CIS 658 – Getting Comfy with Ruby!
Winter 2020

Work in pairs during class to complete as many of the problems below as possible. You may use any Ruby interpreter version 2.0 or higher

Due: Jan 16, 2019.

Write the Ruby code necessary to accomplish the following:

1. Print out the ubiquitous "Hello, World" message.

2. For the string "Hello, World," find and print the index of the word "World".

3. Write a loop that prints the string "This is funny monkey #1!" 10 times where the number 1 changes from 1 to 10. Implement this in 3 *different* ways using Ruby (including both "normal" and functional styles).

4. Write a simple game that generates a random number 1 – 1000. Let a player guess the number. If the guess is wrong print out whether the guess is low or high and let the player guess again. Repeat this until the user guesses the number. Print the number of attepts needed to guess the number. Hint: `rand(1000)` will generate a random number random number 0 .. 999. The function `gets` will read a string from the keyboard that can be translated into an integer.

5. Write a method named `odd_even` that takes an array of integers as input and returns an array of symbols as output indicating the odd/even sequence. For example

```
odd_even([8, 6, 7, 5, 3, 0, 9])
```

should return

```
[:even, :even, :odd, :odd, :odd, :even, :odd]
```

6. Given an array of symbol values representing an ensemble, write a method called `tabulate_sections` that produces a hash that maps instrument sections to the number of instruments in that section of the ensemble. For example, the input array:

```
ensemble = [:piano, :clarinet, :oboe, :trumpet, :frenchhorn, :violin, :piano,
  :oboe, :cello]
```

produces the output hash:

```
 {percussion: 2, woodwind: 3, brass: 2, strings: 2}
```

You may assume that your method has access to the `section` hash below. Print an error message if the input refers to an instrument not present in `section`.

```
section = {
   piano: :percussion,
   clarinet: :woodwind,
   oboe: :woodwind,
   trumpet: :brass,
   frenchhorn: :brass,
   violin: :string,
   oboe: :woodwind,
   cello: :string,
   viola: :string,
   timpani: :percussion
}
```

7. Write a `Car` class with the following:

- Instance variables `odometer`, `gas_in_tank`, and `miles_per_gallon.`
- Accessor methods for the instance variables. (Accessors allow users to read, but not modify, the instance variables.)
- A constructor that takes the miles per gallon as a parameter and sets both the odometer and gas in tank to 0.
- A method `add_gas` that takes the number of miles to add to `gas_in_tank`.
- A method `drive` that takes the number of miles to drive as a parameter and adjusts the odometer and gas in tank accordingly. If there isn't enough gas in the tank, only drive the number of miles possible with the amount of gas available. Return `true` if the car was able to drive the distance requested; return `false` otherwise.

Submission instructions:

Combine your individual solutions of Ruby into a single text file with a simple comment indicating which homework problem each fragment of Ruby solves. Have one team member email the file to the instructor. Make sure all team member names are included in the file.

For more practice: https://www.w3resource.com/ruby-exercises/