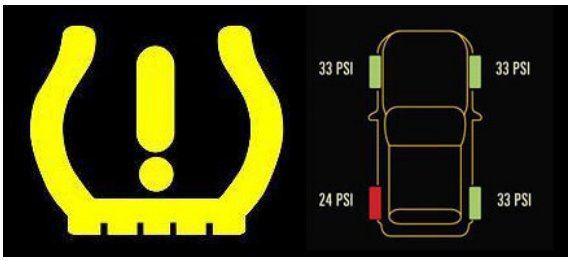
Exercise: Tire pressure monitor system

In these exercises, you will practice design, implementation and unit test.

Some modern cars have a tire pressure monitor systems, which continuously monitors the pressure in each tire. The pressure is displayed on the cars info-screen and a warning light is turned on in the dashboard, if the pressure becomes low.



Figur 1 - TPMS illustration from: http://www.rematiptop.com/tpms/img/TPMS-warning-light.jpg

---

Exercise 1: (Design)
Design software to read the pressure from a single tire and display the pressure.
Create a UML class diagram and any other diagrams, you find relevant for the design.
Explain your design to one or two of your fellow students.

Exercise 2: (Design and implementation)
Implement your design as a Console application and create a simulated version of the sensor.
The simulated sensor should be polled once a second, i.e. you have to ask the sensor for the tire pressure.
The sensor should provide a random value between 20 and 40 psi.
Simulate the warning light by printing to the console, when the light is on.
For this exercise, "low tire pressure" is defined as below 30 psi.

Exercise 3: (SRP consideration)
Does your design adhere to the Single Responsibility Principle?
If not, how can you improve your design?
Update the UML diagram(s) and the implementation.

Exercise 4: (Unit test)
Write NUnit tests for the part of the software, which decides if the pressure is low.
Make sure to consider boundary values and equivalence partitions.

Exercise 5: (Filtering, design)
Sometimes the pressure sensor provides a wrong value, e.g. the pressure spikes, when the car drives over a bump in the road. Add this behavior to your sensor simulation, so it has 2% chance of outputting a value, which is 10 psi higher than it would otherwise have been.
Design and implement at filter, which evens out the spikes, by outputting the average value of 15 seconds of measurement. Add the filter to your software.

Exercise 6: (Unit test)
Write the unit tests for you filter.

Exercise 7: (Filtering, design)
The averaging filter is really not very good… what we want is to remove the erroneous measurements, not even them out.
Design a new filter, which removes a measurement if it is more than 5 psi higher than the previous measurement. Use this filter in your software.

Exercise 8: (Unit test)
Write the unit tests for you filter.

Exercise 9: (Design pattens)
Modify your design, so you can change the filter used. Implement your new design.

Exercise 10: (Configuration)
Identify the hard-coded configuration values used in the software.
Implement configuration by adding one or more configuration classes and use the XMLSerializer to load the configuration.

Exercise 11: (Design)
Consider how your design should be modified, to accommodate all four tires and to display the pressure for each tire on the cars info-screen.
Create a UML class diagram and any other diagrams, you find relevant for the design.
Explain your design to one or two of your fellow students.