a beginner's guide to:

RUBY

g.brown

skillstopractice.com

WARNING: This is not meant to be read like a book.

It’s meant to be explored, tinkered with, and ruminated over.

The important ideas covered here will show up again and again, with increased nuance the deeper into the materials you go.

Allow them to slowly and joyfully unfold.

Now let’s get into some code.

-g

0x01 – 0x0F

0x01 – A <RUBY PROGRAM> SAYS "HELLO WORLD"

```ruby
puts "Hello World"
```

```
Hello World
```

We use a METHOD called puts to display a STRING on the CONSOLE using a STREAM called STDOUT ("Standard Output").

METHOD	A useful action that a program can perform when called by its name. (This often also involves passing along a message that the METHOD will make use of in some way to do its job)
STRING	A free-form message which can be almost anything. (i.e. plain English, specially formatted text like a phone number or email address, or even arbitrary data that's machine-readable only.)
CONSOLE	The software tool that lets you run system commands and see their results. This is where you typically run Ruby programs from.
STREAM	Something computer programs can read from and write to that allows them to interact with the outside world. Streams can be used to work with files, to send and receive information over the internet, and also to interact with developer via their CONSOLE.
STDOUT	The "Standard Output" stream. Anything written to it will show up on the developer's CONSOLE.

0x02 – A <RUBY PROGRAM> TELLS YOU YOUR LUCKY NUMBER.

```ruby
puts "Your lucky number is #{rand(1..100)}."
```

```
Your lucky number is 42.
```

Q1 There are two METHODS that are run in this tiny Ruby program, what are they?

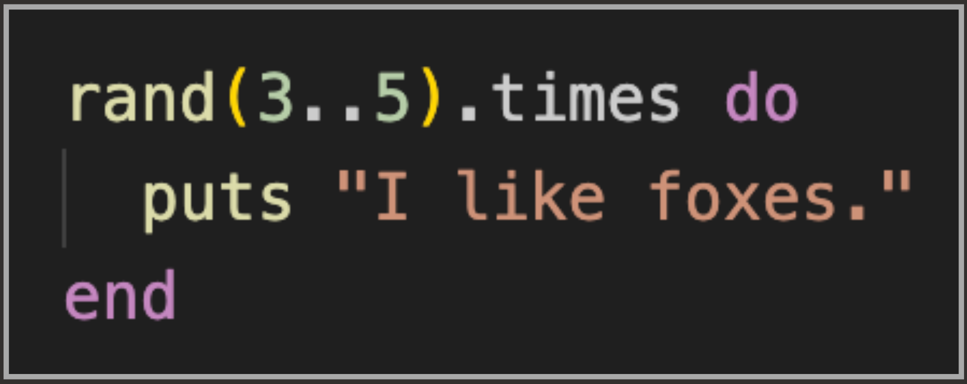
Q2 Which of the two METHODS runs first?

Q3 What is the purpose of the #{ } notation in the STRING shown in this code sample?

Q4 If the #{ } notation was left out and the string was written as “Your lucky number is rand(1..100)” – what would show up on the CONSOLE after this program ran?

RANGE	Represents a span of numbers between a lower and upper limit. E.g. (1..100) means “from 1 to 100”
#{ }	This notation allows you to run some code and then take its end result and stick it into a STRING. The technical term for this handy STRING building technique is Interpolation.

0x03 – A <RUBY PROGRAM> GETS REPETITIVE



```ruby
rand(3..5).times do
  puts "I like foxes."
end
```

Q1 What three METHODS are used in this program?

Q2 What will be displayed on the CONSOLE when this program is run?

Q3 What is the lower limit of the RANGE used in this program?

Q4 What is the upper limit of the RANGE used in this program?

Q5 What is the purpose of the do   end notation?

E1 Update the program so that it ends by saying: "And that's all you need to know for now."

(This should be shown only *once* per run)

E2 Use what you've learned to write a small program that simulates rolling a six sided die ten times.

. operator – Method Call

TODO – EXPLAIN DO/END

(And bound to)

Almost like a method without a name

A “building block” made of code.

0x04 – A <RUBY PROGRAM> VISITS THE FARMER'S MARKET.

```ruby
items_in_stock = ["Apples", "Tomatoes", "Eggs", "Pints of Milk"]

puts "We sell #{items_in_stock.count} types of items"

items_in_stock.each do |item|
  puts "We have #{rand(20..40)} #{item} in stock"
end
```

Q1 What four METHODS are used in this program?

Q2 Of the four, which METHOD is making use of a BLOCK?

Q3 What will be the first line of text shown on the CONSOLE when this program is run?

Q4 What is the purpose of the [, , ,] notation used on the first line of this program?

Q5 How many lines of text in total will be shown on the CONSOLE after this program runs?

Q6 Would the program work the same way, or differently if items_in_stock was renamed to items_for_sale?

Q7 What does item refer to in this program?

```
Variable
```

```
Array
```

```
ruby x.rb | say
```