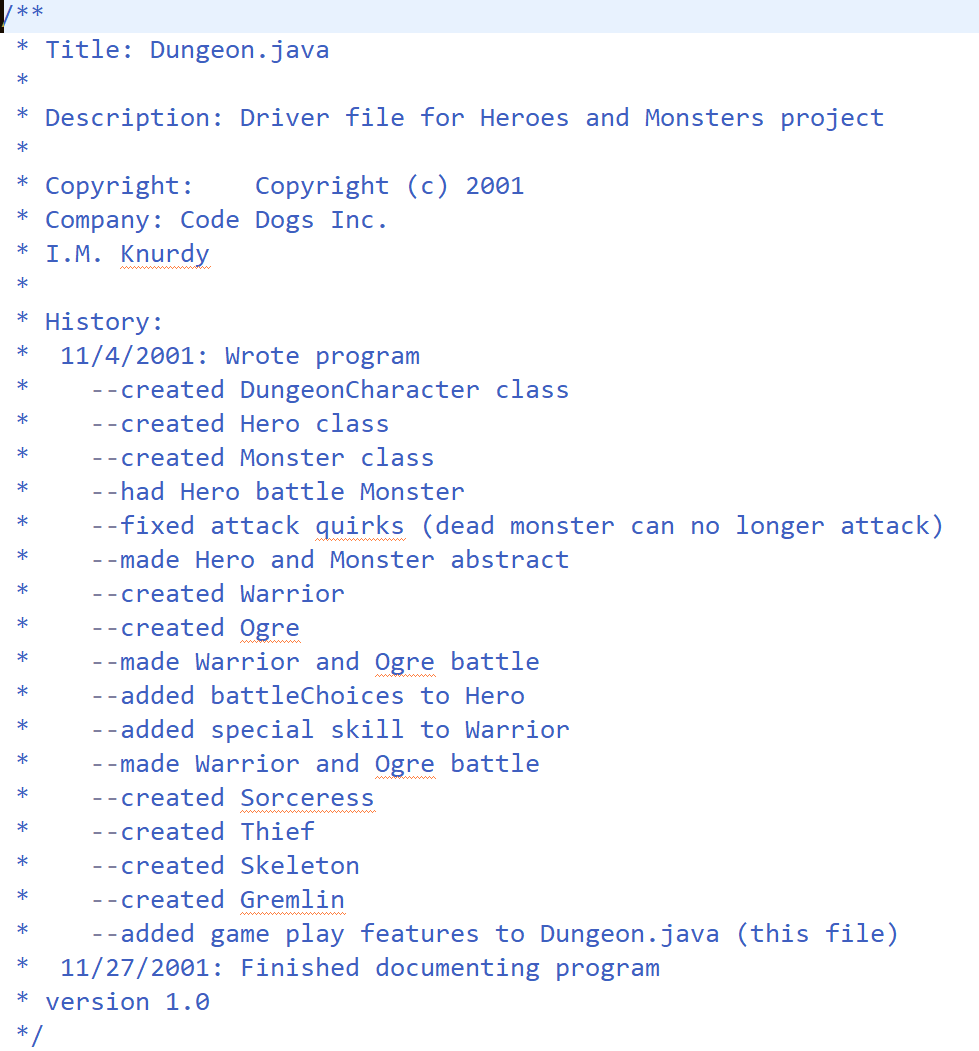
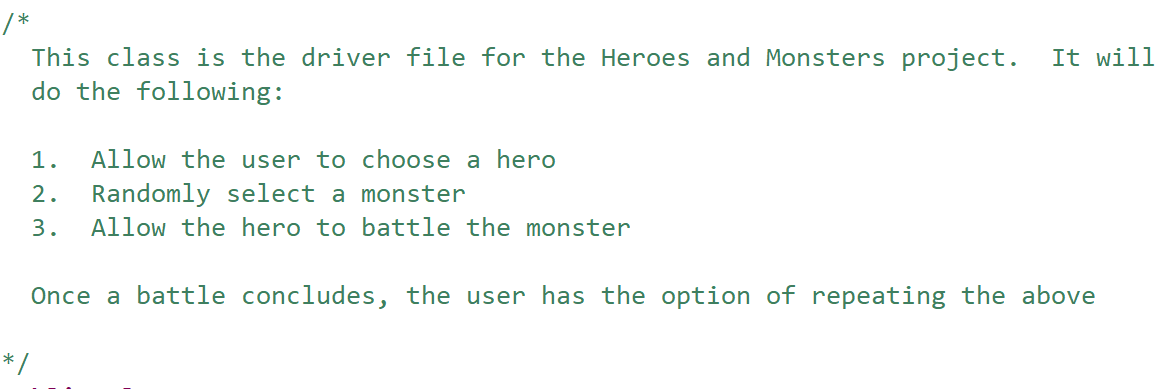
Group: Marie Taylor, Aleks Chernega, Jay Brown

Refactors:

1. Delete comments: deleted all excessive and messy comments including descriptions and copyrights. From the example below, all comments were deleted. This cleans up the code immensely and minimizes the length of the classes.

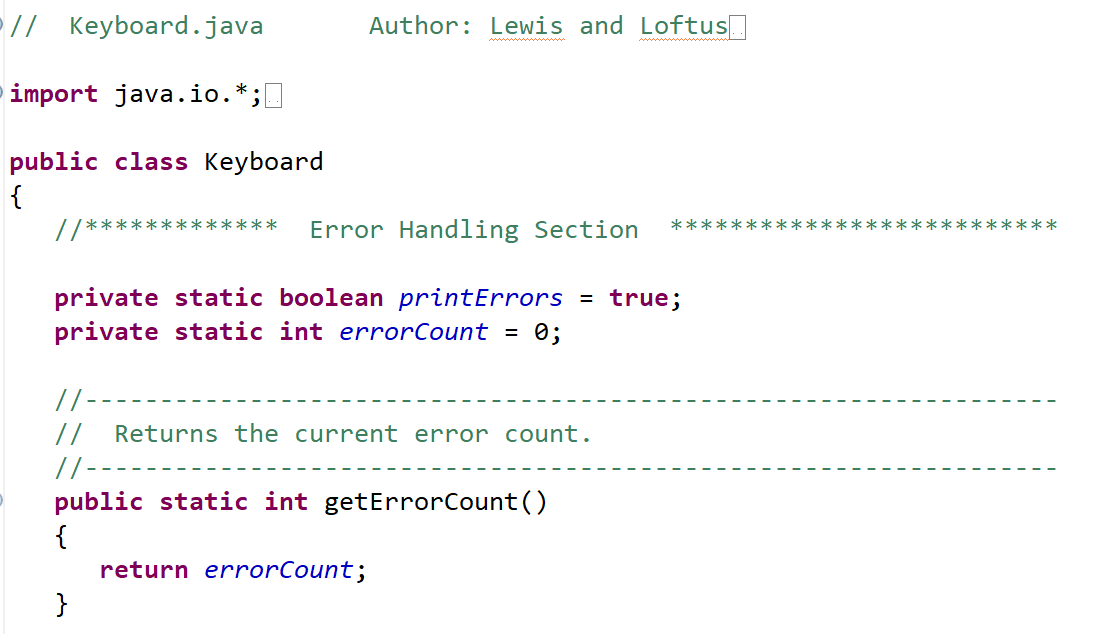
An example of the offending code:

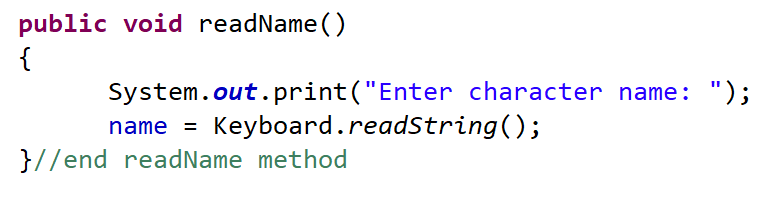




1. Change Keyboard to Scanner: deleted the Keyboard class entirely and replaced it with the built in Scanner class. This improves the execution of reading input from the user and removes an entire class from the class structure.

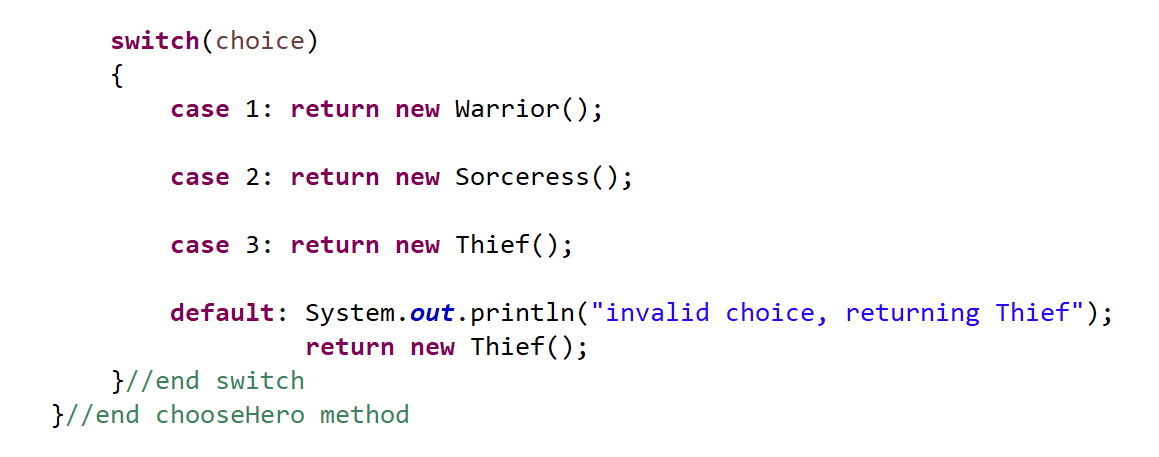
An example of the offending code:

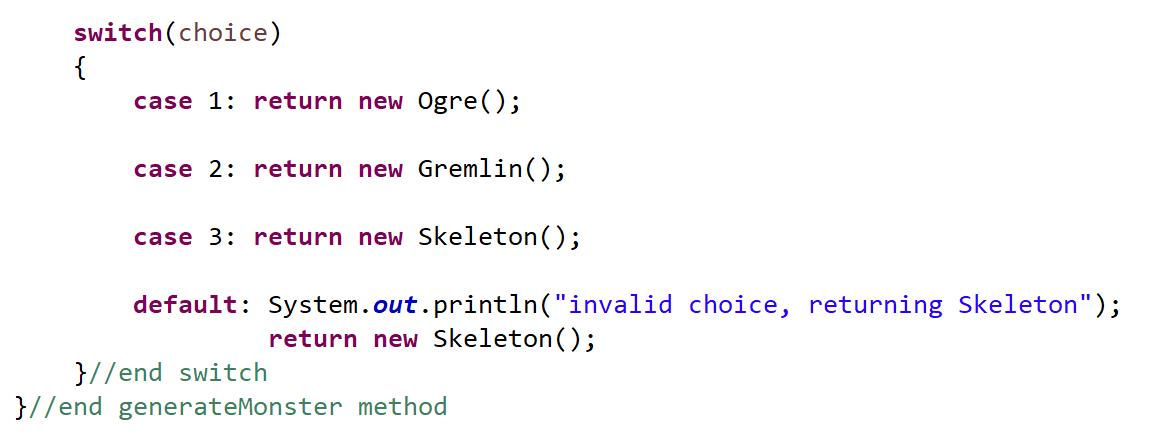




1. Create a Hero and Monster Factory: created a simple factory for both Hero and Monster. The code snippets below are of the old way of creating a Hero and Monster and were replaced with the Hero and Monster factories. This encapsulates object creation.

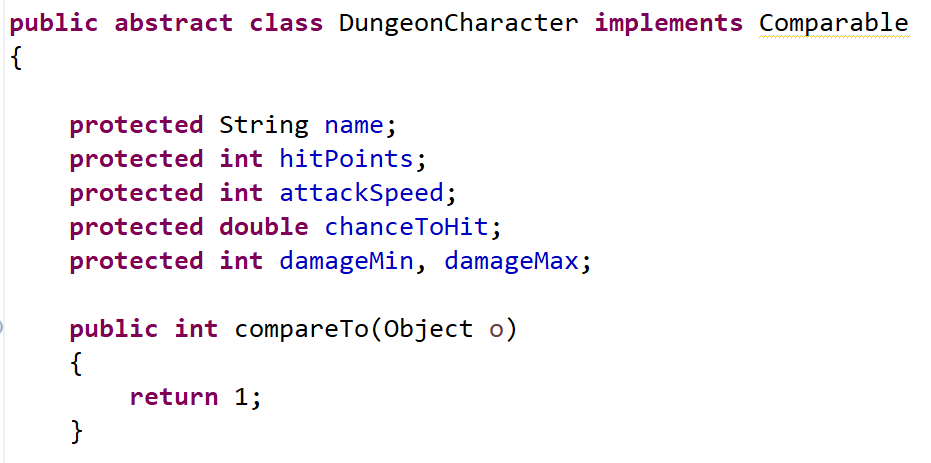
An example of the offending code:





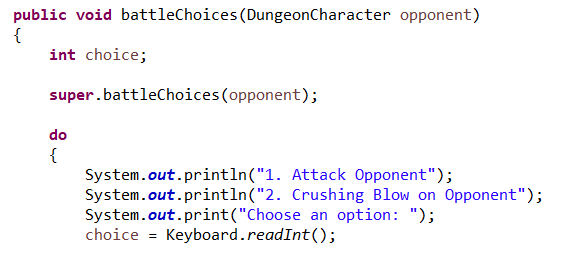
1. Delete Dead Code: deleted code that was not being used. From the example below, the compareTo method was removed from the DungeonCharacter class since it was never used. In turn, DungeonCharacter no longer implements Comparable. This cleans up the code and minimizes the length of classes.

An example of the offending code:



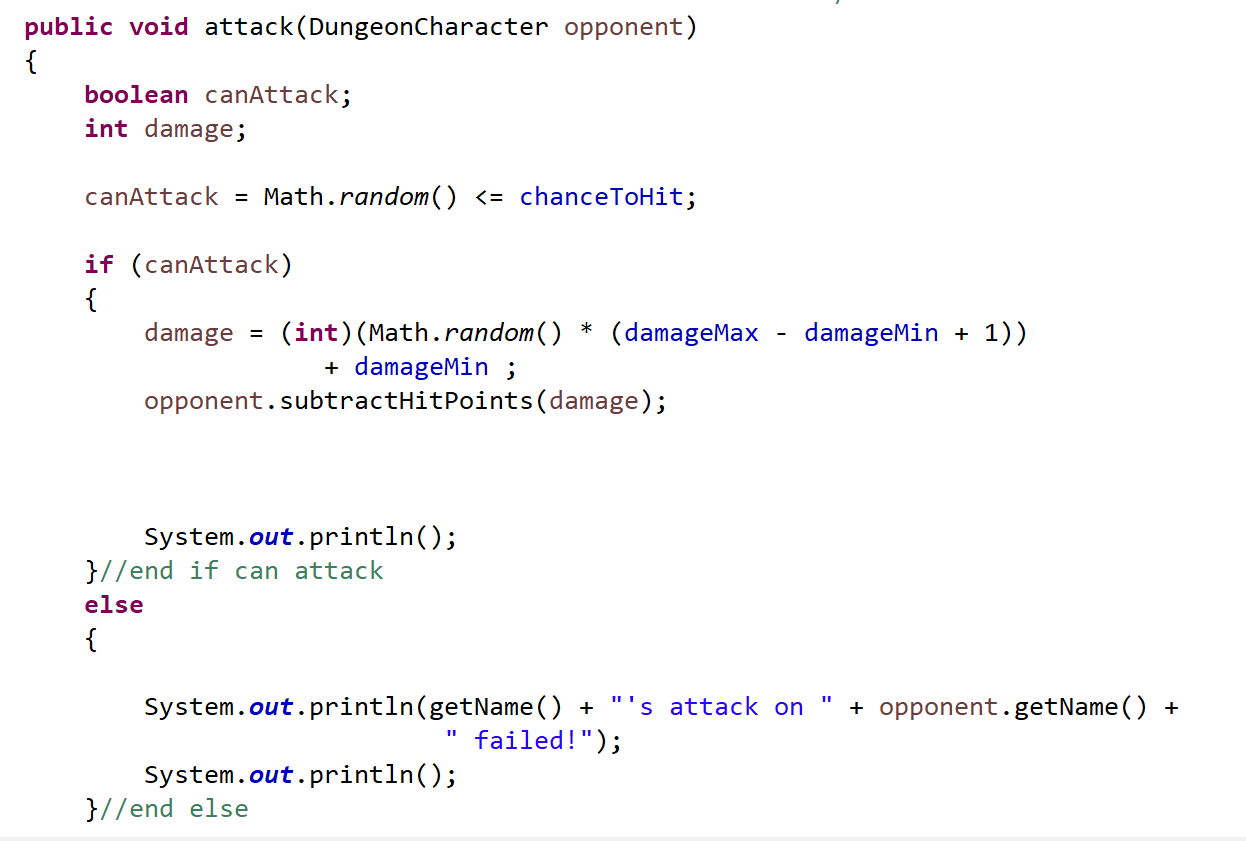
1. Remove Excessive Print Lines: combined unnecessary print lines into one line when possible. From the example below, option one and two were condensed to one print line with a new line character. This cleans up the code by eliminating redundancy and minimizes the length of classes.

An example of the offending code:

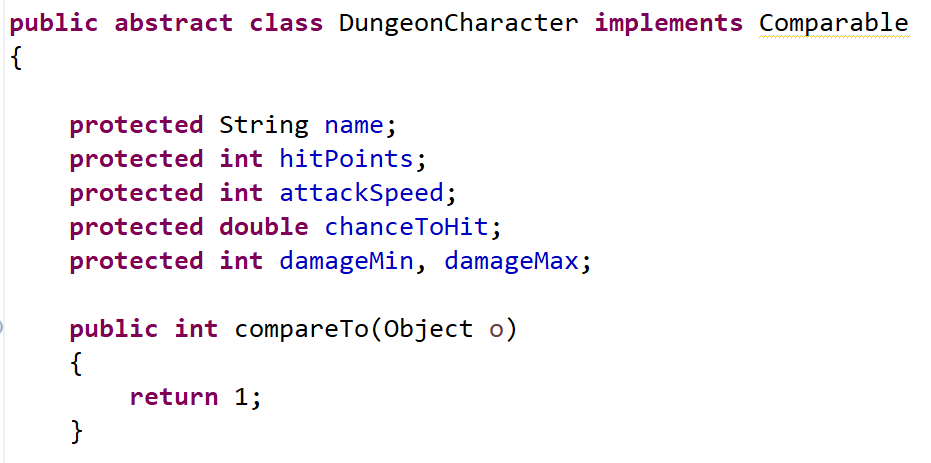


1. Homogenize Inconsistent Formatting: deleted all the excessive white space in the code. One example of this is below in the attack method from DungeonCharacter class. There is unneeded white space in-between opponent.subtractHitPoints call and the print line call. Deleting all unneeded white space cleans up the code and minimizes the length of classes.

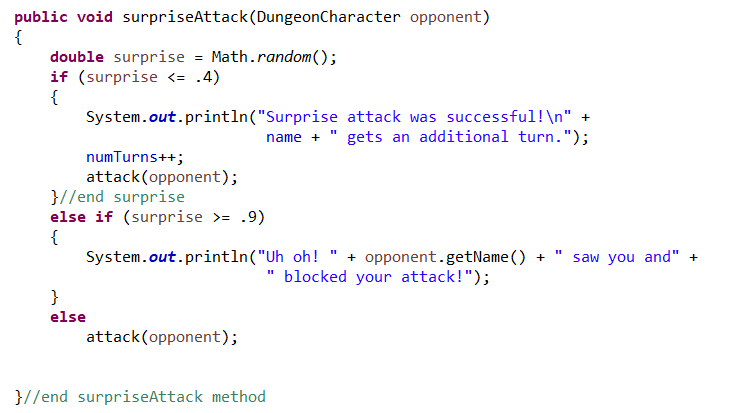
Offending code:



1. Change Visibility: reduced the visibility of relevant fields from protected to private and added the appropriate getters and setters for each field. From example below, all fields were change to private and getters and setters were generated. This ensures no outside source can access the data.  
     
   An example of the offending code:



1. Create Attack Interface: created an interface called SpecialPower to implement the Strategy Pattern. Every Hero’s special attack now has its own class that implements the SpecialPower interface. The below snippet of code is the supriseAttack method from the Thief class. This has been moved its own class that implements SpecialPower.  
     
   Offending code:



1. Restructure Main Method: moved the game setup and battle method into their own class. This vastly simplifies the main method and reduces static calls. The snippet below is from the Dungeon class and shows some of the battle method which has been moved to its own class.  
     
   Offending code:

