CS 3330 OOP Final Project: Bannerlord

Mount and Blade II: Bannerlord is a new video game that reached public beta just over a month ago, after ten years since the last release and seven years since its announcement. Its open world gameplay itself is quite unique, but the story line is about a medieval world in which you are a knight trying to make a name for yourself and move up in the world. Made by a small indie dev team, it is extremely mod-friendly. The game is written mostly in C#, a language that I'll be learning next semester, which is very similar to Java. In the meantime, while I cannot develop mods, I will make this standalone program to perhaps be ported into C# and into the game itself at a future date.

Required Elements:

1. Object oriented elements that you write the code for:
   - a. Classes.
     - -Army class
   - b. Subclasses.
     - -All subunits (subclasses of Unit class)
   - c. At least one abstract class.
     - -Unit class
   - d. At least one Interface.
     - -BannerlordCollection interface
2. Code elements that you utilize:
   - a. One or more collection classes.
     - -Army class
   - b. Exception Handling.
     - -Try/catch in handleAboutPage() & addToArmy() in FXMLDocumentController
3. The application must have a clearly defined model (as in the M in MVC). ^
4. The UI must utilize multiple scenes and at least one of the scenes will have the contents of the scene graph changed based on the application state.
   - -Battle Simulator*: Simulate your auto-resolve chances to determine if it's worth it.
   - -Army Planner: Plan your armies out ahead of time to determine its cost.

*Ran out of time to implement, will have to take off points. Sorry, was very busy!

5. There must be a way to access "About" information that includes information about you and the application.
   - -Help→About in the menu bar; takes user to GitHub page.
6. The application must save data and load data. The target for saving/loading data can be files, a network service, and/or a database.
   - -Object Serialization*: Save & load armies locally

*Tried everything I could to get it to work, rewatched the class lectures, youtube videos, etc but just couldn't get it to work properly no matter what. Anyhow, the feature is still there but catches an IOException that gets thrown every time.

Expectations:

1. The application is functional for a defined activity, task, and purpose. The goal is to develop a complete application not just a pile of code that doesn't serve a purpose.

-The function of this program is help a player of Mount & Blade II: Bannerlord plan out their armies. In the game you build armies that can get very large, which without careful planning could bankrupt you. As Bannerlord continues its development they will add more features such as army cohesion: how well your troops fight together which will depend on their mix of culture. Luckily, with the power of Object Oriented Design & Java, it will be very easy to update these features into my program when they are added to the game.

2. The user interface is useable, organized, and understandable.

-I would say so myself.

3. The code is well-structured and logically organized.

-While it may not be the cleanest or up to industry standard, I think I did pretty well for my first full scale Java program. As I practice more and more, my code will become cleaner & more concise. In fact, in the development cycle of this program alone, I was able to reduce many lines of redundant copy & paste code after realizing I could polymorphism to help me with that. This allowed me to reimplement many functions into one function that could handle everything, cutting our several hundreds line of line. (Though I wasn't sure if the same could be done with @FXML functions so I did not mess with it.) This helped me learn how powerful a tool polymorphism & Object Oriented Design as a whole is, and gave me a new found respect for the Java programming language.

4. The application you build is not to be trivial in simply meeting the requirements set forth in this document. Yes, you are to meet the requirements but you are also to build an application that has a purpose and delivers functionality or capability. The requirements are parameters to be used in design and implementation of the application. They are not intended to be the end product.

-While the program is a little light & simplistic (besides the overwhelming amount of Unit subclasses), this was in no way intentional; I was not trying to do the least amount of work possible, the program is just light by nature of its function. It was difficult for me to find the proper place for an interface in my program, though, as my abstract class was really all I needed.

5. You should design and build an application that you would be happy to show a prospective employer or client.

-While I doubted my ability to produce a respectable program after my first introduction class to Java, I am now proud to show this program to prospective employers & as such have listed it on my personal GitHub page. As I add more features onto the program as more features are added to the video game, it will become more impressive. I'm now considering joining the video game modding community, and possibly modding professionally in the future. Many people are paid by crowdfunding sources like Patreon to produce mods for moddable games.

MoSCoW

Must have: Required Elements from First Page

Should have: Pleasing UI

Could have: Reflection: Custom units you can make yourself (because of user mods, etc)

Want to have but cannot: Program imbedded into game itself