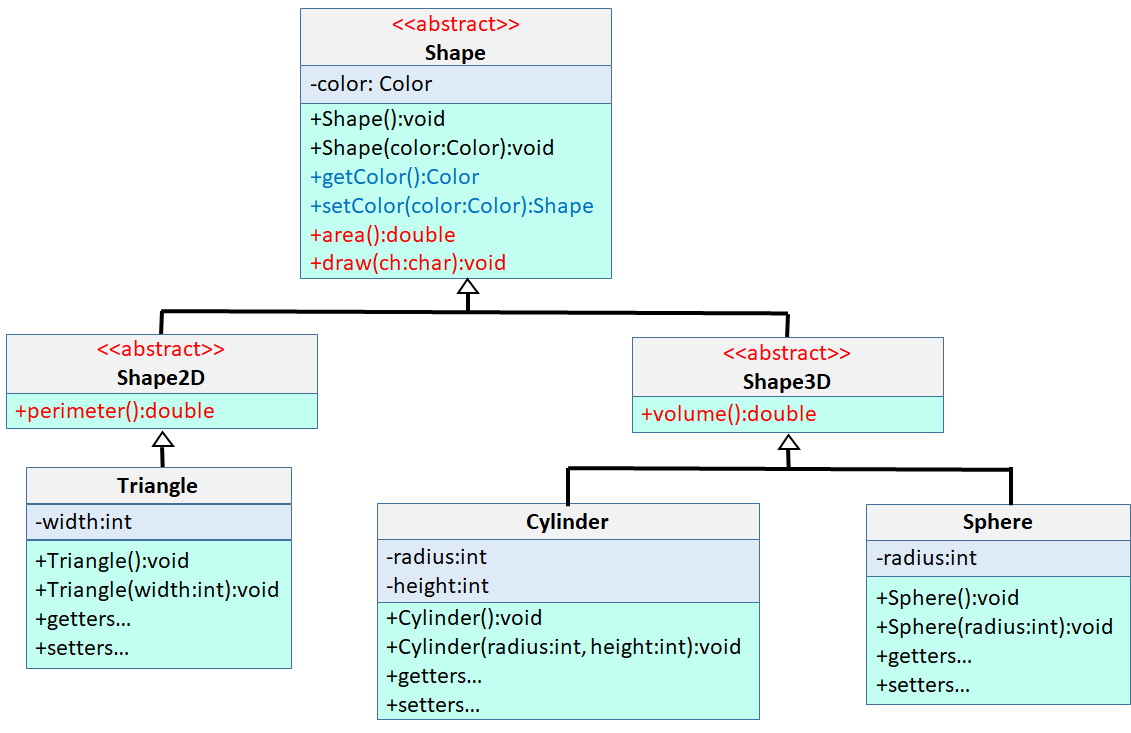
Lab 18 & 19

**Rules:**

1. You are given the following files: **Color.java**, **Test1.java, Test2.java** and **Test3.java**, and are asked to implement the following files: **Shape.java,** **Shape2D.java**, **Shape3D.java, Triangle.java, Cylinder.java**, and **Sphere.java** as described below.
2. Do not forget to take your work with you when you leave the lab by either copying your work files to your own USB flash disk, or by e-mailing them to yourself.

You are asked to implement the classes in the following UML class diagram:



**Shape, Shape2D & Shape3D**

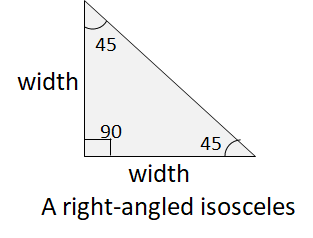
At the top of our class hierarchy we have the abstract **Shape** class, which has a **private color attribute**, and public getter and setter for Color. You must implement getColor and setColor methods. Shape also has 2 abstract methods, area() and draw(), which are common to all shapes.

We then have a **Shape2D** class that extends Shape, and has a single abstract method, **perimeter()**, used to compute the perimeter of a 2D shape. All 2D shapes will be extending Shape2D.

Finally, we have a **Shape3D** class that extends Shape, and has a single abstract method, volume(), used to compute the volume of a 3D shape. All 3D shapes will be extending Shape3D.

**Triangle**

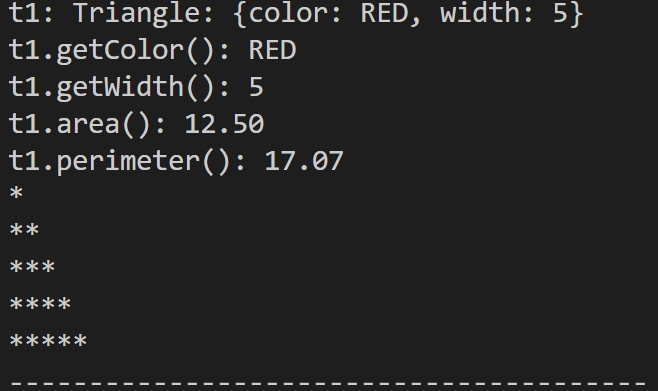
Triangle is a 2D shape, i.e., extends Shape2D, that represents a right-angled isosceles triangle having a side length of “width” as shown below:

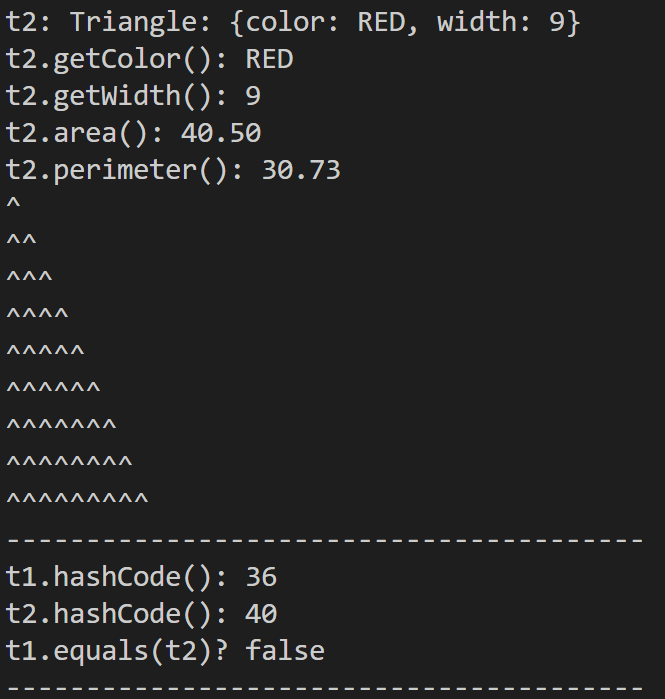


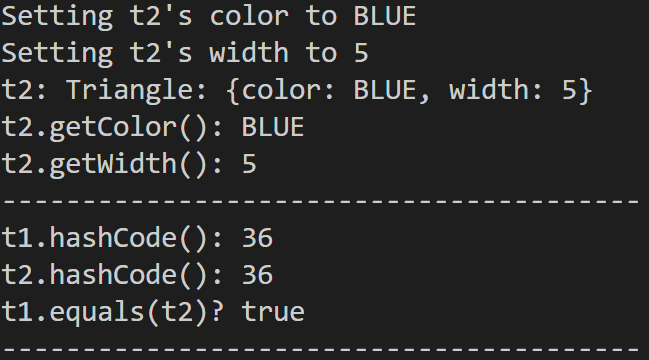
Here is the list of methods you must implement for Triangle:

|  |  |
| --- | --- |
| Triangle() | No-args constructor: Sets the width to 1, color to RED |
| Triangle(int width) | Parametrized constructor: Sets the width to the user-supplied value, and color to RED. If the user-supplied width <=0, throw an IllegalArgumentException() |
| int getWidth() | Getter for width |
| Triangle setWidth(int width) | Setter for width: Sets the width to the user-supplied value, and returns a reference to the object for method chaining. If the user-supplied width <=0, throw an IllegalArgumentException() |
| Triangle setColor(Color color) | Setter for color: Sets the color to the user-supplied value, and returns a reference to the object for method chaining |
| double area() | Returns the area of the triangle |
| double perimeter() | Returns the perimeter of the triangle |
| String toString() | Returns a nice string representation of the object |
| boolean equals(Object o) | Returns true if the triangle is equal to the user-supplied Object “o”. Two triangles are equal if their widths are equal even if their color may be different. |
| int hashCode() | Returns the hashCode of the triangle.  The hashCode for two equal triangles must be the same. |
| void draw(char ch) | Draws the triangle using the user-supplied char. |

In order to test your Triangle.java, we have given you a driver test code **Test1.java** that creates two Triangle objects and tests all methods that you need to implement. Here is how your output must look like when you run **Test1.java** against your Triangle implementation:

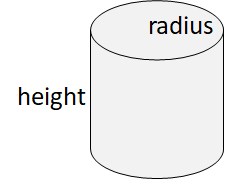






**Cylinder**

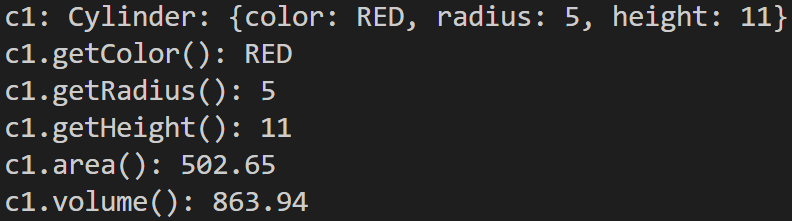
Cylinder is a 3D shape, i.e., extends Shape3D, that represents a cylinder with a radius and height as shown below:

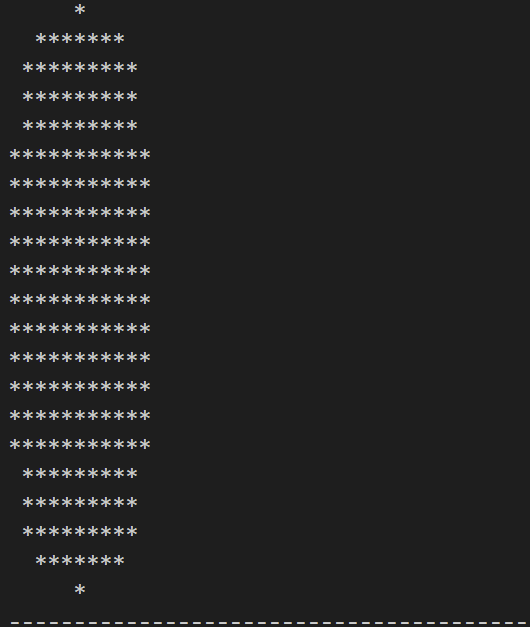


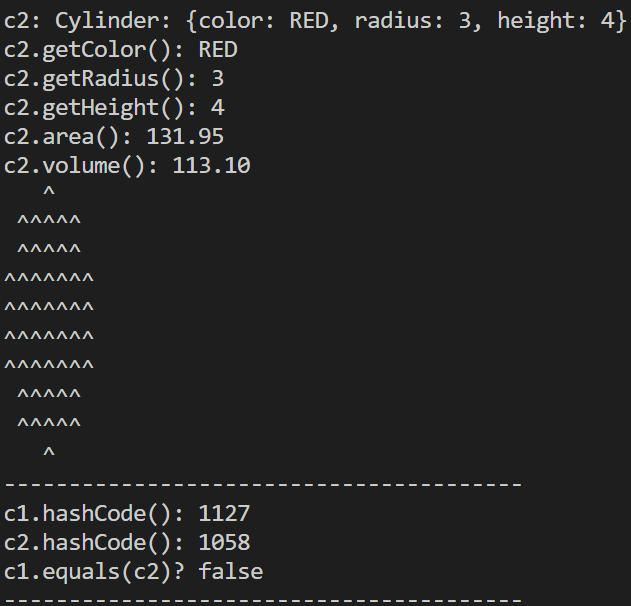
Here is the list of methods you must implement for Cylinder:

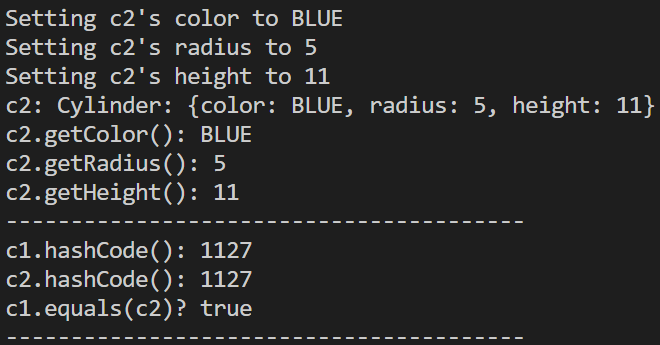
|  |  |
| --- | --- |
| Cylinder() | No-args constructor: Sets the width to 1, color to RED |
| Cylinder(int radius, int height) | Parametrized constructor: Sets the radius and height to the user-supplied values, and color to RED. If the user-supplied radius or height <= 0, throw an IllegalArgumentException() |
| int getRadius() | Getter for radius |
| Int getHeight() | Getter for height |
| Cylinder setRadius(int radius) | Setter for radius: Sets the radius to the user-supplied value, and returns a reference to the object for method chaining. If the user-supplied radius <= 0, throw an IllegalArgumentException() |
| Cylinder setHeight(int height) | Setter for height: Sets the height to the user-supplied value, and returns a reference to the object for method chaining. If the user-supplied height <= 0, throw an IllegalArgumentException() |
| Cylinder setColor(Color color) | Setter for color: Sets the color to the user-supplied value, and returns a reference to the object for method chaining |
| double area() | Returns the area of the cylinder, which is computed as 2π\*radius² + 2π \* radius\*height |
| double volume() | Returns the volume of the cylinder, which is computed as π\*radius²\*height |
| String toString() | Returns a nice string representation of the object |
| boolean equals(Object o) | Returns true if the cylinder is equal to the user-supplied Object “o”. Two cylinders are equal if their radius and height are equal even if their color may be different. |
| int hashCode() | Returns the hashCode of the cylinder. The hashCode for two equal cylinders must be the same. |
| void draw(char ch) | Draws the cylinder using the user-supplied char. Here is how you should draw a cylinder: First draw the half circle representing the upper part of the cylinder using the radius. Then draw the body using the height and radius. Finally, draw the bottom half circle representing the lower part of the cylinder using radius. |

In order to test your Cylinder.java, we have given you a driver test code **Test2.java** that creates two Cylinder objects and tests all methods that you need to implement. Here is how your output must look like when you run **Test2.java** against your Cylinder implementation:







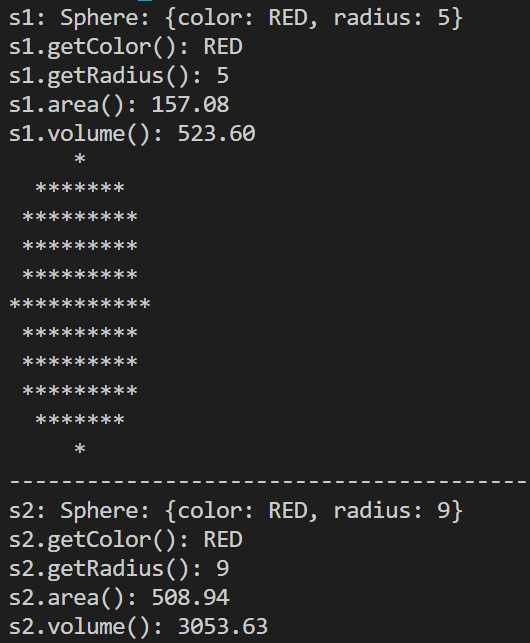


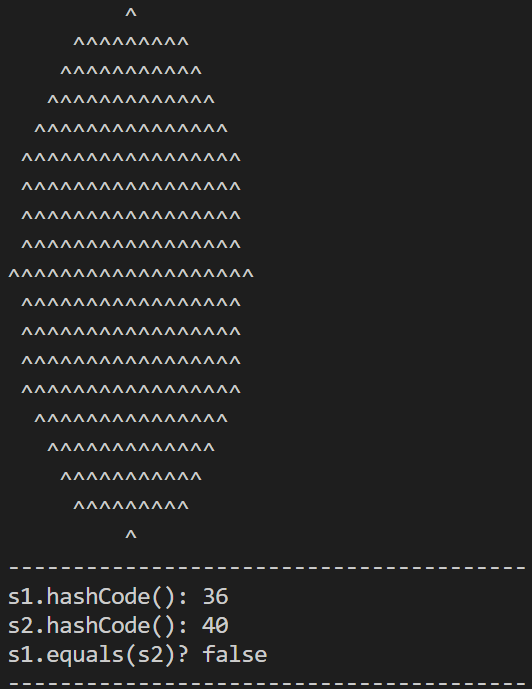
**Sphere**

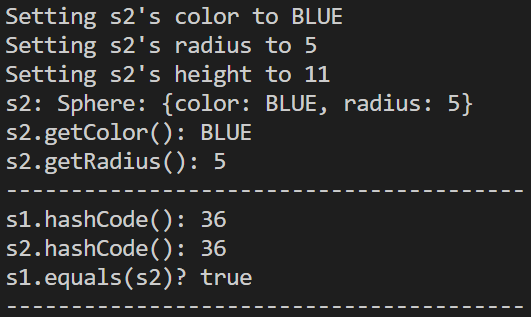
Sphere is a 3D shape, i.e., extends Shape3D, that represents a sphere with a radius. Here is the list of methods you must implement for Shere:

|  |  |
| --- | --- |
| Sphere() | No-args constructor: Sets the width to 1, color to RED |
| Sphere (int radius) | Parametrized constructor: Sets the radius to the user-supplied value, and color to RED. If the user-supplied radius <= 0, throw an IllegalArgumentException() |
| int getRadius() | Getter for radius |
| Sphere setRadius(int radius) | Setter for radius: Sets the radius to the user-supplied value, and returns a reference to the object for method chaining. If the user-supplied radius <= 0, throw an IllegalArgumentException() |
| Sphere setColor(Color color) | Setter for color: Sets the color to the user-supplied value, and returns a reference to the object for method chaining |
| double area() | Returns the area of the cylinder, which is computed as 4\*π\*radius² |
| double volume() | Returns the volume of the cylinder, which is computed as 2\*π\*radius² |
| String toString() | Returns a nice string representation of the object |
| boolean equals(Object o) | Returns true if the sphere is equal to the user-supplied Object “o”. Two spheres are equal if their radius is equal even if their color may be different. |
| int hashCode() | Returns the hashCode of the sphere. The hashCode for two equal spheres must be the same. |
| void draw(char ch) | Draws the sphere using the user-supplied char. You must draw the sphere as if you are drawing a circle with “radius”. |

In order to test your Sphere.java, we have given you a driver test code **Test3.java** that creates two Spheres objects and tests all methods that you need to implement. Here is how your output must look like when you run **Test3.java** against your Sphere implementation:







You are advised to implement your own test code. When grading, we may use a different Test. Make sure that your code works under all circumstances.

Lab Work Submission:

* You can continue to work on this lab after our lab class, on your own, at home.
* Submit your lab work via Blackboard on or before: **Wednesday, November 8, 2023, 11:59pm**.
* The only accepted submission method!
* Once you submit your assignment you will not be able to resubmit it!
* Make absolutely sure the Java files you want to submit are the Java files you want graded.
* You will not be able to submit your lab work under any circumstances once **Lab18** disappears at **12:00 a.m.** on **Thursday, November 9, 2023**.
* There will be **NO** exceptions to these rules!
* To submit your lab work, upload **Shape.java, Shape2D.java, Shape3D.java, Triangle.java, Cylinder.java** & **Sphere.java** files (**with .java extension**) you did for this lab to the **Lab18** assignment in the **Labs** tab in your Lab section’s presence in Blackboard.
* Then, make sure you click the **Submit** button to submit your lab work.
* This lab is worth **13 points** as follows:

|  |  |
| --- | --- |
| Shape.java | 2 points |
| Shape2D.java | 1 point |
| Shape3D.java | 1 point |
| Triangle.java | 3 points |
| Cylinder.java | 3 points |
| Sphere.java | 3 points |