Hello Igor, COMPLETE your daily stand up report :-)

Ruby Quest01

Subject | 1 Solution | Additional Resources (0)

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)`.

Ruby Quest01	My First Script
Submit directory	ex00
Submit file	my_first_script.rb

Control Center

roup formation - (November 9, 2022 10:05pm)

Progress - (November 9, 2022 10:05pm)

ubmitted - (November 9, 2022 10:33pm)

est Review - November 9, 2022 10:33pm

Description

Write your first script and have it print `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!"`

Ruby Quest01	My First Variable Integer
Submit directory	ex01
Submit file	my_first_variable_integer.rb

Description

Finished - November 9, 202 2 10:33pm

Go To DoCode

Access:

READ WRITE

Go To Gitea

Keep Working On This Soluti

Also working on the project




valiboy e_d shoerg as_r jorayev _f kituku_ j

Just finished




qosimo v_ab rasulov _az mustap ha_a ayomid e_s




babatu nd_k sharipo v_u xabibul l_r davron ku_j



In most languages, you have `types`. A good example is that a letter is different from a number.
In a computer, everything is numbers (0 and 1) but we, as humans, interact with letters (and words) to make them usable there are `conventions`. A letter is a number and one of these conversion tables is ASCII. (You should Google man ASCII.)

Enough talking!

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

Function prototype (ruby)

```
XX = 34
puts(person_age)
```

Example 00

```
Input:
Output: 34

Return Value: nil
```

Ruby Quest01	My First Variable Char
Submit directory	ex02
Submit file	my_first_variable_char.rb

Description

Let's continue with a `char variable`. Same as the previous exercise, but this time create a `char variable` containing a letter.

Type: Project
Group Size: 1 Participant
Review system: Test Review (Gandalf)
Difficulty: Initiation
Average duration: 1 Week

Project's Metadata

Project

id: 41

name: ruby-quest01

visible: True

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

Function prototype (ruby)

```ruby
XX = 'c'
puts(my_letter)
```

Example 00

```
Input:
Output: c

Return Value: nil
```

Ruby Quest01	My First Variable String
Submit directory	ex03
Submit file	my_first_variable_string.rb

Description

What is a `string`? Is it a word? How can a computer create a string?
It could be defined by "multiple letters", 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.
(The 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
```

Ruby Quest01	**My Multiple Variables Multiple Type**
Submit directory	ex04
Submit file	my_multiple_variables_multiple_type.r

Description

Replace/Complete the following code. Create multiple variables with (if needed) the right type.
(The 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
```

Ruby Quest01	My First Incrementation
Submit directory	ex05
Submit file	my_first_incrementation.rb

Description

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

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

Function prototype (ruby)

```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
```

Ruby Quest01	My First If Else
Submit directory	ex06
Submit file	my_first_if_else.rb

Description

`If statements` linked to `else statements` are part of the fundamentals of coding. The challenge is to design the best `condition`.

A condition controls which part of your code is executed. `if` contains what to do if the condition is true, and `else` contains what to do if the condition is not met.

An example:

```
let earth_is_flat = false;

if earth_is_flat {
  println!("Science doesn't exist");
} else {
  println!("Phew.");
}
```

Replace/Complete the following code so that we know whether 10 is less than or greater than 20.
(The 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
```