Implementation and Testing Evidence

Matthew Shield

27th October 2017

I.T. 1 - Encapsulation

```
package com.example.user.blackjack;

import java.util.Random;

public class Player {

    private int points;
    private boolean stuck;
    protected Hand hand;

    public Player() {
        this.points = 0;
        this.stuck = false;
        this.hand = new Hand();
    }

    public int getPoints() {
        return points;
    }
```

I.T. 2 - Inheritance

Super Class:

```java
package animal_management;
import behaviors.*;

abstract class Animal implements Petable, Sellable{
  private String species;
  private int age;
  private String colour;
  private int buyPrice;
  private int sellPrice;

  public Animal(String species, int age, String colour, int buyPrice, int
      this.species = species;
      this.age = age;
      this.colour = colour;
      this.buyPrice = buyPrice;
      this.sellPrice = sellPrice;
  }

  public int calculateMarkup() {
    int result = sellPrice - buyPrice;
    return result;
  }

}
```

Child Class that inherits from the super class:

```java
package animal_management;

public class Dog extends Animal{


  public Dog(String species, int age, String colour, int buyPrice, int
  sellPrice) {
    super(species, age, colour, buyPrice, sellPrice);
  }
  public String pet() {
    return "Woof!";
  }

}
```

Object of Child Class:

```java
import static org.junit.Assert.*;
import org.junit.*;
import behaviors.*;
import animal_management.*;

public class DogTest {

  Dog dog;

  @Before
  public void before() {
    dog = new Dog("Dog", 5, "Black", 60, 80);
  }

  @Test
  public void canGetMarkup() {
    assertEquals(20, dog.calculateMarkup());
  }

}
```

Method using information inherited from another class:

```java
package animal_management;
import behaviors.*;
import java.util.*;


public class Shop{

  private ArrayList<Sellable> stock;

  public Shop(){
    this.stock = new ArrayList<Sellable>();
  }

  public int profitMargin(){
    int profitMargin = 0;

    for(Sellable animal: this.stock){
      profitMargin += animal.calculateMarkUp();
    }
    return profitMargin;
  }

}
```

I.T. 3 – Searching Data

```ruby
def displayOddNumbers(numbers)
    odds = numbers.find_all { |n| n % 2 != 0}
    p odds
end

numbersArray = [1, 2, 3, 4, 5, 5, 2, 2, 8, 7]

displayOddNumbers(numbersArray)
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  1: powershell

C:\Users\Matthew\Documents\codeclan_work\pda_searching_sorting_hashes_arrays\searching> ruby .\ap
p.rb
[1, 3, 5, 5, 7]
```

I.T. 4 – Sorting Data

```ruby
def sortByLength(array)

    result = array.sort {|left, right| left.length <=> right.length}
    p result

end

names = ["Bobby", "Del", "John", "Timothy"]

sortByLength(names)
```

```
C:\Users\Matthew\Documents\codeclan_work\pda_searching_sorting_hashes_arrays\sorting> ruby .\app.
rb
["Del", "John", "Bobby", "Timothy"]
```

I.T. 5 - Array

```ruby
def reverse_lineup(array_of_players)
    return array_of_players.reverse
end

p reverse_lineup(["John", "Del", "Bob", "Tim"])
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  1: powershell

C:\Users\Matthew\Documents\codeclan_work\pda_searching_sorting_hashes_arrays\array> ruby .\app.rb

["Tim", "Bob", "Del", "John"]
```

I.T. 6 - Hash

```ruby
def players()
    hash_of_players = {
        "John" => "Defender",
        "Del" => "Striker",
        "Bob" => "Goalie",
    }
end

def get_positions(player)
    return players()[player]
end

p get_positions("Del")
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  1: powershell

C:\Users\Matthew\Documents\codeclan_work\pda_searching_sorting_hashes_arrays\hash> ruby .\app.rb
"Striker"
```

I.T. 7 – Polymorphism

Sellable.java

```java
package behaviors;

public interface Sellable {
  public double calculateMarkup();
}
```

Dog.java

```java
package animal_management;
import behaviors.*;


public class Dog extends Animal implements Sellable {

  private String description;
  private String breed;
  private double buyPrice;
  private double sellPrice;


  public Dog(String species, int age, String colour, String description,
  String breed, double buyPrice, double sellPrice) {
    super(species, age, colour);
    this.description = description;
    this.breed = breed;
    this.buyPrice = buyPrice;
    this.sellPrice = sellPrice;
  }

  public String getDescription() {
    return this.description;
  }

  public int getBreed() {
    return this.breed;
  }

  public double getBuyPrice() {
    return this.buyPrice;
  }

  public double getSellPrice() {
    return this.sellPrice;
  }

  public double calculateMarkup() {
    double result = sellPrice - buyPrice;
    return result;
  }

  public String pet() {
    return "Woof!";
  }

}
```

Shop.java

```java
package animal_management;
import behaviors.*;
import java.util.*;


public class Shop{

  private ArrayList<Sellable> stock;

  public Shop(){
    this.stock = new ArrayList<Sellable>();
  }

  public int stockCount(){
    return this.stock.size();
  }

  public void addStock(Sellable item){
    this.stock.add(item);
  }

  public Sellable removeStock(){
    if(stockCount() > 0){
      return stock.remove(0);
    }
    return null;
  }

}
```

Petable.java

```java
package behaviors;

public interface Petable {
  public String pet();
}
```

Animal.java

```java
package animal_management;
import behaviors.*;

abstract class Animal implements Petable{
  protected String species;
  protected int age;
  protected String colour;

  public Animal(String species, int age, String colour) {
      this.species = species;
      this.age = age;
      this.colour = colour;
  }

  public String getSpecies() {
    return this.species;
  }


  public String getAge() {
    return this.age;
  }

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

}
```