BEWD 13

LESSON 4

3 LEARNING GOALS

ITERATORS - EXPLORING EACH

COLLECTIONS - HASH BASICS

RUBY CORE - COMBINING CONCEPTS

LAB TIME - CAR LOT

GIT TIME

GIT TIME

YOU SHOULD KNOW HOW TO

- CREATE A REMOTE BRANCH
- PULL FROM THE UPSTREAM REPO
- PUSH THE CODE TO YOUR FORKED MASTER BRANCH

CODE CHALLENGE!

LOVE POST

LOVE POST!

1 - CAPTURE THE NAME OF A LOVE INTEREST

2 - CAPTURE RESPONSE FROM USER

3 - PROVIDE ADVICE USING A SWITCH STATEMENT

3A- MANAGE `EDGE CASES` WHEN AN INVALID RESPONSE IS PROVIDED

KEYS TO SUCCESS

- ONE BRICK AT TIME
- DEBUG WITH PRY EVERY TIME
- CODE PROLIFICALLY

LOVE POST - SOLUTION

```ruby
def get_love_interest
  puts "Who do you love? \n"
  love_interest = gets.strip
  capture_love_interest_response(love_interest)
end

def capture_love_interest_response(love_interest)
  puts "Are you thinking of #{love_interest}?\n"
  puts "Answer 'Yes' or 'No' \n"
  user_answer = gets.strip.downcase
  get_valid_answer(user_answer, love_interest)
end

def get_valid_answer(user_answer, love_interest)
  case user_answer
    when "yes"
      puts "Maybe you should call #{love_interest}?\n"
    when "no"
      puts "Ok, maybe call them soon. You love #{love_interest}!"
    else
      puts "Your answer is the not valid \n"
      puts "Please put 'Yes' or 'No'\n "
      capture_love_interest_response(love_interest)
    end
end

get_love_interest
```

LET'S CODE!

LOVE POST

COLLECTION

<hash review>

HASH REVIEW: LEARNING GOALS

1 - WHAT IS A HASH

2 - HOW TO CREATE A HASH

3 - HOW TO USE 5 HASH METHODS

WHAT'S A HASH?

- A HASH IS A COLLECTION OF UNIQUE KEYS & THEIR VALUES.
- A HASH PRIMARILY USES A STRING OR A SYMBOL AS A KEY.

KEY

VALUE

```
cars = {}
cars[:tesla] = { year: 2016, model: "Model X", price: "80000", electric: true }
cars[:ford] = { year: 1967, model: "Mustang", price: "30000", electric: false }
```

KEY / VALUE PAIR

```
cars[:tesla]
  { year: 2016, model: "Model X", price: "80000", electric: true}
```

HASHES

3 WAYS TO CREATE A HASH

```
1 - Via Instantiation
    Hash.new

2 - Using the Literal Hash Constructor (curly brackets)
    car = {}

3 - Use the Literal Hash Constructor with keys
     car = {"name" => "Tesla", "model" => 'Model X', "year" => 2017}
           ************** OR ************
     car = {name: "Tesla", model: 'Model X', year: 2017}
```

HASHES

COMMON METHODS

```
1 - .length
2 - .merge and merge!
3 - .select
4 - .keys, .values
5 - .values, values_at
6 - .has_key? , .has_value?
```

RUBY DOCS FOR THE HASH CLASS

HTTP://RUBY-DOC.ORG/CORE-2.3.0/HASH.HTML

HASHES

DISCOVER NEW METHODS

HTTP://RUBY-DOC.ORG/CORE-2.3.0/HASH.HTML

CORE

<combining concepts>

CORE: LEARNING GOALS

1 - CREATE AN ARRAY OF HASHES

2 - ITERATE USING THE .EACH METHOD

3 - PRACTICE USING `IF` && `UNLESS`

4 - BUILD YOUR VERY OWN CAR_LOT

LAB TIME

<car_lot.rb>

KEYS TO SUCCESS

- ONE BRICK AT TIME
- DEBUG WITH PRY EVERY TIME
- CODE PROLIFICALLY

LET'S CODE!

LAB TIME - CAR_LOT