Evidence for Implementation and Testing Unit.

Hamish Hoad

Cohort E-21

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

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

I.T 3 - Example of searching

searching_data_in_a_program.rb

```ruby
# data
chicken_hashes = [
  { name: "Margaret", age: 2, eggs: 0 },
  { name: "Hetty", age: 1, eggs: 2 },
  { name: "Henrietta", age: 3, eggs: 1 },
  { name: "Audrey", age: 2, eggs: 0 },
  { name: "Mabel", age: 5, eggs: 1 },
]
# searching data
def find_chicken_by_name(array, name)
  for chicken in array
    if (chicken[:name] == name)
      return "I found #{name}"
      return chicken
    end
  end
  return "not found"
end
# result
result1 = find_chicken_by_name(chicken_hashes, "Audrey")
result2 = find_chicken_by_name(chicken_hashes, "Hetty")
p result1
p result2
```

hamishhoad@Hamishs-MBP — ..week_01/day_3 — -zsh

```
➜ day_3 ruby searching_data_in_a_program.rb
"I found Audrey"
"I found Hetty"
➜ day_3
```

I.T 4 – Example of sorting

sorting_data.rb

```ruby
# data
chicken_hashes = [
  { name: "Margaret", age: 2, eggs: 0 },
  { name: "Hetty", age: 1, eggs: 2 },
  { name: "Henrietta", age: 3, eggs: 1 },
  { name: "Audrey", age: 2, eggs: 0 },
  { name: "Mabel", age: 5, eggs: 1 },
]
# counter
total_eggs = 0
# sort and print
for chicken in chicken_hashes
  if (chicken[:eggs] > 0)
    p "Great, #{chicken[:name]} has laid eggs!"
  end
  total_eggs += chicken[:eggs]
  chicken[:eggs] = 0
end
p "That makes a total of #{total_eggs} eggs collected."
```

hamishhoad@Hamishs-MBP — ~/codeclan_work — -zsh

```
➜ codeclan_work ruby sorting_data.rb
"Great, Hetty has laid eggs!"
"Great, Henrietta has laid eggs!"
"Great, Mabel has laid eggs!"
"That makes a total of 4 eggs collected."
➜ codeclan_work
```

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

an_array_in_a_program.rb

```ruby
# array
participants = ["Margaret", "Hetty", "Henrietta", "Audrey", "Mabel"]

# for loop function returning participants
for participant in participants
  p "Participant name: " + participant
end

# function returning second list
p "Participant list: " + participants.reduce{ |list, participant| list + ", " + participant}
```

hamishhoad@Hamishs-MBP — ..week_01/day_3 — -zsh

```
➜ day_3 ruby an_array_in_a_program.rb
"Participant name: Margaret"
"Participant name: Hetty"
"Participant name: Henrietta"
"Participant name: Audrey"
"Participant name: Mabel"
"Participant list: Margaret, Hetty, Henrietta, Audrey, Mabel"
➜ day_3
```

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

hash.rb

```ruby
# Hash
my_hash = Hash.new()
  meals = {
  'breakfast' => 'yogurt',
  'lunch' => 'roll',
  'dinner' => 'steak'
}

p 'For breakfast, ' + meals['breakfast']

# nested hash
countries = {
  uk: {
  capital: 'London',
  population: 8_787_892
  },
  germany: {
  capital: 'Berlin',
  population: 3_605_000
  }
}

p 'Londons population is ' + countries[:uk][:population].to_s
p 'Berlins Population is ' + countries[:germany][:population].to_s
```

hamishhoad@Hamishs-MBP — ..week_01/day_3 — -zsh

```
➜ day_3 ruby hash.rb
"For breakfast, yogurt"
"Londons population is 8787892"
"Berlins Population is 3605000"
➜ day_3
```

I.T 7 - Example of polymorphism in a program