**Ruby Quest01**

Remember to git add && git commit && git push each exercise!

We will execute your function with our test(s), please DO NOT PROVIDE ANY TEST(S) in your file

For each exercise, you will have to create a folder and in this folder, you will have additional files that contain your work. Folder names are provided at the beginning of each exercise under submit directory and specific file names for each exercise are also provided at the beginning of each exercise under submit file(s).

**My First Script**

* Submit directory: ex00
* Submit file: ["my\_first\_script.rb"]

Write your first script, printing Hello World!. Create a file my\_first\_script.rb. Add a printing function (see tips)

**Example 00 (In Javascript)**

$>node my\_first\_script.js

Hello World!

$>

**Example 01 (In Python)**

$>python my\_first\_script.py

Hello World!

$>

**Example 02 (In Ruby)**

$>ruby my\_first\_script.rb

Hello World!

$>

*Tip* (In Ruby) It will contain puts "Hello World!"

**My First Variable Integer**

* Submit directory: ex01
* Submit file: ["my\_first\_variable\_integer.rb"]

First part of coding is to create variable. Let's get started with an integer variable. What does it mean "integer variable?" In most languages you have "types", a good comparaison is a letter is different from a number. In a computer everything is numbers (0 and 1). But we, as human, interact with letter (and words) to make it usuable there are "convention": a letter is a number and one of this table of convertion is: ASCII (you should google man ascii)

Enought talking!

Replace/Complete the following code. Create a variable with (if needed) the right type. (XX is what you need to replace)

**Function prototype** (ruby)

XX = 34

puts(person\_age)

**Example 00**

Input:

Output: 34

Return Value: nil

**My First Variable Char**

* Submit directory: ex02
* Submit file: ["my\_first\_variable\_char.rb"]

The first part of coding is to create a variable. Let's get started with an integer variable. What does an "integer variable” mean? In most languages, you have "types". A good comparison is that a letter is different from a number. In a computer, everything is numbered (0 and 1). But we, as humans, interact with a letter (and words) to make it useable there is "convention": a letter is a number and one of this table of conversion is: ASCII (you should google man ASCII)

Enough talking!

Replace/Complete the following code. Create a variable with (if needed) the right type. (XX is what you need to replace)

**Function prototype** (ruby)

XX = 'c'

puts(my\_letter)

**Example 00**

Input:

Output: c

Return Value: nil

**My First Variable String**

* Submit directory: ex02
* Submit file: ["my\_first\_variable\_string.rb"]

What is a string?, a word? How a computer is creating a string? It could be defined by "multiple letter", which is translated to multiple "characters." Is it an array of characters? :-)

Replace/Complete the following code. Create a variable with (if needed) the right type. (XX is what you need to replace)

**Function prototype** (ruby)

XX = "Learning is growing"

puts(my\_string)

**Example 00**

Input:

Output: Learning is growing

Return Value: nil

**My Multiple Variables Multiple Type**

* Submit directory: ex03
* Submit file: ["my\_multiple\_variables\_multiple\_type.rb"]

Replace/Complete the following code. Create multiple variables with (if needed) the right type. (XX is what you need to replace)

**Function prototype** (ruby)

XX = 34;

XX = "Luke";

XX = ',';

puts("Hello #{my\_name}#{my\_comma} I'm #{my\_age} years old.")

**Example 00**

Input:

Output: Hello Luke, I'm 34 years old.

Return Value: nil

**My First Incrementation**

* Submit directory: ex04
* Submit file: ["my\_first\_incrementation.rb"]

Incrementation and decrementation depending of the language it's either ++ (--) or += 1 (-= 1).

Replace/Complete the following code. (XX is what you need to replace)

**Function prototype** (ruby)

my\_index = 0

// replace this comment with an increment

puts(my\_index)

// replace this comment with an decrement

// replace this comment with an decrement

puts(my\_index)

// replace this comment with an increment

// replace this comment with an increment

// replace this comment with an increment

puts(my\_index)

**Example 00**

Input:

Output: 1

-1

2

Return Value: nil

**My First If Else**

* Submit directory: ex05
* Submit file: ["my\_first\_if\_else.rb"]

if statement is linked to else are part of the fundamental of coding. Key is to put our effort to design the best condition.

Replace/Complete the following code. (XX is what you need to replace)

**Function prototype** (ruby)

nbr = 10

if (XX)

puts("nbr is greater than 20")

else

puts("nbr is less than 20")

end

**Example 00**

Input:

Output: nbr is less than 20

Return Value: nil