Evidence for Implementation and Testing Unit.

Katy Preston
Cohort E21

I.T. 1 - Demonstrate one example of encapsulation that you have written in a program.
The hotel has an ArrayList of Rooms as a property, which has been made private.

```
public class Hotel {
    private ArrayList<Room> allRooms;

    public Hotel(ArrayList<Room> allRooms){
        this.allRooms = allRooms;
    }
```

I.T. 2 - Example the use of Inheritance in a program.
Room is an Abstract Superclass

```
public abstract class Room {
    private int capacity;
    private ArrayList<Guest> guests;

    public Room(int capacity, ArrayList<Guest> guests){
        this.capacity = capacity;
        this.guests = guests;
    }



    public int getCapacity(){
        return this.capacity;
    }
```

Bedroom class inherits from Room class

```
public class Bedroom extends Room {
    private int roomNumber;
    private BedroomType type;
    private double nightlyRate;

    public Bedroom(int capacity, ArrayList guests, int roomNumber, BedroomType type, double nightlyRate){
        super(capacity, guests);
        this.roomNumber = roomNumber;
        this.type = type;
        this.nightlyRate = nightlyRate;
    }
```

Bedroom object inheriting properties from Room

```
guest = new Guest( name: "Antoin");
guests = new ArrayList();
bedroom = new Bedroom( capacity: 2, guests , roomNumber: 1, BedroomType.DOUBLE, nightlyRate: 80.00);
```

Able to get the capacity of the bedroom using the method getCapacity() inherited from Room class (seen in first screenshot)

```
@Test
public void canGetCapacity(){
    assertEquals( expected: 2, bedroom.getCapacity());
}
```

I.T. 3 - Example of Searching.

A function to search for an animal by its ID, followed by the function running in the terminal.

```
def self.find(id)
  sql = "SELECT * FROM animals
  WHERE id = $1"
  values = [id]
  result = SqlRunner.run(sql, values).first
  animal = Animal.new(result)
  return animal
end
```

```
[3] pry(main)> Animal.find(3)
=> #<Animal:0x007faca465b898
 @admission_date="December 2017",
 @adoptable=true,
 @adopted=true,
 @id=3,
 @name="Django",
 @type="Hamster",
 @url=
  "http://absfreepic.com/absolutely_free_photos/small_photos/cute-
hamster-with-yellow-hair-2516x1667_93382.jpg">
```

I.T. 4 - Example of Sorting.

A function to find all animals that are ready for adoption, followed by the function running in terminal.

```
def self.ready_for_adoption()
  sql = "SELECT * FROM animals WHERE adoptable = 't' AND
  adopted = 'f'"
  animal_data = SqlRunner.run(sql)
  animals = map_items(animal_data)
  return animals
end
```

```
[4] pry(main)> Animal.ready_for_adoption
=> [#<Animal:0x007faca4623038
  @admission_date="April 2018",
  @adoptable=true,
  @adopted=false,
  @id=5,
  @name="Beyonce",
  @type="Giant Rabbit",
  @url=
   "http://flemish-giant.com/wp-content/uploads/2015/10/Flemish-Gi
ant-Rabbit-Light-Grey.jpg">,
 #<Animal:0x007faca4622e80
  @admission_date="February 2018",
  @adoptable=true,
  @adopted=false,
  @id=6,
  @name="Pinky",
  @type="Cat",
  @url=
   "https://www.pets4homes.co.uk/images/breeds/23/large/3514efe61d
990b82bbc37bed00eea52a.jpg">,
 #<Animal:0x007faca4622cc8
  @admission_date="May 2018",
  @adoptable=true,
  @adopted=false,
  @id=7,
  @name="Monkey",
  @type="Guinea Pig",
  @url=
   "https://www.petmd.com/sites/default/files/guide-to-guinea-pigs
.jpg">,
 #<Animal:0x007faca4622b10
  @admission_date="April 2018",
  @adoptable=true,
  @adopted=false,
  @id=8,
  @name="Edgar",
  @type="Rat",
  @url=
   "https://thumbor.forbes.com/thumbor/960x0/https%3A%2F%2Fblogs-i
mages.forbes.com%2Fjudystone%2Ffiles%2F2018%2F02%2Fhantavirus-mous
e-cdc.jpg">]
```

I.T. 5 - Example of an array, a function that uses the array and the result

Array of fish in a river

```ruby
fish = ["Salmon", "Goldfish", "Zebra"]
@river = River.new("Chilko", fish)
```

Function using the array

```ruby
def number_of_fish_in_river
  @fish.length
end
```

Testing the function

```ruby
def test_count_fish_in_river
  assert_equal(3, @river.number_of_fish_in_river)
end
```

```
1 runs, 1 assertions, 0 failures, 0 errors, 0 skips
```

I.T. 6 - Example of a hash, a function that uses the hash and the result

hashes.rb

```ruby
def pets_by_type(shopname, type)
  count = []
  for pets in shopname[:pets]
    if (pets[:pet_type] == type)
      count.push(pets)
    end
  end
  return count
end
```

hashes_spec.rb

```ruby
class MyHashTest < MiniTest::Test
  def setup
    @le_shop = {
      pets: [
        {
          name: "Lupe",
          pet_type: :cat,
          breed: "Lupidiam",
          price: 50
        },
        {
          name: "Basilius",
          pet_type: :cat,
          breed: "Bastium",
          price: 500
        },
        {
          name: "Le manche",
          pet_type: :doggo,
          breed: "Good boy",
          price: 1000,
        }
      ]
    }
  end

  def test_all_pets_by_type
    pets = pets_by_type(@le_shop, :doggo)
    assert_equal(1, pets.count)
  end
end
```

Test result:

```
# Running:

.

Finished in 0.001169s, 855.4320 runs/s, 855.4320 assertions/s.

1 runs, 1 assertions, 0 failures, 0 errors, 0 skips
```

I.T. 7 - Example of polymorphism in a program

Piano and Clarinet classes inherit from Instrument class

```
public class Piano extends Instrument{
    private String size;

    public Piano(String type, String colour, double priceBought, double priceSell, String size){
        super(type, colour, priceBought, priceSell);
        this.size = size;
    }
```

```
public class Clarinet extends Instrument{
    private Mouthpiece mouthpiece;
    private Reed reed;

    public Clarinet(String type, String colour, double priceBought, double priceSell, Mouthpiece mouthpiece, Reed reed){
        super(type, colour, priceBought, priceSell);
        this.mouthpiece = mouthpiece;
        this.reed = reed;
    }
```

Instrument superclass implements the interfaces IPlay and ISell

```
public abstract class Instrument implements IPlay, ISell {
    private String type;
    private String colour;
    private double priceBought;
    private double priceSell;

    public Instrument(String type, String colour, double priceBought, double priceSell){
        this.type = type;
        this.colour = colour;
        this.priceBought = priceBought;
        this.priceSell = priceSell;
    }


    public abstract String play();

    public double calculateMarkup(){
        return this.priceSell - this.priceBought;
    }

    public String getType(){
        return this.type;
    }

    public String getColour(){
        return this.colour;
    }

    public double getPriceBought() {
        return this.priceBought;
    }

    public double getPriceSell() {
        return this.priceSell;
    }
}
```

ISell Interface:

```
public interface ISell {
    double calculateMarkup();
    double getPriceBought();
    double getPriceSell();
}
```

The piano can be considered as a piano, an instrument, an iPlay object or an iSell object. In the method below, it is being passed in as an iSell object, and added to the ArrayList called stock, which can contain different items including other instruments, and other music shop products.

```
public void buy(ISell iSell){
    this.funds -= iSell.getPriceBought();
    this.addItemToStock(iSell);
}
```

```
@Test
public void canSellItem(){
    shop.addItemToStock(piano);
    shop.addItemToStock(piano);
    shop.sell(piano);
    assertEquals( expected: 1, shop.getStockCount());
    assertEquals( expected: 1400, shop.getFunds(), delta: 0.1);
}
```