# Project 3

Objectives

* Practice using files and classes in Python
* Learn about sorting algorithms

Introduction

Write a python program that reads a file containing comma-separated data about several pet animals, with each line containing data about one pet. Once read in, the data is stored in a different object class depending on the type of animal. After a data-object is created, it is added to one big pets list and all operations are performed on that list.

Steps:

* Manually create text file pets.csv
* Create python file to define classes (pets.py)
* Create python file to process pet data (petfinder.py

Part 1: Create file pets.csv

Manually create a data file pets.csv with pets, in CSV (comma separated values) format, like the following, but the pets listed in the file can have any names and attributes you want, for example:

* Fish, Nemo, April 2nd, Goldfish, Orange,
* Cat, Shadow, July 3rd, American Bobtail, White,
* Dog, Dug, August 1st, Golden Retriever, Brown,
* Bird, Nigel, January 15th, Pelican, Brown,

Part 2: Create python file pets.py to define classes

Create a second file pets.py, that defines four pet classes (or whatever pet classes you want):

• Dog class with the attributes: name, birthdate (store as string), breed, color.

• Cat class with the attributes: name, birthdate (store as string), breed, color.

• Fish class with the attributes: name, birthdate (store as string), type, color.

• Bird class with the attributes: name, birthdate (store as string), type, color.

• … (whatever other types you want, e.g., Reptile)

Each class must have \_\_init\_\_(), private data attributes, accessors/mutators for all attributes, and \_\_str\_\_() methods.

Part 3: Create a python file petfinder.py to process pet data

In a third file, write a program petfinder.py that:

* Imports pets.py
* Creates an empty list: lstPets
* Defines a main() function

Function fReadData() reads the pets.csv data file:

* Each line of the data file contains information about a single pet with its individual attributes separated by commas.
* For every line in the data file, read in all fields for a single pet and create a new object for that pet depending on its type (the first string in each line). For example, data about a dog would be stored in a dog object.
* After each pet object is created, the function fReadData() adds it to lstPets.
* When the list of pet objects has been created, its length (number of objects) should be the same as the number of pets in the data file.
* When the fReadData() function has read the file and built the lstPets, it returns the list.

Function main():

* Calls function fReadData()
* Stores the returned list in its own variable – perhaps also named lstPets, but not necessarily -- and uses the list for everything that follows
* Calls function fShowMenu()

Function fShowMenu():

* Displays the following menu:

1. Show pet names: displays only the names of all the pets
2. Search for pet: Ask the user for the name of a pet. If the pet is in the list, print all the details for that pet and the index in the list where it was found. If the pet is not in the list, display a message telling the user that the pet is not in the list.
3. Show list: Display the whole list of pets using a loop to print each pet (using \_\_str\_\_ method). Don’t just print the list directly; that will just display a list of unidentified objects.
4. Show pets of a certain type: Asks users what kind of pet (dog/cat/fish/bird /…) to display, then display a list of only pets of that type.
5. Sort the list of pets alphabetically by pet name (use Python function sorted).
6. Exit the program.

* Include input validation wherever you get input from the user
* Each of the menu option functionality must be written in a separate function (except menu option 6), pass pet list object into each of these functions.

Screenshots of all menu and all menu options:

|  |  |
| --- | --- |
| Menu |  |
| 1. Show Pet Names |  |
| 2. Search for Pet |  |
| 3. Show List |  |
| 4. Show pets of certain type |  |
| 5. Sort pets |  |

**Required files to be uploaded to Canvas:**

* Upload the files individually (no zipping, compressing)
* README.txt file - A description of your project with your name and your student ID. Please include any problems you faced, any resources you used, names of friends/tutors you received help from
* Screenshot of the executed code in command line/terminal window of all menu choices (either pasted into a Word document or as an image)
* The main python file named petfinder.py
* The class python file pets.py
* The pets.csv file
* Make sure to complete all tasks in each section
* No global variables, no phantom values
* Your code should contain some meaningful comments
* Your code should be well organized and formatted