BLG332E
Object Oriented Programming

Practice Session 6

Exercise 9

In this exercise, you will create a hierarchy of animals that is rooted in an abstract class Animal. Several of the animal classes will implement an abstract class called Pet. You will experiment with variations of these animals, their methods, and polymorphism.

Start by changing your working directory to chap08/exercise1 on our computer.

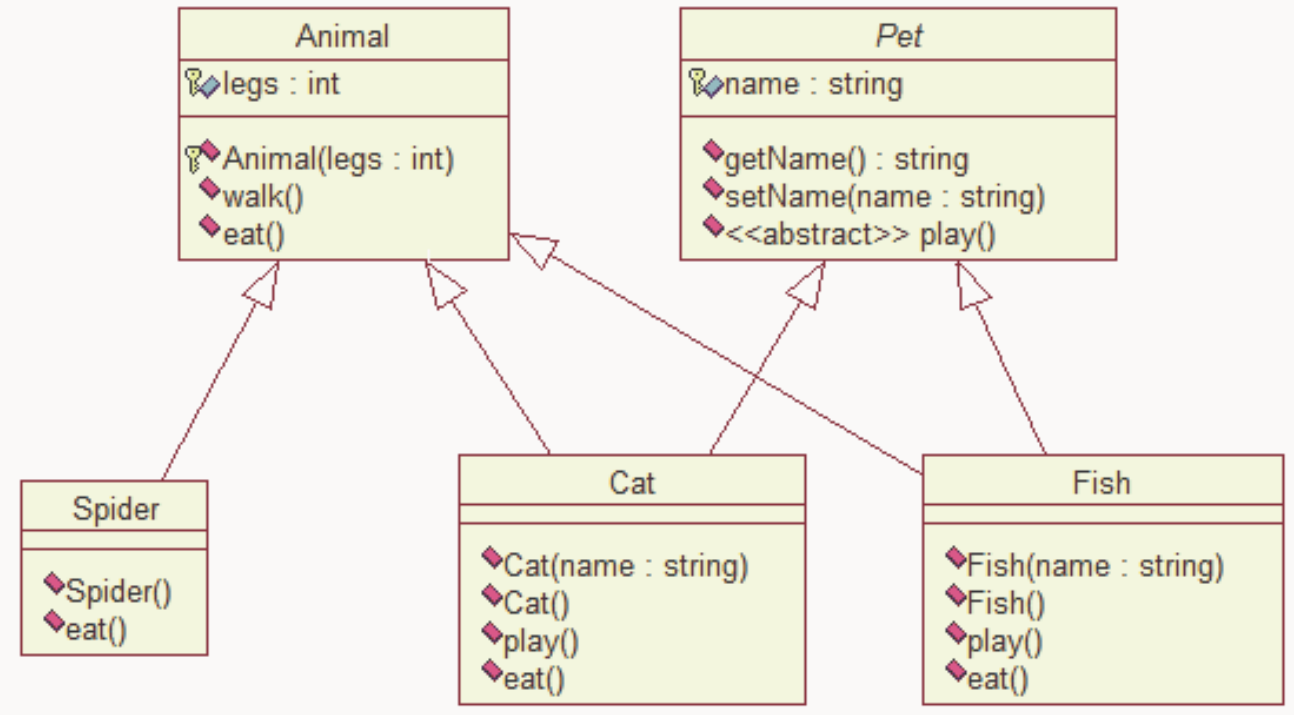
Task 1: Create the `Animal` class, which is the abstract base class of all animals.

a. Declare a protected integer attribute called `legs`, which records the number of legs for this animal.

b. Define a protected constructor that initializes the `legs` attribute.

c. Declare an abstract method `eat`.

d. Declare a concrete method `walk` that prints out something about how animals walks (include the number of legs)



Task 2: Create the `Spider` class.

a. The `Spider` class extends the `Animal` class.

b. Define a default constructor that the constructor of the base class to specify that all spiders have eight legs.

c. Implement the `eat` method.

Task 3: Create the `Pet` class specified by the UML diagram.

Task 4: Create the `Cat` class that inherits `Animal` and `Pet` classes.

a. Define a constructor that takes one string parameter that specifies the cat's name. This constructor must also call the constructor of the base class to specify that all cats have four legs.
b. Define another constructor that takes no parameters.
c. Implement the `play` method.
d. Implement the `eat` method.

Task 5: Create the `Fish` class that inherits `Animal` class. Override the `Animal` methods to specify that fish cannot walk and do not have legs.

Task 6: Create a `TestAnimal` progam. Have the main function create and manipulate instances of the classes you created above.

Start with:

```
Fish *d = new Fish();
Cat  *c = new Cat("Tekir");
Animal *a = new Fish();
Animal *e = new Spider();
Pet *p = new Cat();
```

Experiment by

a. calling the methods in each object,
b. casting objects,
c. using polymorphism, and
d. using scope operator to call base class methods.