|  |  |
| --- | --- |
| 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 №** | 9 |

**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

**Задание**

При помощи Javascript модифицировать код ЛР 8 таким образом, чтобы для отображения результатов вычисления браузер не выполнял полную перезагрузку страницы.

Сформировать тесты для проверки работы программы при помощи Katalon Recorder / Selenium Webdriver.

**Код программы**

**Файл “** **app/controllers/calc\_controller.rb”**

|  |
| --- |
| class CalcController < ApplicationController  def input  render layout: false  end  def view  input = params[:array]  @longest\_sequence = []  @sequences = []  return unless input&.match?(/\A\s\*\d+(\s+\d+)\*\s\*\z/)  array = input.split  start = 0  max\_len = 0  curr\_len = 1  array.length.times do |i|  next if i.zero?  if array[i - 1].to\_i <= array[i].to\_i  curr\_len += 1  else  @sequences << array[start, curr\_len] if curr\_len > 1  if curr\_len > max\_len  @longest\_sequence = @sequences[-1]  max\_len = curr\_len  end  start = i  curr\_len = 1  end  end  @sequences << array[start, curr\_len] if curr\_len > 1  if curr\_len > max\_len  @longest\_sequence = @sequences[-1]  max\_len = curr\_len  end  @result = {  longest\_sequence: @longest\_sequence,  sequences: @sequences  }  respond\_to do |format|  format.html  format.json do  render json: {  type: @result.class.to\_s,  value: @result  }  end  end  end  end |

**Файл “public/index.html”**

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta http-equiv="Content-Type" content="text/html charset=utf-8">  <title>AJAX Calc</title>  <script>  calc\_url= location.protocol + "//" + location.host + "/calc/view.json"    function send\_query() {  var frm = document.forms.calc\_form;  var param\_str = "";  if (frm.array.value?.trim()) {  param\_str = "?" + "array=" + frm.array.value;  } else {  alert("Enter the array of numbers!");  return false;  }    var my\_JSON\_object = {};  var http\_request = new XMLHttpRequest();  http\_request.open("GET", calc\_url + param\_str, true);  http\_request.onreadystatechange = function () {  var done = 4, ok = 200;  if (http\_request.readyState == done && http\_request.status == ok) {  my\_JSON\_object = JSON.parse(http\_request.responseText);  show\_result(my\_JSON\_object);  }  };  http\_request.send(null);    return false;  }  function create\_row(title, sequence) {  let row = document.createElement("tr");    let title\_item = document.createElement("td");  title\_item.innerHTML = title;    let value\_item = document.createElement("td");  if (sequence) {  value\_item.innerHTML = sequence.join(" ");  }    row.appendChild(title\_item);  row.appendChild(value\_item);  return row;  }  function show\_result(data) {  let table = document.getElementById("result\_table");  table.innerHTML = "";  for (const sequence of data.value.sequences) {  table.appendChild(create\_row("Sequence Found:", sequence));  }  if (data.value.longest\_sequence) {  table.appendChild(create\_row("Longest Sequence:", data.value.longest\_sequence));  } else {  table.appendChild(create\_row("Longest Sequence:", ["Not Found"]));  }  }  </script>  </head>  <body>  <form name="calc\_form" method="get">  <label>Array</label><br>  <input id="array" type="text">  <br><br>  <input name="commit" type="submit" value="Get Result" onclick="return send\_query();">  </form>  <div id="result"><br><table id="result\_table"></table></div>  </body>  </html> |

**Файл “test/controllers/calc\_controller\_test.rb”**

|  |
| --- |
| require 'test\_helper'  class CalcControllerTest < ActionDispatch::IntegrationTest  test 'should get input' do  get calc\_input\_url  assert\_response :success  end  test 'should get view' do  get calc\_view\_url  assert\_response :success  end  test 'should get 5 6 8 for view with 5 6 8 5 6 3' do  get calc\_view\_url, params: { array: '5 6 8 5 6 3' }  assert\_equal assigns[:longest\_sequence], %w[5 6 8]  end  test 'should get unknown for view with letters' do  get calc\_view\_url, params: { array: 'j 3 5 e 4' }  assert\_equal assigns[:longest\_sequence], []  end  test 'should get unknown for view with empty array' do  get calc\_view\_url, params: { array: '' }  assert\_equal assigns[:longest\_sequence], []  end  end |

**Файл “test/spec/regular\_spec.rb”**

|  |
| --- |
| require 'selenium-webdriver'  require 'json'  describe 'Regular' do  before(:each) do  @driver = Selenium::WebDriver.for :chrome  @vars = {}  end  after(:each) do  @driver.quit  end  it 'regular' do  @driver.get('http://localhost:3000/')  @driver.find\_element(:id, 'array').send\_keys('3 6 5 17 26 21 9 15 7')  @driver.find\_element(:name, 'commit').click  sleep(0.2)  html\_content = @driver.find\_element(:id, 'result\_table')['innerHTML']  expect(html\_content).to include('<tr><td>Sequence Found:</td><td>3 6</td></tr><tr><td>Sequence Found:</td><td>5 17 26</td></tr><tr><td>Sequence Found:</td><td>9 15</td></tr><tr><td>Longest Sequence:</td><td>5 17 26</td></tr>')  end  end |

**Файл “test/spec/onenumber\_spec.rb”**

|  |
| --- |
| require 'selenium-webdriver'  require 'json'  describe 'Onenumber' do  before(:each) do  @driver = Selenium::WebDriver.for :chrome  @vars = {}  end  after(:each) do  @driver.quit  end  it 'onenumber' do  @driver.get('http://localhost:3000/')  @driver.find\_element(:id, 'array').click  @driver.find\_element(:id, 'array').send\_keys('1')  @driver.find\_element(:name, 'commit').click  expect(@driver.find\_element(:id, 'result\_table').text).to eq('Longest Sequence: Not Found')  end  end |

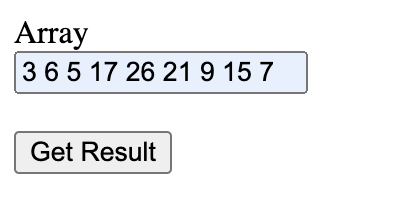
**Файл “test/spec/empty\_spec.rb”**

|  |
| --- |
| require 'selenium-webdriver'  require 'json'  describe 'Empty' do  before(:each) do  @driver = Selenium::WebDriver.for :chrome  @vars = {}  end  after(:each) do  @driver.quit  end  it 'empty' do  @driver.get('http://localhost:3000/')  @driver.find\_element(:id, 'array').click  @driver.find\_element(:name, 'commit').click  expect(@driver.switch\_to.alert.text).to eq('Enter the array of numbers!')  end  end |

**Файл “test/spec/spacesonly\_spec.rb”**

|  |
| --- |
| require 'selenium-webdriver'  require 'json'  describe 'Spacesonly' do  before(:each) do  @driver = Selenium::WebDriver.for :chrome  @vars = {}  end  after(:each) do  @driver.quit  end  it 'spacesonly' do  @driver.get('http://localhost:3000/')  @driver.find\_element(:id, 'array').click  @driver.find\_element(:id, 'array').send\_keys(' ')  @driver.find\_element(:name, 'commit').click  expect(@driver.switch\_to.alert.text).to eq('Enter the array of numbers!')  end  end |

**Работа программы**



**Рисунок 1** – ввод значений

**A screenshot of a computer

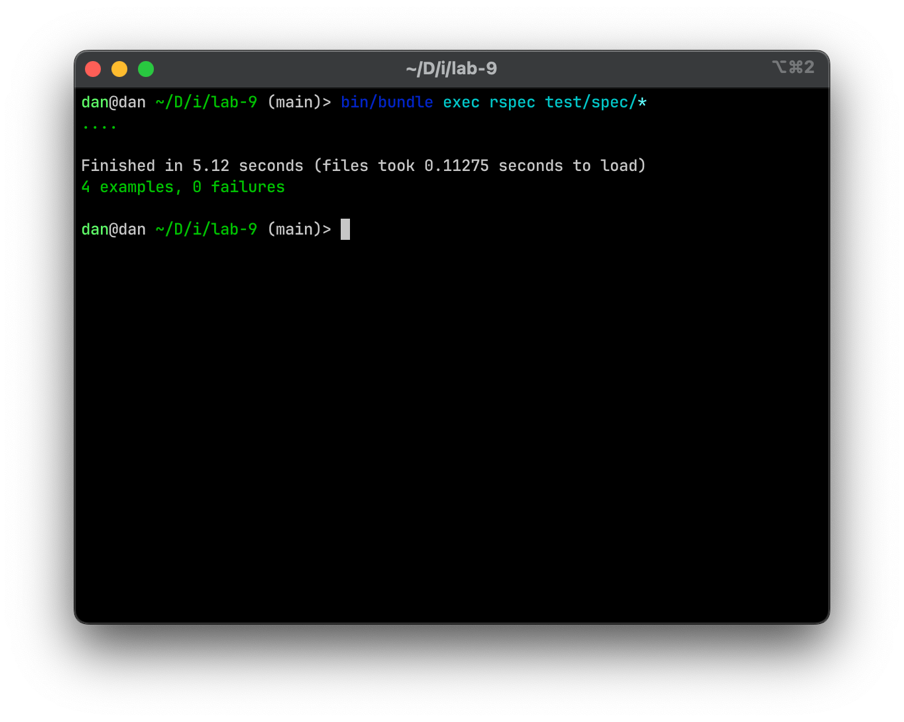
Description automatically generated**

**Рисунок 2** –вывод результатов

**A computer screen shot of a program

Description automatically generated**

**Рисунок 3** – результат выполнения функциональных тестов

****

**Рисунок 4** – результат выполнения selenium тестов

**Вывод**

Изучена технология AJAX для динамического отображения контента на web-странице; полученные знания отработаны при создании простого web-сайта; написаны и выполнены функциональные тесты, а также тесты, записанные в selenium.