Final Project: RPG Game

Prompt:
Implement a basic RPG game with at least three (3) characters types (wizard, warrior, etc.) You will implement a duel where two character instances will fight each other. As an example, a warrior character fights enemies with a weapon of choice; swinging a weapon costs the character stamina. A wizard character fights by casting spells; casting spells inflict damage and expends magical power or mana. The two will face off and take turns selecting actions, which change the status of themselves and their opponent, until one or both are defeated.

Design Requirements:
Implement the Character class
- Create the base Character class. It should manage variables that keep track of every character's state, such as name, level, or health. Include three (3) interaction functions that change those variables. For example, resting or using a healing item should increase health. Adding xp should increase a players level and allow for additional or more powerful abilities.
- Include a function that updates the character's state values in combat as well as after combat.
- The Character class should have a method to save and load all of its data onto or from a file.

Implement Types of Characters
- Create at least three (3) new derived classes that inherits from the above Character class
- Each new type (class) should be unique, from printing a unique message for each interaction to the particulars of their attack and defense. Add actions that the characters can take in support of a duel system where the characters alternate selecting actions and attacking/defending/casting spells until one is defeated (zero health).
- Make sure to override the file save and load methods to make sure the new variables associated with the derived classes are updated in the save file.

Implement the Game Loop
- After the game starts, the player should be presented with a menu of the list of available character types to choose from, perhaps with a brief description, or allowed to load a character from a file.
- When creating a new character, the player should be asked to select a character type and to provide a name. Random starting attributes may be assigned to the new character. Any further game message for the character should address it by its new name.
- The game will then present the player with a duel against an opponent with interactions (attack/cast a spell, defend, flee, …), save or load files, or exit the game.
- It should handle player actions, until a player flees or the duel is complete, and repeat with the menu options and provide duels until the player exits.

System Requirements:
- Must use file I/O for saving and loading game data
- Must implement a character base class and at least three (3) different character type classes derived from the base class, with each having some unique attributes, abilities, and/or interactions.
- Implement a duel system between two character instances.
- Use comments to explain the sections and classes and to clarify actions and abilities.
- Consider use polymorphism for easier programming.
- Try using text styling (ASCII) to create an aesthetic menu system.

Final Project: RPG Game

Submission:

1. **Prepare a Preliminary Project Report**. This report should include a description of how the program is (or will be) designed, including the design of the classes and their inheritance relationships. Be sure to describe the main functionality (in the form of pseudo code or step-by-step instructions) of the program. Please also provide the menu options and their use cases for your program. You can refer to ZyBooks 11.10 to learn how to use UML to describe your OOP design.
2. **Prepare a short two to three-page report** explaining how the program was designed and developed, what classes were used, any special instructions to operate the program, and how the final implementation differed from the plan. **Submit the report on Blackboard**
3. **Submit your source code as a single .zip file on Blackboard.**
4. **Provide a link to a video clip (aim for 5 minutes in length) demonstrating all the above required functionality. Submit this to Blackboard.**

Grading:

- Mid-term report **25%**.
- Peer-grading (based on the uploaded video) **25%**.
- Final product graded by instructors and TAs **50%**.
    - Final report 20% (out of 50)
    - Demo via video clip 40% (out of 50)
    - Source code 40% (out of 50)