Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

О Т Ч Е Т

по лабораторной работе № _ 6

Название: Итераторы, Enumerable, Enumerator в Ruby.

Дисциплина: Языки интернет программирования

Студент ИУ6-31Б _____ Т.Е. Старжевский
(Группа) (Подпись, дата) (И.О. Фамилия)

Преподаватель _____ Д. В. Малахов
(Подпись, дата) (И.О. Фамилия)

Москва, 2022

Текст задания:

Часть 1

Решить задачу, организовав итерационный цикл. Вычислить сумму ряда $S = \sum_{k=1}^{\infty} \frac{1}{k(k+1)(k+2)(k+3)}$ с точностью $\xi = 10^{-2}, 10^{-3}$. Точное значение: $\frac{1}{3 \cdot 3!}$. Определить, как изменяется число итераций при изменении точности.

Часть 2

Решить предыдущее задание с помощью Enumerable или Enumerator.

Часть 3

Составить метод minmax, отыскивающую $x \in [a, b]$, для которого функция $y = f(x)$ принимает максимальное и минимальное значение с точностью $0,01$. В основной программе использовать этот метод для математических функций $y = \frac{x-1}{x+2}; x \in [0, 2]$ и $y = sin(\frac{x}{2} - 1), x \in [-1, 1]$.

Реализовать вызов метода двумя способами: в виде передаваемого lambda-выражения и в виде блока.

Часть 1

Текст программ:

main.rb:

```
class Solution
attr_accessor :iterations, :sum

def initialize(eps = 0.1)
@eps = eps
@iterations = 0class Solution
attr_accessor :iterations, :sum

def initialize(eps = 0.1)
@eps = eps
@iterations = 0
@sum = 0
rek
rescue SystemStackError
puts "\t!!!#{SystemStackError}: Stack Overflow"
end

def count(eps)
initialize eps
end
```

```
def rek(k = 1, pred_sum = 0)
@sum += 1.0 / (k * (k + 1) * (k + 2) * (k + 3))
@iterations += 1
if (@sum - pred_sum).abs < @eps
@sum
else
rek k + 1, @sum
end
end
end



@sum = 0
rek
rescue SystemStackError
puts "\t!!!#{SystemStackError}: Stack Overflow"
end

def count(eps)
initialize eps
end

def rek(k = 1, pred_sum = 0)
@sum += 1.0 / (k * (k + 1) * (k + 2) * (k + 3))
@iterations += 1
if (@sum - pred_sum).abs < @eps
@sum
else
rek k + 1, @sum
end
end
end
```

user.rb:

```
require_relative 'main'

print 'Введите точность: '
solve = Solution.new(ep = gets.to_f)
print 'Сколько раз уменьшать точность? '
count = gets.to_i
i = 0

puts "\t================================ OUTPUT BEGIN ================================"
count.times do
solve.count ep
puts "#{i+=1}) With ep = #{ep}\tTotal sum: #{solve.sum}, Iterations: #{solve.iterations}"
ep /= 10
end
puts "\t================================ OUTPUT END ================================"
```

test.rb:

```ruby
require_relative 'main'

RSpec.describe Solution do
it 'Should rescue SystemStackError' do
2.times do
ep = 0.00000000000000001
test_solution = Solution.new ep
5.times do
expect(test_solution.count(ep)).to eq(nil)
end
end
end

it 'Should return correct value' do
2.times do
ep = 0.01
test_solution = Solution.new ep
5.times do
expect(test_solution.count(ep).floor(2)).to eq(0.05)
end
end
end

end
```

Результаты работы и тестов:

```
timofey@timofey-ASUS:~/Projects/BMSTU_WEB/Labs/lab6/part_1$ ruby user.rb
Введите точность: 0.01
Сколько раз уменьшать точность? 4
        ================================ OUTPUT BEGIN ================================
1) With ep = 0.01       Total sum: 0.049999999999999996, Iterations: 2
2) With ep = 0.001      Total sum: 0.05456349206349206, Iterations: 5
3) With ep = 0.0001     Total sum: 0.05530303030303029, Iterations: 9
4) With ep = 1.0e-05    Total sum: 0.055506822612085754, Iterations: 17
        ================================ OUTPUT END ================================
timofey@timofey-ASUS:~/Projects/BMSTU_WEB/Labs/lab6/part_1$ rspec test.rb
        !!!SystemStackError: Stack Overflow
        !!!SystemStackError: Stack Overflow
        !!!SystemStackError: Stack Overflow
        !!!SystemStackError: Stack Overflow
        !!!SystemStackError: Stack Overflow
        !!!SystemStackError: Stack Overflow
        !!!SystemStackError: Stack Overflow
        !!!SystemStackError: Stack Overflow
        !!!SystemStackError: Stack Overflow
        !!!SystemStackError: Stack Overflow
        !!!SystemStackError: Stack Overflow
..

Finished in 0.02456 seconds (files took 0.06505 seconds to load)
2 examples, 0 failures

timofey@timofey-ASUS:~/Projects/BMSTU_WEB/Labs/lab6/part_1$
```

Часть 2
Текст программ:

main2.rb:

```
# frozen_string_literal: true

# Solution 2
class Solution2
def self.go_lazy(value, debug: false)
Enumerator::Lazy.new(0..Float::INFINITY) do |yielder, idx|
puts "Value: #{value} iteration: #{idx}" if debug
value += 1.0 / ((idx + 1) * (idx + 1 + 1) * (idx + 1 + 2) * (idx + 1 + 3))
yielder << value
end
end

def self.show (ep = 0.01, debug: false)
pred_sum = 0
go_lazy(0, debug: debug).take_while do |sum|
flag = (sum - pred_sum).abs
pred_sum = sum
flag > ep
end.inject(:+)
end
end
```

user.rb:

```
require_relative 'main'

print 'Введите точность: '
solve = Solution.new(ep = gets.to_f)

puts "\t================================= OUTPUT BEGIN ep = #{ep}
================================="
solve.each do |sum|
puts "Sum: #{sum}"
end
puts "\t================================= Output ep = 0.01 ================================="
solve.ep = 0.01

solve.each_with_index do |sum, index|
puts "Sum: #{sum} iteration: #{index + 1} "
end
puts "\t================================= Output ep = 0.001 ================================="
solve.ep = 0.001

solve.each_with_index do |sum, index|
puts "Sum: #{sum} iteration: #{index + 1} "
end
puts "\t================================= OUTPUT END ================================="
```

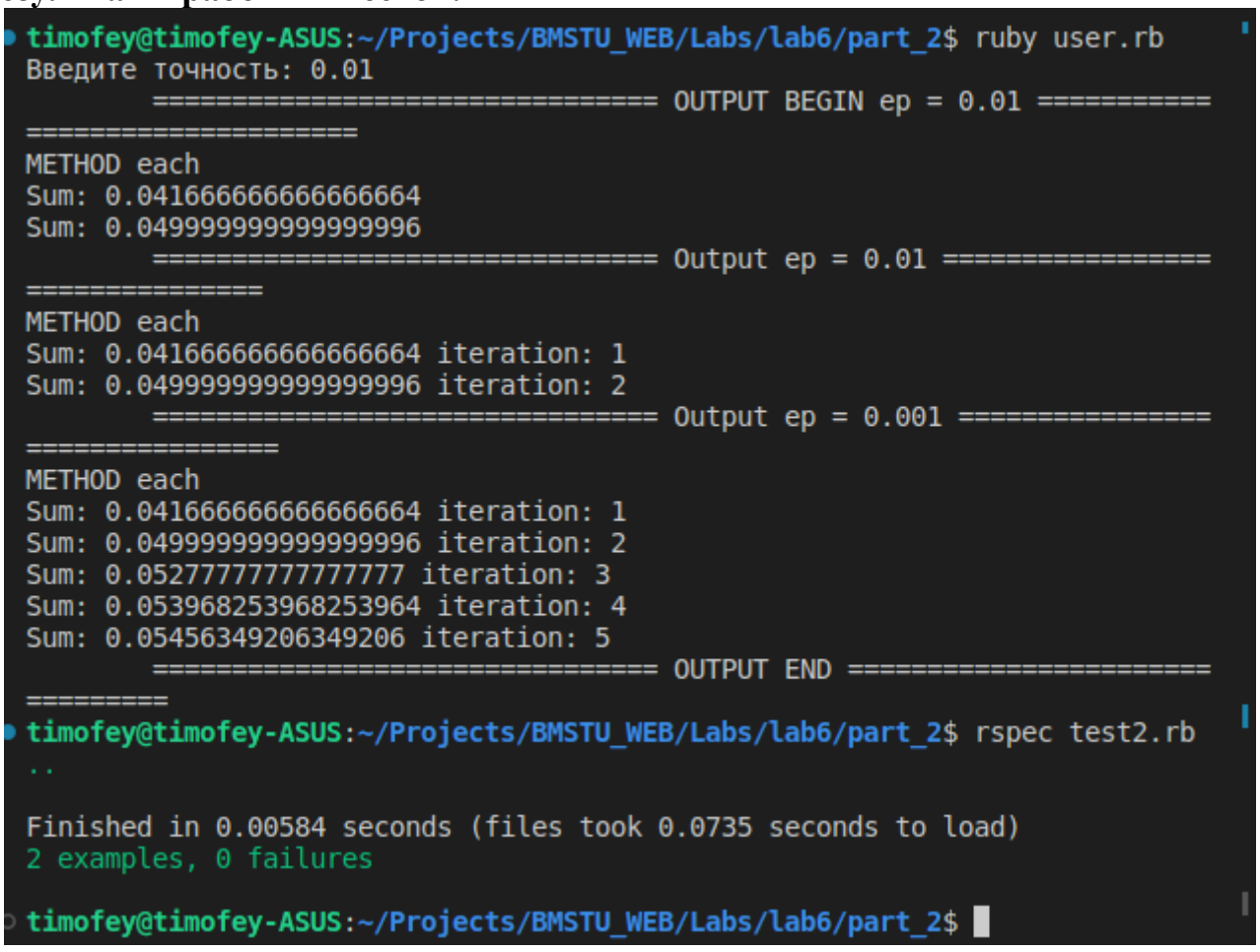
test2.rb:

```
require_relative 'main2'
```

```
RSpec.describe Solution2 do
describe '#show' do
it 'should return correct results' do
results = [0.041, 0.198, 0.418, 0.861, 1.639]
ep = 0.1
results.each do |result|
ep /= 10
expect(Solution2.show(ep).floor(3)).to eq result
end
end

it 'should return nil if zero' do
expect(Solution2.show(1)).to be_nil
end
end
end
```

Результаты работы и тестов:



Часть 3

Текст программ:

main.rb:

```
class Solution
include Enumerable

def initialize(a = 0, b = 5, step = 0.01)
```

```
@a = a
@b = b
@step = step
end

def y(x)
(x**2)
end

def l_minmax(func, a = @a, b = @b)
arr = []
while a <= b
arr.push func.call(a)
a += @step
end
arr.minmax
end

def my_minmax(a = @a, b = @b)
arr = []
while a <= b
arr.push yield a
a += @step
end
arr.minmax
end

def each
x = @a
while x <= @b
yield y(x)
x += @step
end
end
end

class Y1 < Solution
def y(x)
(x - 1) / (x + 2)
end
end

class Y2 < Solution
def y(x)
Math.sin(x / 2 - 1)
end
end
```

```
def make_correct_with_count(strings = @strings)
```

```
strings.map! do |string|
selected = string.split.select do |el|
@counter += 1 unless el.match(@reg_exp)
el.match(@reg_exp)
end
selected.map! do |word|
@counter += 1 if word.match(/^\d/)
word.sub(/^\d/, '_')
end
selected.join(' ')
end

strings.reject(&:empty?)
end

def make_correct(strings = @strings)
strings.map! do |string|
string.split.select { |el| el.match(@reg_exp) }.map! { |word| word.sub(/^\d/, '_') }.join(' ')
end
strings.reject(&:empty?)
end
end
```

user.rb:

```
require_relative 'main'

y1 = Y1.new 0, 2
y2 = Y2.new(-1, 1)

puts "\t================================ OUTPUT BEGIN ================================"
puts '============= MINMAX for (x - 1)/(x + 2) ============'
puts y1.minmax
puts '============= MINMAX for sin(x / 2 - 1) ============'
puts y2.minmax
puts "\t================================ OUTPUT with block & proc & lambda
================================"

puts '============= MINMAX block for (x - 1)/(x + 2) ============'
puts y1.my_minmax { |x| ((x - 1) / (x + 2)) }
puts '=============MINMAX Proc for sin(x / 2 - 1) ============'
puts y2.my_minmax(&proc { |x| Math.sin(x / 2 - 1) })
puts '============ MINMAX Lambda ============'

func = ->(x) { Math.sin(x / 2 - 1) }
puts y2.l_minmax func
puts "\t================================ OUTPUT END ================================"
```

test.rb:

```
require_relative 'main'
```

```
RSpec.describe Solution do
it 'Should return [-1, 0.24] for (x - 1)/(x + 2) like lambda' do
y = Y1.new 0, 2
results = [-1, 0.24]
y.my_minmax { |x| ((x - 1) / (x + 2)) }.each_with_index do |el, index|
expect(el.floor(2)).to eq results[index]
end
end

it 'Should return [-0.99, -0.48] for sin(x / 2 - 1) like proc' do
y = Y2.new(-1, 1)

results = [-0.9971288334080497, -0.4838074403239595]
y.my_minmax { |x| Math.sin(x / 2 - 1) }.each_with_index do |el, index|
expect(el).to eq results[index]
end
end
end
```

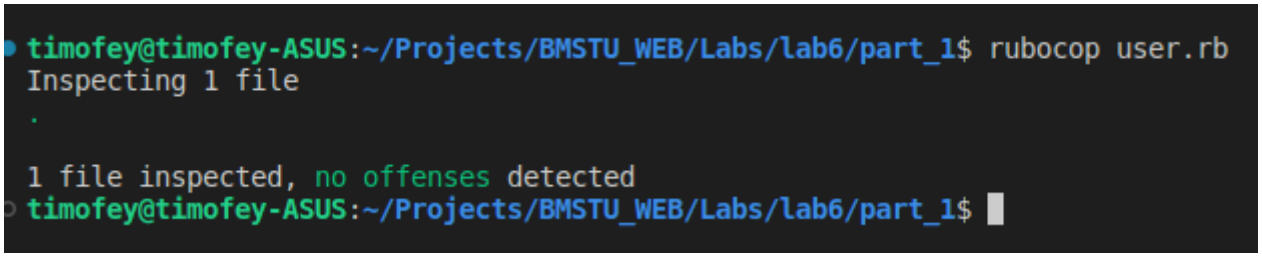
Результаты работы и тестов:

```
timofey@timofey-ASUS:~/Projects/BMSTU_WEB/Labs/lab6/part_3$ ruby user.rb
        ================================ OUTPUT BEGIN ======================
===========
============= MINMAX for (x - 1)/(x + 2) ============
-1
0.24812030075188
============= MINMAX for sin(x / 2 - 1) ============
-0.9971288334080497
-0.4838074403239595
        ================================ OUTPUT with block & proc & lambda
================================
============= MINMAX block for (x - 1)/(x + 2) ============
-1
0.24812030075188
=============MINMAX Proc for sin(x / 2 - 1) ============
-0.9971288334080497
-0.4838074403239595
============ MINMAX Lambda ============
-0.9971288334080497
-0.4838074403239595
        ================================ OUTPUT END ======================
=========
timofey@timofey-ASUS:~/Projects/BMSTU_WEB/Labs/lab6/part_3$ rspec test.rb
..

Finished in 0.00167 seconds (files took 0.05855 seconds to load)
2 examples, 0 failures

timofey@timofey-ASUS:~/Projects/BMSTU_WEB/Labs/lab6/part_3$
```

Rubocop:



Установлен мною в редактор кода в котором пишу, остальное проверял внутри, ошибок нет.

Вывод: Изучил работу Enumerable и Enumerator.