This module requires the creation of a Java class named Fan that encapsulates the properties and behaviors of a fan. The class should include constants for different speeds, private fields for the Fan's state, and methods to manipulate and retrieve these properties. Additionally, a test code should be provided to demonstrate the functionality of the Fan class. The Fan class is designed to represent a fan with various attributes and functionalities.

1. **Constants**: The class defines four constants (STOPPED, SLOW, MEDIUM, and FAST) that represent the different speeds of the fan. These constants are assigned integer values from 0 to 3.

A screen shot of a computer program

AI-generated content may be incorrect.

1. **Private Fields**: The class contains private fields:
   * speed: An integer that holds the current speed of the fan, initialized to STOPPED.
   * on: A boolean that indicates whether the fan is operational.
   * radius: A double representing the radius of the fan, defaulting to 6.
   * color: A string that holds the color of the fan, defaulting to "white".

A screen shot of a computer

AI-generated content may be incorrect.

1. **Constructors**:
   * The no-argument constructor initializes the fan with default values.
   * The parameterized constructor allows the creation of a fan with specified attributes.

A computer screen shot of a program code

AI-generated content may be incorrect.

1. **Getter and Setter Methods**: These methods provide access to the private fields, allowing for encapsulation and data integrity.

A screen shot of a computer program

AI-generated content may be incorrect.

1. **toString Method**: This method overrides the default toString() method to provide a meaningful string representation of the fan's state. It indicates whether the fan is on or off and displays its speed, radius, and color when it is on.

A black background with white text

AI-generated content may be incorrect.

1. **Test Code**: The main method serves as a test harness for the Fan class. It creates two instances of the Fan class: one using the default constructor and another using the parameterized constructor. The state of each fan is printed to the console, demonstrating the functionality of the class methods.

A screen shot of a computer code

AI-generated content may be incorrect.

The Fan class encapsulates the properties and behaviors of a fan effectively, providing a clear and concise implementation that adheres to object-oriented programming principles.

Output:

A screenshot of a computer program

AI-generated content may be incorrect.