COMP201: Software Engineering I
Assignment 1.2 (2022/23)
(100% mark for Assignment 1.2 is 5% of COMP201 grade)

Deadline for Assignment 1.2: 2nd of December 2022, 17:00

OBJECTIVE
This assignment is mainly about "design/implementation".
You should be implementing a simulation of part of the security system. This simulation will follow the requirements that have been defined in coursework 1 and follow the use cases that you defined. This simulation will not be a full implementation of the original requirement but only implement the card/code entry part of the system. For this case, there is no UI code required. You are strongly encouraged to start this work as soon as possible.

The purpose of this exercise is to understand the challenges of implementing a design from requirements and the state modelling required. This code will be implemented using a simulated version of the hardware, it would then be possible later to implement a security system by replacing the simulation code with software drivers connected to real hardware.

Assignment number	1 of 2 (part 2)
Weighting	5%
Assignment Circulated date provided to class	2/11/2022
Deadline Day & Date & Time	2nd of December 2022 at 17:00
Submission Mode	Electronic submission/Canvas
Learning outcome assessed	1. Realise the problems in designing and building significant computer systems 2. Understand the need to design systems that fully meet the requirements of the intended users 3. Be able to apply these principles in practice 4. Be able to demonstrate how to effectively implement an OO design in an O-O language such as Java or Python.
Submission necessary in order to satisfy Module requirements	No

Purpose of assessment	To assess the students ability to analyse a problem and implement it in code

Marking criteria	See end of document
Late Submission Penalty	Standard UoL Policy

Description of problem
Produce code to support a security system simulation in Java using the existing code as a base. (This has been made available to you as a zip file).

NOTE You only need to fulfil the requirements of this coursework not the whole of the requirement detailed in coursework 1.

Card handling

All the code you will implement is done as part of files Authenticator.java and Card.java, you will need to modify both these files. It is very important you do NOT modify the public interface of these Java classes.

Look at the comments in the source files for information on what has to be done. Everywhere there is a TO DO comment, please complete the code as requested.

checkFireCode
The behaviour should be as follows, when calling checkFireCode(). If the card is locked, it will return CARD_LOCKED_FIRE. If the code is correct, the method returns the OK status. If the code is invalid the method returns INVALID_FIRE_CODE. If the code is incorrect, the method returns BAD_FIRE_CODE. Getting the code wrong 3 times will lock the card and return CARD_LOCKED_FIRE.

checkBurglaryCode
The behaviour should be as follows, when calling checkBurglaryCode(). If the card is locked, it will return CARD_LOCKED_BURGLAR_ALARM. If the code is correct, the method returns the OK status. If the code is invalid the method returns INVALID_BURGLARY_CODE. If the code is incorrect, the method returns BAD_BURGLARY_CODE . Getting the code wrong 3 times will lock the card and return CARD_LOCKED_BURGLAR_ALARM.

For this functionality you will have to had behaviour to the card simulator which allows it to count the number of wrong code counts and save them persistently.

Marking criteria

This code will be marked using automatic testing. The structure and format of the code will **NOT** be marked for this assignment, **however** you are strongly encourage to format and comment your code correctly.