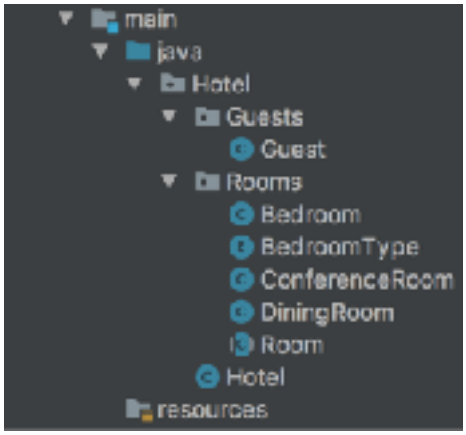
Evidence for Implementation and Testing Unit

Rob Gathergood
E19

I.T.1 - Demonstrate one example of encapsulation that you have written in a program




```java
package Hotel.Guests;

public class Guest {
    private String name;

    public Guest(String name) { this.name = name; }

    public String getName() { return this.name; }
}
```

I.T.2 - Example of the use of inheritance in a program

- **A Class**

```java
package Shop.Instruments;

import Shop.ISell;

public abstract class Instruments implements IPlay, ISell{
    private String model;
    private String brand;
    private Type type;
    private double buyPrice;
    private double sellPrice;

    public Instruments(String model, String brand, Type type, double buyPrice, double sellPrice) {
        this.model = model;
        this.brand = brand;
        this.type = type;
        this.buyPrice = buyPrice;
        this.sellPrice = sellPrice;
    }

    public String getModel() { return model; }

    public String getBrand() { return brand; }

    public Type getType() { return type; }

    public double getBuyPrice() { return buyPrice; }

    public double getSellPrice() { return sellPrice; }

    public String play() { return this.type.getSound(); }

    public double getMarkUp() { return sellPrice - buyPrice; }
}
```

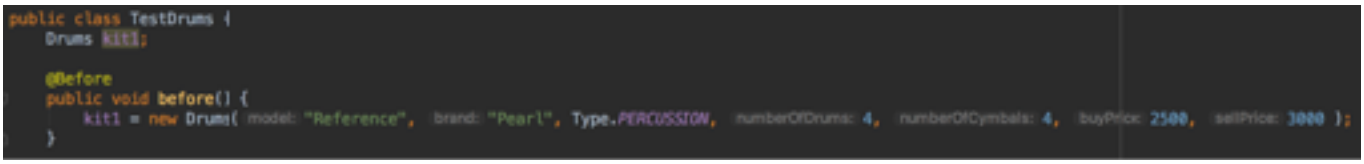
- **A Class that inherits from the previous class**

```java
package Shop.Instruments;

public class Drums extends Instruments {
    private int numberOfDrums;
    private int numberOfCymbals;

    public Drums(String model, String brand, Type type, int numberOfDrums, int numberOfCymbals, double buyPrice, double sellPrice) {
        super(model, brand, type, buyPrice, sellPrice);
        this.numberOfDrums = numberOfDrums;
        this.numberOfCymbals = numberOfCymbals;
    }
```

- **An object in the inherited class**



- **A Method that uses the information inherited from another class**

I.T.3 - Example of searching

```ruby
brands = ["Pearl", "DW", "Gretsch", "Tama", "ludwig"]

def search(brands, brand)
  if brands.include?(brand)
    print "The brand #{brand} exists!"
  end
end

search(brands, "Pearl")
```

```
→  pda git:(master) ✗ ruby week2_3_evidence.rb
The brand Pearl exists!%
```

I.T.4 - Example of sorting

```ruby
brands = ["Pearl", "DW", "Gretsch", "Tama", "ludwig"]

def sort(array)
  print array.sort
end

sort(brands)
```

```
→  pda git:(master) ✗ ruby week2_3_evidence.rb
["DW", "Gretsch", "Pearl", "Tama", "ludwig"]
```

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

```ruby
brands = ["Pearl", "DW", "Gretsch", "Tama", "ludwig"]

def list_brands(brands)
  puts brands
end

def count_brands(brands)
  puts brands.count
end

list_brands(brands)
count_brands(brands)
```

```
→  pda git:(master) ✗ ruby week2_3_evidence.rb
Pearl
DW
Gretsch
Tama
ludwig
5
```

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

```ruby
products = {"Pearl" => "Reference", "DW" => "Collector
Series", "Gretsch" => "USA Custom", "Tama" => "Starclassic",
"Ludwig" => "Legacy"}
```

```ruby
def list_products(products)
  puts products
end

def add_product(products)
  products["Ddrum"] = "Runner"
end

list_products(products)
add_product(products)
list_products(products)
```

```
→  pda git:(master) ✗ ruby week2_3_evidence.rb
{"Pearl"=>"Reference", "DW"=>"Collector Series", "Gretsch"=>"USA Custom", "Tama"
=>"Starclassic", "Ludwig"=>"Legacy"}
{"Pearl"=>"Reference", "DW"=>"Collector Series", "Gretsch"=>"USA Custom", "Tama"
=>"Starclassic", "Ludwig"=>"Legacy", "Ddrum"=>"Runner"}
```

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

```java
package Shop.Instruments;

import Shop.ISell;

public abstract class Instruments implements IPlay, ISell{
    private String model;
    private String brand;
    private Type type;
    private double buyPrice;
    private double sellPrice;

    public Instruments(String model, String brand, Type type, double buyPrice, double sellPrice) {
        this.model = model;
        this.brand = brand;
        this.type = type;
        this.buyPrice = buyPrice;
        this.sellPrice = sellPrice;
    }

    public String getModel() { return model; }

    public String getBrand() { return brand; }

    public Type getType() { return type; }

    public double getBuyPrice() { return buyPrice; }

    public double getSellPrice() { return sellPrice; }

    public String play() { return this.type.getSound(); }

    public double getMarkUp() { return sellPrice - buyPrice; }
}
```

```java
package Shop;

import java.util.ArrayList;

public class Shop {
    private String name;
    private ArrayList<ISell> items;
    private double profit;

    public Shop(String name) {
        this.name = name;
        this.profit = 0;
        this.items = new ArrayList<>();
    }

    public String getName() { return name; }

    public double getProfit() { return profit; }

    public ArrayList<ISell> getItems() { return items; }

    public void addToStock(ISell item) { this.items.add(item); }

    public int getStockCount() { return this.items.size(); }

    public void sellItem(ISell item) { this.items.remove(item); }

    public double potentialProfit() {...}
}
```

```java
package Shop.Instruments;

public class Drums extends Instruments {
    private int numberOfDrums;
    private int numberOfCymbals;

    public Drums(String model, String brand, Type type, int numberOfDrums, int numberOfCymbals, double buyPrice, double sellPrice) {
        super(model, brand, type, buyPrice, sellPrice);
        this.numberOfDrums = numberOfDrums;
        this.numberOfCymbals = numberOfCymbals;
    }

    public int getNumberOfDrums() {
        return numberOfDrums;
    }

    public int getNumberOfCymbals() {
        return numberOfCymbals;
    }
}
```

```java
package Shop.Instruments;

public class Keyboard extends Instruments {
    private String material;

    public Keyboard(String model, String brand, Type type, String material, double buyPrice, double sellPrice) {
        super(model, brand, type, buyPrice, sellPrice);
        this.material = material;
    }

    public String getMaterial() { return material; }
}
```

```java
package Shop;

public interface ISell {
    double getMarkUp();
}
```

```java
package Shop.Instruments;

public interface IPlay {
    String play();
}
```