**Final project: The Cleanup**

**Groups of 2 – same members as before**

**Date Due – Finals week**

**Problem and Specification**

All programs have to go through a final cleanup stage, now that you have some design patterns to work with, see which ones can be used to help tidy up your program. In addition, the following new universal commands are to be added. Using your previous game, you will introduce a Session mechanic to it. This will allow you to resume from where you left off or switch between users

**Additional Commands**

|  |  |
| --- | --- |
| **FORMAT** |  |
| REGISTER <NAME> | Registers a new user with a default state or loads and existing user |
| <COMMAND> <PARAMS> | <Reply based on your game > |
| QUIT | Quits the game |
| SAVE | saves the current user’s state |
| LOAD | Reloads the current user’s state |
| RUN | Runs a file of commands |

**Implementation**

State

* Until the user registers a new or existing player, the game will not process any commands except REGISTER or QUIT
* After this is done, all commands can be processed as usual

Memento

* Multiuser support -- assume the name is the unique identifier for each session, REGISTER <NAME> will switch to an existing session if present or make a new one if not
* Session information should be saved to a file and loaded as needed
* Use the Memento Design pattern to handle sessions saving

Strategy

* Use the Strategy pattern to switch between a command line mode to a file reading mode
* Provide a file to traverse and complete the entire game (so you wont have to type everything)
* This file can be used to register users, load, etc.
  + Each command is one line in the file

Try and use other design patterns where you see fit and can clean up the code (e.g. Command) The final cleanliness of the code and design will factor into your grade

* Provide a UML of your original program AND final program
  + This will act as the basis of comparison
* Keep an eye out for those huge if-elses … I certainly will