Martin Hynes

16390836

Assignment 4

Animal.java

```
//import java.io
import java.io.*;
//Animal class (Serializable, Comparable)
public abstract class Animal implements Serializable, Comparable<Animal>{

	//Instance Variables
	private static final long serialVersionUID = 4L;;
	private String name;
	private String breed;
	private int size;

	//Constructor
	public Animal(){
		this.name = "unassigned";
		this.breed = "unassigned";
		this.size = 0;
	}

	//Overload Constructor
	public Animal(String Name,String Breed,int Size){
		this.name = Name;
		this.breed = Breed;
```

```
        this.size = Size;
}

//compareTo method
public int compareTo(Animal animal){
        int compare = name.compareTo(animal.name);
        return compare;
}

//setter/getter methods
public void setName(String Name){
        this.name = Name;
}
public String getName(){
        return this.name;
}

public void setBreed(String Breed){
        this.breed = Breed;
}
public String getBreed(){
        return this.breed;
}

public void setSize(int Size){
        this.size = Size;
}
```

```
	public int getSize(){
		return this.size;
	}

	//Abstract method sound
	public abstract String sound();

	//toString override
	public String toString(){
		return "Name: "+this.getName()+". Breed: "+this.getBreed()+". Size: "+this.getSize()+".";
	}
}
```

Cat.java

```
public class Cat extends Animal{

	public Cat(String Name,String Breed,int Size){
		super(Name, Breed, Size);
	}

	public String sound(){
		return "Meow!";
	}
}
```

Dog.java

```
public class Dog extends Animal{
```

```
	public Dog(String Name,String Breed,int Size){
		super(Name, Breed, Size);
	}

	public String sound(){
		return "Woof!";
	}
}
```

Tester.java

```
//import java.util and java.io
import java.util.*;
import java.io.*;

//Tester class
public class Tester{

	//main method
	public static void main(String[] args){

		//Create Cat and Dog objects
		Cat cat1 = new Cat("Tiger","Persian",2    );
		Dog dog1 = new Dog("Buddy","Poodle",1);

		//Create LinkedList
		LinkedList<Animal> list = new LinkedList<Animal>();
```

```
//Add Animal objects to list
list.add(cat1);
list.add(dog1);

//Print contents of list
for(Animal animal:list){
	System.out.println(animal.toString());
}

//Print information about sorting
System.out.println();
System.out.println("Sorting by Name.");

//Sort by name
Collections.sort(list);

//Print contents of list
for(Animal animal:list){
	System.out.println(animal.toString());
}

//Print information about sorting
System.out.println();
System.out.println("Sorting by Breed.");

//Sort by Breed
```

```
Collections.sort(list, new BreedCompare());

//Print contents of list
for(Animal animal:list){
	System.out.println(animal.toString());
}

//Print information about sorting
System.out.println();
System.out.println("Sorting by Size.");

//Sort by Size
Collections.sort(list, new SizeCompare());

//Print contents of list
for(Animal animal:list){
	System.out.println(animal.toString());
}

//Serialize application
System.out.println();
System.out.println("Serializing application.");
serialize(list);
list = null;

//Deserialize application
System.out.println();
```

```
		System.out.println("Deserializing Application");
		deserialize(list);
	}

	//Static class BreedCompare
	static class BreedCompare implements Comparator<Animal>{
		//Override compare method to compare by Breed
		public int compare(Animal a1, Animal a2){
			int comp = a1.getBreed().compareTo(a2.getBreed());
			return comp;
		}
	}

	//Static class SizeCompare
	static class SizeCompare implements Comparator<Animal>{
		//Override compare method to compare by Size
		public int compare(Animal a1, Animal a2){
			return a1.getSize() - a2.getSize();
		}
	}

	//serialize method
	public static void serialize(LinkedList<Animal> list){
		//try catch statement
		try{
			//create FileOutputStream, ObjectOutputStream
			FileOutputStream filestream = new FileOutputStream("animal.txt");
```

```
                ObjectOutputStream os = new ObjectOutputStream(filestream);

                //Write list to file
                os.writeObject(list);

                //close stream
                os.close();

        //Catch exceptions
        }catch(Exception e){
                e.printStackTrace();
        }
}
//Suppress warnings
@SuppressWarnings("unchecked")

//deserialize method
public static void deserialize(LinkedList<Animal> list){

        //try catch statement
        try{
                //Create FileInputStream, ObjectInputStream
                FileInputStream filestream = new FileInputStream("animal.txt");
                ObjectInputStream os = new ObjectInputStream(filestream);

                //Read LinkedList from file
                list = (LinkedList<Animal>)os.readObject();
```

```
                    //For every Animal object in LinkedList, print toString of object
                    for(Animal animal:list){
                        System.out.println(animal.toString());
                    }

                    //Close stream
                    os.close();
                //Catch exceptions
                }catch(Exception e){
                    e.printStackTrace();
                }
        }
}
```