CS/SE 2337 – Homework 6 – Operation Kindness

Dr. Doug DeGroot's C++ Classes
Due: Sunday, November 19, 2023 by midnight

How to submit: Upload your entire code in a zip file to the eLearning site. The project must compile on either Codeblocks or MS Visual Studio 2015 (or later). Include your name in the file/project name, like so:

HW6-CS2337-Jimi-Hendrix.zip

Include all project files (*.cpp and *.h files) plus a single txt/doc/pdf file of all your program's reports.

Maximum Number of Points: 100

Objective:

The goal of this homework is to write a small database system for an Animal Shelter. This is a no-kill shelter like the one just west of the Addition Airport. You will accept animal "donations" to the shelter, but mostly only dogs and cats. (But yes, there may be the occasional hamster or other non-dog, non-cat animal donation.) At present, all we need to do is write the skeleton of a database that will track all the animals in the shelter. We will add animals to the database and print reports based on the database. (Clearly there is a lot more functionality we could add to this program, but for now, we will just do the above.)

Approach:

1. Use three classes for this project: Animal, Cat, and Dog. Cat and Dog will be subclasses of Animal.
2. Each animal in the database will have the following attribute, some with and some without values (something like the following):
    a. Type (dog, cat, hamster, etc.)
    b. Name
    c. Age
    d. Weight
    e. Breed
    f. Color
    g. Health
    h. Sound
3. There is a CSV data file that you will need to read and parse to create the animal objects and set their data attributes in the database. An example file input will look something like this:

    AnimalType,Name,Age,Weight,Breed,Color,Health,Sound ← Example header.
    cat,Morris,,3,mixed,yellow,good,meow
    cat,Mittens,1,,Calico,brown and white,good,Mew mew
    cat,Junior,1,2,Tabby,black,needs shots,Meow
    chipmunk,Chippy,,,white and gold,good,sniff sniff
    cat,Charcoal,1,2,Siamese,white and yellow,good

    The data file will be a CSV (comma-separated-values) file saved from an Excel file; thus, it will be a plain text file with commas used as field separators. Note as in the above example that some of the values might be empty, meaning "unknown." Each line of the data file will represent one animal – a cat, a dog, or some other animal. Dogs and Cats will be created using their own Class definitions; other animals will simply be Animals and only animals. The final data file that you are to use will be uploaded to the eLearning site and called

    HW 6 - Operation Kindness.csv.
4. Using static variables, keep track of the number of objects created for each type; thus you will know how many cats have been created, how many dogs, and how many total animals (you don't have to count "other" animals since you can always compute that with

    Nbr "others" created = nbr Animals created – (nbr Dogs created + nbr Cats created)

5. Use a separate static variable within each class to count the objects created for that particular class. Create a method to keep count of the number of animals and to assign a unique number to every newly created animal. Also, keep a private Dog/Cat/etc counter for each dog, cat, and other animal. So, for example, there will be both a *numberOfCats* member and a *myCatNumber* member in the Cat class.
6. Every time you create a Dog, bump both the Dog counter and the Animal counter by having the Dog constructor(s) call the appropriate Animal constructor. Do the same for Cats, etc.
7. Create a .h file and a .cpp file for each of the three (or more) classes.
8. Create and maintain three separate **vectors** of objects – one each for Animal, Cat, and Dog. Store every animal in the Animal vector. The Animal vector will store all three types of objects – animals, cats, and dogs, exhibiting polymorphism. (See Section 15.6 of your text).
9. Store all cats in the Cat vector and all dogs in the Dog vector (that's in addition to storing them in the Animal vector). You won't need any other classes, but if you feel like it, you can create other animal classes.
10. Read the data input file line by line, collecting the animal attributes. Create an object of the appropriate type and set all the attributes to the values you read from the file. If there are missing values, you ***could*** use the default values you create in the base (Animal) class, but try to not always default to the parent class.
11. For each class, create an Introduction method that will have the animal/cat/dog/other speak and then say its name, age, weight, breed, color, health, etc.
12. Once you've read the entire database and created all objects and vectors, create 4 reports:
    a. Report 1: Number of animals created, number of cats created, number of dogs created.
    b. Report 2: An Animal report. Have each animal in the Animal vector introduce itself (by "speaking", giving its name, age, etc.)
    c. Report 3: A Cat report: same as above but using all Cat attributes.
    d. Report 4: A Dog report: same as above but using all Dog attributes.
    e. Other Reports: You can optionally create other reports, perhaps such as one listing all animals that need medical attention, etc.
13. Print the reports to both the console and to a disk file. Submit the disk file with your project submission. Remember, both a disk file output stream and the console are *ostream* objects! So all you need is one set of output routines that you can use for both, as we saw in class.

Annotations for the code:

1. Use information hiding, don't DRY, focus on single-responsibility functions/procedures, etc.
2. Add comments at the very top of your program file to include your name, the name of the program, and your creation date. Then and any relevant notes on how your design works when executed.
3. Add a change log in your comments that list the major changes you make to your logic and when – nothing too terribly detailed, but a list of breadcrumbs that will remind you and others of what you've done as your program becomes more sophisticated and/or nearly complete. ***If you are so inclined, experiment with CodeBlock's auto-changelog creation feature.***
4. Point out (in the comments at the top of your program) any special features or techniques you added using a comment saying something like "// Special Features:"
5. Comment your code effectively, as we have discussed in class several times. Use descriptive variable names everywhere so that the code becomes as self-documenting as possible. Use additional commentary only to improve readability and comprehensibility by other people.
6. You absolutely MUST use consistent indentation and coding styles throughout the program.
7. If the program does not work at all, or works incorrectly, at least 10 points will be deducted.
8. No late submissions will be accepted since this is near the end of the semester.

Codes on eLearning

There are some example Class codes on eLearning that you might want to explore, not only an example of using Animal classes but also all the Pearson codes for the chapters on Classes. I think you might find it valuable to explore these programs.

Some photos of Operation Kindness!

You can find more information and more photos here:

https://www.operationkindness.org/