CSE 215L: Programming language II Lab
Faculty: Dr. Mohammad Rashedur Rahman (RRn)
Lab - 10 [Inheritance]
Fall-2022
Lab Instructor: Md. Mustafizur Rahman

<u>**Objective:**</u>
- To define a subclass from a superclass through inheritance
- To invoke the superclass's constructors and methods using the super keyword
- To override the instance methods in the subclass

<u>**Inheritance:**</u> In Object-oriented programming (OOP), new classes can be defined from the existing classes which are called inheritance. More simply, Inheritance is a mechanism in which one object acquires all the properties and behaviors of a parent object. It represents the IS-A relationship which is also known as a parent-child relationship. Inheritance has two purposes: reuse existing code, and reduce code duplication.

<u>**Superclasses and Subclasses:**</u> Inheritance enables us to define a general class (i.e., a superclass) and later extends it to more specialized classes (i.e., subclasses). Generally, when common traits are found among two classes, define one as super/parent/base class and the other as sub/child/extended/derived class. A child class can inherit the properties of the parent class and also can add its own properties.

<u>**Using the extends keyword:**</u> The extends keyword has been used to relate the subclass with the superclass. For example, let's say we have two classes, named class A and B, respectively, where class A is the superclass and B is the subclass. Then the Java syntax will be,
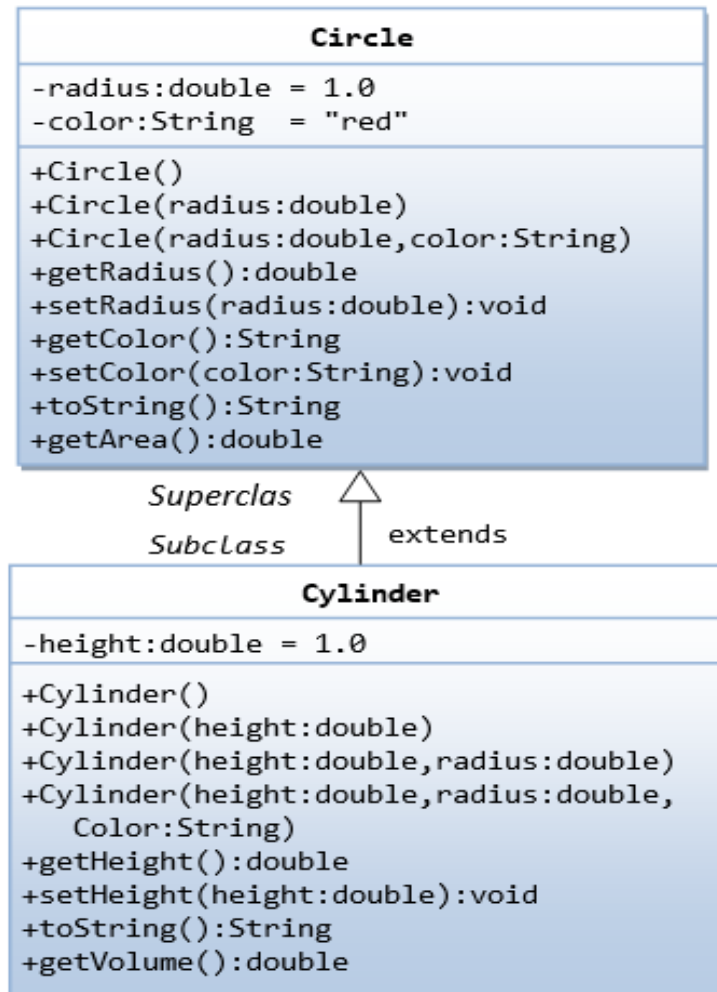
<p align="center">public class B extends A</p>

<u>**Using the super keyword (this vs. super):**</u> The keyword **this** refers to the object itself. It can also be used inside a constructor to invoke another constructor of the same class. For example, the following code shows the uses of this keyword.

| public class **Circle** {
      private double radius;

      public Circle(double radius) {
      this.radius = radius;
      }

      public Circle() {
      this(1.0);
      }
  ………
   ………
} | • **this** keyword is used to reference the hidden data field radius of the object being constructed.

• **this(1.0)**, in this statement, **this** keyword is used to invoke another constructor. |

Tasks:
1. Implement the following classes and test their methods.

Circle
-radius:double = 1.0 -color:String = "red"
+Circle() +Circle(radius:double) +Circle(radius:double,color:String) +getRadius():double +setRadius(radius:double):void +getColor():String +setColor(color:String):void +toString():String +getArea():double

Superclas   △
SubcLass   extends

Cylinder
-height:double = 1.0
+Cylinder() +Cylinder(height:double) +Cylinder(height:double,radius:double) +Cylinder(height:double,radius:double,   Color:String) +getHeight():double +setHeight(height:double):void +toString():String +getVolume():double

2. Implement the following classes and test their methods.

Circle

- radius: double

+ Circle()
+ Circle(radius: double)
+ Circle(radius: double,
        color: String,
        filled: boolean)
/* getters – setters */
+ getPerimeter(): double
+ getDiameter(): double
+ getArear(): double
+ printCircle(): void

GeometricObject

- color: String
- filled: boolean

+ GeometricObject()
+ GeometricObject(color: String,
           filled: boolean)
+ getColor(): String
+ getFilled(): boolean
+ setColor(color: String): void
+ setFilled(filled: boolean): void
+ toString(): String

Rectangle

- width: double
- height: double

+ Rectangle()
+ Rectangle(width: double,
        height: double)
+ Rectangle(width: double,
        height: double,
        color: String,
        filled: boolean)
/* getters – setters */
+ getPerimeter(): double
+ getDiameter(): double
+ getArear(): double
+ printRectangle(): void