|  |
| --- |
| GeometricObject |
| -color: String  -filled: Boolean  -dateCreated: Date |
| #GeometricObject()  #GeometricObject(color: String, filled: boolean)  +getColor(): String  +setColor(color: String): void  +isFilled(): boolean  +setFilled(filled: boolean): void  +getDateCreated(): Date  +toString(): String  *+getArea(): double*  *+getPerimeter(): double*  +compareTo(o: GeometricObject): int  +max(o1: GeometricObject, o2: GeometricObject): GeometricObject |

public abstract class GeometricObject implements Comparable<GeometricObject> {

private String color = "white";

private boolean filled;

private java.util.Date dateCreated;

protected GeometricObject() {

dateCreated = new java.util.Date();

}

protected GeometricObject(String color, boolean filled) {

dateCreated = new java.util.Date();

this.color = color;

this.filled = filled;

}

public String getColor() {

return color;

}

public void setColor(String color) {

this.color = color;

}

public boolean isFilled() {

return filled;

}

public void setFilled(boolean filled) {

this.filled = filled;

}

public java.util.Date getDateCreated() {

return dateCreated;

}

@Override

public String toString() {

return "created on " + dateCreated + "\ncolor: " + color + " and filled: " + filled;

}

public abstract double getArea();

public abstract double getPerimeter();

@Override

public int compareTo(GeometricObject o) {

if (this.getArea() > o.getArea()) {

return 1;

} else if (this.getArea() < o.getArea()) {

return -1;

} else {

return 0;

}

}

public static GeometricObject max(GeometricObject o1, GeometricObject o2) {

if (o1.compareTo(o2) > 0) {

return o1;

} else {

return o2;

}

}

}

public class TestGeometricObject {

public static void main(String[] args) {

Circle circle1 = new Circle(5);

Circle circle2 = new Circle(7);

Circle maxCircle = (Circle) GeometricObject.max(circle1, circle2);

System.out.println("The larger circle has radius " + maxCircle.getRadius() + " and area " + maxCircle.getArea());

Rectangle rectangle1 = new Rectangle(3, 4);

Rectangle rectangle2 = new Rectangle(5, 6);

Rectangle maxRectangle = (Rectangle) GeometricObject.max(rectangle1, rectangle2);

System.out.println("The larger rectangle has dimensions of " + maxRectangle.getWidth() + " x " + maxRectangle.getHeight() + " and area " + maxRectangle.getArea());

}

}

Text

Description automatically generated