Submission: See course shell for how and when to submit

You will practice working with properties and using List.

Creating a Pet class

Create the following class

```
Pet
Class
Properties
  + «property setter absent» Name              : string
  + «property setter private» Owner            : string
  + «property setter absent» Age               : int
  + «property setter absent» Description       : string
  + «property setter private» IsHouseTrained   : bool

Methods
  + «constructor» Pet(
        name            : string,
        age             : int,
        description     : string)
  + ToString()                                 : string
  + Train()                                    : void
  + SetOwner(newOwner : string)                : void
```

Description of members:

Fields:

There are no fields.

Properties:

1. The properties are self-explanatory. The getter is public and the setter is mostly absent.

Constructor:

1. `public Pet(string name, int age, string description)` – This constructor takes three arguments and assigns them to the appropriate properties. It also initializes the fields owner to “no one” and **isHousedTrained** to `false`

Methods:

Remember the ToString() method is needed to produce a sensible output on the screen

1. `public override string ToString()` – This method returns a string fully describing this object.

2. `public void SetOwner(string owner)` – This method simply assigns the argument to the appropriate field.
3. `public void Train()` – This method sets the property `IsHouseTrained` to `true`.

Test Harness

In your main method write the code to do the following:

1. Create four objects. You decide on the arguments
2. Create a List to store all the above objects.
3. Use some of the methods on some of the objects.
4. Using a suitable looping statement, display all the objects in the collection.
5. Prompt the user for an owner's name and then display only the pets belonging to a particular person.