**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

## Дисциплина: языки интернет-программирования

**Вариант №15**

Студент гр. **ИУ6-33Б** _____ В.К. Залыгин

(Подпись, дата) (И.О. Фамилия)

Преподаватель _____ В.Д. Шульман

(Подпись, дата) (И.О. Фамилия)

Москва, 2023

## Цель

Изучить расширенные возможности языка Ruby (инструменты потоковой обработки, примеси из стандартной библиотеки).

## Задание

Задание выполняется как консольное приложение Ruby. Результат следует предоставлять в виде трех отдельных файлов:

- основная программа;
- программа для взаимодействия с пользователем;
- программа для тестирования на основе

  `MiniTest::Unit::TestCase`

Каждое задание сдавать в виде:

- отдельного архива, содержащего указанные файлы;
- pdf-файл с отчётом.

Отчет должен содержать:

1. заполненный титульный лист
2. текст задания
3. тексты программ с подписанными именами файлов
4. результаты выполнения
5. результаты проверки анализаторами rubocop и reek

# ЛР 6

## Часть 1

Решить задачу с точностью $\xi = 10^{-2}, 10^{-4}$, организовав итерационный цикл. Найти первый член последовательности $y = \frac{n}{n^2+2}$, для которого у $y < \xi$. 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 метод для 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.

Рисунок 1 - задание для 15 варианта

## Выполнение

Main.rb

```ruby
# frozen_string_literal: true

require_relative 'task1'
require_relative 'task2'
require_relative 'task3'

def nav_to_task
  puts 'select task number'
  case gets.to_i
  when 1
    io_task1
  when 2
    io_task2
  when 3
    io_task3
  else
    puts 'unknown task number'
  end
end

nav_to_task
```

```
@vzalygin ➜/workspaces/ipl.labs/lab6 (master) $ rubocop
Inspecting 5 files
CC.CC

Offenses:

main.rb:7:1: C: Metrics/MethodLength: Method has too many lines. [11/10]
def nav_to_task ...
^^^^^^^^^^^^^^^
task1.rb:3:1: C: Metrics/MethodLength: Method has too many lines. [16/10]
def find1(eps) ...
^^^^^^^^^^^^^^
task3.rb:3:1: C: Metrics/MethodLength: Method has too many lines. [17/10]
def minmax(l, r, f) ...
^^^^^^^^^^^^^^^^^^^
task3.rb:3:12: C: Naming/MethodParameterName: Method parameter must be at least 3 characters long.
def minmax(l, r, f)
           ^
task3.rb:3:15: C: Naming/MethodParameterName: Method parameter must be at least 3 characters long.
def minmax(l, r, f)
              ^
task3.rb:3:18: C: Naming/MethodParameterName: Method parameter must be at least 3 characters long.
def minmax(l, r, f)
                 ^
tests.rb:8:1: C: Style/Documentation: Missing top-level documentation comment for class Task1Tests.
class Task1Tests < Minitest::Test
^^^^^^^^^^^^^^^^
tests.rb:24:1: C: Style/Documentation: Missing top-level documentation comment for class Task2Tests.
class Task2Tests < Minitest::Test
^^^^^^^^^^^^^^^^
tests.rb:40:1: C: Style/Documentation: Missing top-level documentation comment for class Task3Tests.
class Task3Tests < Minitest::Test
^^^^^^^^^^^^^^^^

5 files inspected, 9 offenses detected
```

Рисунок 2 - ран рубокопа

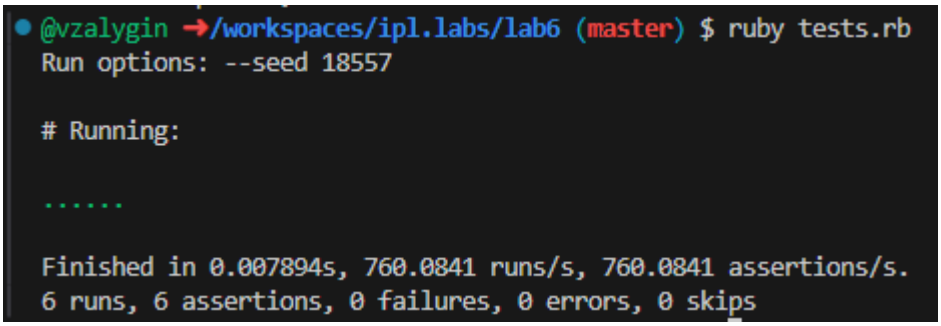


Рисунок 3 - ран тестов

## Часть 1

Task1.rb

```ruby
# frozen_string_literal: true

def find1(eps)
  func = ->(n) { 1.0 * n / ((n**2) + 2) }
  left = 0
  right = 1
  while func.call(right) >= eps
    left = right
    right *= 2
  end
  while right - left > 1
    cur = (right + left) / 2
    if func.call(cur) < eps
      right = cur
    else
      left = cur
    end
  end
  right
end

def io_task1
  puts 'enter the eps'
  eps = gets.to_f
  puts "the first element smaller then #{eps} is on #{find1(eps)} place"
end
```

Тестирование:

```ruby
# frozen_string_literal: true

require 'minitest/autorun'
require_relative 'task1'
require_relative 'task2'
require_relative 'task3'

class Task1Tests < Minitest::Test
  def test_big_eps
    eps = 0.01
    expected = 100
```

```
    assert_equal(expected, find1(eps))
  end

  def test_small_eps
    eps = 0.0001
    expected = 10_000

    assert_equal(expected, find1(eps))
  end
end
```

## Часть 2

Task2.rb

```
# frozen_string_literal: true

def find2(eps)
  sequence = Enumerator.new do |y|
    n = 1
    loop do
      y << [1.0 * n / ((n**2) + 2), n]
      n += 1
    end
  end

  sequence.take_while { |value_key| value_key[0] >= eps }[-1][1] + 1
end

def io_task2
  puts 'enter the eps'
  eps = gets.to_f
  puts "the first element smaller then #{eps} is on #{find2(eps)} place"
end
```

Тестирование:

```
class Task2Tests < Minitest::Test
  def test_big_eps
    eps = 0.01
    expected = 100

    assert_equal(expected, find2(eps))
  end

  def test_small_eps
    eps = 0.0001
    expected = 10_000

    assert_equal(expected, find2(eps))
  end
end
```

## Часть 3

```
# frozen_string_literal: true

def minmax(l, r, f)
```

```ruby
  eps = 0.01
  max_value = -Float::INFINITY
  max_i = 0
  min_value = Float::INFINITY
  min_i = 0
  (0..((r - l) / eps).to_i).each do |i|
    i = i * eps + l
    if max_value < f.call(i)
      max_value = f.call(i)
      max_i = i
    end
    if min_value > f.call(i)
      min_value = f.call(i)
      min_i = i
    end
  end
  [min_i, max_i]
end

def io_task3
  fs = [[0, 2, ->(x) { (x - 1) / (x + 2) }],
        [-1, 1, proc { |x| Math.sin(x / 2 - 1) }]]

  puts 'choose f1 (print 1) or f2 (print 2)'
  min, max = minmax(*fs[gets.to_i - 1])
  puts "min: #{min}, max: #{max}"
end
```

Тестирование:

```ruby
class Task3Tests < Minitest::Test
  def test_real
    f = [0, 2, ->(x) { (x - 1) / (x + 2) }]
    expected = [0, 2]

    assert_equal(expected, minmax(*f))
  end

  def test_sin
    f = [-1, 1, proc { |x| Math.sin(x / 2 - 1) }]
    expected = [-1, 1]

    assert_equal(expected, minmax(*f))
  end
end
```

## Вывод

Изучены расширенные возможности языка Ruby (инструменты потоковой обработки, примеси из стандартной библиотеки).