CSCI 218: Programming II (Spring 2025)

Week 4 Lab Activity: Exploring Inheritance, Overriding, Overloading Concepts

---

Section 1

Objective:

To understand the concepts of inheritance, method overriding, method overloading, and how to use the `this` and `super` keywords in object-oriented programming.

Part 1: Inheritance

Inheritance allows one class (subclass or child class) to inherit the properties and behaviors (methods) from another class (superclass or parent class).

Task 1: Create a Parent Class and a Child Class

1. Create a class named `Animal` with the following attributes:
    - `name`
    - `age`
2. Add a method named `speak()` to the `Animal` class that outputs a general message like "Animal makes a sound."
3. Create a subclass `Dog` that inherits from `Animal`. In the `Dog` class, override the `speak()` method to display a message like "Dog barks."

Questions:

- What happens when the `speak()` method is called on an instance of `Dog`?
- How is method overriding achieved here?

Part 2: Method Overloading

Method overloading occurs when you define multiple methods in the same class with the same name but different parameters (different number or type of parameters).

Task 2: Overload Methods in the Parent Class

1. In the `Animal` class, add multiple `speak()` methods:
    - One that takes no arguments and prints "Animal makes a sound".

    - One that takes a `String` parameter for a specific animal sound.
2. In the `Dog` class, use the overloaded `speak()` method.

Questions:

- How does method overloading differ from method overriding?
- How can you call the overloaded methods in the `Dog` class?

Part 3: Using the `this` Keyword

The `this` keyword refers to the current object instance of a class. It is commonly used to refer to instance variables when their names conflict with parameter names or when calling another constructor in the same class.

Task 3: Use `this` Keyword

1. Add a constructor to the `Animal` class that initializes `name` and `age`.
2. In the `Dog` class, use the `this` keyword to call the parent class's constructor.

Questions:

- What does the `this` keyword do in the constructor of the `Dog` class?
- How does `super()` differ from `this`?

Part 4: Using the `super` Keyword

The `super` keyword is used to access the members (methods and variables) of the parent class from the subclass.

Task 4: Use `super` Keyword

1. In the `Dog` class, call the parent class's `speak()` method using the `super` keyword.
2. Create another method in the `Dog` class that calls the `speak()` method of the parent `Animal` class.

Questions:

- What does the `super.speak()` call do in the `Dog` class?
- When would you use `super` in a subclass method or constructor?

Part 5: Testing the Implementation

Create a `Main` class to test the inheritance, overriding, overloading, and the use of `this` and `super`.

Task 5: Main Class

1. Instantiate objects of both `Animal` and `Dog` classes.
2. Call the `speak()` method on both instances.
3. Call the overloaded `speak()` method for the `Dog` object.
4. Print the values of `name` and `age` for both objects.

Questions:

- What is the output of the program when you run it?
- How does the `speak()` method behave differently when called on `Dog` versus `Animal`?

Part 6: Extras

- **Extend the program** to include another subclass, such as `Cat`, which overrides the `speak()` method and demonstrates both method overloading and overriding.
- **Modify the code** to use the `this` keyword to avoid any ambiguity between instance variables and constructor parameters.

Conclusion

This lab demonstrates the core principles of inheritance in object-oriented programming. By completing these tasks, you will gain hands-on experience with method overriding, overloading, and the `this` and `super` keywords, which are essential for writing flexible and maintainable code in object-oriented languages.

Section 2

Part 1

Consider a system for modeling geometric shapes in a drawing application. Define a base class called Shape with the following attributes and methods:

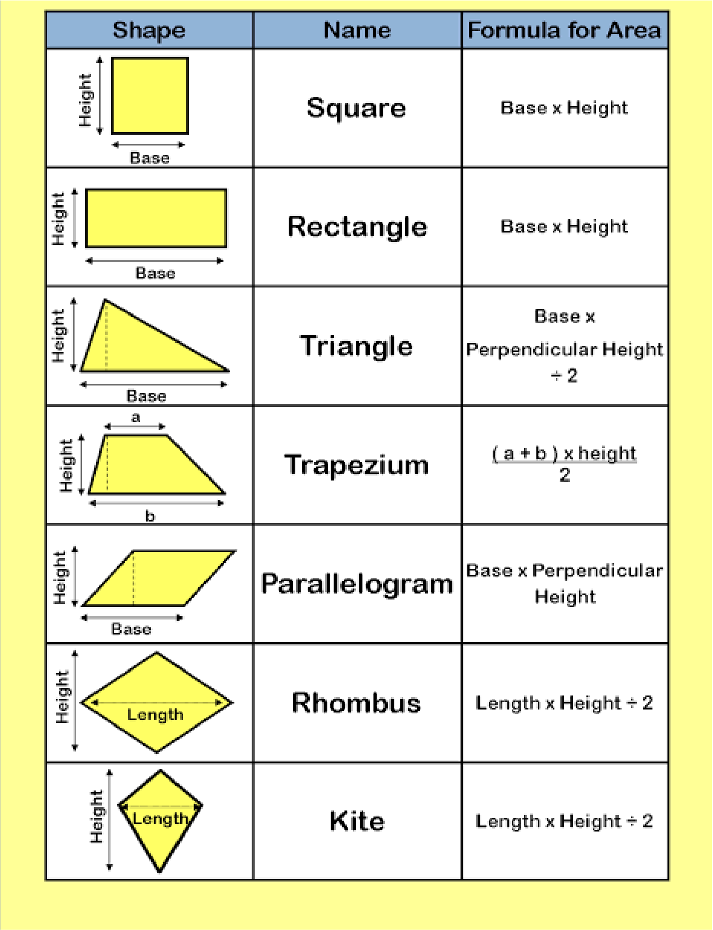
Attributes: name, color

Methods:

- getArea – returns 0

- displayInfo – prints the name, color and area

Using this base class, create the following subclasses and implement them accordingly.



Shape	Name	Formula for Area
Height Base	Square	Base x Height
Height Base	Rectangle	Base x Height
Height Base	Triangle	Base x Perpendicular Height ÷ 2
a Height b	Trapezium	$\frac{(a + b) \text{ x height}}{2}$
Height Base	Parallelogram	Base x Perpendicular Height
Height Length	Rhombus	Length x Height ÷ 2
Height Length	Kite	Length x Height ÷ 2