

**Федеральное государственное бюджетное образовательное учреждение высшего
профессионального образования**



**«Московский государственный технический
университет
имени Н.Э. Баумана»**

(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Компьютерные системы и сети

О т ч е т

Лабораторная работа 10

Студент гр.ИУ6-32 _____
(Подпись, дата)

Огданец Е.П.
(И.О. Фамилия)

Преподаватель _____
(Подпись, дата)

Самарев Р.С.
(И.О. Фамилия)

Задание 1

Модифицировать код ЛР 8 таким образом, чтобы по запросу с указанными параметрами выдавался результат в формате XML (средствами стандартной сериализации ActiveSupport).

- Проверить формирование XML и сохранить в файл для отладки XSLT и второго приложения.
- Написать функциональный тест, проверяющий формат выдаваемых данных при запросе RSS.

Разработать XSLT-программу преобразования полученной XML в HTML.

Добавить в проверяемый XML-файл строку привязки к преобразованию `<?xml-stylesheet type="text/xsl" href="some_transformer.xslt"?>`. Проверить корректность отображения браузером результата преобразования.

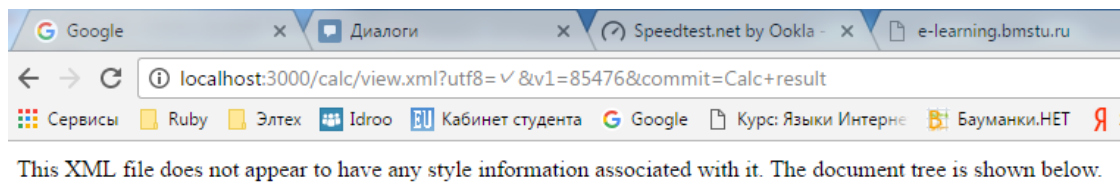
Проверить на автономной Ruby-программе корректность преобразования, используя следующий фрагмент кода:

```
require 'nokogiri'
```

```
doc = Nokogiri::XML(File.read('some_file.xml'))
```

```
xslt = Nokogiri::XSLT(File.read('some_transformer.xslt'))
```

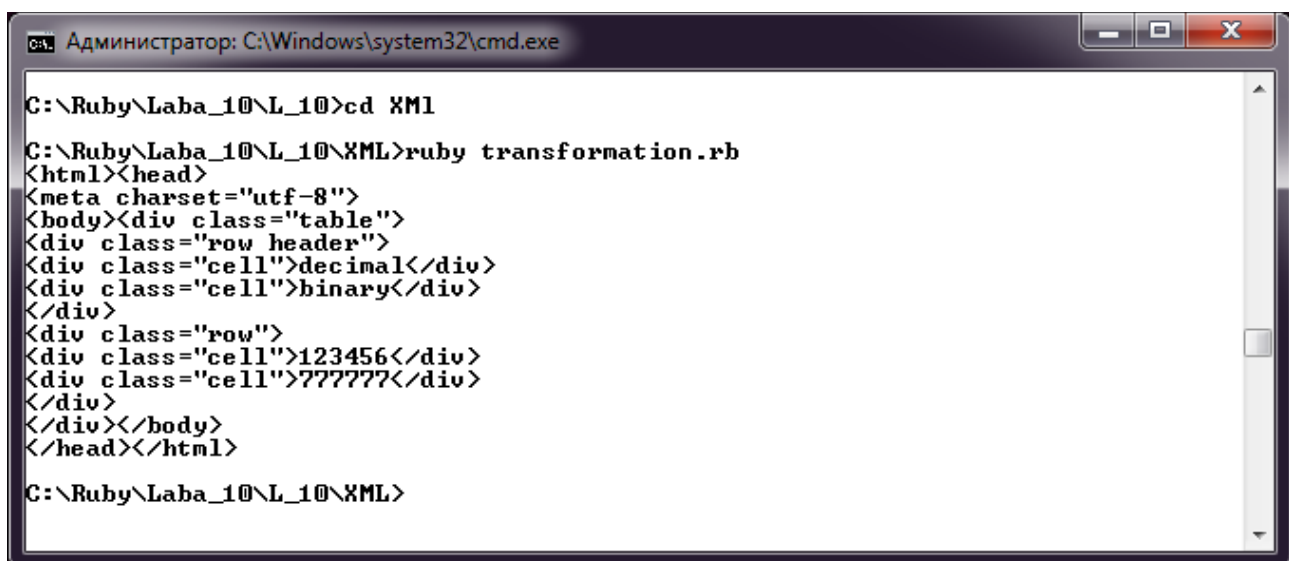
```
puts xslt.transform(doc)
```



```
<?xml>
<hash>
  <result>
    <description>Result</description>
    <type>Array</type>
    <value type="array">
      <value>85476</value>
      <value>152934</value>
      <value>592185</value>
      <value>1173480</value>
      <value>2017191</value>
      <value>3934293</value>
      <value>7858686</value>
      <value>14727273</value>
      <value>52000014</value>
      <value>93000039</value>
    </value>
  </result>
  <text>
    <decimal>
      <type>String</type>
      <value>decimal</value>
    </decimal>
    <binary>
      <type>String</type>
      <value>binary</value>
    </binary>
    <repeatcalculation>
      <type>String</type>
      <value>repeat_calculation</value>
    </repeatcalculation>
  </text>
</hash>
```

Transformer.xslt

```
<?xml version='1.0' encoding='UTF-8'?>
<xsl:stylesheet version='1.0' xmlns:xsl='http://www.w3.org/1999/XSL/Transform'>
  <xsl:template match='/>
    <html>
      <head>
        <meta charset="utf-8">
      </head>
      <body>
        <div class='table'>
          <div class='row header'>
            <div class='cell'><xsl:value-of
select="//text/decimal/value"/></div>
            <div class='cell'><xsl:value-of
select="//text/binary/value"/></div>
          </div>
          <xsl:for-each select='//result/value'>
            <div class='row'>
              <div class='cell'><xsl:value-of
select="value"/></div>
              <div class='cell'><xsl:value-of
select="value/following-sibling::*"/></div>
            </div>
          </xsl:for-each>
        </div>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```



The screenshot shows a Windows command prompt window titled "Администратор: C:\Windows\system32\cmd.exe". The user is in the directory "C:\Ruby\Laba_10\L_10" and has run the command "cd XML". They then executed "ruby transformation.rb", which produced the following XML output:

```
C:\Ruby\Laba_10\L_10\XML>cd XML
C:\Ruby\Laba_10\L_10\XML>ruby transformation.rb
<html><head>
<meta charset="utf-8">
<body><div class="table">
<div class="row header">
<div class="cell">decimal</div>
<div class="cell">binary</div>
</div>
<div class="row">
<div class="cell">123456</div>
<div class="cell">777777</div>
</div></body>
</html>
C:\Ruby\Laba_10\L_10\XML>
```

Задание 2

Разработать второе приложение, являющееся посредником между клиентом и первым приложением, задачей которого является преобразование XML в HTML или передача в неизменном виде браузеру для отображения браузером.

Приложение должно запускаться с указанием номера порта TCP, отличным от номера порта первого приложения (например rails server -p 3001)!

- Подготовить каркас приложения, а также форму формирования запроса, форму отображения результата и соответствующие действия контролера.
- Добавить в контроллер преобразование XML в HTML с помощью ранее разработанного XSLT-файла.
- Подключить запрос XML с первого приложения и проверить работу приложений в связке.
- Написать функциональный тест, проверяющий что при различных входных данных результат генерируемой страницы различен.
- Доработать код контроллера и представлений данного приложения для выдачи браузеру XML-потока в неизменном виде (организовать возможность выбора формата выдачи для пользователя).
- Проверить, что браузер получает XML первого приложения в неизменном виде.
- Доработать код контроллера приложения таким образом, чтобы XML-поток первого приложения получал дополнительную строку, указывающую xsl. Модифицировать форму запроса параметров таким образом, чтобы браузер получал в ответ XML. При этом разместить XSLT-файл в директории public.
- Проверить, что браузер производит преобразование XML->HTML в соответствии с xlt.
- Реализовать функциональные тесты второго приложения. Проверить результаты, формируемые приложением, на соответствие выбранному формату выдачи.

Итоговая форма ввода параметра должна содержать кнопки или селектор, позволяющие проверить два варианта преобразования:

- Серверное xml+xslt->html
- Клиентское xml+xslt->html

Factorial_controller.rb

```
class CalcController < ApplicationController
  def input
    end

  def ifPalindrom(num)
    #mas = num.split(/[1-9]/)
    mas = num.split("")
    #p mas
    n = mas.length / 2
    #1234321
    ifpolindrom = true
    for i in 0..n
      if mas[i] != mas[mas.length - 1 - i]
        ifpolindrom = false
      end
    end
    ifpolindrom
  end

  def Change(num)
    mas = num.split("")
    n = mas.length / 2
    n -= 1
    for i in 0..n
      c = mas[i]
      mas[i] = mas[mas.length - 1 - i]
      mas[mas.length - 1 - i] = c
    end

    (mas.join.to_i + num.to_i).to_s
  end

  def view
    str = params[:v1].to_s

    resul_array = []
    count = 0
    resul_array[count] = str
    while !ifPalindrom(str)

      str = Change(str)
      count += 1
      resul_array[count] = str
      #p str
    end
    #v1 = params[:v1].to_s
    @result = resul_array
    respond_to do |format|
      format.html
      format.xml do
        render xml: {
```

```

        result: {
          description: 'Result',
          type: @result.class.to_s,
          value: @result
        },
        text: {
          decimal: { type: 'String', value: 'decimal' },
          binary: { type: 'String', value: 'binary' },
          repeatcalculation: { type: 'String', value: 'repeat_calculation' }
        }
      }
    end
  end
end
end
end

```

Input.html.erb

```

<h1>Calc#input</h1>
<p>Find me in app/views/calc/input.html.erb</p>
<%= form_tag("/calc/view.xml", :method => "get") do %>
  <%= label_tag("Value number:") %>
  <%= text_field_tag(:v1) %> <br/>
<br/>
  <%= submit_tag("Calc result") %>
<% end %>

```

View.html.erb

```

<h1>Calc#view</h1>
<p>Find me in app/views/calc/view.html.erb</p>

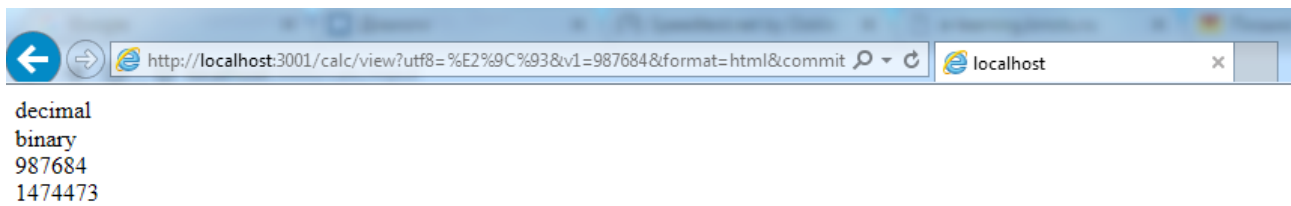
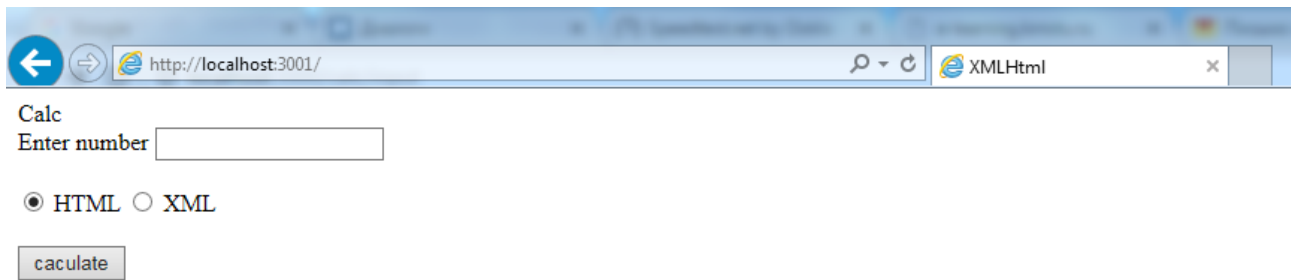
<table border="1">
<caption>Table of result</caption>
  <tr>
    <th>i</th>
    <th>number</th>
  </tr>
  <% for i in 0..@result.length-1 %>
    <tr><td><%=i+1%></td><td><%=@result[i]%></td></tr>
  <%end%>

</table>

<%= link_to "Repeat calculation", :calc_input %>

```

Интерфейс



Вывод: мы научились организовывать вывод информации в формате XML. Проверять формирование XML. Разрабатывать XSLT-программу преобразования XML в HTML.