1. Create a Book class with private attributes for title, author, and price. Implement getters and setters, ensuring that the price cannot be set to a negative value.

HINT

- Declare title, author, and price as private fields.
- Write a constructor that accepts and assigns all three values.
- In setPrice(), check if the incoming value is negative before assigning it.
- If the value is negative, print a message and leave the existing price unchanged.

2. Create a Student class containing an array of numerical marks. Implement a method to calculate the average mark and another to return a pass or fail status string.

HINT

- Loop through the marks array and sum all the values.
- Divide the sum by marks.length to get the average.
- Decide on a pass threshold (for example, average of 40 or above) and return “Pass” or “Fail” accordingly.

3. Design a Car class with attributes make, model, and currentSpeed. Implement accelerate() and brake() methods that change the speed safely.

   HINT

   - In accelerate(), add the given amount to currentSpeed.
   - In brake(), subtract the given amount, but check if the result would go below zero.
   - If braking would make the speed negative, set currentSpeed to 0 instead.

4. Create a Time class that stores hours, minutes, and seconds. Use setters to validate that inputs fall within correct time constraints, for example minutes and seconds must be between 0 and 59.

   HINT

   - Write a separate setter for each field: setHours(), setMinutes(), and setSeconds().
   - Validate hours to be between 0 and 23.
   - Validate minutes and seconds to be between 0 and 59.
   - Print a message and skip the assignment when a value fails validation.

5. Create a Patient class with name, age, and illness. Write three overloaded constructors and use this(...) to chain them so default values are assigned if some data is missing.

HINT

- Write the full constructor first, taking name, age, and illness.
- Write a constructor that takes only name, and have it call this(name, 0, "Not specified").
- Write a no-argument constructor that calls this("Unknown") so it chains through the second constructor.