Ruby CLI Project Foundations

CLI Project Checklist https://goo.gl/forms/3prZGTSze1uf5gf73

Variables and Data Types

1. Create a variable called `name` and set it equal to a string of your first and last name
2. Create a variable called `age` and set it equal to your age
3. Change the value of `age` to be how old you will be in 2020
4. Write a program that greets a user and asks what their name is. It should store their name in a local variable called `name`
5. Next, have you program ask the user their age. Store the age in a local variable called `age`.
6. Finally, print a message to the user saying hello to them using their name, telling them their age and what their age will be in 2020.

Reference:
https://en.wikibooks.org/wiki/Ruby_Programming/Syntax/Variables_and_Constants

Control Flow

1. Create a program that asks the user the name of their hometown. If the hometown has more than 9 characters in it, print “Wow, that’s a long hometown!”. Otherwise, print, “I heard that is a nice place”

Reference:
Learn.co: About Ruby Conditionals

Methods and Arguments

1. Break the program above into separate methods
    a. One called “greet_user” that prints a greeting
    b. One called “age_in_twenty_twenty” which takes in an age and returns how old the person will be in 2020
    c. One called “hometown_is_long?” that takes in a string and returns “true” if the string is longer than 9 characters, false otherwise
    d. One called “respond_to_hometown” that takes in the hometown and returns the correct greeting as described above

Reference:
Learn.co: Procedural Ruby - Methods and Arguments

Arrays

1. Assign a new array to a variable, `lyric`
2. Add the following strings to the array: `"laughter"`, `"it's"`, `"free"`
3. How would you get the first element of the lyric array?
4. What does `lyric.length` return?
5. Replace the last element in the array with the string, `"fun"`
6. How would you add `"a lot of"` to the ***front*** of the array?

Reference:
Learn.co - Array Basics

Hashes

1. Assign a new hash to a variable, `snowy_owl`
2. How would you add the following to the `snowy_owl` hash? `"type"=>"Bird", "bird_type" => "Owl", "diet"=>"Carnivore", "life_span"=>"10 years"`
3. After adding the above values to the `snowy_owl` hash, how would I access the value, `"Bird"`?
4. What does `snowy_owl.keys` return?
5. How would I access the key, `"type"`?

References:
Learn.co - Intro to Hashes
http://ruby-for-beginners.rubymonstas.org/built_in_classes/hashes.html

Classes, Instances, and Object Instantiation (+ some method scope)

1. Create a Baby class
    a. Create a new instance of the Baby class
        i. What is the process of creating a new instance called?
    b. Assign that instance to a local variable
2. Create an instance method writer for a name attribute
3. Create an instance method reader for a name attribute
    a. What is another way to create instance reader and writer methods in Ruby?
4. Create an instance method, "cry", where it "puts's" "Waaaaaa!"
    a. Say if I create a local variable `wah` and assign it the string "Waaa!" - what is the scope of that local variable?
    b. If I interpolate that local variable like so (`puts #{wah}`), what would you expect to happen?
5. Say that you want the cry method to be called when new instances of the Baby class are instantiated. How would you do this using the initialize method?

6. And, wait a second, once a baby is born, do we typically ever change the baby's name? No!
    a. How would you change the code to reflect this reality?

References:
Learn.co: OO Ruby - Classes and Instances
Learn.co: OO Ruby - Class Variables and Methods
Learn.co: OO Ruby - Instance Methods
Learn.co: Procedural Ruby - Method Scope

Classes, Scope, and Self

1. In your text editor, create a Dog class in the same file as the Baby class (This is for illustration. Typically, classes would live in separate files - dog.rb and baby.rb)
2. Create an array and assign it to a class variable, @@all
3. Create attr_reader for a name attribute
4. Create an initialize method and enable it to
    a. Take in a name argument and assign it to @name
    b. Shovel the new instance into the @@all array
5. Go to irb
6. Create a lot of Dog instances
7. Try to read the @@all variable - we can't do it. How would you enable the @@all variable to be exposed?
    a. Write a class method, `self.all`
    b. What does `self` represent here?
8. Do the same for the Baby class
9. Can any of the class variables there be seen by the Dog class?
10. How would you enable the Dog class to "see" the value of @@all in the Baby class?

References:
Learn.co: OO Ruby - Classes and Instances
Learn.co: OO Ruby - Class Variables and Methods
Learn.co: OO Ruby - Self
Learn.co: OO Ruby - Private Methods

Object Relationships

Please review the resources below and be able to show how these concepts are used within your CLI project.

References:
Learn.co: Intro to Object Relationships
Learn.co: Has Many Object
Learn.co: Collaborating Objects
Learn.co: Collaborating Objects Review
Learn.co: Has Many Objects Through

Iteration

Please review the resources below and use iteration within your CLI Project.

References:
Learn.co: Intro to Loops
Learn.co: Iteration and Abstraction
Learn.co: Intro to Ruby Iterators