CECS 174 – Assignment 14

Virtual Pet – Part I

Create a class that represents a type of pet. Come up with whatever type of pet you would like (cat, dog, dragon, Pokémon, anything). Think of different attributes your pet might have. Ones that are required are: Name, Age, Gender, Hunger, Cleanliness, Happiness. Come up with any others that you would like. Think of different actions your pet does. Ones that are required are: Feed, Clean, Play. Come up with several more to increase the functionality of your pet. Be creative, the more functions the better.

After you have made a list of attributes and actions, decide what types and ranges of values each of your attributes will have. For example: Cleanliness might be an integer that ranges from 0 to 10, where 0 is dirty and 10 is squeaky clean.

Then decide what effects each of your actions will have on the attributes (ex. Feeding your pet can make them happier and less hungry, bathing your pet will make them less happy but a lot cleaner), you can create methods for different amounts of each attribute (ex. feeding the pet a snack, vs. feeding it dinner), again, be creative with this.

After you have listed this information, begin writing your pet class. Each of your attributes becomes an instance variable, and each of your actions become a method.

Notes:

1. Make sure that your instance variables are private, and your methods are public, unless you have a good reason for them being otherwise.
2. Create a constructor that sets the name and gender through the parameter list. Once these are set, they should not be changed. Initialize all other variables to a default starting value, set the age to 0.
3. Most of your instance variables won't have a set method since your action methods will be the ones modifying them by either adding or subtracting from their values. However, each of them should have a get method.
4. Make sure that each of the methods that change your instance variables check to make sure that the variables stay within the range of values that you already pre-determined. For example: Cleanliness' value can only be between 0 and 10. If a method is called that increases your pet's cleanliness when it is already at 10, then it stays at 10. It would be easier to create a method to check the levels all at once.
5. Write a method that increases your pet's age by one.
6. Test your code by creating a small testbench file that will test and run your methods to make sure that they are implemented correctly.
7. Write comments describing what each method does.
8. Your class should NOT interact with the user (no input or output). That will be left to the implementation.