GitHub link :

Class Documentation:

The real-world object I decided to create a custom class for was a dog. I chose to create a class revolving around a dog because I feel like there are a lot of variations between dogs; thus a lot of different things I could do with a dog class. In this case, I wanted there to be multiple dogs for my demo program so the dog class helped keep track of data variables in relation to a dog. The class variable for the Dog class is “Good Dog”. I chose this since a good dog is something that can be applied to any dog no matter its data variables.The dog class takes in a name, size, and breed as data variables. These data variables are private from encapsulation. The name variable is a variable that gives a name to the dog. The size variable is a variable that gives a size for the dog. The breed variable is a variable that gives a breed for the dog. These three simple variables are the building blocks for being able to create multiple dogs with different names, sizes, and breeds. The following methods I decided to use in my class were set/get methods for each data variable, personality method, and age method. Starting with the set/get methods, I allowed each private data variable like name, to be set using the set method for that variable. The get method will return the set variable. The personality method will take in itself and breed. This method will return a personality depending on the breed. This method uses random.choice which gets personality traits for certain breeds for every new dog so they are not the same. The final method is age, which takes in size and breed. This method will consider the size of the dog and its breed to determine if the dog is a puppy, teen, or adult. The goal for the dog class was to create indefinite combinations for a dog to be used in my demo program.

Demo Program Documentation:

The demo program I decided to create utilizes the dog class to create multiple dogs for the user to choose from in a virtual adoption center. The user is able to input many different options to narrow down the options of dogs at the virtual adoption center to adopt their favorite dog. The program is interactive and gets inputs from the user as choices are made. Each input from the user is set using the set methods for each variable. These set variables are gathered later using get methods for the set variables. There are a lot of options for the user to choose from in terms of how each dog is displayed depending on the user’s inputs.

The program starts off by greeting the user. From there, the user is given the choice from certain dogs available at the adoption center that day. Each day there will be new dogs available, and the user is given the option to keep coming back on different days to see what new dogs are there. The user decides what size they want for their chosen breed. The size and breed will affect the age and personality of the dog. If the user does not like a dog shown to them at the adoption center, they can keep looking for a new dog. Each dog is given a new name and personality to simulate an adoption center as close as possible. At the end, the user is given a certificate that displays all the data variables for the dog, date adopted which is accurate, and picture of the dog.

This program is used in the terminal and requires no additional files or downloads of certain modules. User friendly with simple inputs, and no need to alter code for different choices.