Method Overriding

1. Problem: Create a class called `Calculator` that demonstrates method overloading for basic arithmetic operations (addition, subtraction, multiplication, and division).
    - Example: Implement the `add`, `subtract`, `multiply`, and `divide` methods in the `Calculator` class, each overloaded with different parameter lists to perform the respective operations.
2. Problem: Design a simple shape hierarchy with classes like `Circle` and `Rectangle`. Implement method overriding to calculate the area of these shapes.
    - Example: Create a superclass `Shape` with an overridden `calculateArea` method in subclasses like `Circle` and `Rectangle`.
3. Problem: Develop a student grading system where you have different grading scales for different courses. Use method overloading to calculate the final grade based on the course's grading scale.
    - Example: Create a `Student` class with overloaded `calculateFinalGrade` methods for each course, allowing custom grading logic.
4. Problem: Build a file manager application that handles various file operations like copying, moving, and deleting files. Use method overloading to implement these operations.
    - Example: Create a `FileManager` class with overloaded methods for operations like `copyFile`, `moveFile`, and `deleteFile`.
5. Problem: Implement a banking system with multiple account types (savings, checking) and use method overriding to customize interest calculation for each account type.
    - Example: Create `Account` classes for savings and checking accounts, both overriding an `calculateInterest` method.
6. Problem: Develop a drawing application with different shapes (e.g., lines, circles, rectangles). Use method overloading to draw each shape based on its attributes.
    - Example: Implement a `Draw` class with overloaded methods for drawing lines, circles, and rectangles, accepting different parameters for each shape.
7. Problem: Create a simple game with different character classes (warrior, mage, archer). Use method overriding to define their unique abilities.
    - Example: Implement a `Character` superclass and subclasses for each character class (e.g., `Warrior`, `Mage`, `Archer`) with overridden methods for their special abilities.
8. Problem: Design a scheduling application that schedules events of various types (meetings, appointments, tasks). Use method overloading to add events to the schedule.
    - Example: Implement a `Scheduler` class with overloaded `addEvent` methods for different event types, each with relevant parameters.

9. Problem: Create a media player application that plays different media types (audio and video). Use method overriding to customize the playback behavior for each media type.
    - Example: Implement a `MediaPlayer` class with overridden methods for playing audio and video files.
10. Problem: Build a restaurant menu system with various dishes and options. Use method overloading to add dishes to the menu with different customization levels.
    - Example: Create a `Menu` class with overloaded methods to add dishes to the menu, allowing customization options for each dish.