

Evidence for Implementation and Testing Unit

Joe Stafford

Cohort: E21

I.T. 5 - Demonstrate the use of an array in a program

- An array in a program (@occupants, @songlist are both arrays in the Room class)

```
room.rb guest.rb song.rb bar.rb drink.rb
1 class Room
2
3   attr_reader(:name, :occupancy_limit, :entry_fee,
4   * :total_cash)
5
6   def initialize(name, occupancy_limit, entry_fee)
7     @name = name
8     @occupancy_limit = occupancy_limit
9     @entry_fee = entry_fee
10    @total_cash = 0
11    @occupants = []
12    @songlist = []
13  end
```

- A function that uses an array (find_guest_by_name method loops through the @occupants array)

```
room.rb guest.rb song.rb bar.rb drink.rb
41
42 def find_guest_by_name(guest_name)
43   @occupants.each do |guest|
44     if guest.name() == guest_name
45       return guest
46     end
47   end
48   return guest = nil
49 end
50
```

- The result of the function running (Below shows successful tests and the test I used to test the above method)


```
[→ weekend_homework_2 git:(master) ruby specs/room_spec.rb
Run options: --seed 54886

# Running:

.....

Finished in 0.002159s, 7410.8383 runs/s, 8800.3704 assertions/s.

16 runs, 19 assertions, 0 failures, 0 errors, 0 skips
→ weekend_homework_2 git:(master) █
```



The screenshot shows a code editor with a dark theme. At the top, there are five tabs: 'room_spec.rb', 'guest_spec.rb', 'song_spec.rb', 'bar_spec.rb', and 'drink_spec.rb'. The 'room_spec.rb' tab is active. The code is written in Ruby and includes line numbers on the left margin. The code defines two test methods: 'test_can_find_guest_by_name__guest_found' and 'test_can_find_guest_by_name__guest_not_found'. The first method adds a guest named 'Joe Stafford' and asserts that the room's find method returns the same guest. The second method adds a guest named 'Jo Stafford' and asserts that the room's find method returns nil.

```
54   end
55
56   def test_can_find_guest_by_name__guest_found
57     @room.add_guest(@guest)
58     assert_equal(@guest, @room.find_guest_by_name("Joe Stafford"))
59   end
60
61   def test_can_find_guest_by_name__guest_not_found
62     @room.add_guest(@guest)
63     assert_nil(nil, @room.find_guest_by_name("Jo Stafford"))
64   end
65
```

I.T. 6 - Demonstrate the use of a hash in a program

- A hash in a program (a hash of a pet, with several key - value pairs)

```
@new_pet = {  
  name: "Bors the Younger",  
  pet_type: :cat,  
  breed: "Cornish Rex",  
  price: 100  
}
```

- A function that uses a hash (this method adds the new_pet hash to the customers array)

-

```
66  
67 def add_pet_to_customer(customer, new_pet)  
68   customer[:pets].push(new_pet)  
69 end
```

- The result of the function running (Below shows successful tests and the test I used to test the above method)

```
[→ weekend_homework_1 git:(master) ruby spec  
s/pet_shop_spec.rb  
Run options: --seed 45031  
  
# Running:  
  
.....  
  
Finished in 0.002424s, 9075.9074 runs/s, 127  
88.7785 assertions/s.  
  
22 runs, 31 assertions, 0 failures, 0 errors  
, 0 skips
```

```
def test_add_pet_to_customer
  customer = @customers[0]
  add_pet_to_customer(customer, @new_pet)
  assert_equal(1, customer_pet_count(customer))
end
```

I.T. 3 - Demonstrate searching data in a program

- Function that searches data (The below method utilises an SqlRunner to run the SQL statement 'SELECT')

```
62
63   def self.all()
64     sql = "SELECT * FROM customers"
65     customer_data = SqlRunner.run(sql)
66     return
67     • Customer.map_items(customer_data)
68   end
```

- The result of the functioning program (Below shows the array of hashes returned when calling the function above)

```
[[1] pry(main)> Customer.all()
=> [#<Customer:0x00007f8d94a24170
  @funds=10,
  @id=7,
  @name="Joe Stafford">,
#<Customer:0x00007f8d94a240a8
  @funds=5,
  @id=8,
  @name="Euan Bell">,
#<Customer:0x00007f8d931abf80
  @funds=20,
  @id=9,
  @name="Ruairidh Grass">,
#<Customer:0x00007f8d931abe68
  @funds=10,
  @id=10,
  @name="Paul Stevenson">]
[2] pry(main)> █
```

I.T. 4 - Demonstrate sorting data in a program

- Function that sorts data (below function shows sorting data using the id of a customer to select all films that customer is going to see)

```
43  
44 def films()  
45     sql = "SELECT films.*  
46     FROM films  
47     INNER JOIN tickets  
48     ON tickets.film_id = films.id  
49     WHERE customer_id = $1"  
50     values = [@id]  
51     film_data = SqlRunner.run(sql, values)  
52     return Film.map_items(film_data)  
53 end
```

- The result of the function running (this shows an array of hashes showing the films that the object 'customer1' is going to see)

```
[[2] pry(main)> customer1.films()  
=> [#<Film:0x00007f8d9412f498  
    @id=7,  
    @price=2,  
    @title="Dr Strangelove">,  
    #<Film:0x00007f8d9412f3d0  
    @id=8,  
    @price=1,  
    @title="The Ladykillers">]
```