Deliverable 1

Scenario:

My project will be a medieval character creation tool which will allow users to customize and create unique characters based on a multitude of different options ranging from character class to what type of gear their character wears.

Design Paradigm:

Through this project I plan to demonstrate functionalities such as:

- Character creation through the use of classes and data fields
- Itemization such as weapons and armor using Lists or HashMaps
- ASCII art visualization to allow the user to see their creation
- Error handling such as preventing invalid choices
- Save/load progress by using file I/O to edit and store creations

Expected Output:

The user will select from a variety of options presented to them at every stage of the process, such as things like what gender their character will be, or how what weapons their character will wield. Through these choices, the user will slowly build their character using text I/O files to store their choices. Finally, the results of the program will be a stored ASCII art image of the character they created. This program, among other things, will allow users to create a visualization of a character they dream of before they actually do so in other games such as Dungeons and Dragons where they might not be allowed to make changes to their character after creation.

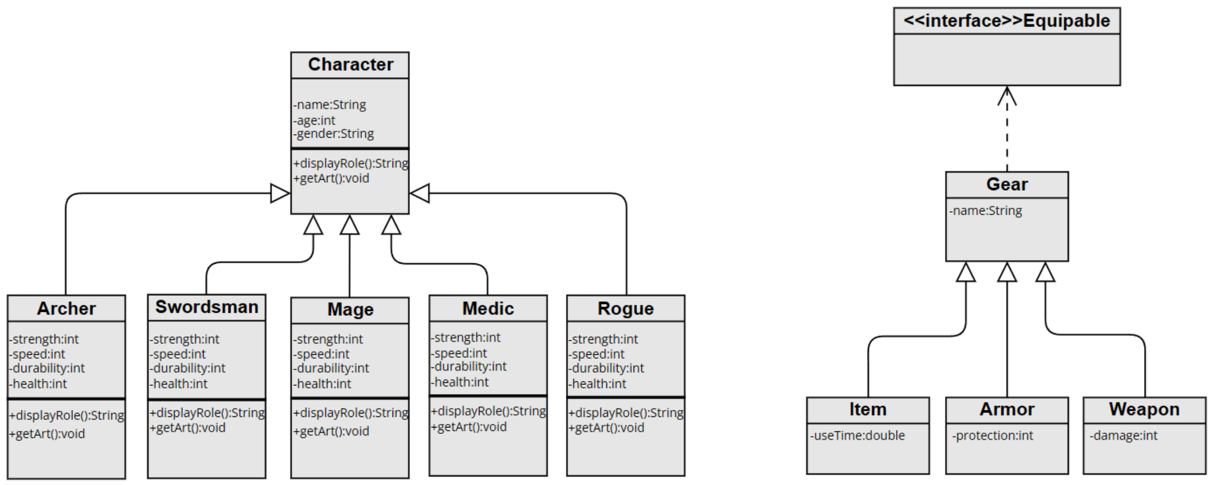
This project will have a Character class at the top of the hierarchy, branching off into multiple other sub-classes which will be the types of roles offered for the medieval character. The Character class will contain data fields such as name, gender, and age, while the sub-classes will be things like archer, swordsman, or mage which contain specific data fields which define the characters unique attributes such as strength, speed, or durability.

The interface will be called equipable, it is implemented by the gear class which is extended by the item class, the weapon class and the armor class, it determines whether something can be equipped by the character or not.

Certain methods will apply runtime-polymorphism such as the "displayRole" method which will be different depending on whether run in the Character class or one of the role classes, the methods which will allow the user to display the statistics of their character which changes based on class, or the "getArt" method which will print out the image of their character at any specific point in the creation process.

The user can use textIO to store these images of their character as well as the statistics and their information.

Comparable will be implemented in things like sorting characters by name, age or health. While the Comparator will be used to compare more complex things such as the details of different characters, like their roles or their weapons.



For deliverable 2, I will add the classes as well as some of the methods as a rough skeleton for my project, I will add details as I progress.