|  |  |
| --- | --- |
| 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:**

Работа с массивами и строками в Ruby

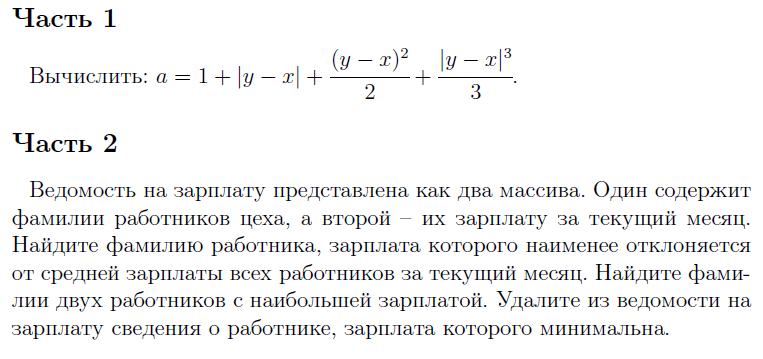
**Дисциплина:** Языки Интернет-программирования

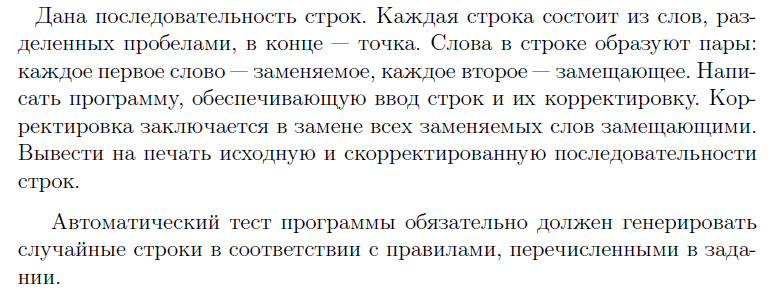


|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-35Б |  | 29.09.2023 | В. И. Мамыкин |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | Е.Ю. Гаврилова |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2023

**Задание:**

****

****

**Цель:** реализовать консольные приложения на Ruby и написать для них тесты. Научиться использовать базовые модули Ruby, работать с массивами и строками.

Часть 1:

**1.rb**

# frozen\_string\_literal: true

require\_relative 'functions'

puts 'Введите x: '

x = gets.chop.to\_f

puts 'Введите y: '

y = gets.chop.to\_f

print "Результат: #{format('%0.5f', my\_function(x, y))}"

**functions.rb**

def my\_function(x, y)

1 + (y - x).abs + (y - x)\*\*2 / 2 + (y - x).abs\*\*3 / 3

End

**tests.rb**

# frozen\_string\_literal: true

require 'minitest/autorun'

require\_relative '1'

class TestFunc < Minitest::Test

def test\_func1

assert\_in\_delta 2.83333, my\_function(1.0, 2.0), 0.001

end

def test\_func2

assert\_in\_delta 4.75000, my\_function(1.0, 2.5), 0.001

end

def test\_func3

assert\_in\_delta 2.83333, my\_function(2.0, 1.0), 0.001

end

end

Часть 2:

**1.rb**

# frozen\_string\_literal: true

require\_relative 'functions'

surnames = Array.new(5)

puts 'Введите фамилии: '

surnames.each\_index { |i| surnames[i] = gets.chop }

paycheks = Array.new(5)

summ = 0

puts 'Введите зарплаты: '

paycheks.each\_index do |i|

paycheks[i] = gets.chop.to\_f

summ += paycheks[i]

end

sr\_arifm = summ / 5

puts 'Фамилия работника, зарплата котрого наименее отклоняется от средней: '

puts operation\_1(surnames, paycheks, sr\_arifm)

puts 'Фамилии работников, зарплаты которых наибольшие: '

puts operation\_2(surnames, paycheks)

puts 'Фамилии работников (исключен работник с наименьшей зарплатой): '

print operation\_3(surnames, paycheks)

**functions.rb**

# frozen\_string\_literal: true

def operation\_1(surnames, paychecks, sr\_arifm)

deviation = 1\_000\_000

index = 0

paychecks.each\_index do |i|

if (paychecks[i] - sr\_arifm).abs < deviation

deviation = (paychecks[i] - sr\_arifm).abs

index = i

end

end

surnames[index]

end

def operation\_2(surnames, paychecks)

maximum = 0

pred\_maximum = 0

paychecks.each\_index do |i|

if paychecks[i] > paychecks[maximum]

pred\_maximum = maximum

maximum = i

elsif paychecks[i] > paychecks[pred\_maximum]

pred\_maximum = i

end

end

[surnames[pred\_maximum], surnames[maximum]]

end

def operation\_3(surnames, paycheks)

minimum = 0

paycheks.each\_index do |i|

minimum = i if paycheks[i] < paycheks[minimum]

end

surnames.delete\_at(minimum)

surnames

end

**tests.rb**

# frozen\_string\_literal: true

require 'minitest/autorun'

require\_relative '1'

class TestFunc < Minitest::Test

def test\_func1

assert\_equal('c', operation\_1(%w[a b c d e], [1, 2, 3, 4, 5], 3))

assert\_equal(%w[d e], operation\_2(%w[a b c d e], [1, 2, 3, 4, 5]))

assert\_equal(%w[b c d e], operation\_3(%w[a b c d e], [1, 2, 3, 4, 5]))

end

end

Часть 3:

**1.rb**

# frozen\_string\_literal: true

require\_relative 'functions'

my\_string = gets.chop

my\_array = my\_string.split

puts "Исходная строка: #{my\_string}"

my\_array[-1] = my\_array[-1].delete\_suffix('.')

# print my\_array

# puts

puts "Полученная строка: #{permutations(my\_array)}"

# print my\_array

# s = ''

# 0.upto(rand(1..10) \* 2) { s += (1..rand(30)).map { rand(97..122).chr }.join + ' '

# s = s.delete\_suffix(' ')

# s += '.'

# puts s

**functions.rb**

# frozen\_string\_literal: true

def permutations(my\_arr)

my\_array = my\_arr.clone

my\_array.each\_index do |i|

my\_array[i] = my\_array[i + 1] unless i.odd?

end

my\_array.join(' ')

end

def proverka(my\_arr)

flag = true

my\_array = my\_arr.clone

my\_array2 = my\_arr.clone

my\_array2.each\_index do |i|

my\_array2[i] = my\_array2[i + 1] unless i.odd?

end

my\_array2.each\_index do |i|

flag = false if !i.odd? && (my\_array2[i] != my\_array[i - 1])

end

if flag

my\_array.join(' ')

else

my\_array2.join(' ')

end

end

**tests.rb**

# frozen\_string\_literal: true

require 'minitest/autorun'

require\_relative '1'

class TestFunc < Minitest::Test

def test\_func1

s = ''

1.upto(rand(1..10) \* 2) { s += (1..rand(30)).map { rand(97..122).chr }.join + ' ' }

s = s.delete\_suffix(' ')

s += '.'

my\_arr = s.split

my\_arr2 = s.split

my\_arr[-1] = my\_arr[-1].delete\_suffix('.')

my\_arr2[-1] = my\_arr2[-1].delete\_suffix('.')

print 'test: ' + s

assert\_equal(proverka(my\_arr), permutations(my\_arr2))

end

end

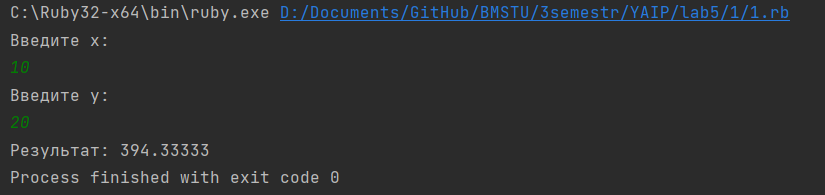


Рисунок 1 – Результат выполнения 1 функции

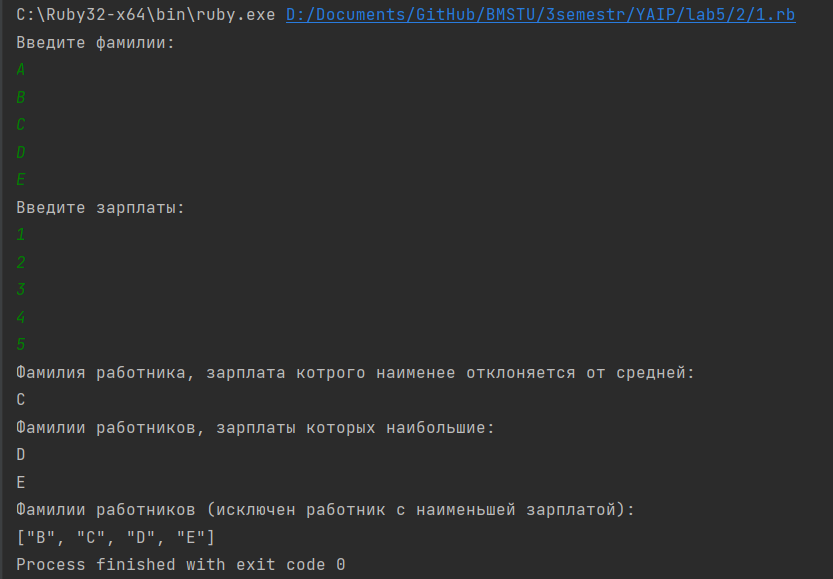


Рисунок 2 – Результат выполнения 2 функции

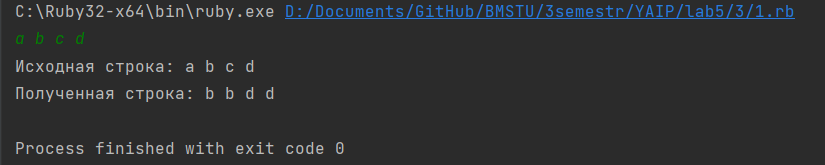


Рисунок 3 – Результат выполнения 3 функции

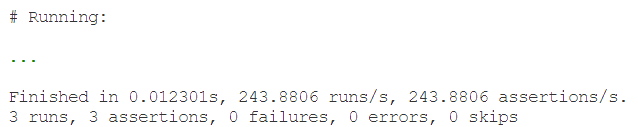


Рисунок 4 – Результат работы тестов 1 части

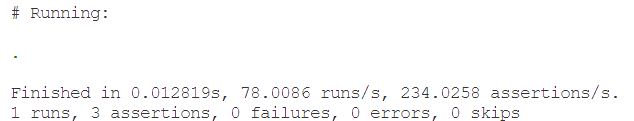
****

Рисунок 5 – Результат работы тестов 2 части

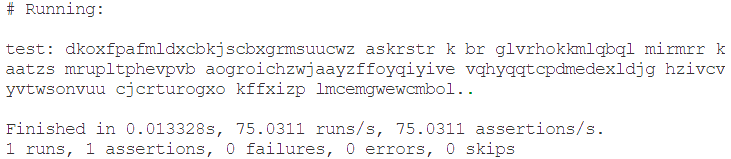
****

Рисунок 6 – Результат работы тестов 3 части (с отладочной печатью)

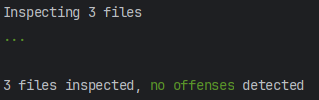
****

Рисунок 7 – Результат работы robocop *–config robocop.yaml (для всех частей задания)*

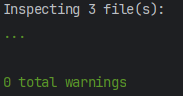
****

Рисунок 8 – Результат работы reek (для всех частей задания)

**Вывод:** были сделаны консольные приложения на Ruby и написаны для них тесты. Изучено использование базовых модули Ruby, а также работа с массивами и строками.