**INTRODUCTION TO RUBY**

**Overview & Sneak Peek**

Ruby is a powerful, flexible programming language you can use in web/Internet development, to process text, to create games, and as part of the popular Ruby on Rails web framework. Ruby is:

* **High-level**, meaning reading and writing Ruby is really easy—it looks a lot like regular English!
* **Interpreted**, meaning you don’t need a compiler to write and run Ruby. You can write it here at Codecademy or even on your own computer (many are shipped with the Ruby interpreter built in—we’ll get to the interpreter later in this lesson).
* **Object-oriented**, meaning it allows users to manipulate data structures called objects in order to build and execute programs. We’ll learn more about objects later, but for now, all you need to know is *everything* in Ruby is an object.
* **Easy to use**. Ruby was designed by Yukihiro Matsumoto (often just called “Matz”) in 1995. Matz set out to design a language that emphasized human needs over those of the computer, which is why Ruby is so easy to pick up.

This course assumes no previous knowledge of Ruby in particular or programming/computer science in general.

**Instructions**

Ready to learn Ruby? Click the Next button to continue!

**script.rb**

# Welcome to Ruby!

**Variables**

One of the most basic concepts in computer programming is the **variable**. You can think of a variable as a word or name that grasps a single value. For example, let’s say you needed the number 25 from our last example, but you’re not going to use it right away. You can set a variable, say my\_num, to grasp the value 25 and hang onto it for later use, like this:

my\_num = 25

Declaring variables in Ruby is easy: you just write out a name like my\_num, use = to assign it a value, and you’re done! If you need to change a variable, no sweat: just type it again and hit = to assign it a new value.

**Instructions**

**1.**

Set the variable my\_num to the value 100, then click the Run button to run your code.

**script.rb**

my\_num = 100

# Write code above this line!

puts my\_num

**Math**

Ruby isn’t limited to simple expressions of assignment like my\_num = 100; it can also do all the math you learned about in school.

There are six arithmetic operators we’re going to focus on:

Addition (+)

Subtraction (-)

Multiplication (\*)

Division (/)

Exponentiation (\*\*)

Modulo (%)

The only ones that probably look weird to you are exponentiation and modulo. Exponentiation raises one number (the base) to the power of the other (the exponent). For example, 2\*\*3 is 8, since 2\*\*3 means “give me 2 \* 2 \* 2“ (2 multiplied together 3 times). 3\*\*2 is 9 (3 \* 3), and so on.

Modulo returns the remainder of division. For example, 25 % 7 would be 4, since 7 goes into 25 three times with 4 left over.

**Instructions**

**1.**

Do a little math practice in the editor. When you’re ready, click Next.

**script.rb**

first\_number = 2

second\_number = 3

result = first\_number + second\_number

puts result

**'puts' and 'print'**

The print command just takes whatever you give it and prints it to the screen. puts (for “put string”) is slightly different: it adds a new (blank) line after the thing you want it to print. You use them like this:

puts "What's up?"  
print "Oxnard Montalvo"

No parentheses or semicolons needed!

**Instructions**

**1.**

In the editor, use at least one print statement and at least one puts statement. You can print out any strings you like! (Make sure to put your strings between quotes, like this: "Hello!".)

**script.rb**

uts "Hello"

print "Andres "

print "Netanyahu"