✔ Done

In this tutorial, we will be altering the design of the solution for the tutorial from last week (Week 4). The relevant Java files are included in the Java code base folder. Load them into IntelliJ, and complete the following tasks:

Part 1

1. Analyse the *Dog* class and ask yourself the following questions:
    1. Does a dog really have a max age and a min age?
    2. Can you see a difference between the nature/domain of the fields? Can you see that some, e.g. the microchip number, are unique to the dog, while others, e.g. breed, are not?

2. Analyse the *FindADog* class:
    1. When the *Dog* class is instantiated in *getUserCriteria* (*dogCriteria*), how many of the fields are *null* or blank or 0? How many are actually used? Does this suggest that we're not using *Dog* properly? Perhaps it indicates that *Dog* is doing too many things! It is both representing a 'real' dog, and someone's idea of the dog they want to adopt, i.e. a concept.
    2. Contrast how the *dogCriteria* object is created to how the *Dog* class is instantiated in the method *loadDogs*. What do you notice? Does this indicate that the *Dog* class should only be used to create 'real' *Dog* objects?

3. If you concluded that we need a *DreamDog* class, well done! That is exactly what you are going to create.

4. Create a class named *DreamDog*, then do the following:
    1. Create fields for the generic features (min age, max age, breed, sex and desexed status).
    2. Create a constructor to initialise all these fields.
    3. Create getters for all the fields.

5. Remove the generic features (min age, max age, breed, sex and desexed status) from *Dog*. Remember to remove the associated getters and setters too.
6. Because all dogs have both unique and generic features, create a *DreamDog* field in *Dog*. Use the breed, sex and desexed parameters in the *Dog* constructor to initialise the *DreamDog* field. (HINT: initialise the min and max age parameters to *0*). Create a getter for the *DreamDog* field too.
7. In *FindADog getUserCriteria*, remove the *Dog* object, as well as the min and max age setters. Instead instantiate the *DreamDog* class and return this object.
   HINT: you'll have to change *getUserCriteria*'s return type
8. This will have created an error in *main*, because *findMatch* expects a *Dog* object, so change *dogCriteria*'s type to *DreamDog*, and then go to the *AllDogs* class, and change the parameter type to *DreamDog*.
9. You will notice red lines under *getBreed*, *getSex*, and *isDesexed*. This is because those methods are no longer in *Dog*. However, because *Dog* has a *DreamDog* as a field, we can access those methods through *Dog*'s *getDreamDog* method, i.e. *dog.getDreamDog().getBreed().*
10. You'll also have to change the *getAllBreeds* method in *AllDogs* to use *Dog*'s *getDreamDog* method too.
11. Back in *FindADog main*, notice there are red lines under *potentialMatch.getSex()* etc. Guess why?
12. If you fixed it by changing it to *potentialMatch.getDreamDog.getSex()*, well done!

Part 2

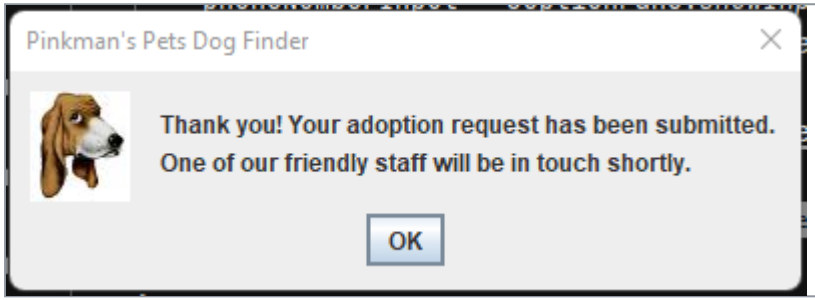
1. To test whether your program works correctly, input the following:
2. ```
   jack russell
   male
   yes
   1
   4
   ```

   You should get the following output:
   ```

Select Ripper, then enter the following:

```
Jesse Morty
0475757575
jmorty@geekmail.com
```

You should get the following output:



Check that the file *Jesse_Morty_adoption_request.txt* exists in your project, and open it. It should contain the following information:

```
Jesse Morty wishes to adopt Pixy (778435462). Their phone number is 0475757575 and their email address is jmorty@geekmail.com
```

Well done! you have successfully used encapsulation and aggregation to improve the design of Pinkman's Pet Finder!

Last modified: Friday, 22 July 2022, 4:24 PM