Assignment 13 - final modifier

Exploring the `final` Modifier in Java

Objective:

To understand and practice the use of the `final` modifier in Java, including its usage with variables, methods, and classes.

Instructions:

Complete each problem by implementing Java code and answer any related questions.

Problem 1: Using `final` with Variables

1. Create a class `Car` with the following:
   - A `final` instance variable `MAX_SPEED` of type `int` initialized to 200.
   - An instance variable `currentSpeed` of type `int`.
   - A method `setSpeed(int speed)` that updates `currentSpeed`. If `speed` is greater than `MAX_SPEED`, set `currentSpeed` to `MAX_SPEED`.
2. Write a main class to create an instance of `Car`, attempt to set a speed greater than `MAX_SPEED`, and print the `currentSpeed`.

Questions:

- What happens if you try to change `MAX_SPEED` in the constructor or any method?
- Explain why `final` is useful here.

Problem 2: Using `final` with Methods

1. Create a base class `BankAccount` with:
   - An instance variable `balance` of type `double`.
   - A `final` method `displayBalance()` that prints the current balance.

- A method `deposit(double amount)` to increase the balance by the given amount.

2. Create a subclass `SavingsAccount` that extends `BankAccount`:
   - Attempt to override `displayBalance()` in `SavingsAccount`. Observe and explain the outcome.

Questions:

- What error message do you get if you attempt to override `displayBalance()`?
- Why might you use `final` on a method in a real-world scenario?

Problem 3: Using `final` with Classes

1. Create a `final` class `LibraryItem` with:
   - A variable `title` of type `String`.
   - A constructor to initialize `title`.
   - A method `displayTitle()` that prints the title.
2. Attempt to create a subclass `Book` that extends `LibraryItem`. Observe and explain the result.

Questions:

- What error message do you get when you try to extend `LibraryItem`?
- When would it make sense to make a class `final` in real-world applications?

Problem 4: Using `final` with Static Constants

1. Create a class `MathConstants`:
   - Define a `public static final` variable `PI` with a value of `3.14159`.
   - Define another `public static final` variable `E` with a value of `2.718`.
2. Write a `main` method that prints the values of `PI` and `E`.

Questions:

- Why is `final` used for these variables?

- Explain the difference between `static` and `final` in this context.

Submission:

Submit your code for each problem along with answers to the questions in comments or a separate text file.

Good luck, and remember to think about why `final` might be helpful for creating constants, preventing inheritance, or restricting method overriding in Java!