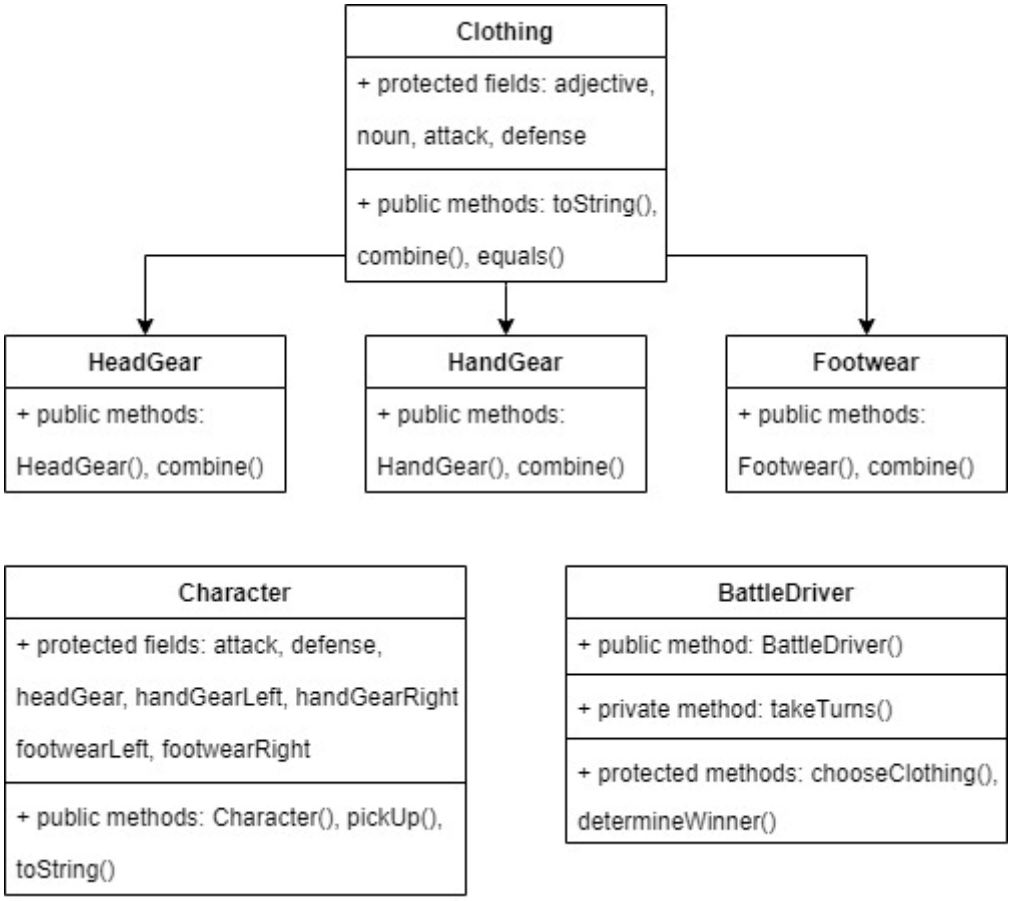
- My design will use 3 subclasses (i.e., HeadGear, HandGear, Footwear) inherited from the class Clothing to represent different types of clothing. Each subclass will also contain its own constructor and combine() method. There will also be a class named Character that has five fields for wearing clothing, including head gear, left hand gear, right hand gear, left footwear, and right footwear. The passed-in clothing can enhance the character's default attack and defense points. Lastly, there will be a BattleDriver class that simulates a battle between two players with a list of available clothing items.
- For each clothing item, it will contain fields including its adjective, its noun, its attack points, and its defense points. They will have "protected" access because it both allows me to access these fields within this package and restrict users from directly manipulating them.
- My strategy for testing will based on each unique method of each class. I will create several clothing items first and then use them as input for BattleDriver's available clothing list. Then I will test if BattleDriver.chooseClothing() and BattleDriver.determineWinner() work fine before testing the standard output of the BattleDriver constructor.



Testing plan:

- Test if the constructors of HeadGear, HandGear, and Footwear work fine, and if their combine() method can successfully combine items of the same type or throw an exception when items of different clothing subclasses are passed in. Finally, test if their toString() output matches the expected format.
- Test if the constructor of Character works fine, and if its pickUp() method allows its instance to equip a clothing item in an appropriate field or combine an existing clothing item. I'll also test if its toString() output matches the expected format.
- Test if BattleDriver throws an exception when the clothing list has an insufficient length (< 10).
- Test if BattleDriver.chooseClothing() can choose the optimal clothing from the passed in list with all the filtering rules successfully applied.
- Test if BattleDriver.determineWinner() can determine a winner or announce a tie accurately when two characters with certain attack and defense points passed in (with the given rule applied).

- Test if the constructor of BattleDriver can simulate the game appropriately, print out the readable content to demonstrate each player's status, and end the game with an accurate announcement (as standard output).
- The following Clothing instances will be used as primary test examples:

Variable name	**Class**	**adjective**	**noun**	**attack**	**defense**
hat	HeadGear	Fashionable	Hat	0	1
helmet	HeadGear	Solid	Helmet	0	6
visor	HeadGear	Portable	Visor	0	0
gloveCotton	HandGear	Cotton	Glove	1	0
gloveMetal	HandGear	Metal	Glove	2	0
sword	HandGear	Sharp	Sword	8	0
shield	HandGear	Heavy	Shield	4	0
pistol	HandGear	Dangerous	Pistol	10	0
boot	Footwear	Leather	Boot	2	2
sneaker	Footwear	Light	Sneaker	1	0
hoverBoard	Footwear	Happy	HoverBoard	3	1
sandal	Footwear	Scurry	Sandal	1	0