Laboratory 9

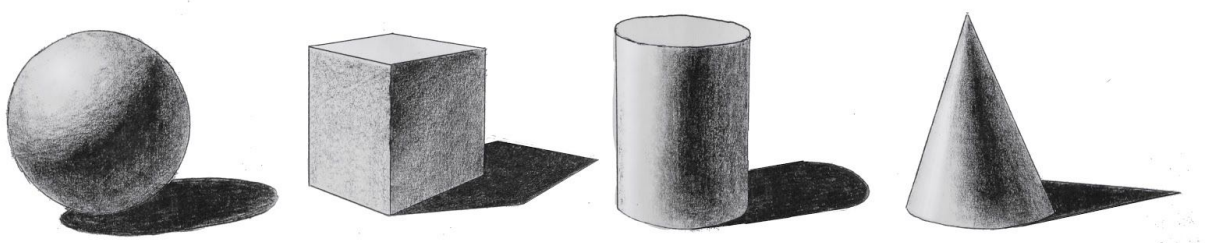
Objective:
This laboratory focuses on problems related to threads synchronization and design patterns.

Problem 1
Imagine a car race, where a number of cars are aligned at the start line. The green light is turned on and the cars will start the race. But not all the cars are able to start in the same moment. Every car has a delay, some of them being delayed more than others, some of them less, but eventually, they are able to pass the finish line.

Create a class `Car`, being a subclass of `threading.Thread`, which has a method `start_car()`. This method is called based on a random delay, let's say the delay is between 1 and 5 seconds.

The green light is modeled here by a `threading.Event` object which is set to true, and all the `Car` objects call the `start_car()` method.

Problem 2
Imagine a system composed of some three-dimensional geometrical objects (a sphere, a cylinder, a cube and a cone).



Think of these figure composing an abstract statue, the cube being at the bottom, then over the cube is laying a sphere, over the sphere a cylinder and on the very top, the cone.

We need to offer to a client of our system the possibility to compute the volume of the whole system. Of course, we can provide it the details of all the components of the system and let him to manage himself with the computation, but we are smart guys so we can provide him a Facade to the system, hiding all the guts of the system, and exposing only a get_volume() method.

Below there is a table with the volumes for the involved objects.

3D-Object	Volume
Sphere	$V = \frac{4}{3} \pi r^3$
Cube	$V = l^3$
Cone	$V = \frac{1}{3} \pi r^2 h$
Cylinder	$V = h \pi r^2$

Create a class for each 3D-object and implement a get_volume() method. Then create a Facade pattern and provide it with a get_volume() method.

The sizes for the objects are:

sphere	radius = 3
cube	size = 3
cylinder	radius = 3, height = 7
cone	radius = 3, height = 5