CSCI 3130: Introduction to Software Engineering

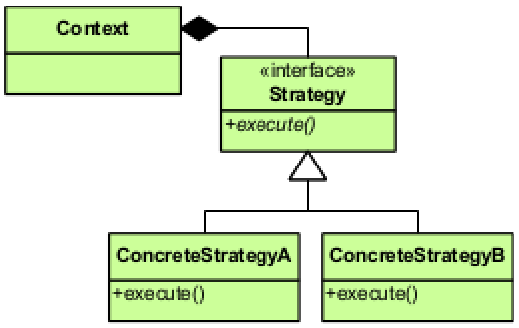
Assignment 4: Refactoring and Patterns

You work for an educational software company that is developing a suite of simple Physics simulations. The code in **pendulum.zip** is the beginning of a suite intended to permit the creation of a series of different types of pendulum simulation. You previously identified a code smell ("trivial layer") in the `AbstractEarthPendulum` class, but figured it was mostly harmless. Now your team wants to be able to dynamically set which planet a pendulum is on during an interactive simulation, and to allow different kinds of gravity models that are impossible to represent with a single constant (e.g., latitude models).

1. Run the main/test routine to see the expected behavior of the code.

2. Apply the refactoring approach "*Collapse Hierarchy*" to reduce the hierarchy levels by one. Ensure that the main/test routine still works as expected. Make a copy of all the code in a folder called "RefactoringStep1"

3. Next, use the **Strategy** design pattern (shown below) to replace the field `g` with a reference to a strategy object (as outlined below). This will allow different kinds of gravity models to be associated with a `Pendulum` instance dynamically.



Name the strategy interface `GravityModel`, with a single method
`public double getGravitationalField ();`

Assume that objects implementing `GravityModel` will be initialized with all state required to produce the gravitational field value under their specific model, and that this occurs before being assigned to a pendulum. In other words, the pendulums will just use the model assigned to it via the `GravityModel` interface. Adjust the pendulum constructors so that a `GravityModel` is assigned on initialization, and add a new method that will allow the `GravityModel` to be changed dynamically.

Define one concrete strategy called `GravityConstant`. This class simply receives a constant value for *g* in its constructor.

4. Run the main/test routine to ensure that the functionality hasn't changed. Copy all the code into a folder called "RefactoringStep2"

5. Modify the provided main/test routine so that it initializes a `SimplePendulum` and a `RegularPendulum` with a `GravityConstant` object, lets the pendulums swing for a little while, then assigns a different `GravityConstant` object. For example, your first `GravityConstant` could represent the Earth’s gravitational field (9.81 m/s^2), and the second could represent Jupiter’s (25 m/s^2).

Submit all code (including the 2 refactoring folders and the final version of the code with the modified test) as a **single zip** file on Brightspace.