Evidence for Implementation and Testing Unit.

Bert Overduin
Cohort E19

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

Item.java

```java
package ShoppingBasket;

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 this.name;
    }

    public double getPrice() {
        return this.price;
    }

    public int getQuantity() {
        return this.quantity;
    }

    public Boolean hasBogof() {
        return this.bogof;
    }

}
```

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

Room.java

```java
package Hotel.Rooms;

import Hotel.Guest;

import java.util.ArrayList;

public abstract class Room {

    private int number;
    private int capacity;
    private ArrayList<Guest> guests;

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

BedRoom.java

```java
package Hotel.Rooms;

import Hotel.Guest;

import java.util.ArrayList;

public class BedRoom extends Room {

    private double rate;
    private Type type;

    public BedRoom(int number, int capacity, double rate, Type type) {
        super(number, capacity);
        this.rate = rate;
        this.type = type;
    }
```

```java
        bedroom = new BedRoom( number: 1,  capacity: 1,  rate: 100.00, Type.SINGLE);
```

```java
    public boolean isVacant() {
        ArrayList<Guest> guests = getGuests();
        return guests.size() == 0;
    }
```

I.T 3 - Example of searching

```ruby
benelux = ['The Netherlands', 'Belgium', 'Luxembourg']

def search_word(list, searched_word)
  for word in list
    return "#{searched_word} found" if word == searched_word
  end
  return "#{searched_word} not found"
end

puts search_word(benelux, "The Netherlands")

puts search_word(benelux, "Italy")
```

```
➜  code git:(master) ✗ ruby searching.rb
The Netherlands found
Italy not found
```

I.T 4 - Example of sorting.

```ruby
benelux = ['The Netherlands', 'Belgium', 'Luxembourg']

def sort_words(unsorted)
  still_unsorted = unsorted
  sorted = []
  while still_unsorted.length > 1
    unsorted = still_unsorted
    still_unsorted = []
    smallest = unsorted.pop
    for word in unsorted
      if word < smallest
        still_unsorted << smallest
        smallest = word
      else
        still_unsorted << word
      end
    end
    sorted << smallest
  end
  sorted << unsorted[0]
end

puts sort_words(benelux)
```

```
➜  code git:(master) ✗ ruby sorting.rb
Belgium
Luxembourg
The Netherlands
```

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

```ruby
benelux = ['The Netherlands', 'Belgium', 'Luxembourg']

def count_characters(list)
  for word in list
    puts "#{word} contains #{word.length()} characters"
  end
end

count_characters(benelux)
```

```
➜  code git:(master) ✗ ruby array.rb
The Netherlands contains 15 characters
Belgium contains 7 characters
Luxembourg contains 10 characters
```

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

```ruby
benelux = [
  {
    name: "The Netherlands",
    population: 17200671,
    capital: "Amsterdam"
  },
  {
    name: "Belgium",
    population: 11358357,
    capital: "Brussels"
  },
  {
    name: "Luxembourg",
    population: 590667,
    capital: "Luxembourg City"
  }
]

def calc_total_population(countries)
  total_population = 0
  for country in countries
    total_population += country[:population]
  end
  return total_population
end

puts calc_total_population(benelux)
```

```
➜  code git:(master) ✗ ruby hash.rb
29149695
```

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

IPlay.java

```java
package Behaviours;

public interface IPlay {

    public String play();

}
```

Cello.java

```java
package Items.Instruments;

import Behaviours.IPlay;

public class Cello extends Instrument implements IPlay {

    public String play() {
        return "The sound of the cello ...";
    }

}
```

Viola.java

```java
package Items.Instruments;

import Behaviours.IPlay;

public class Viola extends Instrument implements IPlay {

    public String play() {
        return "The sound of the viola ...";
    }

}
```