Министерство образования и науки Российской Федерации

ФГБОУ ВО «Ульяновский государственный технический университет»

Кафедра «Вычислительная техника»

Лабораторная работа №2

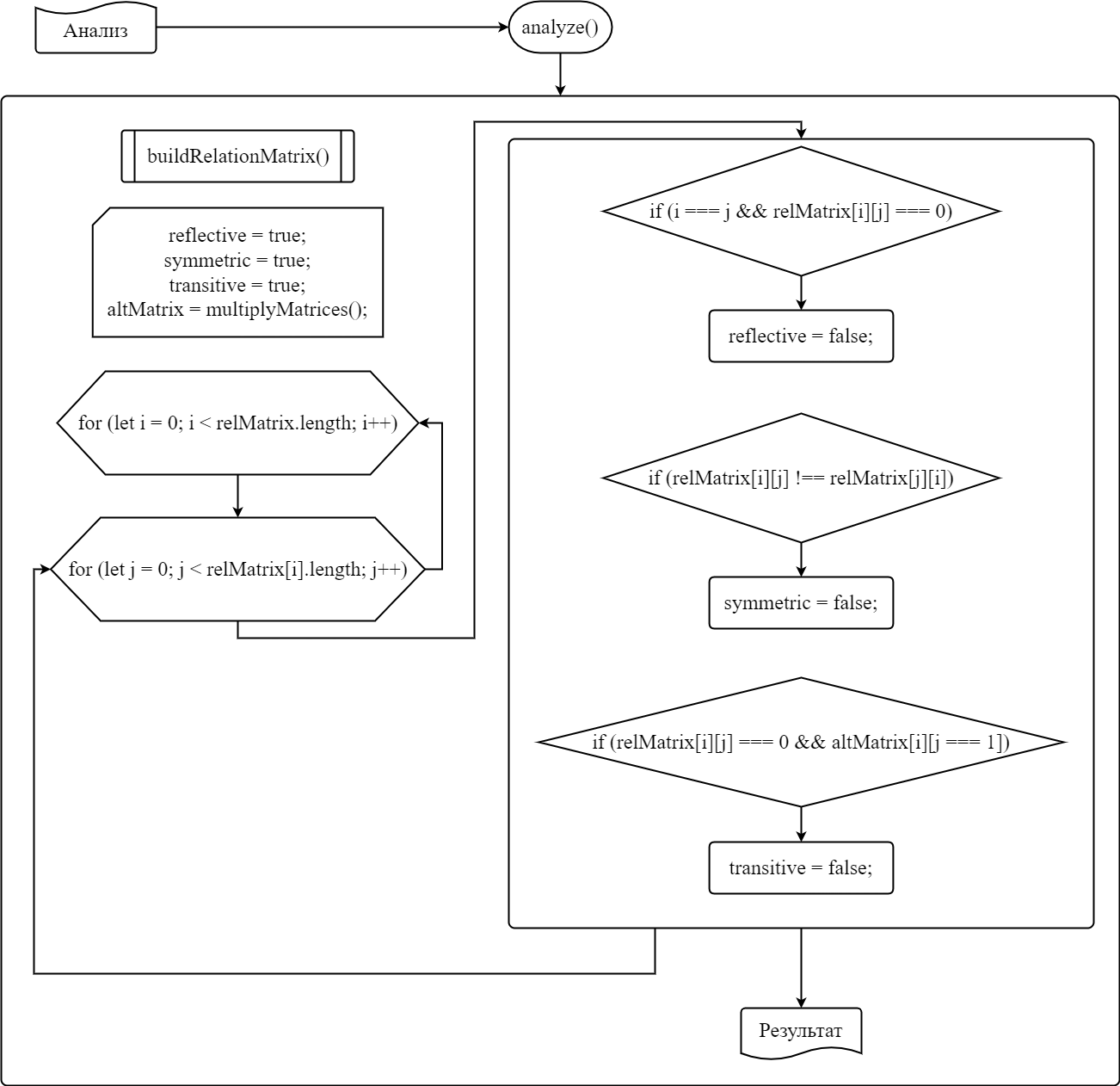
«Отношения»

Выполнил студент

группы ИВТАСбд-11

Нуждов Д. А.

Ульяновск, 2022

**Постановка задачи**Разработать программу, определяющую свойства отношения. Необходимо выполнить проверку корректности ввода.  
  
**Особенности реализации**Работа выполнена в виде сайта. По нажатии на кнопку запускается функция, заданная в параметре onсlick, которая сначала передает управление алгоритму построения матрицы, а потом анализирует ее. В конце результат выводится на экран. В случае ошибки ввода появляется оповещение.  
  
**Блок-схема**

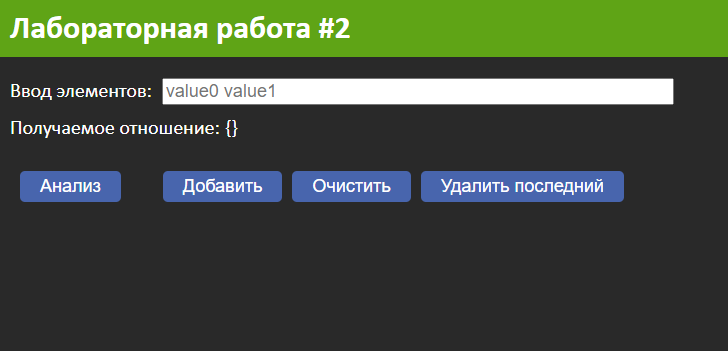
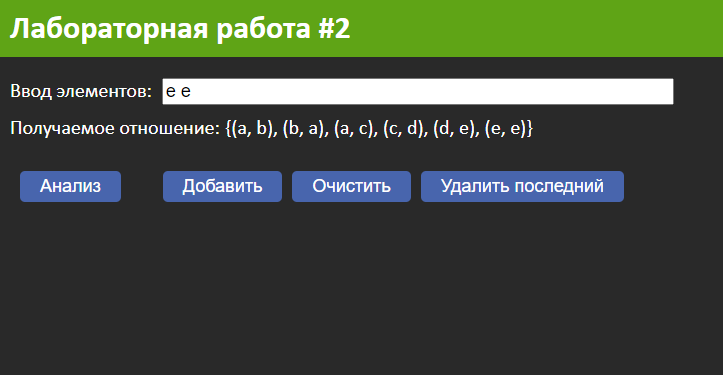
**Интерфейс**Интерфейс сайта состоит из заголовка, поля ввода, поля промежуточного отношения (обновляется при добавлении/удалении элемента из массива), нескольких кнопок и поля результата.

Рис. 1 – внешний вид

Рис. 2 – работа с промежуточным множеством

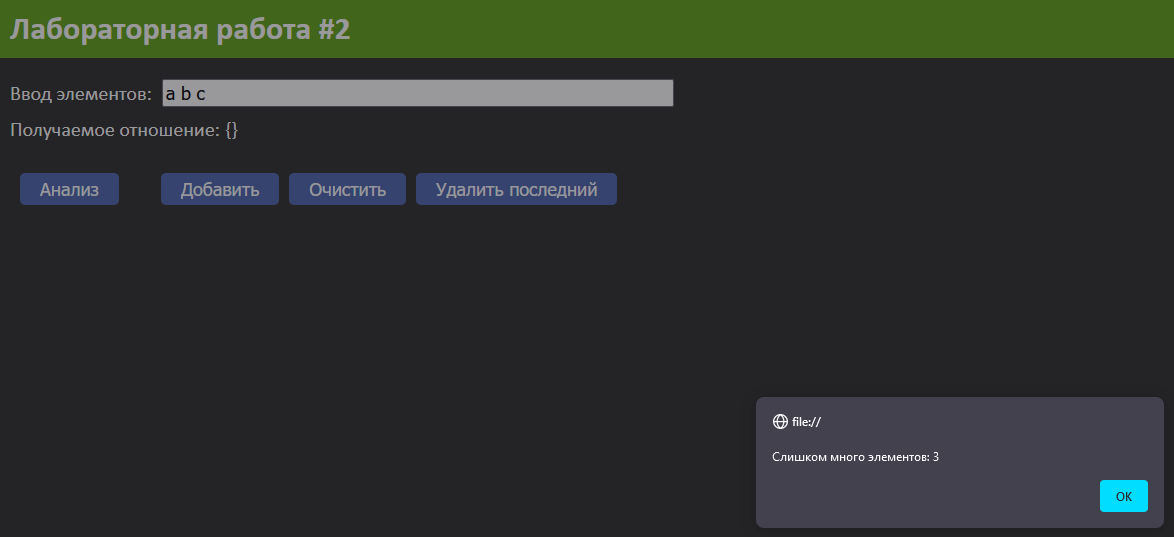
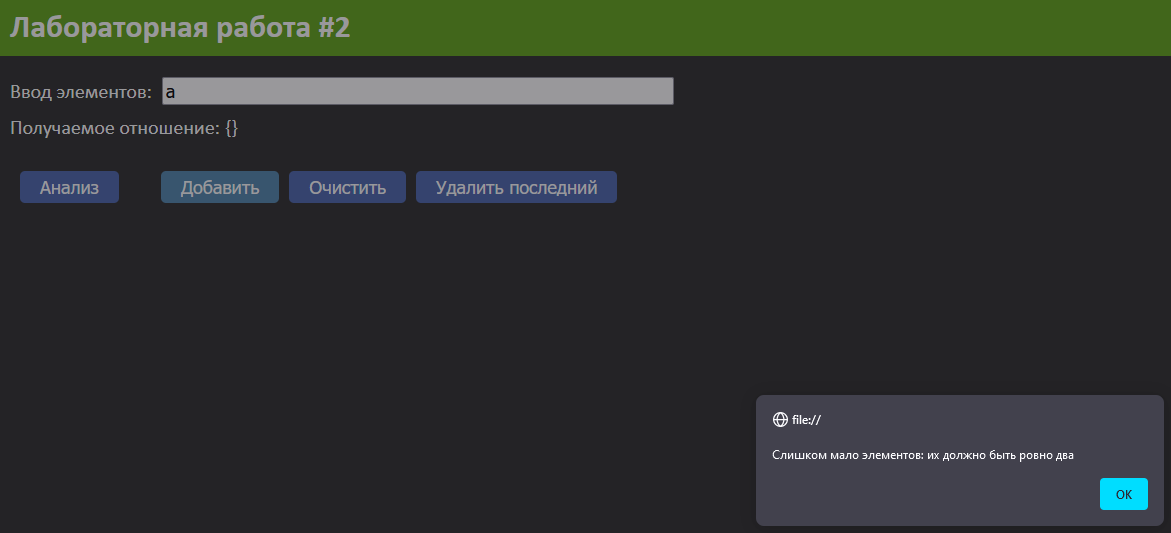
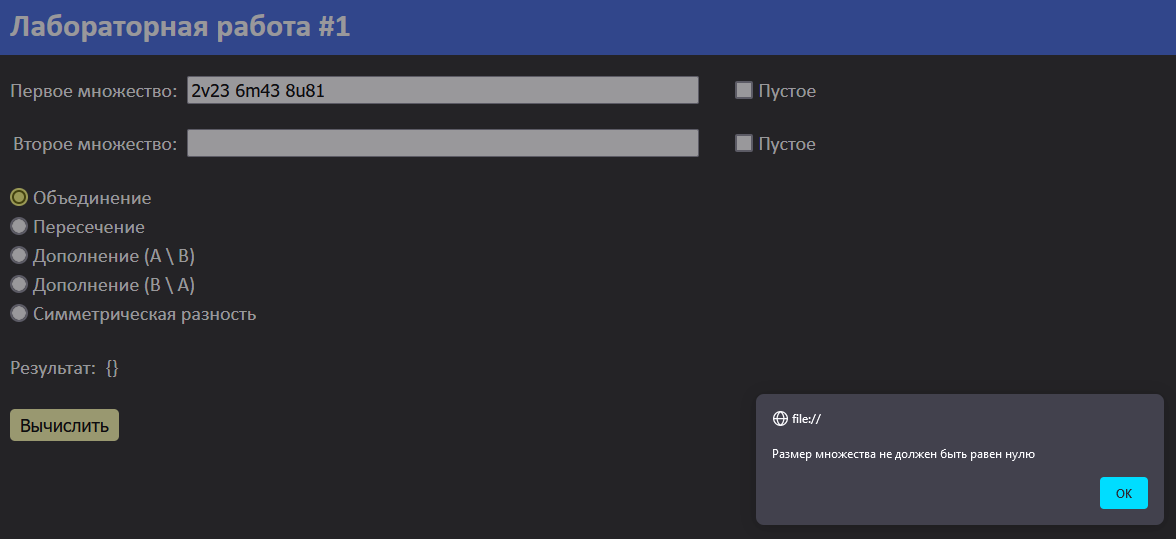
****  ****Рис. 3 – ошибка «Мало элементов»

Рис. 4 – ошибка «Много элементов»

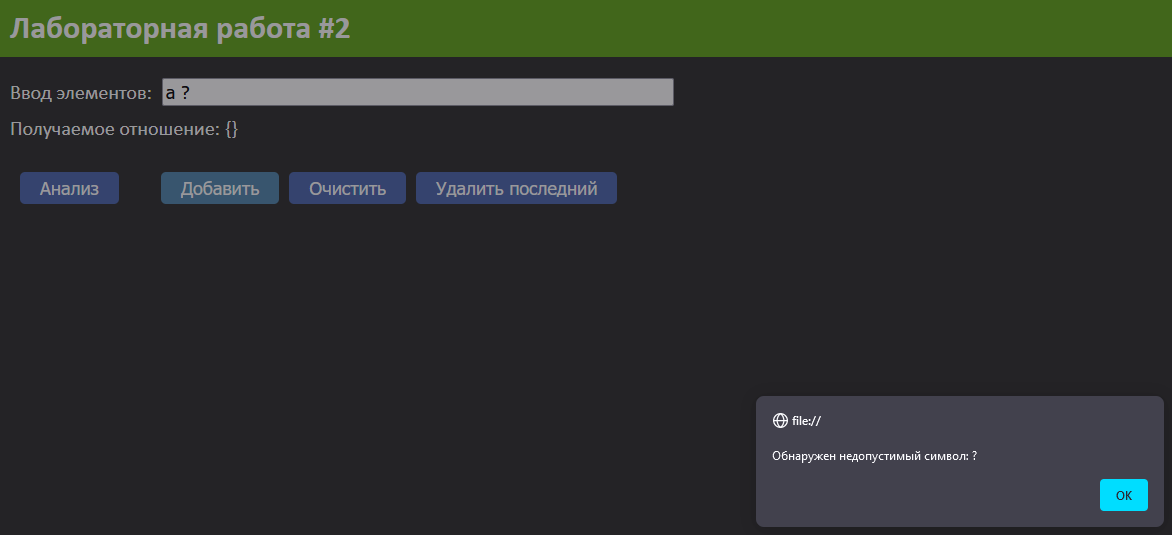
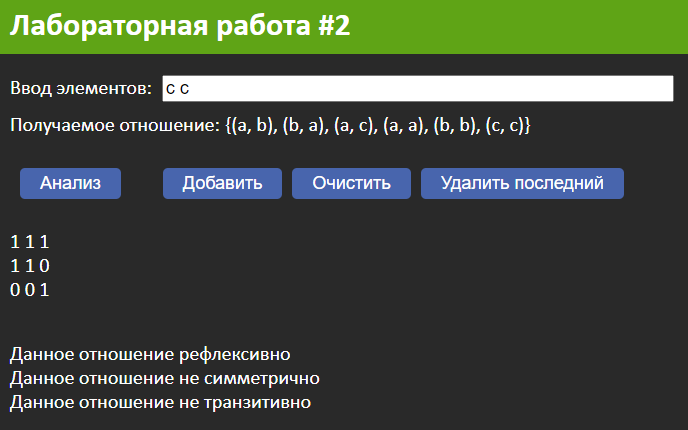
Рис. 5 – ошибка «Недопустимый символ»

Рис. 6 – работа программы

**Выводы о проделанной работе**Программа справляется с поставленной задачей, матрица строится верно, и программа верно анализирует ее. Были получены знания о том, что такое отношения и каким свойствами они обладают. Также закреплены знания о веб-разработке и работе с двумерными массивами (матрицами).  
  
**Список литературы**1) Веб-технологии для разработчиков (https://developer.mozilla.org/ru/docs/Web)  
2) Цикл «Отношения» / Хабр (https://habr.com/ru/post/515014)

**Приложение***index.html*<html lang="ru">

<head>

<meta charset="UTF-8">

<title>Лабораторная работа #2</title>

<link href="styles/style.css" rel="stylesheet">

</head>

<body>

<h1>Лабораторная работа #2</h1>

<div class="if">

<label for="rf">Ввод элементов: </label><input id="rf" type="text" placeholder="value0 value1">

</div>

<div class="prv">

<label>Получаемое отношение: </label><label id="pre">{}</label>

</div>

<div class="cv">

<button id="commit" onclick="analyze();">Анализ</button>

<button id="add" onclick="add();">Добавить</button>

<button id="clear" onclick="clearAll();">Очистить</button>

<button id="clearLast" onclick="clearLast();">Удалить последний</button>

</div>

<div class="rv">

<p id="resultMatrix"></p>

<p id="result"></p>

</div>

</body>

<script src="scripts/mod.js"></script>

</html>

*style.css*body {

margin: 0;

padding: 0;

background-color: #292929;

height: 100%;

}

h1 {

font-family: "Calibri", serif;

background-color: #5fa416;

padding: 10px;

color: white;

}

button {

font-size: 18px;

margin-left: 10px;

border-width: 0;

border-radius: 5px;

background: #4865ad;

color: white;

padding: 5px 20px 5px 20px;

}

button:hover {

background: #4f85ad;

}

button:active {

background: #4f65fd;

}

#commit {

margin-right: 32px;

}

div {

padding-left: 10px;

display: flex;

align-items: baseline;

margin-bottom: 10px;

}

.cv {

margin-top: 32px;

}

label {

margin-right: 5px;

font-size: 20px;

font-family: "Calibri", serif;

color: white;

}

#rf {

width: 512px;

font-size: 18px;

margin-right: 32px;

margin-left: 5px;

}

p {

font-size: 20px;

font-family: "Calibri", serif;

color: white;

}

.rv {

display: flex;

flex-direction: column;

}  
  
*mod.js*let relMatrix = [];

let elements = [];

let e\_set = [];

let restrictedSymbols = ["`", "~", "!", "@", '"', "#", "№", "$", ";", "%", "^", ":", "&", "?", "\*", "(", ")", "-", "\_", "=", "+", "[", "]", "{", "}", "'", "<", ">", ",", ".", "/", "|", "\\"];

function formatView() {

let s = "";

for (let i = 0; i < e\_set.length; i++) {

for (let j = 0; j < e\_set.length; j++) {

if (j !== e\_set.length-1) {

s += relMatrix[i][j] + " ";

} else {

s += relMatrix[i][j];

}

}

s += "<br>";

}

return s;

}

function buildRelationMatrix() {

relMatrix = [];

e\_set = [];

elements.forEach(v => {

e\_set.push(v[0]);

e\_set.push(v[1]);

});

e\_set = Array.from(new Set(e\_set)).sort();

for (let i = 0; i < e\_set.length; i++) {

let matrixRow = [];

for (let j = 0; j < e\_set.length; j++) {

if (elements.some(e => [e\_set[i], e\_set[j]].every((v, idx) => v === e[idx]))) {

matrixRow.push(1);

} else {

matrixRow.push(0);

}

}

relMatrix.push(matrixRow);

}

document.getElementById("resultMatrix").innerHTML = formatView();

}

function getHRV() {

let v0 = "";

elements.forEach((e, idx) => {

let t0 = `(${e[0]}, ${e[1]})`;

if (idx === 0) {

v0 += t0;

} else {

v0 += `, ${t0}`;

}

});

return `{${v0}}`;

}

function strip(v) {

let r = [];

let idx = 0;

if (v.length !== 0) {

while (idx-1 !== v.length) {

let s = "";

while (v[idx] !== " ") {

s += v[idx];

idx++;

if (idx === v.length) {

break;

}

}

if (s !== "") {

r.push(s);

}

idx++;

if (idx === v.length) {

break;

}

}

}

return r;

}

function validate(v0) {

let isAlreadyFailed = false;

restrictedSymbols.some(s => {

if (v0.includes(s)) {

alert(`Обнаружен недопустимый символ: ${s}`);

isAlreadyFailed = true;

}

});

if (!isAlreadyFailed && v0.length > 2) {

alert(`Слишком много элементов: ${v0.length}`);

isAlreadyFailed = true;

}

if (!isAlreadyFailed && v0.length < 2) {

alert(`Слишком мало элементов: их должно быть ровно два`);

isAlreadyFailed = true;

}

return isAlreadyFailed === false;

}

function add() {

let v0 = strip(document.getElementById("rf").value);

if (validate(v0)) {

elements.push(v0);

document.getElementById("pre").innerHTML = getHRV();

}

}

function clearAll() {

elements = [];

document.getElementById("pre").innerHTML = "{}";

}

function clearLast() {

if (elements.length > 0) {

elements.pop();

document.getElementById("pre").innerHTML = getHRV();

}

}

function multiplyMatrices() {

let r = [];

for (let i = 0; i < relMatrix.length; i++) {

r[i] = [];

}

for (let k = 0; k < relMatrix.length; k++) {

for (let i = 0; i < relMatrix.length; i++) {

let e = 0;

for (let j = 0; j < relMatrix.length; j++) {

e += relMatrix[i][j]\*relMatrix[j][k];

}

r[i][k] = e;

}

}

return r;

}

function analyze() {

buildRelationMatrix();

let reflective = true;

let symmetric = true;

let transitive = true;

let altMatrix = multiplyMatrices();

for (let i = 0; i < relMatrix.length; i++) {

for (let j = 0; j < relMatrix[i].length; j++) {

if (i === j && relMatrix[i][j] === 0) {

reflective = false;

}

if (relMatrix[i][j] !== relMatrix[j][i]) {

symmetric = false;

}

if (relMatrix[i][j] === 0 && altMatrix[i][j] === 1) {

transitive = false;

}

}

}

document.getElementById("result").innerHTML = `Данное отношение ${reflective ? "" : "не "}рефлексивно<br>Данное отношение ${symmetric ? "" : "не "}симметрично<br>Данное отношение ${transitive ? "" : "не "}транзитивно`;

}

document.getElementById("rf").addEventListener("keyup", function(event) {

if (event.key === "Enter") {

document.getElementById("add").click();

}

});