Programming Project #2

CIS 2151 – Prof. John P. Baugh
Oakland Community College - OR

Points: _____ / 125

Objectives

- To use inheritance
- To understand polymorphism

Introduction

For this program, you will use a random number generator to determine what enemy type will be created and stored in an ArrayList. This program could be used as a subcomponent of a computer game.

Instructions and Information

You must create a general abstract **Enemy** class with the following fields:

- weight (an integer)
- height (an integer)

The Enemy class must also have the following methods:

- Getters for both the weight and height
- Setters for both the weight and height
- A constructor that takes the weight and the height
- An abstract method, attack, that (when implemented) will simply print out (using System.out methods) the specific attack of the subclass type

In addition to the Enemy class, you should have four different concrete (non-abstract) classes that extend the Enemy class:

- Goblin
  - Weight should be generated randomly between 5 and 10
  - Height should be generated randomly between 70 and 100
  - The attack method should print "Gurgle!"
- Ghost
  - Weight should be 0
  - Height should be generated randomly between 90 and 150
  - The attack method should print "Boo!"
- Ogre
  - Weight should be generated randomly between 120 and 200
  - Height should be generated randomly between 200 and 300
  - The attack method should print "Ugh!"

- Dragon
    - Weight should be generated randomly between 1000 and 1500
    - Height should be generated randomly between 750 and 2000
    - The attack method should print "Rawr!"

Your program should create an ArrayList capable of holding any kind of Enemy. Then, you should generate 100 different enemies using random number generators (RNGs) to select a particular enemy, and also to set their specifications (e.g., weight and height) when necessary.

If the RNG returns a 1, then create a Goblin and add it to the ArrayList. If the RNG returns a 2, create a Ghost. If it returns a 3, create an Ogre, and if it returns a 4, create a Dragon.

Thus, the number used from the RNG should ultimately fall between 1 and 4 (inclusive) – which can be done using appropriate scaling and shifting of the values returned by the RNG.

Add each of the 100 enemies to the ArrayList, and then loop through the array list and print out the attacks from each of the enemies in the array list.

So output might look something like:

```
Boo!
Gurgle!
Gurgle!
Gurgle!
Rawr!
Ugh!
Rawr!
Rawr!
Ugh!
…
```

Deliverables

To turn in the assignment, please upload a zip file of a folder containing:

- The **.java files** necessary for the program to run.
    - Upload them to the appropriate assignment directory in Assignments on D2L.
- **Screenshots** of your program running
- **Everything should be in a single zip file when submitted**