



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

**О Т Ч Е Т**

по лабораторной работе № 5

Дисциплина: Языки интернет-программирования

Студент

ИУ6-35 Б

(Группа)

(Подпись, дата)

И.А. Дулина

(И.О. Фамилия)

Преподаватель

Е.Ю. Гаврилова

(Подпись, дата)

(И.О. Фамилия)

## Вариант 8

**Цель работы:** научиться использовать язык программирования Ruby

### Часть 1

Вычислить:  $y = \frac{\sin(a) - b}{|b| + \cos(b^2)}$ .

**Задание:**

**Код программы:**

*Файл пользователя (client.rb):*

```
#frozen_string_literal: true
```

```
require './main'
puts('Введите a')
a = gets.chomp
puts('Введите b')
b = gets.chomp
puts('Y: ')
puts(calc(a, b))
```

*Файл основной программы (main.rb):*

```
#frozen_string_literal: true
```

```
def calc(aaa, bbb)
  (Math.sin(aaa.to_f) - bbb.to_f) / (bbb.to_f.abs + Math.cos(bbb.to_f * bbb.to_f))
end
```

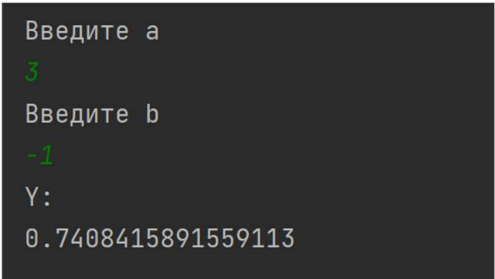
*Файл тестов (test.rb):*

```
#frozen_string_literal: true
```

```
require 'minitest/autorun'

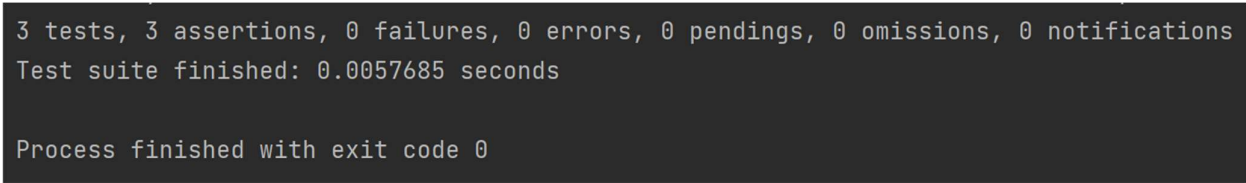
require './main'

class TestCalc < MiniTest::Test
  def test_calc
    assert_in_delta(-0.925, calc(10, 5), 0.01)
    assert_in_delta(-1.225, calc(-3.5, 2), 0.01)
    assert_in_delta(0.574, calc(11, -1.5), 0.01)
  end
end
```



```
Введите a
3
Введите b
-1
Y:
0.7408415891559113
```

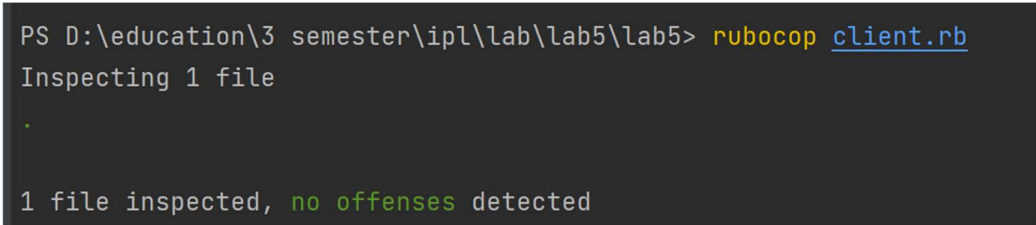
Рисунок 1.1 – результат работы программы



```
3 tests, 3 assertions, 0 failures, 0 errors, 0 pendings, 0 omissions, 0 notifications
Test suite finished: 0.0057685 seconds

Process finished with exit code 0
```

Рисунок 1.2 – результат работы тестов



```
PS D:\education\3 semester\ipl\lab\lab5\lab5> rubocop client.rb
Inspecting 1 file
.

1 file inspected, no offenses detected
```

Рисунок 1.3 – результат работы rubocop client.rb

```
PS D:\education\3 semester\ipl\lab\lab5\lab5> rubocop main.rb
Inspecting 1 file
.

1 file inspected, no offenses detected
```

Рисунок 1.4 – результат работы rubocop main.rb

```
Offenses:

test.rb:6:1: C: Style/Documentation: Missing top-level documentation comment for class TestCalc.
class TestCalc < MiniTest::Test
^^^^^^^^^^^^^^^^
1 file inspected, 1 offense detected
```

Рисунок 1.5 – результат работы rubocop test.rb

## **Часть 2**

### **Задание:**

Дана строка, состоящая из символов латиницы. Необходимо проверить, образуют ли прописные символы и числа из этой строки палиндром.

### **Код программы:**

*Файл пользователя (client2.rb):*

*#frozen\_string\_literal: true*

```
require './main2'
puts('Введите строку')
s = gets.chomp
if prov(s) == true
  puts('Является палиндромом')
else
  puts('Не является палиндромом')
end
```

*Файл основной программы (main2.rb):*

```
#frozen_string_literal: true

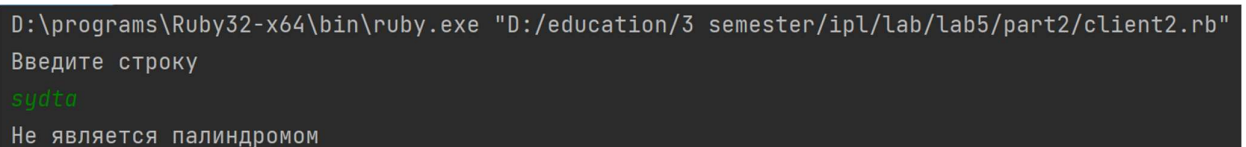
def prov(str)
  str == str.reverse
end
```

*Файл тестов (test2.rb):*

```
#frozen_string_literal: true

require 'test/unit'
require './main2'

class TestPal < Test::Unit::TestCase
  def test_pal
    assert_equal(true, prov('gretypyterg'))
    assert_equal(false, prov('idfsfbk'))
    assert_equal(true, prov('typpyt'))
  end
end
```



```
D:\programs\Ruby32-x64\bin\ruby.exe "D:/education/3 semester/ipL/lab/lab5/part2/client2.rb"
Введите строку
sydta
Не является палиндромом
```

Рисунок 2.1 – результат работы программы

```
D:\programs\Ruby32-x64\bin\ruby.exe "D:/education/3 semester/ipl/lab/lab5/part2/client2.rb"
Введите строку
abba
Является палиндромом
```

Рисунок 2.2 – результат работы программы

```
1 tests, 3 assertions, 0 failures, 0 errors, 0 pendings, 0 omissions, 0 notifications
Test suite finished: 0.0014169 seconds

Process finished with exit code 0
```

Рисунок 2.3 – результат работы тестов

```
PS D:\education\3 semester\ipl\lab\lab5\part2> rubocop client2.rb
Inspecting 1 file
.

1 file inspected, no offenses detected
```

Рисунок 2.4 – результат работы rubocop client.rb

```
PS D:\education\3 semester\ipl\lab\lab5\part2> rubocop main2.rb
Inspecting 1 file
.

1 file inspected, no offenses detected
```

Рисунок 2.5 – результат работы rubocop main.rb

```
test2.rb:6:1: C: Style/Documentation: Missing top-level documentation comment for class TestPal.
class TestPal < Test::Unit::TestCase
^^^^^^^^^^^^^^^^
1 file inspected, 1 offense detected
```

Рисунок 2.6 – результат работы rubocop test.rb

### Часть 3

## Задание:

Дана последовательность строк. Строки содержат зашифрованную информацию и состоят из слов, разделенных пробелом. Пробел записан без шифра. Написать программу, обеспечивающую ввод строк и их расшифровку. Для расшифровки каждая из букв слова заменяется буквой, которая находится через  $n$  букв дальше по алфавиту ( $n$  – вводится с клавиатуры). Буквы, находящиеся на расстоянии, меньшем, чем  $n$  от конца, заменяются после соответствующей корректировки на начальные буквы алфавита. Вывести на печать зашифрованную и подвергнутую дешифровке последовательности строк.

1

---

МГТУ им. Н.Э. Баумана. Каф. ИУ-6. 2022 г.  
Языки Интернет-программирования. Задания по теме Ruby.

---

Автоматический тест программы обязательно должен генерировать случайные строки в соответствии с правилами, перечисленными в задании.

## Код программы:

Файл пользователя (client.rb):

```
#frozen_string_literal: true

require './main'

puts('Введите n - на сколько букв вперёд будет производиться замена')
n = gets.chomp.to_i
puts('Введите количество строк')
count = gets.chomp.to_i
puts('Введите каждую из них, разделяя слова пробелами')
```

```

massive = []
massive_former=[]
i = 0
count.times do
  str = gets.chomp
  str.tr!('0-9', '')
  str.downcase!
  massive_former << str
  massive << shifr(str, n)
end
puts('Зашифрованные строки: ')
massive_former.each { |s| puts s }
puts('Расшифрованные строки: ')
massive.each { |s| puts s }

```

Файл основной программы (main.rb):

```

# frozen_string_literal: true

def shifr(str, number)
  abc = ('a'..'z').zip(1..26).to_h
  strmap = str.split
  id = 0 # индекс слова в строке
  strmap.each do |slovo|
    newstr = ""
    slovo.each_char do |bukva|
      newbukva = zamnab(abc, bukva, number)
      newstr += newbukva
    end
    strmap[id] = newstr
    id += 1
  end

```



```

end
  strmap.join(' ')
end

def zamenab(abc, bukva, number)
  n = abc[bukva].to_i
  n -= 26 if n + number > 26
  n += 26 if n + number < 1
  abc.key(n + number)
end

```

Файл тестов (test.rb):

```

#frozen_string_literal: true

require 'test/unit'
require './main'

def random_str(len = 10, character_set = ['a'..'z'])
  characters = character_set.map(&:to_a).flatten
  characters_len = characters.length
  (0...len).map { characters[rand(characters_len)] }.join
end

def rand_words(num, len = 10, character_set = ['a'..'z'])
  str = ""
  num.times do |_i|
    str += "#{random_str(len, character_set)} "
  end
  str.strip
end

```

end

def prov(*str*, *number*)

*str\_map* = *str*.split

*newstr* = "

*str\_map*.each do |*slovo*|

*newslovo* = zamena(*slovo*, *number*)

*newstr* += "#{*newslovo*} "

  end

*newstr*.chop

end

def zamena(*word*, *number*)

*abc* = ('a'..'z').to\_a

*new\_abc* = *abc*.zip(newabc(*abc*, *number*)).to\_h

*word*.chars.map { |*c*| *new\_abc*.key?(*c*) ? *new\_abc*[*c*] : *c* }.join

end

def newabc(*abc*, *number*)

*newabc* = []

*abc*.each\_with\_index do |\_bukva, *index*|

*number* -= 26 if *index* + *number* > 25

*newabc* << (*abc*[*index* + *number*]).to\_s

  end

*newabc*

end

# *str*='odunxklkn ncpjzqomab utmwfnbdjv nyfstfvfix'

# test

class TestShift < Test::Unit::TestCase

```

def test_shift1
  n = 1
  str = rand_words(4)
  assert_equal(prov(str, n), shifr(str, n))
end

def test_shift2
  n = 10
  str = rand_words(2)
  assert_equal(prov(str, n), shifr(str, n))
end

def test_shift3
  n = -1
  str = rand_words(5)
  assert_equal(prov(str, n), shifr(str, n))
end
end

```

```

D:\programs\Ruby32-x64\bin\ruby.exe "D:/education/3 semester/ipl/lab/lab5/part3/lab53/client.rb"
Введите n - на сколько букв вперёд будет производиться замена
4
Введите количество строк
2
Введите каждую из них, разделяя слова пробелами
djfh sdjfh
askdh fs
Зашифрованные строки:
djfh sdjfh
askdh fs
Расшифрованные строки:
hnjl whnjl
ewohl jw

Process finished with exit code 0

```

Рисунок 3.1 – результат работы программы

