CSE 215L: Programming Language II Lab

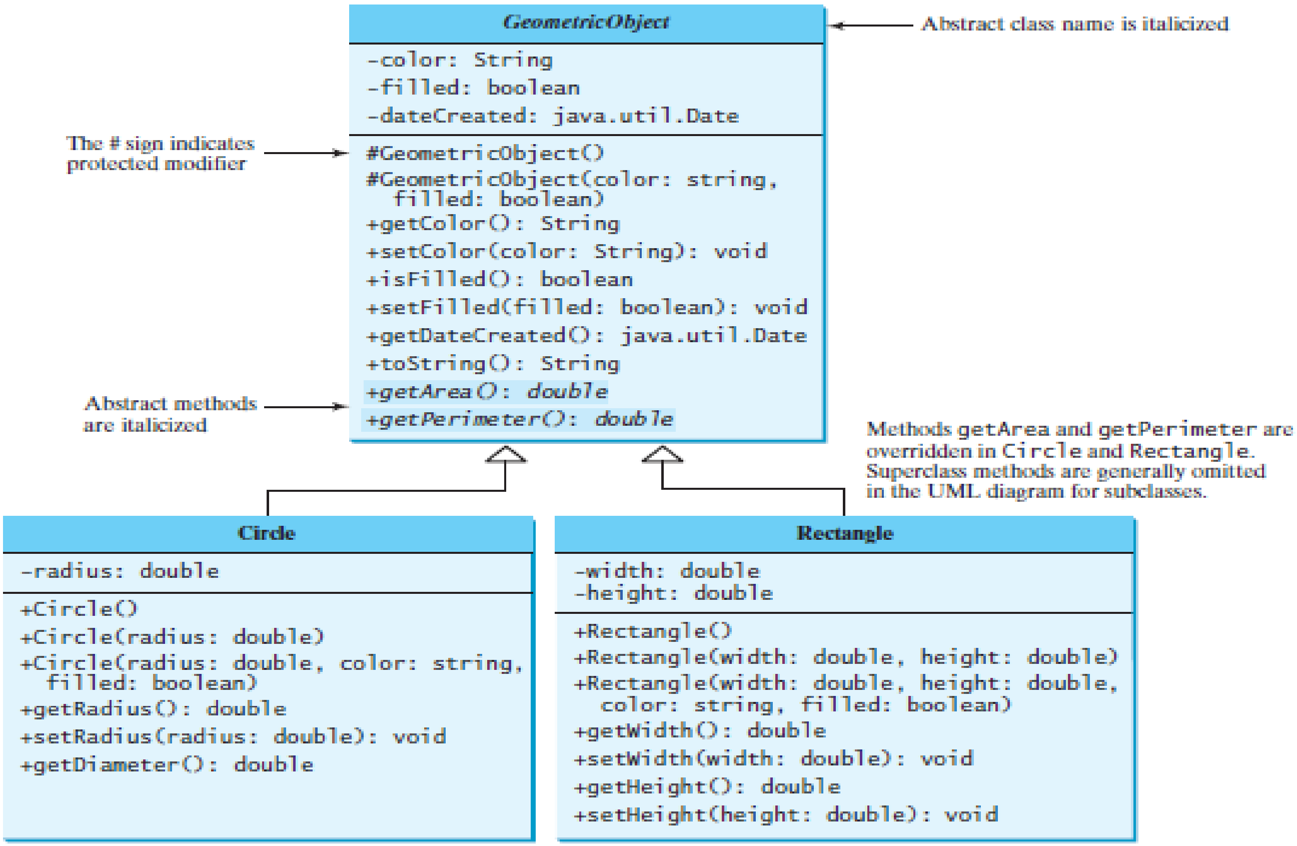
Faculty: Silvia Ahmed, Sec – 4, 5

Lab 11 – Fall 2019

Objective:

After today's lab, the students should be able:

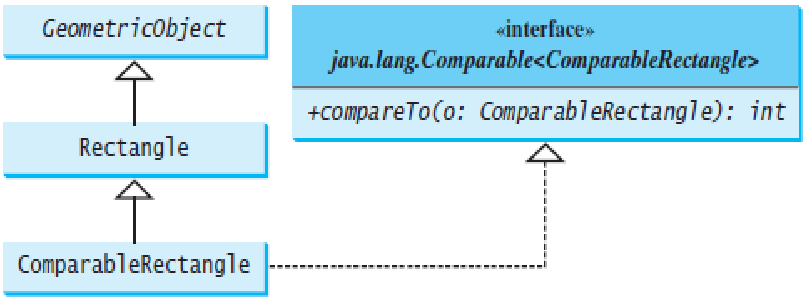
- To design and use abstract classes
- To process a calendar using the **Calendar** and **GregorianCalendar** classes
- To specify common behavior for objects using interfaces
- To define interfaces and define classes that implement interfaces



GeometricObject
-color: String
-filled: boolean
-dateCreated: java.util.Date
#GeometricObject()
#GeometricObject(color: string, filled: boolean)
+getColor(): String
+setColor(color: String): void
+isFilled(): boolean
+setFilled(filled: boolean): void
+getDateCreated(): java.util.Date
+toString(): String
+getArea(): double
+getPerimeter(): double
Abstract class name is italicized
The # sign indicates protected modifier
Abstract methods are italicized
Methods getArea and getPerimeter are overridden in Circle and Rectangle. Superclass methods are generally omitted in the UML diagram for subclasses.
Circle
-radius: double
+Circle()
+Circle(radius: double)
+Circle(radius: double, color: string, filled: boolean)
+getRadius(): double
+setRadius(radius: double): void
+getDiameter(): double
Rectangle
-width: double
-height: double
+Rectangle()
+Rectangle(width: double, height: double)
+Rectangle(width: double, height: double, color: string, filled: boolean)
+getWidth(): double
+setWidth(width: double): void
+getHeight(): double
+setHeight(height: double): void

Interfaces

modifier **interface** InterfaceName {
/** Constant declarations */
/** Abstract method signatures */
}



GeometricObject
Rectangle
ComparableRectangle
«interface»
java.lang.Comparable<ComparableRectangle>
+compareTo(o: ComparableRectangle): int

FIGURE **ComparableRectangle** extends **Rectangle** and implements **Comparable**.

Task – 1

(*Triangle class*) Design a new **Triangle** class that extends the abstract **GeometricObject** class. Draw the UML diagram for the classes **Triangle** and **GeometricObject** and then implement the **Triangle** class. Write a test program that prompts the user to enter three sides of the triangle, a color, and a Boolean value to indicate whether the triangle is filled. The program should create a **Triangle** object with these sides and set the color and filled properties using the input. The program should display the area, perimeter, color, and true or false to indicate whether it is filled or not.

Task – 2

(*Display calendars*) Write the **PrintCalendar** class to display a calendar for a specified month using the **Calendar** and **GregorianCalendar** classes. The program prompts the user to enter the year and the month, and then displays the entire calendar for the month. For example:

```
Enter full year (e.g., 2012): 2012
Enter month as number between 1 and 12: 3
                        March 2012
                  -----------------------------
                Sun Mon Tue Wed Thu Fri Sat
                                      1    2    3
                  4    5    6    7    8    9    10
                 11   12   13   14   15   16   17
                 18   19   20   21   22   23   24
                      25   26   27   28   29   30
```

Figure: The program displays a calendar for March 2012.

You also can run the program without the year. In this case, the year is the current year. If you run the program without specifying a month and a year, the month is the current month.

Task – 3

Design an interface named **Colorable** with a **void** method named **howToColor()**. Every class of a colorable object must implement the **Colorable** interface. Design a class named **Square** that extends **GeometricObject** and implements **Colorable**. Implement **howToColor** to display the message **Color all four sides**.

Draw a UML diagram that involves **Colorable**, **Square**, and **GeometricObject**. Write a test program that creates an array of five **GeometricObject**s. For each object in the array, display its area and invoke its **howToColor** method if it is colorable.

Task – 4

(*Enable **Circle** comparable*) Rewrite the **Circle** class in diagram to extend **GeometricObject** and implement the **Comparable** interface. Override the **equals** method in the **Object** class. Two **Circle** objects are equal if their radii are the same. Draw the UML diagram that involves **Circle**, **GeometricObject**, and **Comparable**.

Task – 5

(*Sum the areas of geometric objects*) Write a method that sums the areas of all the geometric objects in an array. The method signature is:

```
public static double sumArea(GeometricObject[] a)
```

Write a test program that creates an array of four objects (two circles and two rectangles) and computes their total area using the **sumArea** method.