Project description and report

In this project you will learn how to create frameworks for developing software applications, in this particularly case: a framework for matrix games (for example snake) with follow proprieties

- The play field is a rectangular matrix of "game objects", which can have different looks
- After the game has started it will develop as a series of moves even if the user is passive
- The users only way to change the progress of the game is through commands of simple keystrokes.

We will use the limits above to create a framework for such games, ie a package of cooperating classes for these applications.

A new game will be implemented by just writing a model class with the rules for the particular game.

Step 1. Understand the problem and **implemen**t a simple framework based on the documentation doc.zip and the description.

Start with exploring the documentation of the classes in the file BaseFrameworkDescription. Download the doc.zip documentation for the basic classes.

Understand how the game works and how everything is connected. This is very important for being able to develop the framework and your own game.

If everything it works you should be able to play the GoldModel Play . Se the implementation of the GoldModel class in BaseFrameworkDescription.

- Prove that your game works so far.

Questions to be answered:

-What are the difference between GoldModel and GameModel, and how are the classes related to each other?

-Which class is performing the actual drawing of the Gold coins in the Gold game?

-Which class calls for that drawing?

Step 2 :

1. By inspecting the code you have implemented, you should find the responsibilities for each class, how to create games and where a new game is started. As it is now, the constructor for the corresponding model class is called. This is not a good idea! It means that if new games or new ways of combining games into an application are thought of, a class of the framework must be modified. Change this by using the *factory design pattern* as follows: model creation should be left to a `GameFactory`.
    1. Define an `interface GameFactory` to describe what methods a game factory should offer. In addition to a method to create game models you should have in mind that all games produced by a given game factory should have the same *model dimension*, that is, the same dimension for the matrix of game objects. Let this be given by a method in `GameFactory`.
    2. Make the necesary changes to the calsses in the framework so that they make use of the game factory.
    3. Define a class implementing a game factory that can create models for at least *gold* so you can prove it works.
    4. Make the necessary changes to `SwingView` so that it uses the concrete factory above. In this way `SwingMain` is an application where you can change between *gold* or other games so you can change on fly!

2. These games usually come with a way of counting points (a *score*) and also record the best *n*-scores with the name of the player. Add this to the framework!

Step 3:

Create your own game using the framework. This should be unique to each student.

Step 4:

The meaning of this last part is to understand the importance of the documentation. You will produce a javadoc for your framework.

Step 5:

Project report

The final design report is due on Thursday, May 25, 2010. Your report should include the following information:

1. A short description of your implemented framework.
List the classes and the main methods with a list of their arguments and their return values.
Also describe in 1-2 sentences what the function does
to its arguments and what it returns (if it returns something).
Describe the relation between classes / advantage and disadvantage with the design of the framework.
Describe the design patterns you use or not use.

2. A short description of your game implementation and how your game uses the framework and a summary of the overall program flow. This could be a flowchart,
a state diagram or a simple textual description.

3. A one page description of the merits and demerits of your design. Include any modifications/enhancements you made to the framework. Mention the key issues you had in implementing the project and how you handled them.