchy

Modify base `Shape` class and several subclasses: `Rectangle`, `Square`, `Circle`, and `Ellipse`.

Shape must be abstract. All its methods must be abstract.

Modify other classes suitably.

Shape implements the Comparable interface

---

1. Functional Requirements

- **Shape Class (Base Class):**
  - It must have an attribute for **color**.
  - It must have **methods** for calculating `area()` and `perimeter()` (or `circumference` for circles and ellipses).
  - It must have a method `displayInfo()` that prints the shape's color and its calculated area and perimeter/circumference.
  - It must have a constructor to initialize the `color`.
- **Rectangle Class (Subclass):**
  - The `Rectangle` class must inherit from the `Shape` class.
  - It must have attributes for `length` and `width`.
  - It must implement the `area()` method, which calculates area as `length * width`.
  - It must implement the `perimeter()` method, which calculates perimeter as `2 * (length + width)`.
  - It must have a constructor to initialize the `length`, `width`, and inherited `color`.
- **Square Class (Subclass):**
  - The `Square` class must inherit from the `Rectangle` class. This demonstrates the **is-a** relationship (a square is a type of rectangle).
  - It must have an attribute for `side`.
  - Its constructor must initialize both `length` and `width` of the `Rectangle` base class to the `side` value.
- **Circle Class (Subclass):**
  - The `Circle` class must inherit from the `Shape` class.
  - It must have an attribute for `radius`.
  - It must implement the `area()` method, calculating area as `π * radius²`.
  - It must implement a `circumference()` method, calculating circumference as `2 * π * radius`.
  - It must have a constructor to initialize the `radius` and inherited `color`.

- **Ellipse Class (Subclass):**
  - The `Ellipse` class must inherit from the `Shape` class.
  - It must have attributes for `major_axis` and `minor_axis`.
  - It must implement the `area()` method, calculating area as `π * major_axis * minor_axis`.
  - It must implement a `perimeter()` method (which, for an ellipse, is an approximation). A common approximation is `2 * π * sqrt((a² + b²) / 2)`, where 'a' and 'b' are the semi-major and semi-minor axes. The program should use this or a similar formula.
  - It must have a constructor to initialize the `major_axis`, `minor_axis`, and inherited `color`.
- **Main Class:**
  - The main program must create instances of each concrete shape (`Rectangle`, `Square`, `Circle`, `Ellipse`).
  - It must store these shapes in a **list or array of type**

---

3. Constraints

- **Input validation** is required to ensure that dimensions like `length`, `width`, `radius`, `side`, and `axes` are **positive** numerical values.
- The constructors should handle invalid inputs gracefully, perhaps by raising an error or setting default values.