EECS3311 SOFTWARE DESIGN – PROJECT

Text-based Game Prototype

ver. 1.0

PRZEMYSLAW PAWLUK

Due: August 1, 2016 **Weight:** 25%

Where: online **Type:** *Group (1-3 students)*

Main Points

Be sure to read and follow all the guidelines from the links on reports and academic honesty from the WWW home page for the course. The specification is the union of this document plus the program text you are given.

Learning objectives.

- The implementation and documentation of abstract data types as classes
- Eiffel programming
- Programming from contracts and test cases
- Application of design patterns

To hand in

Include your name(s) on the report cover page and in each .e file as a comment. The report is required for your work to be evaluated. The electronic submission is used to run your system but with no report, the code is ignored. Both code and report should be submitted electronically.

A report should be describing the project design. You should describe the general idea, entities/classes that you used and patterns that you did apply.

For all pre- and post-conditions and other contracts, explain briefly why they are important. The structure is to be based on the slide set Abstract Data Types Documentation. Use the cover page posted on the course web site. Keep your documentation brief.

You also have to select and apply 2 design patterns. Your report should describe your selection and explain why you think selected patterns are appropriate. You should also illustrate the application of the patterns in the design with diagrams.

Before the deadline, submit all .e and .ecf files for the system and a pdf with the project report. Use relative addressing in your system so your program will compile no matter where the files exist on Prism or Windows systems. No other files should be submitted (e.g. EIFGEN files etc). You should use eclean (on Prism) before submitting to clear away the unnecessary files. Submit also the report. *There is no need to print the report!*

To submit, use the following command on Prism

```
submit 3311 project [list of files]
```

Files cannot be deleted the submit command can only add or replace files so be very careful to clean up your directory before any submission. While you can develop your system on your personal computer, be sure your program will compile and execute on Prism using estudioXX.XX (where XX.XX is current version installed in the lab e.g. `estudio15.12`).

Tasks

Your task is to create a prototype of a text-based game. The game does not need to be complex, however, it should provide enough complexity to show your design skills. You can find some ideas for games in wiki (https://en.wikipedia.org/wiki/List_of_text-based_computer_games). Minimum requirements that your game has to satisfy are provided below.

Requirements

Please read carefully the following specification. It provides you with detail instructions along with points assigned to each requirement (in brackets).

The total number of points you can get for the project is **485** (implementation and design: **175** points, patterns: **200** points, report: **110**).

1. Scores: game should consistently keep track of scores of some sort (money, points etc.) depending on your game (design: 10, implementation: 10)
2. Game should provide an element of luck/randomization used in player moves or events that happen to the player (design: 10, implementation: 10)
3. Game controller class
   - 3.1. The class has only one instance in the system (design: 10, implementation: 10)
   - 3.2. It is the randomization source, i.e., if any other class needs random number, it asks game controller. This way, you will have more control on the randomization for testing purposes (implementation: 10)
4. Player interface
   - 4.1. Player should communicate with the game using keyboard and screen (text mode) (design: 5, implementation: 10)
   - 4.2. Game should provide a player with a list of options or react to the input provided by the player (see below for details)
5. Game world
   - 5.1. There should be clear start and game over communicated to the player (design: 10, implementation: 10)

   5.2. Should provide multiple states/fields that player can be in/standing on (design: 15, implementation: 15)
   5.3. In each state should offer a player with a list of possible options (design: 10, implementation: 10)
   5.4. Should offer player some pick-up elements that will affect the player state e.g., add points, deduct money or health (design: 10, implementation: 10)
6. Patterns: your project should utilize at least three design patterns. One of them would be Singleton required by the Game controller.
   6.1. Appropriateness of the selection (2 x 20)
   6.2. Implementation (3 x 20)
7. Report
   7.1. List of students (10)
   7.2. Game description (10)
   7.3. Brief class ADT documentation (30)
   7.4. Contracts documentation and implementation (documentation: 10, implementation: 20)
   7.5. Diagrams of the system showing an overview (10 points) and patterns (2 x 10 points)