

Evidence for Implementation and Testing Unit.

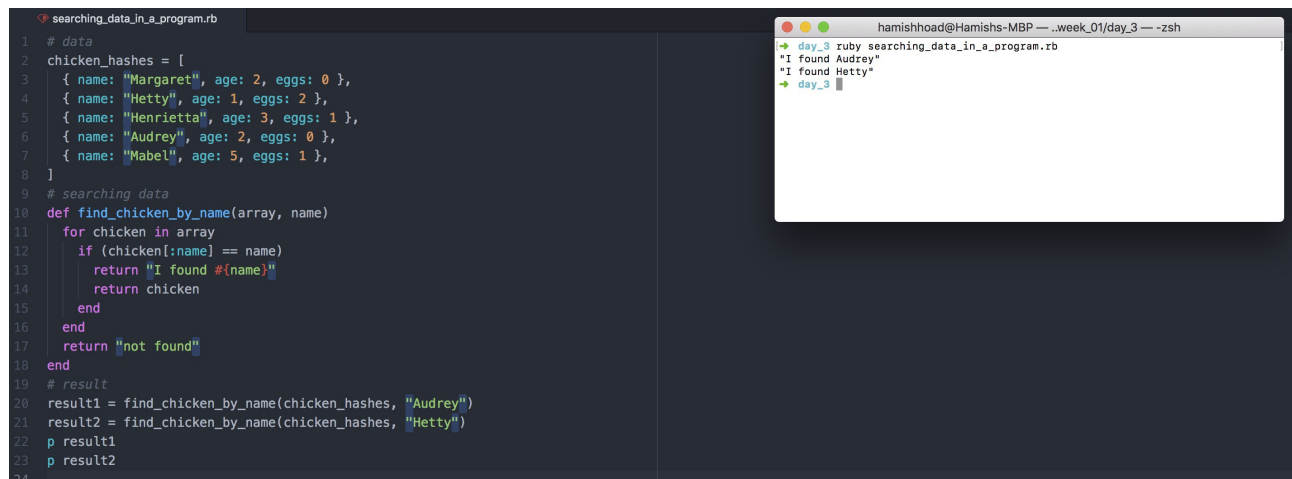
Hamish Hoad

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.

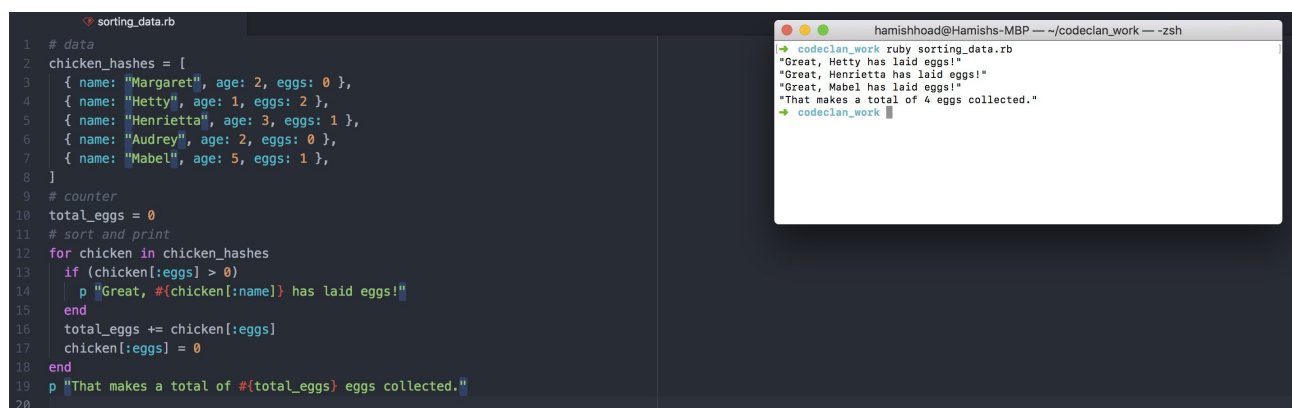
I.T 3 - Example of searching



```
1 # data
2 chicken_hashes = [
3   { name: "Margaret", age: 2, eggs: 0 },
4   { name: "Hetty", age: 1, eggs: 2 },
5   { name: "Henrietta", age: 3, eggs: 1 },
6   { name: "Audrey", age: 2, eggs: 0 },
7   { name: "Mabel", age: 5, eggs: 1 },
8 ]
9 # searching data
10 def find_chicken_by_name(array, name)
11   for chicken in array
12     if (chicken[:name] == name)
13       return "I found #{name}"
14       return chicken
15     end
16   end
17   return "not found"
18 end
19 # result
20 result1 = find_chicken_by_name(chicken_hashes, "Audrey")
21 result2 = find_chicken_by_name(chicken_hashes, "Hetty")
22 p result1
23 p result2
24
```

```
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



```
1 # data
2 chicken_hashes = [
3   { name: "Margaret", age: 2, eggs: 0 },
4   { name: "Hetty", age: 1, eggs: 2 },
5   { name: "Henrietta", age: 3, eggs: 1 },
6   { name: "Audrey", age: 2, eggs: 0 },
7   { name: "Mabel", age: 5, eggs: 1 },
8 ]
9 # counter
10 total_eggs = 0
11 # sort and print
12 for chicken in chicken_hashes
13   if (chicken[:eggs] > 0)
14     p "Great, #{chicken[:name]} has laid eggs!"
15   end
16   total_eggs += chicken[:eggs]
17   chicken[:eggs] = 0
18 end
19 p "That makes a total of #{total_eggs} eggs collected."
20
```

```
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
1 # array
2 participants = ["Margaret", "Hetty", "Henrietta", "Audrey", "Mabel"]
3
4 # for loop function returning participants
5 for participant in participants
6   p "Participant name: " + participant
7 end
8
9 # function returning second list
10 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
1 # Hash
2 my_hash = Hash.new()
3 meals = {
4   'breakfast' => 'yogurt',
5   'lunch' => 'roll',
6   'dinner' => 'steak'
7 }
8
9 p 'For breakfast, ' + meals['breakfast']
10
11 # nested hash
12 countries = {
13   uk: {
14     capital: 'London',
15     population: 8_787_892
16   },
17   germany: {
18     capital: 'Berlin',
19     population: 3_605_000
20   }
21 }
22
23 p 'Londons population is ' + countries[:uk][:population].to_s
24 p 'Berlins Population is ' + countries[:germany][:population].to_s
25
```

```
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