Practical Lab Session: Week 1

For this practical lab session, please first ensure you have viewed the **Week 1** video sessions **1.1, 1.2** and **1.3**. Within these sessions, you should attempt the development exercises presented as **Challenges** during this lab session. Please ensure you complete the set of challenges prior to the next lab session and upload screenshots of your results to the Progress Management section on Blackboard as directed.

Session 1.2 Challenge: Pet Hierarchy

In a new Java project called **Pets**, define a new class `Pet` with instance variables `name` and `age`. Provide a suitable constructor and accessor/mutator methods for the class.

Now, define two new classes `Cat` and `Dog` as subclasses of `Pet`. Each sub-class has an additional instance variable "`breed`" that can store the particular type of cat (e.g. Persian, Tabby, etc.) or dog (e.g. Spaniel, terrier, etc.).

Each sub-class should contain a method called `speak()` that returns a typical animal noise, plus a description of the animal such as…

> "*Miaow! I am Pixel, a 4 year old tabby*", or
> "*Woof! I am Rex, a 9 year old terrier*".

Provide a further class `PetTest` that implements a `main()` method to test your new class hierarchy.

Note: an example of the output from the program is provided in the video session

Session 1.3 Challenge: Talking Pets

In your **Pets** project, modify the class `PetTest` with functionality as follows.

- The application should create an array of 5 animals which is populated by repeatedly prompting the user whether they want to add a cat or dog to the collection.
- Each animal should be created by selecting a random `age` (within a sensible range) and a random `breed` selected from lists specified as arrays. The animal's `name` should be provided by the user when promoted.
- When all animals have been created, the application should output the number of cats and the number of dogs that have been generated and should prompt the user for the name of a pet.
- When a pet name is provided, the application should search for the pet in the collection and call its `speak()` method if it is found, or provide a suitable message if not. The application should repeat this process until the special pet name "exit" is provided by the user.

Note: an example of the output from the program is provided in the video session.