Assignment Seven

Student's Name: Tara Aida **Course Name: Computer Science 01100101**

Teacher's Name: Joe Puccio

Create a program that tracks the motion of a ball accelerating toward the ground due to a gravitational force. Allow the user to specify a horizontal (perpendicular to the gravitational force) velocity. Have the program print out the x (horizontal) and y (vertical) position of the ball, as well as the y velocity (this should be originally negative if you use the convential reference frame of the ground being the origin) at regular intervals (resulting to at least 100 values of each output) until the ball hits the ground.

Example of using constructors to create objects from a class:

Many classes you make may call for what's called a constructor method. This method is called as soon as you create an instance of a class (which is computer jargon for "creating an object from a class"). You should only put things in this method that every object that's created from the class should have. So to continue our prettyGirl example:

```java
public class prettyGirl {

      String name = "Tara Aida";
      int prettiness = 10;
      String looksBestIn;
      Scanner scan = new Scanner(System.in);

      public void whatDoesSheLookBestIn(){
       System.out.println("Tell me, what does "+this.name+" look best in?");
       this.looksBestIn=scan.next();
      }
}
```

We know that every prettyGirl should have a name and a prettiness score, but the reality is that not every prettyGirl is going to have a prettiness of 10, and a name of "Tara Aida". So, we create a method that takes in the name and prettiness the user desires the prettyGirl to have, and then set up the prettyGirl's parameters behind the scenes for the user. There's a specific but simple syntax for the headers of constructor methods. And that's detailed below:

```java
public prettyGirl(String name,int prettiness){
```

Assignment Seven

Things to notice about this header is that there is no return type (such as void or int), and the method's name is the same as the class that contains its name. The previous two noticing must be true (I believe) for the constructor method to be called when a new instance of the class is created.[1] The last thing to notice is that this method takes arguments (which are values passed in from the user[2]) that make it easier for the user to assign custom properties to the prettyGirl (we will see why soon).

And by soon, I mean now! Without knowing, you've ben using constructer methods all along:

```
Scanner scan = new Scanner(System.in);
```
[3]

and

```
prettyGirl physicsGirl = new physicsGirl();
```

The highlighted portion of the line above is actually calling a prettyGirl constructor to generate a new prettyGirl object. Every time you make a new class, Eclipse pretty much sets up a default constructor that will create an object with any properties you've set up for it (in our example above, such properties are `name` and `prettiness`). But because we can't allow the user to customize the properties using the default constructor, we want to overide the default constructor with our own (this happens as soon as you create a method with no return type with the same name as the class). Here is the constructor we'll use:

```
public prettyGirl(String name,int prettiness){
this.name= name;
this.prettiness= prettiness;
}
```

All this does is takes in the name and prettiness that the user specified when they created the instance, and then assigns them to their corresponding properties. Now when we create a new prettyGirl we write:

```
prettyGirl physicsGirl = new prettyGirl("Tara Aida",3);
```

In the next assignment, we may talk about `try` and `switch` statements and maybe even superclasses and inheritance.

Send your source code to the following email address for grading:

Joe@pooch.us

Extra credit: Use constructor methods in your code and once the ball meets the ground, allow for it to bounce. Choose some constant for the elasticity of collision/how much energy is lost each time the ball hits the ground.

Notes on the grading method: I will be inputting the printout of your program into grapher to check for errors.

[1] object is created from the class

[2] by user, I mean anyone or any program that messes around with/ the object from the class

[3] Note that this constructor takes in an argument. You've even changed this argument even though you probably didn't know how the computer dealt with what you were feeding into it.