# Assignment 0

At a high level, this assignment is designed to review some object-oriented concepts and how they work in java, plus some utilitarian things, like file input and output.

# Description

You will complete a program that reads in a data file of pet information, creates Pet objects, writes a report to file which lists all the pets plus the number of each type of pet and the average age for each type of pet (cats and dogs only). Any errors encountered should be listed on the system monitor or console, but should not otherwise halt the processing.

The main program is already partially implemented. It will compile and run, but not completely or properly. Your job is to complete it.

The portions that are already complete relate to file input and output, because, depending on your prior coursework, you may or may not have been exposed to it. Therefore, those portions of code may be more instructional than review. Also, there are multiple ways in java to read and write files. These are simply two examples.

# Overview

The project zip file contains the following three files:

1. PetList.java, which contains the partially working program.
2. Pet.java, an abstract class which you will need to define.
3. pet\_data.csv, a data file that PetList.java reads and processes.

## Files Overview and requirements

Details on the the three files.

### Pet.java

Pet.java is simply defined as an abstract class at this point. You will need to support two kinds of pets: cats and dogs, both of which have a non-negative age and a non-blank name. Both respond to a “speak” request. Cats say “meow” and dogs say “woof”.

### pet\_data.csv

pet\_data.csv is a sample input file of comma-separated values. When a line of input is properly formed, the first element is either “Cat” or “Dog” (without quotes). The second element is the pet’s name, and the 3rd element is the pet’s age. Malformed input lines should be rejected.

### PetList.java

PetList.java is partially implemented already. It reads in the contents of pet\_data.csv, and stores each line in an array of strings named “petData”. There is also an array of Pets defined, and part of your job will be to convert the input strings in petData into Pet objects, and insert those Pet objects into the array of Pets. To complement the arrays, there is an integer variable named “petCount” that keeps track of the number of array elements (note: these are not meant to be “parallel arrays”, and in the case where, for example, two of ten input lines are invalid for some reason, there will only be 8 pet objects). There are also two constants defined for the input and output files (you may change these to try different input files to test your work, but your submitted work should use the original file names to facilitate grading).

## Methods in PetList.java

There are six methods already defined or declared in PetList.java. Some are completed or partially completed. You may, of course, add more. The methods are ordered in roughly their sequence of execution, and a description of each follows (in the same order):

### main

The main method is fairly straight-forward, and simply invokes the next three methods at this point (you may modify it if you deem that necessary).

### retrievePetData

This method is already implemented for you. It reads the data from the input file and populates the petData array. By way of review (or instruction), it does so with a BufferedReader object, which has an anonymous FileReader composed within it.

### convertPetDataToPetObjects

This method is declared, but you will need to define its behavior. As the name implies, it should process the elements in petData, instantiate Pet objects (assuming the input is valid), and insert them into the pets array.

### writeOutput

As the name implies, this method is designed to write output. It opens a PrintWriter object that has an anonymous FileOutPutStream object composed within it. It writes to the output file. Once the output file is opened, and before it is closed, the method invokes the two following methods to actually create the output.

### writePetObjects

This method simply iterates through the array of pets, and invokes their toString methods, printing each’s returned value.

### writePetStatistics

This method is partly implemented, primarily to indicate what output is expected. There are variables defined for the number of cats and dogs, and for the average age of cats and dogs (four variables total). There are four output lines already defined, so you only need to determine the values (but keep in mind that if a given input file has no cats, for example, that the average age for cats would be undefined).

# Requirements

Complete PetList.java, and design and implement Pet.java (and any subtypes you deem appropriate) to make PetList.java work properly as defined above. Your solution should be object-oriented.

# Deliverables

You should submit a zip file containing only your java source code (PetList.java, Pet.java, etc).