Nathan Hamilton
CSCD 349-01
Winter 2018
Refactoring

1. Removed theHero variable from chooseHero() method within Dungeon class, because it is unused.

```
public static Hero chooseHero()
{
	int choice;
	Hero theHero;
```

2. Added constant values for all six monster and hero classes that represent hitpoints, attack speed, etc. to eliminate hardcoded number in class constructors.

Example:

```
public Warrior()
{

	super("Warrior", 125, 4, .8, 35, 60, .2);


}//end constructor
```

3. Within the Sorceress class, added “static” to MIN_ADD and MAX_ADD and changed both visibilities to “private” in favor of encapsulation.

```
public final int MIN_ADD = 25;
public final int MAX_ADD = 50;
```

4. Created CharacterAttack interface to ensure all current or future Monster and Hero classes implement their unique version of the attack(DungeonCharacter) method. (Strategy Pattern)

```
public interface CharacterAttack
{
	public void attack(DungeonCharacter opponent);
}
```

5. Removed comments describing each method (code smell), because the methods are written well enough that they do not need descriptions.

Example:

```
/*-------------------------------------------------------
readName obtains a name for the hero from the user
```

```
        Receives: nothing
        Returns: nothing

        This method calls: nothing
        This method is called by: hero constructor
        ----------------------------------------------------------*/
```

6. Removed Comparable implementation and compareTo() method from DungeonCharacter class, because it is unused.

```
public abstract class DungeonCharacter implements Comparable
{
        public int compareTo(Object o)
        {
                return 1;
        }
```

7. Removed unused methods from Keyboard class. (i.e. readBoolean(), readLong(), readFloat(), readDouble())

```
public static boolean readBoolean()
{
        String token = getNextToken();
        boolean bool;
        try
        {
                if (token.toLowerCase().equals("true"))
                bool = true;
                else if (token.toLowerCase().equals("false"))
                bool = false;
        else
        {
                error ("Error reading boolean data, false value returned.");
                bool = false;
        }
}
catch (Exception exception)
{
        error ("Error reading boolean data, false value returned.");
        bool = false;
}
return bool;
}
```