Create a Visual Studio solution that contains a class library

Start by creating the following class in the library:

```
public abstract class Shape
{
public abstract Vector3 Center {get; }
public abstract float Area {get; }
}
```

Then create these two public abstract classes **Shape2D** and **Shape3D** as both inherit from Shape. In Shape2D you should add a public abstract property **Circumference** of the float type, and in Shape3D you should add a public abstract property **Volume** of type float. For both of these properties, you only have to be able to read their values (ie you should not be able to write to them).

Five classes of geometric shapes:

Now you will create classes that inherit from either Shape2D or Shape3D. All the classes must implement the abstract properties that they inherit from each class in such a way that they give correct values for center, perimeter, area and volume. If you do not know how to calculate certain things, such as the volume of one globe, so try to find the formulas on the internet and implement them in your code.

All classes must also override: a ToString () method to print the name of the shape together with the position of the center point, followed by other values (e.g. radius, or height / width).

The five classes to be implemented are **Circle, Rectangle** and **Triangle** as will inherit from **Shape2D**, as well as **Sphere** and **Cuboid** who will inherit from **Shape3D**.

Circle (Shape2D)

The Circle class must have one (1) constructor that takes a Vector2 center first parameter, and a float radius as other parameters. It should override: a ToString () in such a way that e.g.

Console.WriteLine (new Circle (new Vector2 (3.0f, 4.0f), 2.0f));

=> "Circle @ (3.0, 4.0): r = 2.0"

Rectangle (Shape2D)

This class must have a constructor that takes parameters: Vector2 center, Vector2 size (ie height / width), and an alternative constructor: Vector2 center, float width (which sets both height and width to the same value).

It will also implement a property IsSquare that returns true in height and width are equal (otherwise false).

ToString () => **"rectangle @ (3.0, 4.0): w = 4.0, h = 5.0"** (square om w == h).

Triangle (Shape2D)

The constructor takes three parameters p1, p2, p3 of type Vector2, which describe the three points that make up the triangle. Keep in mind that Center also needs to be calculated.

ToString () => **"triangle @ (3.0, 1.0): p1 (0.0, 0.0), p2 (3.0, 3.0), p3 (6.0. 0.0)"**

Cuboid (Shape3D)

This class must have a constructor that takes parameters: Vector3 center, Vector3 size (ie height / width / depth), and an alternative constructor: Vector3 center, float width (which sets height, width depth to the same value).

It will also implement a property IsCube that returns true in height, width and depth are equal (otherwise false).

ToString () => **"cuboid @ (3.0, 4.0, 5.0): w = 4.0, h = 5.0, l = 2.0"** (cube om w == h == l).

Sphere (Shape3D)

Constructor with parameters Vector3 center, float radius.

ToString () => **“sphere @ (0.0, 1.0, 0.0): r = 3.2”**

Add method to generate random shapes

When you have finished all the classes, add a public static method GenerateShape in the base class **Shape**. This method should randomize a number 0-6 to instantiate one of 7 figures (circle, rectangle, square, triangle, cuboid, cube, sphere) and return in the form of a Shape.

The values of the different figures must also be randomized. GenerateShape is coming need to call the constructor for the different classes with random values on all Parameters.

But, we also want GenerateShape to be available in two versions, one that does not take some parameters (ie all values are randomized), and one where you can submit one Vector3 position to indicate the center of the figure (then only the others are randomized values such as width, radius, etc.). Keep in mind that to classes that inherit from Shape3D can you submit the position as it is, while you get to create a Vector2 of X and Y, for them classes that inherit Shape2D (and thus take a Vector2 for the midpoint).

Also keep in mind that for Triangle you can only randomize two of the points. The third the point must be calculated to get the center point in the right place.

Create a project in the same solution that uses your class library

This project will create a list of 20 random shapes, by calling

GenerateShape (). Then loop through the list and print all the shapes for the console.

Simultaneously calculate the sum of the perimeters of all triangles in the list,

the average area of all Shapes in the list,

and find the Shape3D that has the largest volume of all in the list.

Present this neatly and clearly on the console when the loop is complete.

Tips & Help

Vector2 and Vector3 are 2- and 3-dimensional vectors that have properties

X, Y, and X, Y, Z (floats).

The length of a vector is the distance between the point the vector describes and the origin

(zero points). Use the Length () method to get the length ex myVec3.Length ();

Vector2.Zero yields a 2D vector corresponding to new Vector2 (0.0f, 0.0f);

Vector3.Zero yields a 3D vector corresponding to new Vector3 (0.0f, 0.0f, 0.0f);

Vector2.One => new Vector2 (1.0f, 1.0f);

Vector3.One => new Vector3 (1.0f, 1.0f, 1.0f);

Multiplying a vector by a float is the same as that

multiply each component of the vector by the same value.

ex. Vector3.One * 5.0f => new Vector3 (5.0f, 5.0f, 5.0f);