Software Development I

Game Project v0.1

Overview

This semester we are building an interactive adventure game in Java in the object-oriented style.

For this second version of your project, you will design a working, albeit rudimentary, framework for your game. This will include several Java Classes and, for now, employ a text-only interface.

Deliverables

- Upon launch, display an introduction including title, back-story, and description of the starting location.
- Upon completion/exit, display a the game credits including at least author and date.
- After each game update, display the following information:
  - current location, number of moves made, directions to move (from the current location)
- Include the separate classes for the following game objects:
  - Player – has at least a name, current location, and inventory.
  - Item – has name and description. Make at least four (4) items; one must be a map of the world.
  - Location – has at least a name, description, and list of items present.
  - World – contains at least eight (8) and uses a matrix to define connections between them.
    - You may choose to employ an *N*x*N* adjacency matrix or a 4x*N* navigation matrix.
- Use Game Engine class to manage operations of the game, including the following basic tasks:
  - Step of a basic game loop – read user input, update the game state, render the game state.
  - The game engine must be able to display the intro and credits.
  - Launch the game via a main method that creates the game objects and starts the game loop.
- Your game must read user commands as case-*insensitive* single characters (for now).
  - Commands correspond to the following player *methods* (command character shown in **bold**):
    - Move *either* **N**orth/**S**outh/**E**ast/**W**est *or* forward/back/turn left/turn right (**WASD**).
    - **T**ake an item from and **D**rop and item to her current location.
    - Check her **I**nventory and look at the **M**ap (if she has found and taken it).
    - **Q**uit the game.
    - Report invalid commands and display help information as needed.

Source Code

Your program design must clearly use separate objects for distinct concerns. Your code must be consistently formatted and demonstrate accepted best practices.

Submitting

Create a new Branch of your BitBucket repository. Push changes frequently with concise and relevant commit messages. Open an Issue if you need any help along the way – I reply quickly.
Finally, send me a Pull Request before the iLearn due date. Once graded, merge your branch it into the *master*.