Lab Exercise #13

Assignment Overview

This lab exercise provides practice with class inheritance in Python.
Hint: Code Listing 12.11 should be very helpful.

You will work with a partner on this exercise during your lab session. Two people should work at one computer. Occasionally switch the person who is typing. Talk to each other about what you are doing and why so that both of you understand each step.

1. Examine the file named “lab13.py”, which contains a simple test bed for the classes in “pets.py”, and then execute the test bed.

2. Examine the file named “pets.py”, which contains an outline of four classes to support the handling of pets within Python programs.

Please note that class “PetError” is a subclass of class “Exception” and is complete as-is. You will complete the remaining three classes as described below.

Class “Pet” (which is a subclass of class “Object”) should include the following methods:

__init__	Accepts three arguments: **self**, **species** (default None), and **name** (default “”). Stores **species** and **name** as data attributes. Raises PetError, if **species** is not one of the following (case insensitive): 'dog', 'cat', 'horse', 'gerbil', 'hamster', 'ferret'.
__str__	Returns a string which depends on whether the pet is named or not. if named: "Species of: Xxx, named Yyy", where Xxx is the **species** data attribute and Yyy is the **name** data attribute, both in “title-case” (first letter upper case and rest lower case—hint: title()). otherwise: "Species of: Xxx, unnamed", where Xxx is the **species** data attribute in “title-case”.

Class “Dog” (which is a subclass of class “Pet”) should include the following methods:

__init__	Accepts three arguments: **self**, **name** (default “”), and **chases** (default “Cats”). Uses the constructor for class “Pet” to store **species** (“Dog”) and **name** as data attributes. Stores **chases** as a data attribute. Hint: check out how MassParticle calls the Particle __init__ in Code Listing 12.11
__str__	Returns a string which depends on whether the dog is named or not. Hint: call the Pet class’ __str__ (see MassParticle’s __str__ in Code Listing 12.11) if named: "Species of: Dog, named Yyy, chases zzz", where Yyy is the name data attribute (as above) and zzz is the **chases** data attribute. otherwise: "Species of: Dog, unnamed, chases zzz", where zzz is the **chases** data attribute.

Class “Cat” (which is a subclass of class “Pet”) should include the following methods:

__init__	Accepts three arguments: **self**, **name** (default “”), and **hates** (default “Dogs”). Uses the constructor for class “Pet” to store **species** (“Cat”) and **name** as data attributes. Stores **hates** as a data attribute.
__str__	Returns a string which depends on whether the cat is named or not. if named: "Species of: Cat, named Yyy, hates zzz", where Yyy is the **name** data attribute (as above) and zzz is the **hates** data attribute. otherwise: "Species of: Cat, unnamed, hates zzz", where zzz is the **hates** data attribute.

3. Revise the methods within class “Pet” to accomplish the required tasks.

4. Revise the methods within class “Dog” and class “Cat” to accomplish the tasks required tasks.

5. Revise the test bed to demonstrate that all of the methods in those three classes are correct. Be sure to demonstrate the use of default argument values and error conditions.

★ **Demonstrate your completed program to your TA. On-line students should submit the completed file (named “pets.py”) for grading via the Mimir system. (Do *not* submit your lab13.py file.)**