**Team 8: CSC-251Group Project: Random Item Generator**

**Members**: Nicholas Pacejka, Anurodh Patrekar, Bryan Scott Promish, Rahul Patel, Dharmikkumar Patel

**Application Description:**

This an application designed to function as a game mechanic that creates random items and weapons (commonly called ‘loot’) for a videogame. In current form it randomly assigns values to an object as stats, and displays the information as though it was being viewed in a menu.

This starts with the ***Weapon*** class and the subclasses that extend it. The Weapon class sets the item’s attributes such as, name, damage, accuracy, etc. It also holds the methods that will be used to randomize the stats; these will be described in greater detail later.

The subclasses for ***Weapon*** contain values that determine the upper and lower limit of the object’s attribute; the range that the random number can be selected from. This is done to prevent both negative numbers and needlessly high values from being assigned. These ranges are different for each subclass.

First, it creates an instance of an item. The subclass assigns it's stats by calling the method ***assignInt***, located in the superclass. The ***assignInt*** method takes 2 integers (basically the lowest and highest points I want that subclass to have) and returns a random number from between them. It looks like this:

public int assignInt(int num1, int num2){ return num1 + (int) (Math.random() \* num2);}

Next it generates a name for each item in a similar way but with a few extra steps. The constructor in the subclass calls its ***assignName*** method. Each subclass has 2 string arrays called titles and prefixes. The ***assignName*** method then passes each array into the ***assignTitle*** and **assignPrefix** methods respectfully: both are located in the super class.

 The ***assignTitle*** and ***assignPrefix*** then uses Math.random, like ***assignInt***, but uses the length of the array as its upper limit. Both methods select one string from their respective arrays and returns them to the ***assignName*** method.

It looks like this:

public String assignTitle(String[] titles){ int index = (int)(Math.random()\*titles.length); return titles[index];}

The ***assignPrefix*** method functions the same as ***assignTitles*** method; just with *titles*replaced with *prefixes*

Once the selected title and prefix are returned to ***assignName***, they are combined into a single string and assigned as the items name. This is for the default constructor with no arguments. Each subclass also has a second constructor that will take arguments and create an object with those stats assigned. Example:

Without arguments:

Pistol myPistol = new Pistol();

With arguments:

With the second one we can implement user input.

Pistol the\_companion = new Pistol("The Companion", 1152, 53, 11.55, 1.5, 14);