Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Факультет Программной инженерии и компьютерной техники

**­­­­­Лабораторная работа №1**

**по дисциплине «Веб-программирование»**

Вариант №2202

Группа: P3212

Выполнил: Балин А. А.

Преподаватель: Харитонова Анастасия Евгеньевна

# Оглавление

[Задание 2](#_Toc146625605)

[Выполнение 3](#_Toc146625606)

[Вывод 14](#_Toc146625607)

# Задание

Разработать PHP-скрипт, определяющий попадание точки на координатной плоскости в заданную область, и создать HTML-страницу, которая формирует данные для отправки их на обработку этому скрипту.

Параметр R и координаты точки должны передаваться скрипту посредством HTTP-запроса. Скрипт должен выполнять валидацию данных и возвращать HTML-страницу с таблицей, содержащей полученные параметры и результат вычислений - факт попадания или непопадания точки в область. Предыдущие результаты должны сохраняться между запросами и отображаться в таблице.

Кроме того, ответ должен содержать данные о текущем времени и времени работы скрипта.

Разработанная HTML-страница должна удовлетворять следующим требованиям:

Для расположения текстовых и графических элементов необходимо использовать блочную верстку.

Данные формы должны передаваться на обработку посредством GET-запроса.

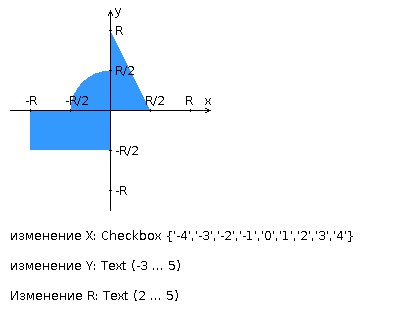
Таблицы стилей должны располагаться в самом веб-документе.

При работе с CSS должно быть продемонстрировано использование селекторов идентификаторов, селекторов псевдоэлементов, селекторов элементов, селекторов классов а также такие свойства стилей CSS, как наследование и каскадирование.

HTML-страница должна иметь "шапку", содержащую ФИО студента, номер группы и новер варианта. При оформлении шапки необходимо явным образом задать шрифт (serif), его цвет и размер в каскадной таблице стилей.

Отступы элементов ввода должны задаваться в пикселях.

Страница должна содержать сценарий на языке JavaScript, осуществляющий валидацию значений, вводимых пользователем в поля формы. Любые некорректные значения (например, буквы в координатах точки или отрицательный радиус) должны блокироваться.



# Выполнение

Код PHP скрипта:

<?php **header**('Access-Control-Allow-Origin: \*');

function **validateY**($y) {

    if (**is\_numeric**($y)) {

        $y = **floatval**($y);

        if (-3 <= $y && $y <= 5) {

            return $y;

        }

    }

    return false;

}

function **validateR**($r) {

    if (**is\_numeric**($r)) {

        $r = **floatval**($r);

        if (2 <= $r && $r <= 5) {

            return $r;

        }

    }

    return false;

}

function **validateX**($x) {

    if (**is\_numeric**($x)) {

        $x = **floatval**($x);

        if (-4 <= $x && $x <= 4) {

            return $x;

        }

    }

    return false;

}

if (**isset**($\_GET['x']) && **isset**($\_GET['y']) && **isset**($\_GET['r'])) {

    $start\_time = **microtime**(true);

    $x = **validateX**($\_GET['x']);

    $y = **validateY**($\_GET['y']);

    $r = **validateR**($\_GET['r']);

    $result = "Failure";

    if ($x !== false && $y !== false && $r !== false) {

        if (**isPointInside**($x, $y, $r)) {

            $result = "Success";

        }

        $time = **number\_format**((**microtime**(true) - $start\_time) \* 1000, 8);

**echo** **json\_encode**(**array**($x, $y, $r, $result, $time));

    } else {

**echo** **json\_encode**(**array**("error" => "Invalid input values"));

    }

}

function **isPointInside**($x, $y, $r)

{

    $ret = false;

    if ($x >= 0 && $y >= 0 && $y <= (-2) \* $x + $r) {

        $ret = true;

    }

    if ($x <= 0 && $y >= 0 && $x ^ 2 + $y ^ 2 <= ($r / 2) ^ 2) {

        $ret = true;

    }

    if ($x <= 0 && $y <= 0 && $x >= -$r && $y >= (-$r / 2)) {

        $ret = true;

    }

    return $ret;

}

Код HTML страницы с каскадными стилями:

<!DOCTYPE *html*>

<html>

  <head>

    <title>Лаб. 1</title>

    <link *rel*="shortcut icon" *href*="../media/favicon.ico" *type*="image/x-icon" />

    <style>

      body {

        display: block;

        min-height: 100vh;

        margin: 0;

        padding: 0;

        text-decoration: 0;

        margin: 0;

        padding: 0;

        font-family: ShantellSans;

        background-position: center;

      }

      .barbie-mode {

        background: **linear-gradient**(#63ecef, #ff22c4);

      }

      .openheimer-mode {

        background-image: **url**("../media/Oppenheimer-first-person-script-01-Mainstyle.webp");

        min-height: 100vh;

        max-height: 100vh;

        background-position: center;

        color: #ffffff;

      }

      @font-face {

        font-family: ShantellSans;

        src: **url**(../css/fonts/ShantellSans-Italic-VariableFont\_BNCE\INFM\SPAC\wght.ttf),

**url**(../css/fonts/ShantellSans-Italic-VariableFont\_BNCE\INFM\SPAC\wght.ttf),

**url**(../css/fonts/static/ShantellSans-Regular.ttf),

**url**(../css/fonts/static/ShantellSans-SemiBold.ttf);

      }

      p,

      h2,

      label {

        text-align: center;

      }

      .mode {

        font-size: 2em;

        font-family: ShantellSans;

        font-style: normal;

      }

      p:hover,

      h2:hover,

      label:hover {

        color: #ff8f0e;

      }

      .covered {

        display: none;

      }

      .header {

        font-family: serif;

      }

      .decart {

        width: 15vw;

        border-radius: 100px;

      }

      .main-container {

        display: flex;

        flex-direction: row;

      }

      .form\_container {

        padding-left: 70px;

      }

      .input-form {

        width: 40vw;

        margin-top: 4vh;

      }

      .input-argument {

        font-size: 20pt;

        font-weight: bold;

        margin-right: 4vh;

      }

      .text-input {

        border: 1px solid #cccccc;

        border-radius: 3px;

        background: #ffffff **!important**;

        height: 15pt;

        width: 30vw;

        color: #cccccc;

      }

      input[type="text"]:hover,

      input[type="text"]:focus {

        color: #4487e5;

        border: 1.5px solid #fffb02;

        border-radius: 4px;

      }

      #form-button {

        margin-top: 5vh;

        width: 35vw;

        color: white;

        background-color: #4487e5;

      }

      .table-container {

        flex-direction: column;

        align-content: center;

      }

      table {

        margin-top: 15px;

        margin-left: 25px;

        margin-right: 15px;

        border-collapse: collapse;

        border-radius: 3px;

        background: #ffffff;

        color: #ff00a6;

        text-align: center;

      }

      td,

      th {

        border: 2px solid #f515b5;

      }

      .argument-column {

        width: 5vw;

      }

      .time-column,

      .result-column {

        width: 10vw;

      }

      p::selection,

      h2::selection,

      label::selection,

      td::selection,

      th::selection {

        color: **rgb**(157, 255, 0);

      }

    </style>

  </head>

  <body *class*="barbie-mode" *onload*="**sendForm**()">

    <h2 *class*="header" *id*="student-info">

      Балин Артем Алексеевич P3212, вариант №2202

    </h2>

    <p *class*="mode" *onclick*="**pic**(1)" *id*="barbie-mode">Barbie mode on</p>

    <p *class*="mode covered" *onclick*="**pic**(2)" *id*="openheimer-mode">

      Oppenheimer mode on

    </p>

    <div *class*="main-container">

      <div *class*="form\_container">

        <form *class*="input-form">

          <p>

            <img *src*="../media/decart.png" *class*="decart" />

          </p>

          <div *class*="input-argument-container">

            <label *class*="input-argument">X: </label>

            <input

*type*="checkbox"

*name*="X-4"

*onchange*="**uncheckboxes**('-4')"

*class*="input-checkbox"

            />

            <label *for*="X-4">-4</label>

            <input

*type*="checkbox"

*name*="X-3"

*onchange*="**uncheckboxes**('-3')"

*class*="input-checkbox"

            />

            <label *for*="X-3">-3</label>

            <input

*type*="checkbox"

*name*="X-2"

*onchange*="**uncheckboxes**('-2')"

*class*="input-checkbox"

            />

            <label *for*="X-2">-2</label>

            <input

*type*="checkbox"

*name*="X-1"

*onchange*="**uncheckboxes**('-1')"

*class*="input-checkbox"

            />

            <label *for*="X-1">-1</label>

            <input

*type*="checkbox"

*name*="X-0"

*onchange*="**uncheckboxes**('-0')"

*class*="input-checkbox"

            />

            <label *for*="X-0">0</label>

            <input

*type*="checkbox"

*name*="X+1"

*onchange*="**uncheckboxes**('+1')"

*class*="input-checkbox"

            />

            <label *for*="X+1">1</label>

            <input

*type*="checkbox"

*name*="X+2"

*onchange*="**uncheckboxes**('+2')"

*class*="input-checkbox"

            />

            <label *for*="X+2">2</label>

            <input

*type*="checkbox"

*name*="X+3"

*onchange*="**uncheckboxes**('+3')"

*class*="input-checkbox"

            />

            <label *for*="X+3">3</label>

            <input

*type*="checkbox"

*name*="X+4"

*onchange*="**uncheckboxes**('+4')"

*class*="input-checkbox"

            />

            <label *for*="X+4">4</label>

          </div>

          <div *class*="input-argument-container">

            <label *class*="input-argument">Y:</label>

            <input

*type*="text"

*class*="text-input"

*name*="Y"

*placeholder*="Y: (-3...5)"

*id*="Y-input"

            />

          </div>

          <div *class*="input-argument-container">

            <label *class*="input-argument">R:</label>

            <input

*type*="text"

*class*="text-input"

*name*="R"

*placeholder*="R: (2...5)"

*id*="R-input"

            />

          </div>

          <input

*type*="submit"

*value*="Send it!"

*onclick*="**sendForm**()"

*id*="form-button"

          />

        </form>

        <p *id*="error-logs"></p>

      </div>

      <div *class*="table-container">

        <table *id*="table-results">

          <tr>

            <th *class*="argument-column">X</th>

            <th *class*="argument-column">Y</th>

            <th *class*="argument-column">R</th>

            <th *class*="result-column">Result</th>

            <th *class*="time-column">In work, ms</th>

            <th *class*="time-column">Received at</th>

          </tr>

        </table>

      </div>

    </div>

  </body>

  <script *src*="../js/client.js"></script>

  <script *src*="../js/formAction.js"></script>

</html>

Код файла client.js:

var audio = new **Audio**("../media/gg.mp3");

function **pic**(x) {

  if (x === 1) {

    document.body.classList = "openheimer-mode";

    document.**getElementById**("barbie-mode").classList = "covered mode";

    document.**getElementById**("openheimer-mode").classList = "mode";

    audio = new **Audio**("../media/gg.mp3");

    audio.**play**();

  }

  if (x === 2) {

    document.body.classList = "barbie-mode";

    document.**getElementById**("barbie-mode").classList = "mode";

    document.**getElementById**("openheimer-mode").classList = "mode covered";

    audio.**pause**();

  }

}

function **uncheckboxes**(x) {

  let id = "X" + x;

  let checkboxes = document.**getElementsByClassName**("input-checkbox");

  [].**forEach**.**call**(checkboxes, (element) => {

    if (element.name !== id) {

      element.checked = false;

    }

  });

}

Код файла formAction.js:

const form = document.**querySelector**(".input-form");

document.**addEventListener**("DOMContentLoaded",(e)=>{

});

form.**addEventListener**("submit", (e) => {

  e.**preventDefault**();

**sendForm**();

});

function **sendForm**(){

  let x = **validateX**();

  let y = **validateY**();

  let r = **validateR**();

  document.**getElementById**("error-logs").innerHTML = "";

  if (**isNumber**(x) && **isNumber**(y) && **isNumber**(r)) {

    let xhr = new **XMLHttpRequest**();

    xhr.**open**(

      "GET",

      "http://127.0.0.1:8000/serv.php?x=" + x + "&y=" + y + "&r=" + r,

      true

    );

    xhr.**onreadystatechange** = function () {

      if (xhr.readyState === 4 && xhr.status === 200) {

        let response = xhr.responseText;

**processResponse**(response);

      }

    };

    xhr.**send**();

  } else {

    if (!**isNumber**(x)) {

      document.**getElementById**("error-logs").innerHTML +=

        "Choose one option for X! ";

    }

    if (!**isNumber**(y)) {

      document.**getElementById**("error-logs").innerHTML +=

        "Type float for Y from -3 to 5! ";

    }

    if (!**isNumber**(r)) {

      document.**getElementById**("error-logs").innerHTML +=

        "Type float for R from 2 to 5! ";

    }

  }

**clearAll**();

}

function **validateFloat**(str) {

  return **parseFloat**(str.**replace**(",", "."));

}

function **validateX**() {

  let checkboxes = document.**getElementsByClassName**("input-checkbox");

  let x = Infinity;

  [].forEach.**call**(checkboxes, (el) => {

    if (el.checked === true) {

      x = **parseInt**(el.name.**replace**("X", ""));

    }

  });

  return x;

}

function **isNumber**(x) {

  return !**isNaN**(x) && **isFinite**(x) && x !== false;

}

function **validateY**() {

  let y = **validateFloat**(document.**getElementById**("Y-input").value);

  if (!(-3 <= y && y <= 5)) {

    y = Infinity;

  }

  return y;

}

function **validateR**() {

  let r = **validateFloat**(document.**getElementById**("R-input").value);

  if (!(2 <= r && r <= 5)) {

    r = Infinity;

  }

  return r;

}

function **clearAll**() {

  [].forEach.**call**(document.**getElementsByClassName**("input-checkbox"), (el) => {

    el.checked = false;

  });

  document.**getElementById**("Y-input").value = "";

  document.**getElementById**("R-input").value = "";

}

function **processResponse**(response) {

  let date = new **Date**();

  let received = date.**toLocaleString**("ru", {

    year: "numeric",

    month: "numeric",

    day: "numeric",

    timezone: "UTC",

    hour: "numeric",

    minute: "numeric",

    second: "numeric",

  });

  let res = JSON.**parse**(response);

  res.**push**(received);

**addRowToTable**(res);

  let savedData = localStorage.**getItem**("data");

  if (savedData) {

    let data = JSON.**parse**(savedData);

    data.**push**(res);

    localStorage.**setItem**("data", JSON.**stringify**(data));

  } else {

    let data = [];

    data.**push**(res);

    localStorage.**setItem**("data", JSON.**stringify**(data));

  }

}

document.**addEventListener**("DOMContentLoaded", function () {

  let savedData = localStorage.**getItem**("data");

  if (savedData) {

    let data = JSON.**parse**(savedData);

    data.**forEach**(function (item) {

**addRowToTable**(item);

    });

  }

});

function **addRowToTable**(data) {

  let table = document.**getElementById**("table-results");

  let newRow = document.**createElement**("tr");

  let xTd = document.**createElement**("td");

  xTd.textContent = data[0];

  let yTd = document.**createElement**("td");

  yTd.textContent = data[1];

  let rTd = document.**createElement**("td");

  rTd.textContent = data[2];

  let resultTd = document.**createElement**("td");

  resultTd.textContent = data[3];

  let timeTd = document.**createElement**("td");

  timeTd.textContent = data[4];

  let receivedTd = document.**createElement**("td");

  receivedTd.textContent = data[5];

  newRow.**appendChild**(xTd);

  newRow.**appendChild**(yTd);

  newRow.**appendChild**(rTd);

  newRow.**appendChild**(resultTd);

  newRow.**appendChild**(timeTd);

  newRow.**appendChild**(receivedTd);

  table.**appendChild**(newRow);

}

# Вывод

В данной лабораторной работе был реализован веб-сервис на PHP, который принимает параметры через GET-запрос.