CS 1331 Homework 4

Due Tuesday, February 5th , 2013 8:00pm

[Note the change from typical Thursday due dates]

Introduction

This week the homework covers the basics of classes, and gives you some more practice with loops, conditionals, and user input.

3.1 Pet.java

Now that we have learned what a class is, we can start writing our own to represent real world objects in our programs (that is what OOP is all about after all). For this portion of the assignment, you will be writing a class that represents a pet.

Every Pet should have a name, and ways to keep track of its hunger and happiness levels. You can store hunger and happiness as ints that represent the percentage of happiness and hunger the Pet is. So, a Pet that is very very hungry would have 100 hunger, and a Pet that is not happy might have 30 happiness.

You should have a constructor that takes in the name of the Pet, and sets the hunger to 0 and happiness to 100.

In addition to knowing these things about itself, a Pet should be able to do certain things. (What an object knows are its instance variables, and what it can do are its methods.)

You will create methods for the actions of playing with the Pet, feeding the Pet, and a method called live(), that simulates time passing (increasing the Pet's hunger, and decreasing it's happiness). You will also create a toString() for the Pet, so that that we can get a String detailing the relevant information about the current state of the Pet. The last method your pet should have is a getter for the name variable.

Since this is your first experience writing a class on your own, we will help by giving you a pattern to follow that will meet all of the requirements of the Pet class. If you don't understand why we are doing certain things the way we are, please ask someone because in the future it will be up to you to design classes on your own.

Note: Make sure that happiness and hunger are never greater than 100 or less than 0

- Instance Variables (make sure to use proper visibility!)
    - String name
    - int happiness
    - int hunger
    - Random rand (this will be useful in the live method)
- Methods
    - constructor that takes in the name:

      ```
      public Pet(String name){...}
      ```

    - a method that plays with the Pet

      ```
      public void play(){…}
      ```

      this method should increase the Pet's happiness by 10, and print something out to the console indicating that the Pet has been played with

    - a method that feeds the pet

      ```
      public void feed(boolean treat){…}
      ```

      this method takes in a boolean parameter that states whether the pet is being fed a treat or just regular food. If the pet is being fed a treat, its happiness should go up by 10. In both cases its hunger should decrease by 10, and some message should be printed to the console. If the pet is not hungry, it should print out a message and hunger should not be decreased

    - a method that simulates time passing

      ```
      public void live(){…}
      ```

      every time this method is called, the Pet's hunger should have a 60% chance of going up by 10, and the Pet's happiness should have a 40% chance of going down by 10. (Hint: this is where that Random object comes in handy :])

    - a toString method

      ```
      public String toString(){…}
      ```

      this method should return a string that contains a line for the Pet's name, happiness, and hunger. These should be written in an easily readable way

    - a getter for the name variable

      ```
      public String getName(){…}
      ```

      this method should return the Pet's name

3.2 VirtualPetCare.java

We provide this file for you (**DO NOT MODIFY IT**) – all you need to do is make sure that you write your Pet class to the specifications listed above. When you compile and run your program, you should get output that looks something like this:

```
Welcome to Virtual Pet Care! We will give you a pet to play with and care for until
you decide to leave.

Would you like to name your pet?
yes
What would you like to name it?
Pikachu
________________________________
Name: Pikachu
Happiness: 90%
Hunger: 10%

Press p to play with Pikachu
Press f to feed Pikachu
Press t to give Pikachu a treat
Press q to quit or press any other letter to ignore Pikachu
t
Yummmm! More? Happiness: 100 Hunger: 0
________________________________
Name: Pikachu
Happiness: 100%
Hunger: 10%

Press p to play with Pikachu
Press f to feed Pikachu
Press t to give Pikachu a treat
Press q to quit or press any other letter to ignore Pikachu
p
Playtime!! Happiness: 100
________________________________
Name: Pikachu
Happiness: 100%
Hunger: 20%

Press p to play with Pikachu
Press f to feed Pikachu
Press t to give Pikachu a treat
Press q to quit or press any other letter to ignore Pikachu
q
Goodbye!
```

Turn-in Procedure

Turn in the following files to T-Square. When you are ready, make sure that you have actually **submitted** your files, and not just saved them as a draft.

- Pet.java
- VirtualPetCare.java

Note** Always submit .java files - never submit your .class files. And be 100% certain that the files you turn in compile and that the program runs - submissions that do not will receive an automatic 0. Also, make sure that your files are in on time; the real deadline is 8 pm. While you have until 2 am to get it turned in, we will not accept homework past 2 am for any reason. Don't wait until the last minute!

Verify the Success of Your HW Turn-in

Practice "safe submission"! Verify that your HW files were truly submitted correctly, the upload was successful, and that the files compile and run. It is solely your responsibility to turn in your homework and practice this safe submission safeguard.

1. After uploading the files to T-Square you should receive an email from T-Square listing the names of the files that were uploaded and received. If you do not get the confirmation email almost immediately, something is wrong with your HW submission and/or your email. Even receiving the email does not guarantee that you turned in exactly what you intended.
2. After submitting the files to T-Square, return to the Assignment menu option and this homework. It should show the submitted files.
3. Download copies of your submitted files from the T-Square Assignment page placing them in a new folder.
4. Recompile and test those exact files.
5. This helps guard against a few things.
   a. It helps insure that you turn in the correct files.
   b. It helps you realize if you omit a file or files. (If you do discover that you omitted a file, submit all of your files again, not just the missing one.)
   c. Helps find last minute causes of files not compiling and/or running.