|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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 **09.03.03 Прикладная информатика**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 5 |

**Название:**

Ruby

**Дисциплина:** Языки Интернет-программирования

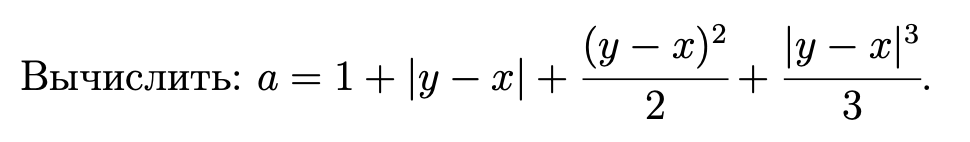
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-35Б |  | 18.10.2022 | В.К.Полубояров |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  |  |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2022

**Цель работы**

Изучить основной синтаксис языка Ruby.

**Часть 1**

****

**Код программы**

**console\_app.rb**

# frozen string\_literal: true

include Equation

require\_relative 'calc\_a'

puts 'Input x'

x = gets.chomp.to\_f

puts 'Input y'

y = gets.chomp.to\_f

p "The result is #{calc\_a(x, y)}"

**calc\_a.rb**

# frozen\_string\_literal: true

# calc the equation

module Equation

def calc\_a(first\_input, second\_input)

mod = (second\_input - first\_input)

1 + mod.abs + ((mod.abs\*\*2) / 2) + ((mod.abs\*\*3) / 3)

end

end

**equation\_spec.rb**

# frozen\_string\_literal: true

require\_relative 'calc\_a'

RSpec.describe Equation do

include Equation

describe '#calc\_a' do

context 'when parameters are equal' do

let!(:first\_parameter) { rand(100) }

let!(:second\_parameter) { first\_parameter }

it 'result is 1' do

expect(calc\_a(first\_parameter,second\_parameter)).to eq(1)

end

end

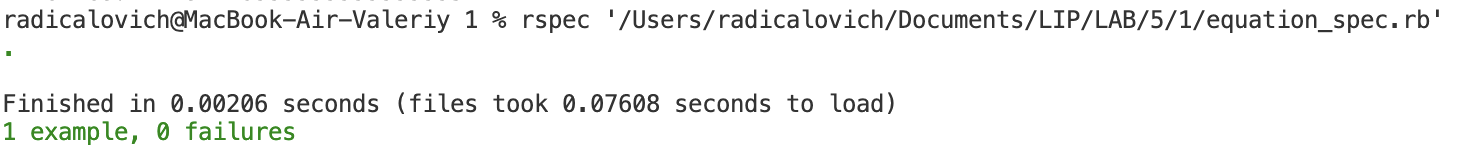
end

end

**Результаты работы программы**

**Изображение выглядит как текст

Автоматически созданное описание**

****

**Часть 2**

**Изображение выглядит как текст

Автоматически созданное описание**

**Код программы**

**console.rb**

# frozen\_string\_literal:true

require\_relative 'functions'

name = []

salary = []

def input(name, salary)

print "\nДля остановки ввода вместо ввода фамилии сотрудника нажмите Enter"

loop do

print "\nВведите фамилию сотрудника: "

name\_input = gets.chomp

break if name\_input == ''

name << name\_input

print "\nВведите зарплату этого сотрудника: "

salary\_input = gets.chomp.to\_i

salary << salary\_input

end

end

def result\_output(name, salary)

print "\n Фамилия сотрудника, зарплата которого наименее отклоняется от средней:

#{Employers.almost\_average(name, salary)}"

print "\n Фамилии двух работников с наибольшей зарплатой: "

Employers.find\_2\_max(name, salary).each { |n| print "#{n}, " }

Employers.delete\_min!(name, salary)

print "\nФамилии сотрудников, после удаления: "

name.each { |n| print "#{n} " }

print "\nЗарплаты сотрудников, после удаления: "

salary.each { |n| print "#{n} " }

end

input(name, salary)

print "\nФамилии сотрудников: "

name.each { |n| print "#{n} " }

print "\nЗарплаты сотрудников: "

salary.each { |n| print "#{n} " }

result\_output(name, salary)

**functions.rb**

# frozen\_string\_literal:true

# all stuff

class Employers

def self.find\_2\_max(name, salary)

array\_of\_2\_max = salary.each\_with\_index.max(2)

[name[array\_of\_2\_max[0][1]], name[array\_of\_2\_max[1][1]]]

end

def self.almost\_average(name, salary)

average\_salary = salary.sum / salary.size

res\_index = salary.index(salary.min\_by { |n| (n - average\_salary).abs })

name[res\_index]

end

def self.delete\_min!(name, salary)

delete\_index = salary.index(salary.min)

salary.delete\_at(delete\_index)

name.delete\_at(delete\_index)

end

end

**employers\_spec.rb**

# frozen\_string\_literal: true

require\_relative 'functions'

RSpec.describe Employers do

describe '#self\_find\_2\_max' do

context 'length is 2' do

let!(:names) { ['Ягмуров', 'Полубояров'] }

let!(:salaries) { [1000, 1600] }

it 'result is array' do

expect(described\_class.find\_2\_max(names, salaries)).to eq(["Полубояров", "Ягмуров"])

end

end

end

end

**Результаты работы программы**

**Изображение выглядит как текст

Автоматически созданное описание**

**Часть 3**

**Изображение выглядит как текст

Автоматически созданное описание**

**Код программы**

**console.rb**

# frozen\_string\_literal: true

require\_relative 'functions'

strings = []

def input(string\_array)

loop do

print "\nВведите строку, оканчивающуюся точкой или нажмите Enter для остановки ввода:\n"

user\_input = gets.chomp

break unless /[.]/.match?(user\_input)

string\_array << user\_input

end

end

def result\_output(string\_array)

print "Результат ввода:\n"

string\_array.each { |n| print "#{n}\n" }

print "Результат вывода:\n"

string\_array.each { |n| puts ChangePositionsOfWords.change\_words(n) }

end

input(strings)

result\_output(strings)

**change\_positions\_of\_the\_words\_spec.rb**

# frozen\_string\_literal: true

require 'faker'

require\_relative 'functions'

RSpec.describe ChangePositionsOfWords do

describe '#self\_change.words' do

context '3 words line' do

let!(:first\_word) { Faker::Lorem.word }

let!(:second\_word) { Faker::Lorem.word }

let!(:third\_word) { Faker::Lorem.word }

let!(:sentence) { "#{first\_word} #{second\_word} #{third\_word}" }

let!(:result\_sentence) { "#{second\_word} #{second\_word} #{third\_word}" }

it 'then the result is' do

expect(described\_class.change\_words(sentence)).to eq(result\_sentence)

end

end

end

end

**functions.rb**

# frozen\_string\_literal: true

# change position

class ChangePositionsOfWords

def self.change\_words(string)

result\_string = ''

temp = ''

string.gsub(/\w+/).with\_index(1) do |n, i|

temp = n

if i.even?

result\_string += "#{temp} #{temp} "

temp = ''

end

end

(result\_string + temp).strip

end

end

**Результаты работы программы**

**Изображение выглядит как текст

Автоматически созданное описание**

**Вывод**

В ходе выполнения лабораторной работы был изучен основной синтаксис языка Ruby, освоены основные методы работы с различными видами объектов (массивы, строки и тд).