



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 № 8

Название: Создание простейших веб-приложений
Ruby on rails

Дисциплина: Языки Интернет-программирования

Студент

ИУ6-35Б

(Группа)

10.11.2023

(Подпись, дата)

В. И. Мамыкин

(И.О. Фамилия)

Преподаватель

Е.Ю. Гаврилова

(Подпись, дата)

(И.О. Фамилия)

Москва, 2023

Задание:

1. Сгенерируйте каркас Rails-приложения в директории, полный путь к которой содержит только символы кодировки ASCII-7bit.
2. С помощью команды **rails generate controller** сформируйте контроллер для реализации логики приложения и двух действий: ввод данных, просмотр результата.
3. Допишите код сформированного контроллера для расчета функции, заданной индивидуально. Предварительно разработайте и отладьте программу вычисления функции вне Rails-приложения и разместите в контроллере уже отлаженный код.
4. Напишите в файле представления (.erb) код для генерации формы ввода данных, необходимых при расчете, а также код для форматирования результатов расчета в виде таблицы с использованием соответствующих элементов разметки.
5. Отладьте и проверьте работу приложения.
6. Замените обращение по корневому адресу на обращение к действиям созданного контроллера.
7. Реализуйте функциональный тест разработанного контроллера приложения на базе каркаса, сформированного при его создании. Проверьте выполнение теста.

Цель: углубление теоретических сведений о принципах проектирования Model-View-Controller и получение практических навыков создания веб-приложения с использованием средств Ruby on Rails, построения простейших форм и выполнения вычислений на стороне серверной части приложения.

input.html.erb

```
<h1>Sequences#input</h1>
<p>Find me in app/views/sequences/input.html.erb</p>
<!-- <%= form_tag("/sequences/view", :method => "get") do %>
  <%= label_tag("Value n:") %>
  <%= text_field_tag(:v1) %> <br/>
  <%= label_tag("Sequence") %>
  <%= text_field_tag(:v2) %> <br/>
<br/>
```

```

    <%= submit_tag("Result") %>
<% end %> -->

<form action="/sequences/view" method="get" accept-charset="UTF-8">
  <label for="v1" > Число: </label>
  <input type="text" id="v1" name="v1" pattern="^([0-9])+" value="10" required/>
  <label for="v2" style="margin-left: 20px"> Последовательность: </label>
  <input type="text" id="v2" name="v2" pattern="^([0-9]+[\s]{0,1})+" value="1 2
3 4 9 16 3 2 4 10" required/>
  <button>Найти!</button>
</form>

```

view.html.erb

```

<h1>Sequences#view</h1>

<p>Find me in app/views/sequences/view.html.erb</p>

<% if @number[0] == @number[1] %>

  <% if @result[2] != '0' %>

    <tr>

      <td style="border: 2px solid darkblue;"> <strong> Введенная
последовательность: </strong> </td>

    </tr>

    <tr>

      <td style="border: 2px solid darkblue;"> <%= @result[3] %> </td>

    </tr>

    <tr>

      <td style="border: 2px solid darkblue;"> <strong> Подпоследовательности:
</strong> </td>

    </tr>

```

```

<% @result[0].each do |res| %>

  <tr>

    <td style="border: 2px solid darkblue;"> <%= res %> </td>

  </tr>

<% end %>

<tr>

  <td style="border: 2px solid darkblue;"> <strong> Самая длинная
последовательность: </strong> </td>

</tr>

<tr>

  <td style="border: 2px solid darkblue;"> <%= @result[1] %> </td>

</tr>

<tr>

  <td style="border: 2px solid darkblue;"> <strong> Ее длина: </strong> </td>

</tr>

<tr>

  <td style="border: 2px solid darkblue;"> <%= @result[2] %> </td>

</tr>

<% else %>

<tr>

  <td style="border: 2px solid darkblue;"> <strong> Введенная
последовательность: </strong> </td>

</tr>

```

<tr>

<td style="border: 2px solid darkblue;"> <%= @result[3] %> </td>

</tr>

<tr>

<td style="border: 2px solid darkblue;"> Подпоследовательностей
квадратов натуральных чисел нет </td>

</tr>

<% end %>

<% else %>

<tr>

<td style="border: 2px solid darkblue;"> Введенная
последовательность: </td>

</tr>

<tr>

<td style="border: 2px solid darkblue;"> <%= @result[3] %> </td>

</tr>

<tr>

<td style="border: 2px solid darkblue;"> Введенное число n:
 </td>

</tr>

<tr>

<td style="border: 2px solid darkblue;"> <%= @number[0] %> </td>

</tr>

```

<tr>

  <td style="border: 2px solid darkblue;"> <strong> Число n не совпадает с
длинной введенной последовательности! </strong> </td>

</tr>

<% end %>

</table>

<br/>

<%= button_to "Вернуться", sequences_input_path, method: :get %>

```

sequences_controller.rb

```

# frozen_string_literal: true

class SequencesController < ApplicationController
  def input; end

  def view
    # Функция, которая проверяет, является ли подпоследовательность
    полными квадратами

    longest_subsequence = [] # Самая длинная подпоследовательность
    current_subsequence = [] # Текущая подпоследовательность
    all_subsequences = [] # Все подпоследовательности
    # unless params[:v2].nil?
    sequence = params[:v2]&.split(' ').&.map(&:to_i)
    sequence.&.each do |number|
      if (Math.sqrt(number) % 1).zero?
        # Если число является полным квадратом

```

```

    current_subsequence << number
  else
    # Если число не является полным квадратом
    longest_subsequence = current_subsequence.clone if
current_subsequence.length > longest_subsequence.length
    cur = current_subsequence.clone
    all_subsequences << cur.join(' ')
    all_subsequences.pop if all_subsequences[all_subsequences.size - 1] == "
    current_subsequence = []
  end
end
cur = current_subsequence.clone
all_subsequences << cur.join(' ') if cur.length.positive?
longest_subsequence = current_subsequence.clone if current_subsequence.length
> longest_subsequence.length
all_subsequences.pop if all_subsequences[all_subsequences.size - 1] == "
subsequence_count = longest_subsequence.length
@number = [params[:v1], sequence&.size.to_s]
@result = [all_subsequences, longest_subsequence.join(' '),
subsequence_count.to_s, sequence&.join(' ')]
# end
# @number = [params[:v1], 0]
# @result = [all_subsequences, ", '0', "]
logger.debug "from view controller: #{ @result}"
end
end

```

sequences_controller_test.rb

```
require 'test_helper'
```

```
class SequencesControllerTest < ActionDispatch::IntegrationTest

  test 'should get input' do
    get sequences_input_url
    assert_response :success
  end

  test 'should get view' do
    get sequences_view_url
    assert_response :success
  end

  test 'should get sequence' do
    get sequences_view_url, params: { v1: '10', v2: '1 2 3 4 9 16 3 2 4 10' }
    assert_equal assigns[:result], [['1', '4 9 16', '4'], '4 9 16', '3', '1 2 3 4 9 16 3 2 4 10']
  end

  test 'should get sequence - 2' do
    get sequences_view_url, params: { v1: '3', v2: '1 4 4' }
    assert_equal assigns[:result], [['1 4 4'], '1 4 4', '3', '1 4 4']
  end

end
```


Sequences#input

Find me in app/views/sequences/input.html.erb

Число: Последовательность:

Рисунок 1 – страница ввода

Sequences#view

Find me in app/views/sequences/view.html.erb

Введенная последовательность: 1 2 3 4 9 16 3 2 4 10 Подпоследовательности: 1 4 9 16 4 Самая длинная подпоследовательность: 4 9 16 Ее длина: 3

Рисунок 2 – страница результата

```
# Running:

.....

Finished in 0.572128s, 6.9914 runs/s, 6.9914 assertions/s.
4 runs, 4 assertions, 0 failures, 0 errors, 0 skips
```

Рисунок 3 – результат работы тестирующей программы

Вывод: были изучены теоретические сведения о принципах проектирования Model-View-Controller и получены практические навыки создания веб-приложения с использованием средств Ruby on Rails, построены простейшие формы и выполнены вычисления на стороне серверной части приложения.