Exercise: Hospital Bed, Producer - Consumer

In these exercises, you will work with C# threads and the producer – consumer design.

Your job is to design and implement software for a hospital bed.

The caretakers at the hospital have trouble with patients leaving their beds, when they are not supposed to.
The hospital has hired *you*, a brilliant young engineer, to solve their problem.
You came up with a solution:

> Install a presence sensor and a buzzer in the beds.
> If the presence sensor detects, that there is no patient in the bed, alarm the caretakers using the buzzer.

Exercise 1:
Design the software to use for the bed.
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:
Implement your design as a Console application and create a simulated version of the sensor.
The simulated sensor should be polled, i.e., you have to ask the sensor if a patient is present.
The sensor should provide a random value (true or false).
Simulate the buzzer by printing to the console, when the buzzer is buzzing.

Exercise 3:
A test has shown, that the sensor sometimes provides false values, i.e., occasionally it reports false, even though a patient is in the bed.
As an engineer, you know that you sometimes have to filter your input signals. So, you come up with a very simple filter: The sensor input must be consistent for 3 consecutive samples, before it can be trusted.
Update your design to include the filter. When you do, have the Single Responsibility Principle in mind.
Implement your solution, after you have updated your design.

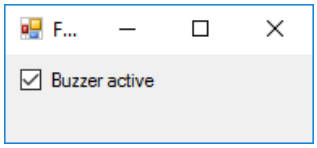
Exercise 4:
Create a Unit test project in your C# solution. Add a unit test for you filter class, and make sure you test cases where the patient is in bed and out of bed, but also when there are changes from one state to another.

Exercise 5:
Did you use the producer – consumer design?
If not, change your code to use this design. Rewriting your code like this is often called *refactoring*.

The next exercises can be solved if you want to practice WPF.

The service technicians who work on the beds have found, that both sensors and buzzers become defective.
They would really like a WPF program, which shows if the buzzer is supposed to be buzzing.

It could just look something like this:



Exercise 5:
Design a WPF program, which shows the buzzer state.
You can probably re-use a lot of the console application from exercise 1 to 4.

Exercise 6:
Implement your design as a WPF application.

Exercise 7:
Add a counter, which counts how many times the buzzer state has changed.