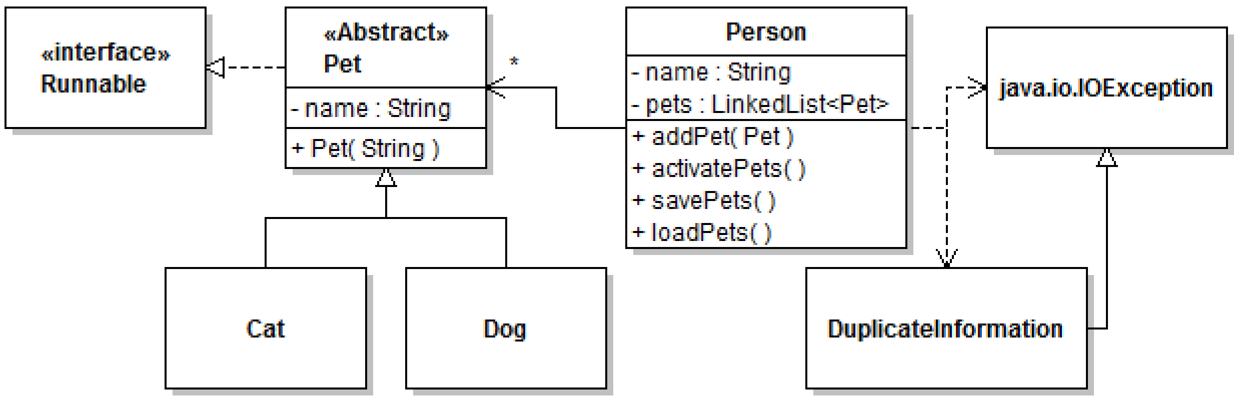
0112562 OBJECT ORIENTED PROG., Section 1, CLASSROOM EXERCISE



Answer these questions according to the UML class schema given above. You may need to extract hidden information from the schema and add necessary code while answering the questions.

Question 1: Write the source code of class Pet.

Question 2: Write the source code of class Cat or Dog. When a pet is passed to a thread, it prints out something such as: Meaow! (Garfield)

Question 3: Write the source code of class DuplicateInformation.

Question 4: Write the source code of class Person. Details of the methods of this class are as follows:

- addPet(Pet): This method adds the given pet among others. The same pet should not be added and two pets with the same should not exist.
- activatePets( ): This method executes the pets in a thread pool.
- savePets( ): This method saves the pets to a file. If the pets are currently active, it must wait for the threads to be completed.
- loadPets( ): This method loads the pets from a file. If there are currently active pets, it must wait for the threads to be completed.

```
public abstract class Pet implements Runnable, java.io.Serializable {
	private static final long serialVersionUID = 1L;
	private String name;
	public Pet(String name) { this.name = name;   }
	public String getName() { return name; }
}
public class Cat extends Pet {
	private static final long serialVersionUID = 1L;
	public Cat(String name) { super(name); }
	public void run() {
		System.out.println("(" + getName() + "): Meaow!");
	}
}
import java.io.*;
public class DuplicateInformation extends IOException {
	public DuplicateInformation( String msg ) {
		super(msg);
	}
}
```

```
import java.util.*;
import java.util.concurrent.*;
import java.io.*;
public class Person {
	private String name;
	private LinkedList<Pet> pets;
	private ExecutorService pool;
	public Person(String name) {
		this.name = name;
		pets = new LinkedList<Pet>();
		pool = Executors.newCachedThreadPool( );
	}
	public Pet searchPet( String petName ) {
		for( Pet pet : pets )
			if( pet.getName().equalsIgnoreCase(petName) )
				return pet;
		return null;
	}
	public boolean searchPet( Pet aPet ) {
		for( Pet pet : pets )
			if( pet == aPet )
				return true;
		return false;
	}
	public void addPet( Pet aPet ) throws DuplicateInformation {
		if( searchPet(aPet) == true )
			throw new DuplicateInformation("Pet already exists!");
		if( searchPet(aPet.getName() ) != null )
			throw new DuplicateInformation("Pet with the name " +
					aPet.getName() + "already exists!");
		pets.add(aPet); //no else is necessary. See Main.
	}
	public void activatePets( ) {
		for( Pet pet : pets ) {
			pool.execute(pet);
		}
		pool.shutdown();
	}
	public String getName() { return name; }
	public void savePets( ) {
		while( !pool.isTerminated() );
		try {
			ObjectOutputStream stream = new ObjectOutputStream(
					new FileOutputStream(name+"sPets.dat") );
			stream.writeObject(pets);
			stream.close();
		}
		catch (IOException e) {
			e.printStackTrace();
		}
	}
	public void loadPets( ) {
		while( !pool.isTerminated() );
		try {
			ObjectInputStream stream = new ObjectInputStream(
					new FileInputStream(name+"sPets.dat") );
			pets = (LinkedList<Pet>) stream.readObject( );
			stream.close();
		}
		catch (IOException e) {
			e.printStackTrace();
		}
		catch (ClassNotFoundException e) {
			e.printStackTrace();
		}
	}
}
```

```java
public class Main {
      public static void main(String[] args) {
            Person jon = new Person("Jon Arbuckle");
            Cat garfield = new Cat("Garfield");
            try {
                  jon.addPet(garfield);
                  jon.addPet(garfield);
            }
            catch (DuplicateInformation e) {
                  e.printStackTrace();
            }
            System.out.println("What happens after exception?");
            jon.activatePets();
      }
}
public class MainV2 {
      public static void main(String[] args) {
            Person jon = new Person("Jon Arbuckle");
            Cat garfield = new Cat("Garfield");
            Dog lassie = new Dog("Lassie");
            try {
                  jon.addPet(garfield);
                  jon.addPet(lassie);
                  jon.activatePets();
                  jon.savePets();
            }
            catch (DuplicateInformation e) {
                  e.printStackTrace();
            }
      }
}
```