Homework 1 : CSCI 2120

This semester will be writing Petite Bête (Little Monster, in French), the Cajun version of a more well-known monster fighting game. Generally, with any non-trivial coding project, once you've mapped out the classes and objects you're going to need to model your problem, coding typically begins with those low level classes that are simply "acted upon" (i.e. their methods are invoked elsewhere in the code, but they do not, themselves, invoke methods of any of your other classes). These classes are referred to as "pure servers" – they are acted upon, but do not act upon others…

Your first assignment, as with any other coding project is to write one of the pure servers of Petite Bête – in this case a class from which we can build objects that represent individual Petite Bête objects.

Submission:

Create a new repository on gitlab.cs.uno.edu (see rubric) for these assignments for this course. Add your instructor (and possibly a grader) as a developer to your repository. Create a new directory within your repository. Name this directory HW1. As you write the program, add, commit, and push your files to the remote repository on the GitLab server. When you have finished the assignment, make sure you've uploaded the most up to date version of your files (check the website to see).

Assignment Statement:

This assignment will consist of a class that defines an object that will represent a single PetiteBete. Also write another class, called Main that contains a main method, builds some PetiteBete objects, and uses all the methods of the PetiteBete objects in some way or another, to show that your PetiteBete objects are functional and behave as advertised.

Writing class PetiteBete:

Some of the basic functionality of a PetiteBete is that it has a tolerance (like the concept of health), a speed (which will dictate, later on, which PetiteBete gets to attack first in a given turn), a base attack power (which will be used to determine how much damage a PetiteBete inflicts when attacking), a name, and a level (which is used later to modify the attack power). Assume that the numeric values are counting numbers.

Provide constructors to allow these to all be set upon build, and an overloaded default constructor that sets default values by invoking the other constructor. Also provide setter methods (commands) to allow a PetiteBete's data to be modified, and getter methods (queries) that allow a PetiteBete to be asked about the values of its internal data. Also, provide a pair of methods that looks like the following, allowing one PetiteBete to attack another PetiteBete:

```
// implementation should invoke other's getHit method
// sending it how much health to take away.
public void attack(PetiteBete other) {}


// implementation should subtract amountOfHit from
// the PetiteBete's tolerance.
public void getHit(int amountOfHit) {}
```

Due Date

You need to complete this assignment and have it pushed to the remote repository on the Git server by the date/time specified on Canvas. Within the context of Canvas, when the homework is complete, post the text "Homework 1 final version submitted".

There should be a README.txt file stating exactly how to compile, run, and test your code. In general IF there are bonuses, and you chose to do them, they should be separate (in subdirectories labeled bonus1, etc.) and fully complete implementations beyond the required one (in other words, start working from a fully implemented original copy in the subdirectory to do the bonus).