

Практическое задание

Генерация html-страниц на основе шаблонов

Задание 1. Дана функция:

$$f(x) = x^3 - 6x^2 + x + 5$$

Для этой функции на интервале $[a, b]$ вывести таблицу значений, состоящую из 30 строк, а также построить график функции на заданном интервале. Информацию сохранить в виде html-страницы, результат генерации страницы показан на рисунке 1 (на рисунке таблица показана в сокращенном виде).

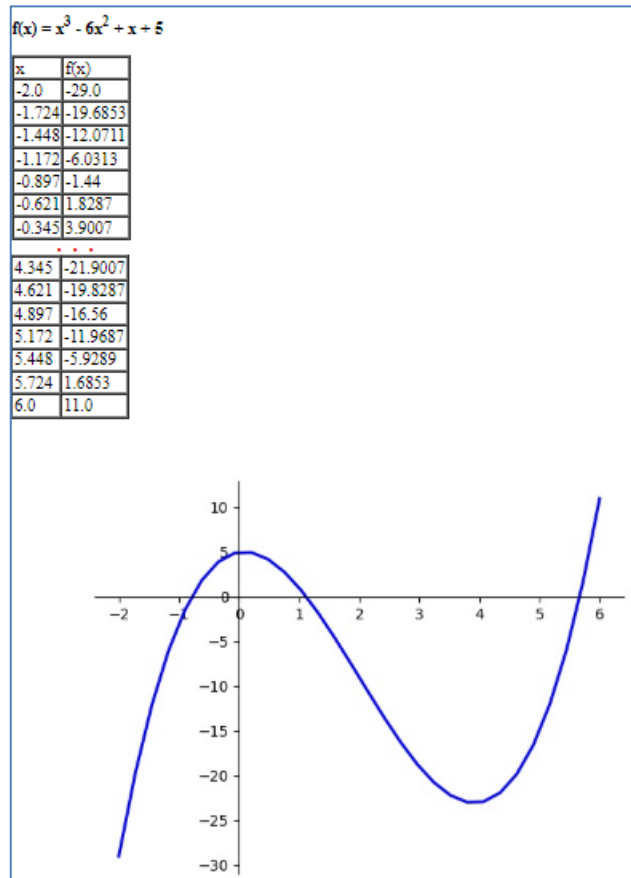


Рисунок 1. Результирующая html-страница

Порядок выполнения работы

1. Создать папку `lab_JINJA`. В этой папке будут размещаться все файлы лабораторной работы.

2. Создать html-шаблон `function_template.html`, который будет использоваться для генерации страницы с функцией. В качестве параметров генерации использовать список значений x (для вывода первого столбца) и список значений y (для вывода второго столбца таблицы значений). В шаблон включить:

- абзац для вывода функции ($f(x) = x^3 - 6x^2 + x + 5$);
- html-таблицу, в которой сначала сформировать заголовок таблицы (первая строка), а затем в цикле сгенерировать все остальные строки.

HTML-код:

`<html>`

```

<head>
  <title> Таблица и график функции </title>
</head>
<body>
  <p>
    <b>f(x) = ...
  </p>
  <table>
    <tr>
      <th>x</th>
      <th>f(x)</th>
    </tr>
    <tbody>
      {% for i in range(len(x)) %}
        <tr>
          <td>{{x[i]}}</td>
          <td>{{y[i]}}</td>
        </tr>
      {% endfor %}
    </tbody>
  </table>
</body>
</html>

```

3. Создать программу function.py, в которой:

```

# Импортировать объект-шаблон из модуля jinja2
from jinja2 import Template

# Описать функцию, в качестве параметра передать значение аргумента x
# функция должна возвращать значение заданной функции, вычисленной от x
def f(...
    ...

# Задать начало a и конец b интервала построения функции,
# количество точек построения.
a = -2
b = 6
n = 30

# Вычислить шаг
h = ...

# Сформировать список со значениями аргумента
x_list = ...

# Сформировать список со значениями функции для
# каждого элемента списка x_list
f_list = ...

# Прочитать шаблон из файла function_template.html
f_template = open('function_template.html','r', encoding='utf-8-sig')
html = f_template.read()
f_template.close()

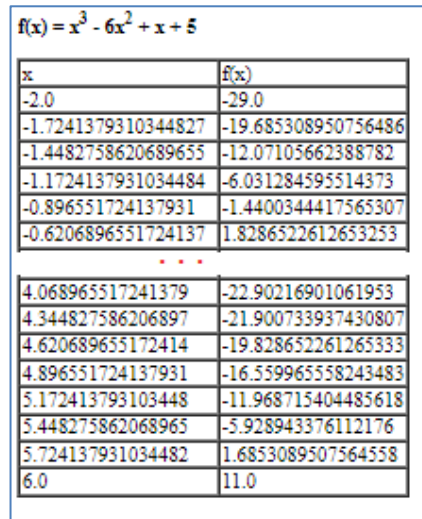
# Создать объект-шаблон
template = Template(html)
# Указать, что в шаблоне будет использована функция len
template.globals["len"] = len

#Создать файл для HTML-страницы
f = open('function.html', 'w', encoding='utf-8-sig')

```

```
# Сгенерировать страницу на основе шаблона
result_html = template....( x = x_list,
                             y = f_list
                           )
# Вывести сгенерированную страницу в файл
f.write(result_html)
f.close()
```

В результате должна получиться страница, показанная на рисунке 2.



$$f(x) = x^3 - 6x^2 + x + 5$$

x	f(x)
-2.0	-29.0
-1.7241379310344827	-19.685308950756486
-1.4482758620689655	-12.07105662388782
-1.1724137931034484	-6.031284595514373
-0.896551724137931	-1.4400344417565307
-0.6206896551724137	1.8286522612653253
...	...
4.068965517241379	-22.90216901061953
4.344827586206897	-21.900733937430807
4.620689655172414	-19.828652261265333
4.896551724137931	-16.559965558243483
5.172413793103448	-11.968715404485618
5.448275862068965	-5.928943376112176
5.724137931034482	1.6853089507564558
6.0	11.0

Рисунок 2. Результат генерации html-страницы

4. Внести изменения в программу (указать, что в шаблоне будет использоваться функция `round()`), а также в шаблоне округлить значения в таблице, как показано на рисунке 1.

5. В программе добавить функцию, которая будет строить график функции по значениям координат x и y , передаваемых в функцию в виде списков. Результат построения сохранить в файл.

```
# В начале программы импортировать модуль для построения графиков
import matplotlib.pyplot as plt

# Определить функции для построения графика по спискам координат x и y
def create_pict(x, y):

    # Построить линию графика, установить для нее цвет и толщину:
    line = plt.plot(x, y)
    plt.setp(line, color="blue", linewidth=2)

    # Вывести 2 оси, установить их в позицию zero:
    plt.gca().spines["left"].set_position("zero")
    plt.gca().spines["bottom"].set_position("zero")
    plt.gca().spines["top"].set_visible(False)
    plt.gca().spines["right"].set_visible(False)

    # Сохранить результат построения в файл:
    plt.savefig("pict.jpg")

    # Вернуть имя созданного файла
    return "pict.jpg"
```

Далее в программе вызвать эту функцию сохранить результат (имя файла) в переменную:

```
name_pict = create_pict(x_list, f_list)
```

А затем передать имя файла в качестве параметра для генерации:

```
result_html = template....( x = x_list,
                             y = f_list,
                             pict = ...
                           )
```

6. В шаблон добавить вывод картинки с графиком (включить переменную с именем картинки в соответствующий тег). В результате должна получиться страница, показанная на рисунке 1.

Самостоятельное задание. Добавьте стили CSS и блоки `<div>...</div>` так, чтобы страница выглядела, как показано на рисунке 3. Форматирование таблицы может быть любым.

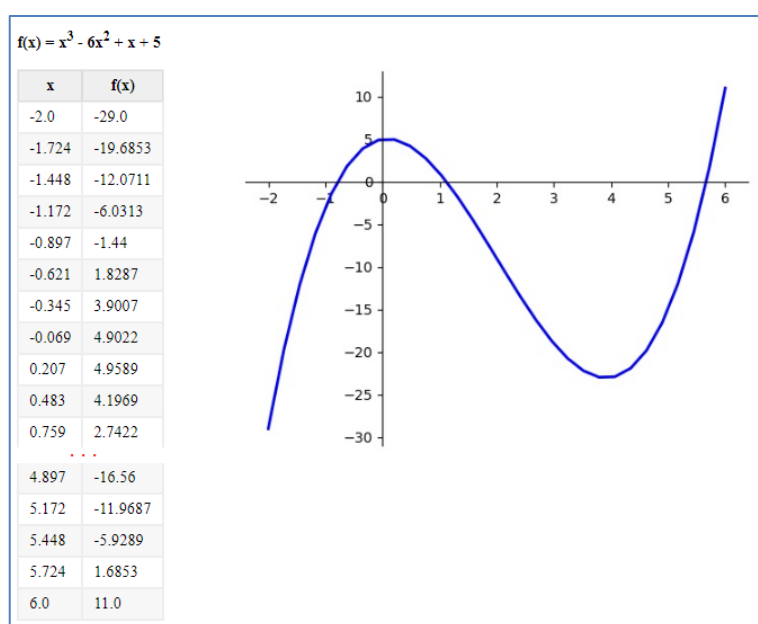


Рисунок 3. Результат генерации html-страницы с CSS стилями

Задание 2. Даны функции:

$$f(x) = x^3 - 6x^2 + x + 5$$

$$y(x) = x^2 - 5x + 1$$

$$z(x) = \frac{1}{x^2 + 1}$$

Создать HTML-страницу, для построения таблиц значений и графиков этих функций (рисунок 4). В окончательном варианте границы блоков убрать (они выведены, чтобы было видно расположение блоков на странице). Страница должна включать:

- блок для выбора функции и настройки параметров построения;
- блок для вывода таблицы значений функции;
- блок для построения графика.

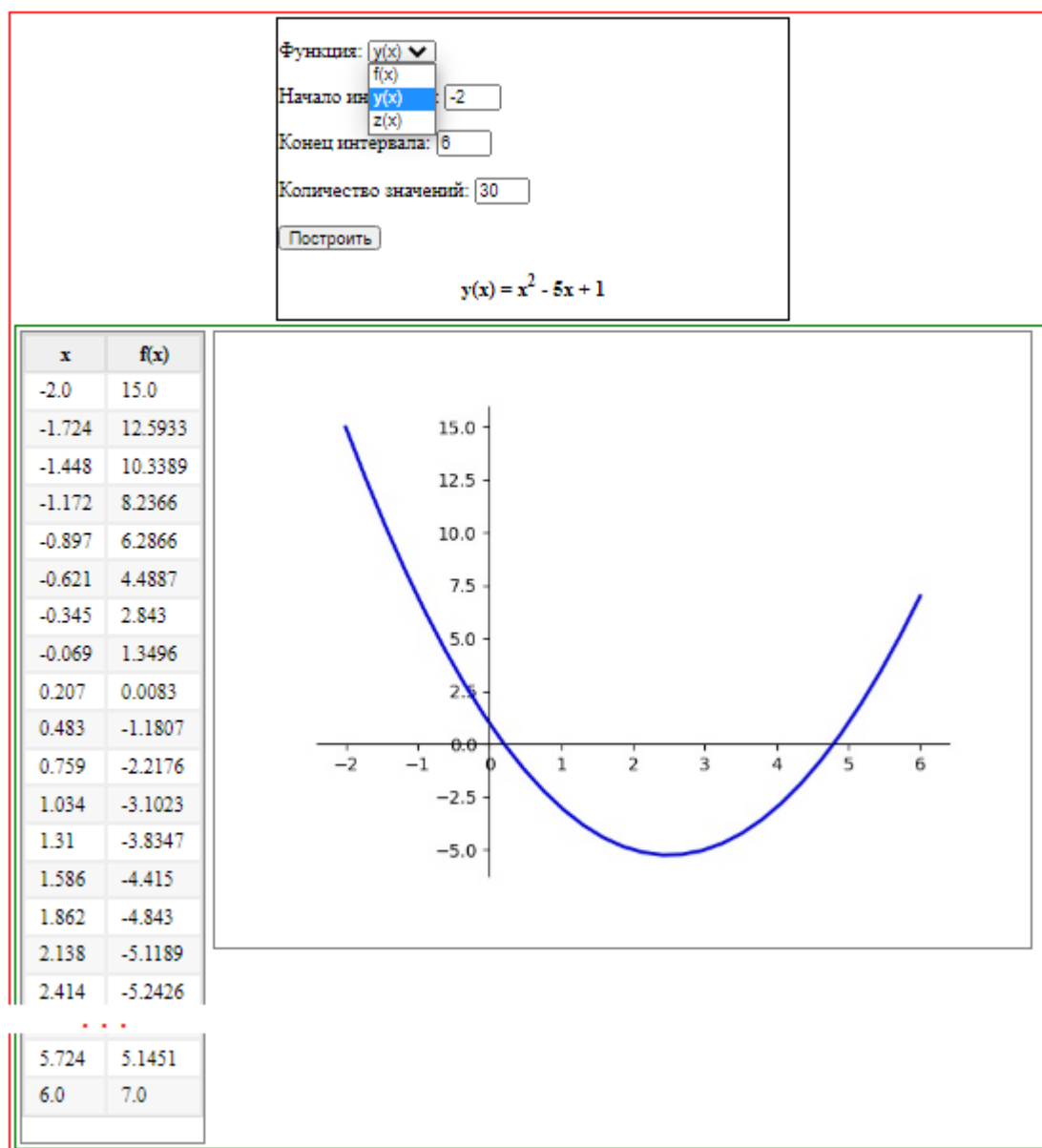


Рисунок 4. Страница для вывода нескольких функций

Порядок выполнения работы

1. Создать html-шаблон `functions_template.html`, который будет использоваться для генерации страницы с выбором функции (структура страницы показана на рисунке 4). В качестве параметров генерации использовать:

- номер варианта функции (`n_var`);
- список с названиями функций (`list_f`);
- начало интервала построения (`a`);
- конец интервала построения (`b`);
- количество значений функции в таблице (`n`);
- список значений x (для вывода первого столбца);
- список значений y (для вывода второго столбца таблицы значений).

Ниже приведен фрагмент шаблона, который располагается в блоке для выбора функции и настройки параметров построения, остальную часть шаблона реализовать самостоятельно.

```

<form>
  <p>Функция:
  <!-- создаем поле со списком для выбора: функция 1, функция 2, функция 3 -->
  <select name=func>
    {% for i in range(count_f) %}
      <!-- если номер варианта совпадает со значением i, устанавливаем
      атрибут selected
      атрибут value для каждой строки будет совпадать с i -->
      {% if n_var == i %}
        <option selected value={{i}}> {{list_f[i]}} </option>
      {% else %}
        <option value={{i}}> {{list_f[i]}} </option>
      {% endif %}
    {% endfor %}
  </select></p>

  <p>Начало интервала: <input type=text name=beg_i value={{a}} size=1></p>
  <p>Конец интервала: <input type=text name=end_i value={{b}} size=1></p>
  <p>Количество значений: <input type=text name=count_p value={{n}}
size=1></p>
  <p><input type=submit value="Построить"></p>
</form>
<p align = center>
  <!-- в зависимости от номера вариант вывести функцию -->
  ...
</p>

```

2. Создать программу `functions.py` на основе программы из предыдущего задания.

В программу внести следующие изменения:

- исправить функцию $f(x)$:

```

def f_x(x, n_var):
    if n_var == 0:
        y = x ** 3 - 6 * x ** 2 + x + 5
    elif n_var == 1:
        y = x ** 2 - 5 * x + 1
    elif n_var == 2:
        y = 1 / (x ** 2 + 1)
    return y

```

- добавить переменную для указания номера варианта функции и список с названиями функций:

```

n_var = 1
list_name_f = ["f(x)", "y(x)", "z(x)"]

```

- добавить параметр `n_var` в вызов функции `f(x)` при формировании списка значений;
- добавить новые параметры в `template.render()`.

3. После генерации должна получиться страница, показанная на рисунке 4.

4. Изменить параметры (начало интервала, конец и пр.), построить другие функции.