|  |  |
| --- | --- |
| 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 **09.03.01 Информатика и вычислительная техника**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 7 |

**Дисциплина:** Языки интернет-программирования

**Вариант:** 23

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-33Б |  |  | Д. И. Мироненко |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | В. Д. Шульман |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2023

**Часть 1**

**Задание**

Сформировать программным путем символьный файл F, содержащий слова. Определить количество слов в файле F, имеющих длину 5 символов.

Автоматический тест программы обязательно должен проверять работу с файлами.

**Код программы**

**Файл “main.rb”**

|  |
| --- |
| # frozen\_string\_literal: true  require\_relative 'utils'  generate\_words('words.txt')  puts "Words with a length of 5: #{count\_words(5, 'words.txt')}" |

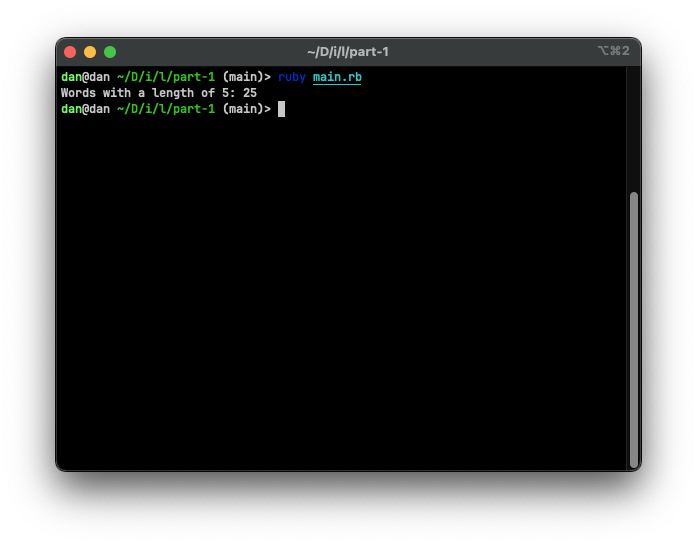
**Файл “utils.rb”**

|  |
| --- |
| # frozen\_string\_literal: true  def count\_words(target\_len, source)  count = 0  File.open(source, 'r') do |file|  file.each do |line|  words = line.split(' ')  words.each do |word|  count += 1 if word.length == target\_len  end  end  end  count  end  def generate\_words(path)  words = []  letters = 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ'  rand(1000).times do  word = ''  rand(20).times do  word += letters[rand(letters.length)]  end  words << word  end  File.open(path, 'w') do |file|  file.write(words.join(' '))  end  end |

**Файл “test\_utils.rb”**

|  |
| --- |
| # frozen\_string\_literal: true  require 'minitest/autorun'  require\_relative 'utils'  # Test 'count\_words'  class TestCountWords < Minitest::Test  @@path = './test\_words.txt'  def setup  File.open(@@path, 'w') do |\_|  end  end  def teardown  File.delete(@@path) do |\_|  end  end  def insert(content)  File.open(@@path, 'w') do |file|  file.write(content)  end  end  def test\_empty  assert\_equal 0, count\_words(5, @@path)  end  def test\_regular  insert("asdf iur\nriwby\nirbfa ebr")  assert\_equal 2, count\_words(5, @@path)  end  end |

**Работа программы**

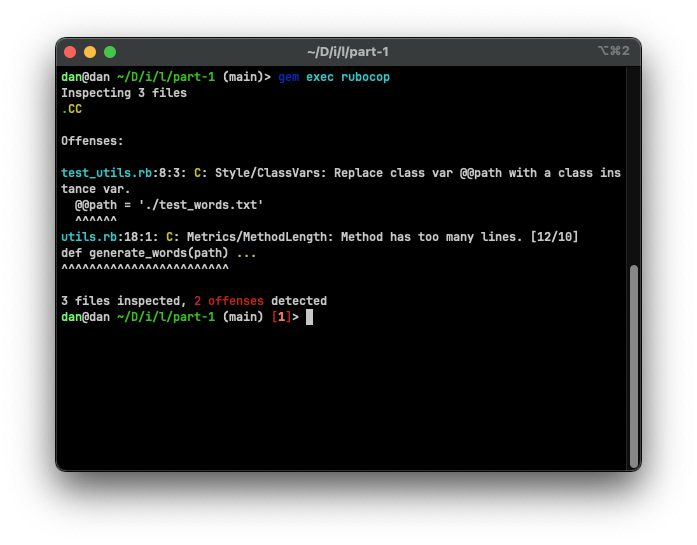


**Рисунок 1** – работа программы

**A screenshot of a computer

Description automatically generated**

**Рисунок 2** – выполнение unit-тестов

****

**Рисунок 3** – проверка кода rubocop’ом

**Часть 2**

**Задание**

Разработать и реализовать иерархию классов для описанных объектов предметной области, используя механизмы наследования. Проверить ее на тестовом примере, с демонстрацией всех возможностей разработанных классов на конкретных данных.

Объект — окно, хранящее размеры и умеющее выводить на экран площадь.

Объект — окно, хранящее размеры и сведения о наличии жалюзи и умеющее выводить на экран площадь и признак наличия жалюзи.

В тестирующей программе обеспечить автоматическую проверку того, что созданные объекты действительно соответствуют заданной иерархии классов.

**Код программы**

**Файл “main.rb”**

|  |
| --- |
| # frozen\_string\_literal: true  require\_relative 'utils'  window = Window.new(5, 6)  puts "square? #{window.square}"  advanced\_window = AdvancedWindow.new(5, 6, true)  puts "jalousie? #{advanced\_window.jalousie?}" |

**Файл “utils.rb”**

|  |
| --- |
| # frozen\_string\_literal: true  # window class  class Window  attr\_reader :width, :height  def initialize(width, height)  if !(width.is\_a?(Integer) || width.is\_a?(Float)) || !width.positive?  raise TypeError, "'width' must be a positive integer!"  end  if !(height.is\_a?(Integer) || height.is\_a?(Float)) || !height.positive?  raise TypeError, "'height' must be a positive integer!"  end  @width = width  @height = height  end  def square  width \* height  end  end  # advanced window class  class AdvancedWindow < Window  attr\_reader :jalousie  def initialize(width, height, jalousie)  raise TypeError, "'jalousie' must be a positive integer!" unless [true, false].include? jalousie  super width, height  @jalousie = jalousie  end  def jalousie?  @jalousie  end  end |

**Файл “test\_utils.rb”**

|  |
| --- |
| # frozen\_string\_literal: true  require 'minitest/autorun'  require\_relative 'utils'  # Test 'Window'  class TestWindow < Minitest::Test  def test\_square  window = Window.new(5, 6)  assert\_equal 30, window.square  end  def test\_negative\_width  assert\_raises TypeError do  Window.new(4, -2)  end  end  def test\_negative\_height  assert\_raises TypeError do  Window.new(-4, 2)  end  end  def test\_zero  assert\_raises TypeError do  Window.new(0, 4)  end  end  def test\_boolean  assert\_raises TypeError do  Window.new(5, false)  end  end  end  # Test 'AdvancedWindow'  class TestAdvancedWindow < Minitest::Test  def test\_jalousie  assert\_equal true, AdvancedWindow.new(5, 6, true).jalousie?  end  def test\_jalousie\_invalid\_type  assert\_raises TypeError do  AdvancedWindow.new(5, 6, 1)  end  end  end |

**Работа программы**

A screenshot of a computer

Description automatically generated

**Рисунок 4** – работа программы

**A screenshot of a computer program

Description automatically generated**

**Рисунок 5** – выполнение unit-тестов

**A screenshot of a computer screen

Description automatically generated**

**Рисунок 6** – проверка кода rubocop’ом

**Вывод**

Реализованы несколько программ на языке программирования ruby; функции всех программ проверена на unit-тестах; исходный код проверен линтером rubocop на наличие ошибок.