COMP319: Software Engineering II

Assignment 1 (2023/2024)

(100% mark for Assignment 1.1 is 20% of COMP319 grade)

Deadline for Assignment 1: 14th of December 2023, 17:00

OBJECTIVE

This assignment involves Object Oriented Pattern design of a game.

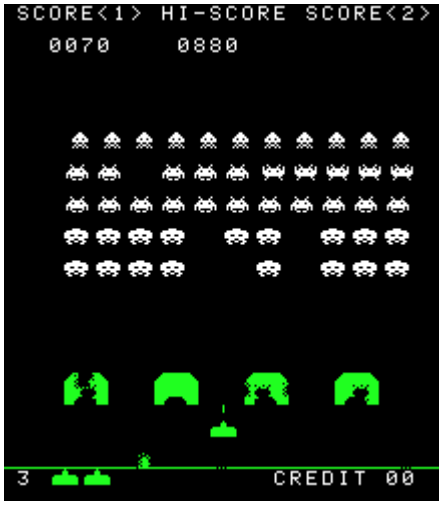
Assignment number	1 of 2
Weighting	20%
Assignment Circulated date provided to class	3/10/2023
Deadline Day & Date & Time	14th of December 2023 at 17:00 (5 PM)
Submission Mode	Electronic submission on Canvas (zip)
Learning outcome assessed	1. Understand the key problems driving research and development in contemporary software engineering (eg the need to develop software for embedded systems). 2. Be conversant with approaches to these problems, as well as their advantages, disadvantages, and future research directions.
Submission necessary in order to satisfy Module requirements	No
Purpose of assessment	To assess the students' ability to understand the use of object orientated patterns.
Marking criteria	See end of document
Late Submission Penalty	Standard UoL Policy

Instructions

This design task requires you to produce a class design for a game of space invaders, see Figure 1. The class design is to be submitted as code only (no UML is required). Each component of the game shown in the Figure must be modelled to a class. The implementation of the game does **NOT** need to include **rendering** of the game, so it does not have to be playable (your code should be model only – no view controller). However, you should have stubs in place where you would 1) expect an object to be drawn to the screen, 2) receive input from the player, 3) Make a sound.

The game does **NOT** have to work to score well, but does have to be a good design which shows relevant use of the patterns required. All classes should be used and have some relationship with other classes in the design.

To help you understand how the game works play it here
https://freeinvaders.org/



You are expected to demonstrate the use of the following OO techniques and patterns.

Factory

Chain of responsibility

Singleton

Open closed principle

Single responsibility

The code must be written in Java™.

Responsibilities

You should make a list of the relevant responsibilities for each class and determine what data should be modelled.

Code style and commenting

Please follow the coding style guidelines given here.

https://cgi.csc.liv.ac.uk/~coopes/comp201/codeStyleGuidelinesCS.pdf

Provide the public interface with each class with Java documentation comments.

This can be done in Eclipse by typing /** then hitting the return key.

Each public method of the class should have a comment and also the class itself should have a class comment.

Deliverables

The deliverable will be a zip file containing all the source code for your game, including all Java source class, interface and other files.

Marking

Relevant use of factory class for the problem given	10%
Relevant use of the chain of responsibility for the problem given	30%
Good application of open/closed principle within the code	15%
Relevant use of Singleton for the problem given	10%
Good coverage of game functionality	10%
Single responsibility conformance	5%
Overall code readability and quality	20%

Note you must use the patterns given to solve the problem given, so not just have for example a piece of factory code which is to solve a different problem.

Compile errors and marking

The code must compile, you will lose marks for compile errors. The more compile errors the more marks you will lose.

Code entry point

Provide a class called Main.java, this should be the entry point for your game and contain a static main method for allow to code to start.

Hints

To help you get started here are some hints. You do not have to follow this method, and you do not need to submit any related documentation however this will help you getting going. To get started write up a description of the game. Include as many details as you can remember to think up a name for the relevant components on the screen, e.g. player, sprite or high score. Once this is written use the noun identification technique to identify candidate classes.

Now start to think the responsibilities (associated methods) that the classes will have to support (for example drawing themselves on the screen, rendering). Also think of the relationships between the classes, aggregation, inheritance, interface implementation etc. This will give you a good start on how to code the game.