RENO COLLECTIVE

FULL SPEED

N V

AHEAD

ACADEMY

Intermediate Ruby

Iteration

for…in loop

```
nums = [1, 2, 3, 4, 5]

for i in nums
  puts i
end
```

.each

```ruby
nums = [1, 2, 3, 4, 5]

nums.each do |n|
 puts n
end
```

Hashes

What is a hash?

```
student_ages = {
      "Jack" => 10,
      "Jill" => 12,
      "Bob" => 14
   }
```

Accessing Values

```
student_ages = {
    "Jack" => 10,
    "Jill" => 12,
    "Bob" => 14
}

student_ages["Jack"]
```

Modifying Values

```
student_ages = {
    "Jack" => 10,
    "Jill" => 12,
    "Bob" => 14
}

student_ages["Jack"] = 11
```

Iterating over a Hash

```
student_ages.each do | student, age |
  puts "#{student}: #{age}"
end
```

Accessing keys & values

```
student_ages.keys

student_ages.values
```

Enumerable

The powerful "Enumerable"

each

map

inject

select

count

include?

any?

map

```
integers = [1, 2, 3, 4]
integers.map { |i| i*i }
# => [1, 4, 9, 16]
```

map

```
nato = {:a => "alpha", :b => "bravo"}
nato.map { |key, value| value.upcase }
# => ["ALPHA", "BRAVO"]
```

inject (reduce)

```
nums = [1, 2, 3, 4, 5]
nums.inject(0) do |accum, element|
  accum + element
end
```

inject (reduce)

```
nums = [1, 2, 3, 4, 5]
nums.inject(:+)
```

inject (reduce)

```
nums = [1, 2, 3, 4, 5]
nums.inject(:+)
```

select (find_all)

```
nums = (1..10)
nums.select do |i|
  i % 3 == 0
end
# => [3, 6, 9]
```

count

```
nums = [1, 4, 5, 6, 7]
nums.count # => 5
```

include?

```
nums = [1, 4, 5, 6, 7]


nums.include?(3) # => false
nums.include?(4) # => true
```

any?

```
nums = [2, 3, 5, 7]
nums.any? do |i|
  i % 2 == 0
end
# => true
```

Methods

The splat operator *

```
def add(*numbers)
  numbers.inject do | sum, num |
    sum + num
  end
end
```

The splat operator *

```
def add(num1, num1, num3)
  num1 + num2 + num3
end

numbers_to_add = [1, 2, 3]
puts add(*numbers_to_add)
```

The splat operator *

```
def add_with_message(message, *numbers)
  "#{message} : #{add(*numbers)}"
end

puts add_with_message("The Sum is", 1, 2, 3)
```

Using a hash for options

https://rubymonk.com/learning/books/1-ruby-primer/chapters/19-ruby-methods/lessons/69-new-lesson#209

Exercise with splat and options

https://rubymonk.com/learning/books/1-ruby-primer/chapters/19-ruby-methods/lessons/69-new-lesson#210

Lambdas

What is a Lambda?

Anonymous functions

Making a Lambda function

```
hello = lambda { "world" }
puts hello.call
```

Making a Lambda function

```
greeting = lambda do | planet |
 "hello #{planet}"
end

puts greeting.call("world")

puts greeting.call("mars")
```

Making a Lambda function

`do…end` **`vs`** **`{ }`**

Classes

What is a class?

`A factory that builds new objects`

`Object.new`

Looking up classes

```
puts 1.class
puts "".class
puts [].class

# Fixnum
# String
# Array
```

We can create a Hash with .new

```
recipes = Hash.new

vowels = Hash.new(0)
```

Checking classes

```
puts 1.is_a?(Integer)
puts 1.is_a?(String)
```

```
# true
# false
```

Making our own class

```
class Rectangle
  def initialize(length, breadth)
     @length = length
     @breadth = breadth
   end
  def perimeter
    2 * (@length + @breadth)
  end
end
```

Adding an area method

```
…
  def perimeter
    2 * (@length + @breadth)
  end


  def area
   @length * @breadth
  end
end
```