ITECH7201 Software Engineering: Analysis and Design (1920)

Assignment 2

Overview

For this assignment, you will logically extend the functionality of the Maze Game introduced during the lectures, via the modification of the code base as well as documentation and implementation of various user stories. **This is a paired assignment**. However, in case of an exceptional situation (if the number of students in a group is 3), the group should take special permission from the Lecturer.

Timelines and Expectations

Percentage Value of Task: 20%

Due: Sun, Oct 27, 2019 - 23:55 (week 11)

Minimum time expectation: 20 hours

Learning Outcomes Assessed

The following course learning outcomes are assessed by completing this assessment:

- Understand the significance of detailed project planning and control, good communication and documentation and the use of appropriate tools in order to provide a quality product
- Understand the distinction between software engineering and programming, and thus the distinction between a software configuration and a program.
- Understand the methods and techniques involved in designing, implementing and maintaining an information system, in particular using an object-oriented approach
- Understand how unit tests are used during software development to assist in agile programming techniques such as refactoring
- Work together in small teams to complete a fully documented, detailed design and implementation of a small business information system
- Demonstrate skills in designing and implementing an information system
- Demonstrate skills in designing Unit tests

Assessment Details

You are required to logically extend the functionality of the Maze Game introduced during lectures, via the modification of the code base as well as documentation and implementation of various user stories. You will use the Boost methodology discussed during lectures, which requires the use of pair programming. All documentation, other than the customized game map, must be completed individually.

The code base provided for this assignment has already implemented the “warm up” and some “sets”. You will be implementing numerous other “sets” for this assignment using the Boost methodology. The “warm down” stage is not required.

You are free to take ideas discussed during lectures and implement these in your own version of the code base provided in Moodle for this assignment.

This assignment will be marked according to the functionality of your code, in addition to the elegance and extensibility of your design and the quality of your documentation.

Note: It is recommended that you spend some time familiarising yourself with the provided code base prior to beginning any work on this assignment. You should start by spending some time exploring the structure of the code to gain an understanding of the roles each class plays within the system and then complete Lab 7-8.

Assignment Requirements

In Pairs:

1. Design and implement a custom environment for your maze game. This requires producing your own unique hand-drawn map of the maze game environment and changing the HardCodedData file to reflect the locations and items on your map. At least *M* (*M*>=5) locations, and at least *N* (*N*>=2) shops with a condition M>N, must be included, as well as enough items to allow for proper testing of the game functionality. For example, there must be sufficient items to verify that a player cannot collect an item if the weight restriction has been met. Note: you should not retain the original locations or items from the provided code base in your version of the game.
2. Implement each of the following deliverables/milestones:
   a. functionality as detailed in Lab 7 and Lab 8
   b. commands to manage the various item management commands:
      i. listItems, getItem, and dropItem, to allow the listing of all held items, collect a new item and drop a held item.
      ii. equipItem and unequipItem, to wear / wield a held item, and to stop wearing or wielding an equipped item.
      iii. purchaseItem and sellItem, to buy an item from a vendor and to sell an item to a vendor.
      iv. getmazestatus to show the status of the maze (i.e., value of *M* and *N*)
      v. Flee/use potion command - used in danger.

   Ensure that weight restrictions are not exceeded and that the context for each command is appropriate. For example, purchasing and selling of items should only occur in a shop.

   c. Basic combat functions, allowing a player to attack or to be attacked by a non-player character (NPC). A player may flee combat or continue to attack until such time as one combatant loses all life points. The end result must not be hard-coded, and neither the hostile NPCs nor player may have their attributes configured in such a way that the final outcome is pre-determined. Note that combat functions should only be available when the player is in the presence of a hostile NPC, and combat may take whatever form is appropriate for your game context rather than being limited to just hits with a weapon.
   d. Collectable items that, when used, restore life points to a player. The number of life points regained should be determined by the roll of 2 six-sided dice up to the player's maximum health. You can elect to allow these to be used during combat if you wish.
3. Write **Three Unit test cases** for each of the methods you are developing in this assignment. You can write the test cases in any Java framework, but Junit is preferred.
4. Prepare a short (5-7 min) video presentation to demonstrate your software/program as if you are going to sell this software/program to a prospective customer. The Power-point presentation must contain the following items:

a. A demonstration of the game – how the game play works and how this was implemented.
b. Explanation of the role of design patterns in the game, clearly identifying the functionality where design patterns have had an impact and how this has occurred.
c. An analysis on the use of a development methodology in creating the game describing how did this assist, or make the development of the game more complicated.

This presentation will be assessed based on the clarity and accuracy of the explanations and how clearly it demonstrates an understanding of how the game was coded, the design patterns used and the impact of using a development methodology.

Individually:

1. Prepare an individual report, to be submitted as a Word document or a PDF, which includes:
    a. The student number and name of each person on your team (including yourself)
    b. User stories for each of the deliverables/milestones
    c. Class diagrams for Lab 7 and Lab 8
    d. Sequence diagrams for two (2) of the item management commands, taken from separate groupings (i.e. you will not get credit for both getItem and dropItem as they are both in group i)
    e. A statement of your own personal contribution to the assignment
    f. A statement of your partner's contribution to the assignment.

Do not work with your partner or any other person to complete your individual report. These must be unique and your own work.

Please note that assignments will NOT be marked and zero marks will be allocated if the individual statements of personal and partner contributions are not submitted.

Submission

Each student must submit a single zip file which contains all assignment files in the Assignment 2 submission box provided in Moodle. Submission files include a photograph or a scanned image of your hand-drawn map, code for each deliverable/milestone, presentation file, and an individual report containing your student number, name, your partner's student number and name, diagrams, user stories and contribution statements. A link to your demonstration video should be in your report.

Marking Criteria/Rubric

Student ID		**Partner ID**	
Student Name		**Partner Name**	

Pre-Requisites For Marking • Statement of personal contribution and partner's contribution	

Task	**Available Marks**	**Student Mark**
Paired Tasks		
a. Functionality as detailed in Lab 7 and Lab 8	6	
b. Keeping the provision of taking the values of *M* and *N* while the program is in execution	6	
c. Hand-drawn map detailing custom game environment, implemented in the game, with at least *M* different locations including exactly *N* shop(s), and items to allow for full testing	6	
d. Item management commands:		
i. listItems / getItem / dropItem	3	
ii. equipItem / unequipItem	3	
iii. purchaseItem / sellItem	3	
iv. getmazestatus	3	
v. flee/ use potion	3	
e. Implementation of combat sequences between a player and hostile NPCs, including variable outcomes and the ability to flee.	5	
f. Implementation of collectable items used to restore a player's life points	5	
g. Unit test	12	
h. Video presentation - by all group members	15	
Individual Work Individual report containing team details and:		
a. User stories for each of the deliverables / milestones	10	
b. Class diagrams for Lab 7 and Lab 8	10	
c. Sequence diagrams for two of the item management commands from different groups.	10	
Total	**100**	
The percentage value of the Task	**20%**	

Feedback

A completed marking guide will be uploaded in Moodle and marks uploaded to fdlGrades within 2 weeks of assignment submission.

Plagiarism:

Plagiarism is the presentation of the expressed thought or work of another person as though it is one's own without properly acknowledging that person. You must not allow other students to copy your work and must take care to safeguard against this happening. More information about the plagiarism policy and procedure for the university can be found at http://federation.edu.au/students/learning-and-study/online-help-with/plagiarism.