I.T 1 Screenshot of an example of encapsulation

```
public class Item {

    private String name;
    private double price;
    private int quantity;
    private boolean bogof;

    public Item(  String name, double price, int quantity, boolean bogof){
        this.name = name;
        this.price = price;
        this.quantity = quantity;
        this.bogof = bogof;
    }


    public String getName() {
        return name;
    }

    public double getPrice() {
        return price;
    }

    public int getQuantity() {
        return quantity;
    }

    public int setQuantity(int i) {
        return this.quantity = 2;
    }

    public boolean getBogof() {
        return bogof;
    }

    public boolean setBogof() {
        return this.bogof = true;
```

I.T 5 Demonstrate the use of an array in a program. Take screenshots of:

songs.rb | guests.rb | rooms.rb

```ruby
def initialize(name, playlist, occupancy, till,
  price)
  @name = name
  @playlist = playlist
  @occupancy = []
  @till = till
  @price = price
end

def add_guest(guest)
  @occupancy << guest
end

def check_wallet(value)
  return @wallet >= value
end

def pay_entrance_fee(fee)
  @wallet -= fee
end

# def add_money_to_till(price)
#   @till += price
# end


end
```

songs_specs.rb | guests_specs.rb | rooms_specs.rb

```ruby
class RoomsTest < MiniTest::Test

  def setup
    @song1 = Songs.new("Born to run")
    @song2 = Songs.new("No Surrender")
    @playlist = [@song1, @song2]

    @guest1 = Guests.new("Prince", 15)
    @guest2 = Guests.new("David Bowie", 15)
    @occupancy = []

    @room = Rooms.new("Jungle", @playlist,
      @occupancy, 100, 5)

  end

  def test_room_name
    assert_equal("Jungle", @room.name)
  end

  def test_room_has_playlist
    assert_equal(@playlist, @room.playlist)
  end

  def test_number_of_songs
    assert_equal(2, @room.playlist().length)
  end

  def test_room_has_till
```

specs/rooms_specs.rb 57:6 | 1 LF UTF-8 Ruby master Fetch 1 file 3 updates

```
➜ specs git:(master) ✗ ruby guests_specs.rb
Run options: --seed 52442

# Running:

....

Finished in 0.001280s, 3125.0001 runs/s, 3125.0001 assertions/s.

4 runs, 4 assertions, 0 failures, 0 errors, 0 skips
➜ specs git:(master) ✗ _
```

I.T 6 Demonstrate the use of a hash in a program. Take screenshots of:

friends_spec.rb — ~/codeclan_work/Week_01/d4/Lab/cc-tdd-pairing-ruby-hashes-master

friends.rb

```ruby
def get_name(person)
  return person[:name]
end
```

specs — names-paul@Paul-Stevensons-MacBook-Pro — ..-master/specs...

```
Finished in 0.001988s, 5030.1811 runs/s, 6036.2173 assertions/s.

10 runs, 12 assertions, 0 failures, 0 errors, 0 skips
➜  specs git:(master) ✗ clear

➜  specs git:(master) ✗ ruby friends_spec.rb
Run options: --seed 27659

# Running:

..........

Finished in 0.002147s, 4657.6617 runs/s, 5589.1941 assertions/s.

10 runs, 12 assertions, 0 failures, 0 errors, 0 skips
➜  specs git:(master) ✗
```

friends_spec.rb

```ruby
  @person5 = {
    name: "Dave",
    age: 20,
    monies: 100,
    friends: [],
    favourites: {
      tv_show: "Scrubs",
      things_to_eat: ["spinach"]
    }
  }

  @people = [@person1, @person2, @person3, @person4, @person5]

end

# 1. For a given person, return their name
def test_getting_name
  result = get_name(@person5)
  assert_equal("Dave", result)
end
```

I.T 3 Demonstrate searching data in a program

pet_shop_spec.rb

```ruby
@pet_shop = {
  pets: [
    {
      name: "Sir Percy",
      pet_type: :cat,
      breed: "British Shorthair",
      price: 500
    },
    {
      name: "King Bagdemagus",
      pet_type: :cat,
      breed: "British Shorthair",
      price: 500
    },
    {
      name: "Sir Lancelot",
      pet_type: :dog,
      breed: "Pomsky",
      price: 1000,
    },
    {
      name: "Arthur",
      pet_type: :dog,
      breed: "Husky",
      price: 900,
    },
```

pet_shop.rb

```ruby
def find_pet_by_name(shop, name)
  for animal in shop[:pets]
    return animal if animal[:name] == name
  end
  return nil
end
```

pet_shop_spec.rb

```ruby
def test_find_pet_by_name__returns_pet
  pet = find_pet_by_name(@pet_shop,
    "Arthur")
  assert_equal("Arthur", pet[:name])
end

def test_find_pet_by_name__returns_nil
  pet = find_pet_by_name(@pet_shop,
    "Fred")
  assert_nil(pet)
end
```

specs — names-paul@Paul-Stevensons-MacBook-Pro — ..t_point/specs — -zsh — 7...

```
➜ Week_02 cd weekend_homework
➜ weekend_homework git:(master) ✗ atom .
➜ weekend_homework git:(master) ✗ ..
➜ Week_02 ..
➜ codeclan_work cd Week_01/Homework/weekend_homework_start_point
➜ weekend_homework_start_point git:(master) atom .
➜ weekend_homework_start_point git:(master) cd specs
➜ specs git:(master) ✗ ruby pet_shop_spec.rb
Run options: --seed 28181

# Running:

...................

Finished in 0.003057s, 6215.2437 runs/s, 6215.2437 assertions/s.

19 runs, 19 assertions, 0 failures, 0 errors, 0 skips
➜ specs git:(master) ✗ _
```

I.T 4 Demonstrate sorting data in a program.

pet_shop_spec.rb

```ruby
@pet_shop = {
  pets: [
    {
      name: "Sir Percy",
      pet_type: :cat,
      breed: "British Shorthair",
      price: 500
    },
    {
      name: "King Bagdemagus",
      pet_type: :cat,
      breed: "British Shorthair",
      price: 500
    },
    {
      name: "Sir Lancelot",
      pet_type: :dog,
      breed: "Pomsky",
      price: 1000,
    },
    {
      name: "Arthur",
      pet_type: :dog,
      breed: "Husky",
      price: 900,
    },
```

pet_shop.rb

```ruby
end

# test8&9
def pets_by_breed(shop, breed)
  total = []
  for pet in shop[:pets]
    total.push(pet) if pet [:breed] ==
      breed
  end
  return total
end
```

pet_shop_spec.rb

```ruby
def test_all_pets_by_breed__found
  pets = pets_by_breed(@pet_shop,
    "British Shorthair")
  assert_equal(2, pets.count)
end

def test_all_pets_by_breed__not_found
  pets = pets_by_breed(@pet_shop,
    "Dalmation")
  assert_equal(0, pets.count)
end
```

specs — names-paul@Paul-Stevensons-MacBook-Pro — ..t_point/specs — -zsh — 7...

```
➜  Week_02 cd weekend_homework
➜  weekend_homework git:(master) ✗ atom .
➜  weekend_homework git:(master) ✗ ..
➜  Week_02 ..
➜  codeclan_work cd Week_01/Homework/weekend_homework_start_point
➜  weekend_homework_start_point git:(master) atom .
➜  weekend_homework_start_point git:(master) cd specs
➜  specs git:(master) ✗ ruby pet_shop_spec.rb
Run options: --seed 28181

# Running:

...................

Finished in 0.003057s, 6215.2437 runs/s, 6215.2437 assertions/s.

19 runs, 19 assertions, 0 failures, 0 errors, 0 skips
➜  specs git:(master) ✗ _
```

I.T 7 Demonstrate the use of Polymorphism in a program

```
import Interfaces.ISell;
import Stock.Stock;

import java.util.ArrayList;

public  class Shop {

    private ArrayList<Stock> stockItems;

    public Shop() {
        stockItems = new ArrayList<>();
    }
```

```
package Interfaces;

public interface IPlay {

    String instrumentGoes();
}
```

```java
package Stock;

import Enums.InstrumentType;
import Interfaces.IPlay;

public abstract class Instrument extends Stock implements IPlay {


    private Enum instrumentType;
    private String make;
    private String model;
    private String description;

    public Instrument(double rrp, double buyIn, Enum instrumentType,
                      String make, String model, String description) {
        super(rrp, buyIn);
        this.instrumentType = instrumentType;
        this.make = make;
        this.model = model;
        this.description = description;

    }
```

```java
package Stock;

import Enums.GuitarType;

import static Enums.GuitarType.ACOUSTIC;

public class Guitar extends Instrument{

    protected Enum numberOfStrings;


    public Guitar(double rrp, double buyIn, Enum instrumentType,
                  String make, String model, String description,
                  GuitarType numberOfStrings) {
        super(rrp, buyIn, instrumentType, make, model, description);
        this.numberOfStrings = numberOfStrings;
    }


    public int getNumberOfStrings() {
        return ACOUSTIC.getNumberOfStrings();
    }

    @Override
    public String instrumentGoes() {
        return "Plink Plonk";
    }
```

I.T 2 Take a screenshot of the use of Inheritance in a program.

Kaiju.java | Johnmon.java | JohnmonTest.java

```java
package Kaijus;

public abstract class Kaiju {

    protected String name;
    protected int health;
    protected int attack;

    public Kaiju(String name, int health, int attack){
        this.name = name;
        this.health = health;
        this.attack = attack;
    }

    public String getName(){
```

Kaiju

Johnmon.java

```java
package Kaijus;

public class Johnmon extends Kaiju {

    public Johnmon(String name, int health, int attack) {
        super(name, health, attack);
    }

    @Override
```

JohnmonTest.java

```java

public class JohnmonTest {

    Johnmon johnmon;
    Tank tank;

    @Before
    public void before(){
        johnmon = new Johnmon( name: "Johnmon",  health: 100,  attack: 30);
    }

    @Test
    public void canGetName(){
        assertEquals( expected: "Johnmon", johnmon.getName());
    }
```

Evidence of Implementation and Testing Unit
Paul McPhail Stevenson Cohort E21