Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Ульяновский государственный технический университет»

Кафедра «Вычислительная техника»

Дисциплина «Математическая логика и дискретная математика»

Лабораторная работа №2

«**Свойства отношений**»

Выполнил студент

группы ИВТАПбд-11

Аронов В. В.

Ульяновск, 2022

**Оглавление**

[**Цель Работы 3**](#_Toc104903381)

[**Описание метода работы 3**](#_Toc104903382)

[**Рефлексивность 7**](#_Toc104903383)

[**Симметричность 8**](#_Toc104903384)

[**Кососимметри́чность 9**](#_Toc104903385)

[**Транзитивность 10**](#_Toc104903386)

[**Вывод результата 11**](#_Toc104903387)

[**Вывод 11**](#_Toc104903388)

[**Листинг кода: 12**](#_Toc104903389)

[**Список использованной Литературы 16**](#_Toc104903390)

# **Цель Работы:**

Разработать алгоритм определений свойств отношений на языке JavaScript, определяющую свойства отношения. Формат ввода определяется заданием. Необходимо выполнить проверку корректности ввода отношения.

Определяемые свойства:

* рефлексивность
* симметричность
* кососимметричность
* транзитивность

# **Описание метода работы:**

**Бинарным отношением между множествами A и В,** называется подмножество R прямого произведения A х В. В том случае, когда A = В, мы говорим просто об отношении R на А.

На странице лабораторной работы создаём форму для ввода данных.

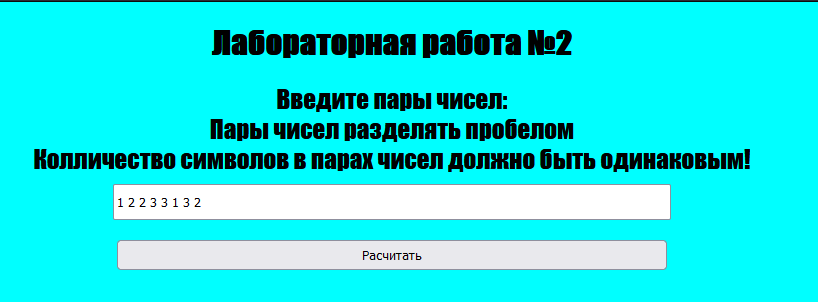
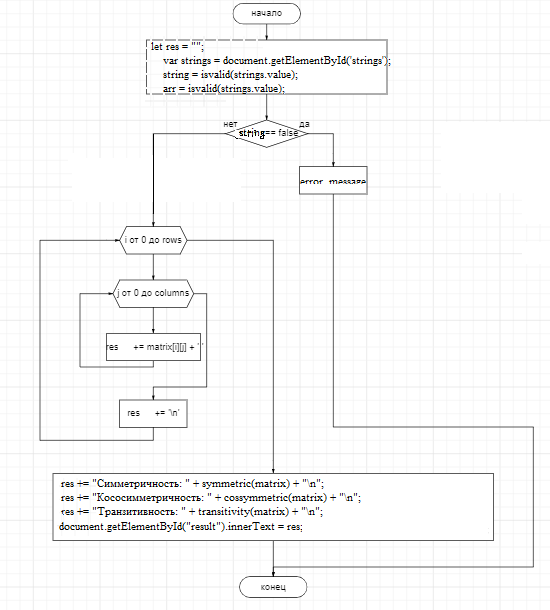


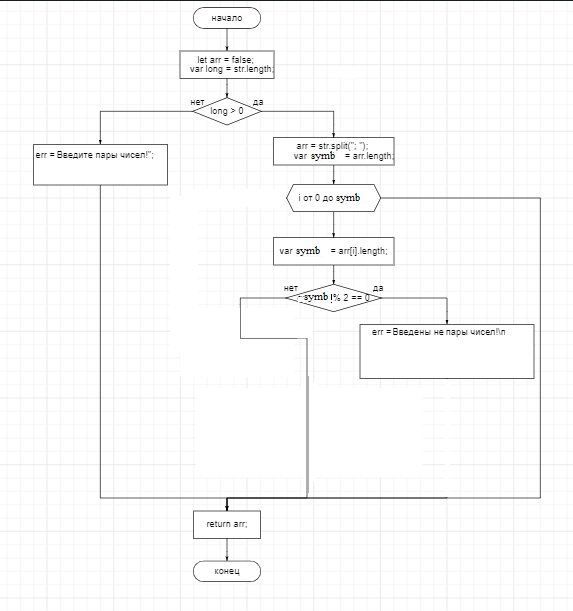
Рис. 1 – форма ввода пар чисел.

По нажатию на кнопку вызывается функция обработки введённых данных. В этой функции вызываются все остальные необходимые для работы функции. А именно, функция проверки ввода данных (валидации), функция построения бинарной матрицы на основании введённых пар, функции проверки свойств отношения уже построенной матрицы.



Блок-схема 1 – основная функция.

В первую очередь производится проверка корректности ввода пар чисел. Условие правильного ввода - это разделение чисел через пробел и отделение пар «;».

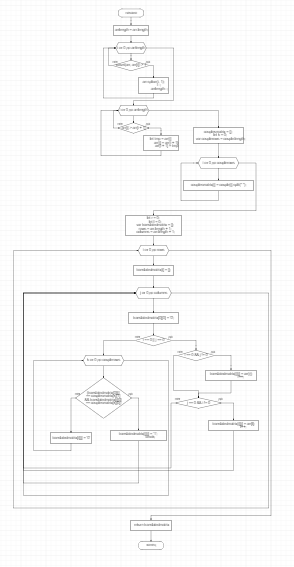


Блок-схема 2 – проверка ввода данных.

Если при воде была допущена ошибка, пользователю выводится сообщение о ней. Иначе строится бинарная матрица, и начинают проверяться свойства бинарного отношения.

**Построение бинарной матрицы**

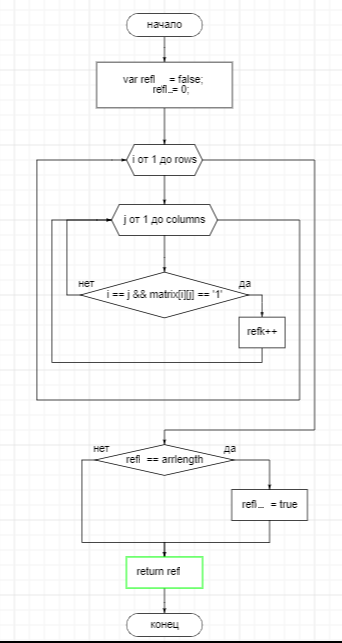
Чтобы проще было определить свойства пар чисел, нужно построить бинарную матрицу по следующему принципу. Сначала в матрицу вносятся по одному элементу пары. Предварительно все эти элементы были внесены в отдельный массив и удалены все повторяющиеся элементы так, что их осталось по одному. После, на пересечении элементов пары ставится «1» или «0» в зависимости от существования пары. Если пара была введена и существует, то ставится «1», иначе – «0». Получим квадратную бинарную матрицу.



Блок-схема 3 – построение бинарной матрицы.

## **Рефлексивность**

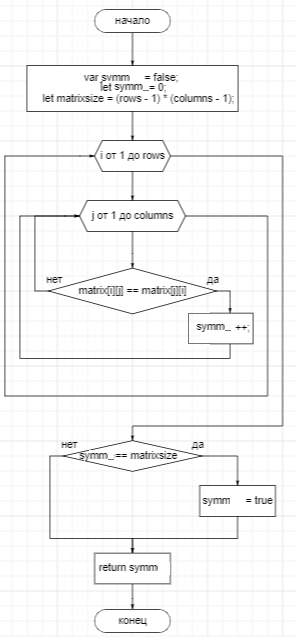
**Рефлексивное** **отношение** в математике - это такое отношение, что любой элемент всегда соотносится с самим собой.



Блок-схема 4 – функция проверки рефлексивности.

## **Симметричность**

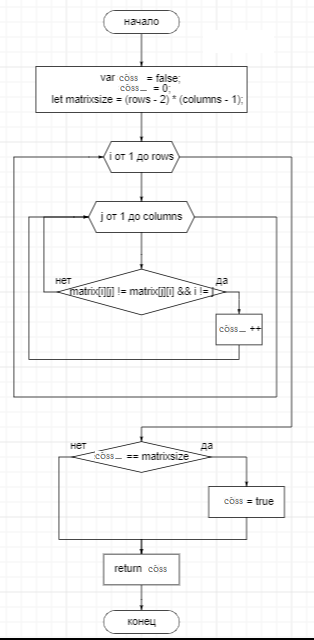
**Симметричное отношение** в математике - это бинарное отношение, не зависящее от порядка аргументов.



Блок-схема 5 – функция проверки симметричности

## **Кососимметри́чность**

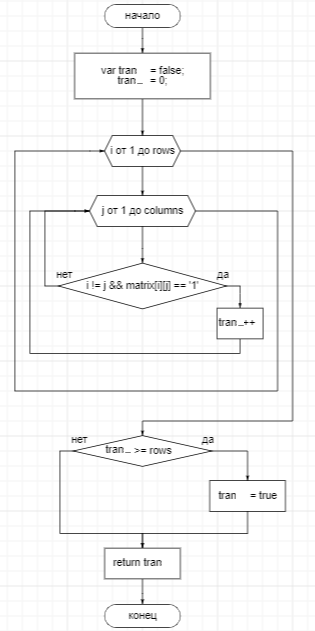
**Кососимметри́чность (или антисимметричность по паре данных аргументов)** — свойство математического объекта, являющегося функцией нескольких аргументов, менять знак (получать множитель −1) при перестановке каких-либо двух аргументов. Например, некоторые квадратные матрицы кососимметричны (антисимметричны) по отношению к перестановке индексов (то есть транспонированию: AT=−A, или Aij=−Aji).



Блок-схема 6 – функция проверки кососимметричности.

## **Транзитивность**

**Транзитивное отношение** в теории множеств - это такое отношение, при котором если один элемент упорядоченной пары соотносится с вторым, а второй элемент другой упорядоченной пары с третьим, то и первый элемент соотносится с третьим и образуют третью упорядоченную пару.



Блок-схема 7 – функция проверки транзитивности.

## **Вывод результата**

После всех операций на странице под кнопкой выводится результат. Если свойство выполняется, то пишется «Есть», иначе – «Нет».

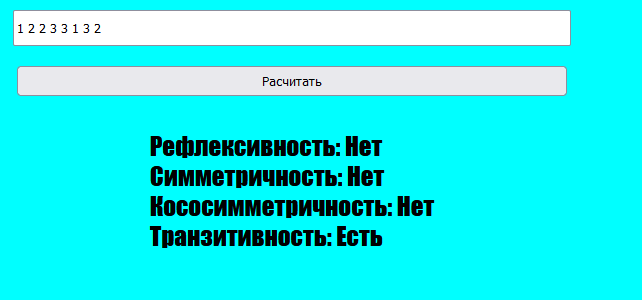


Рис. 2 – вывод результатов

## **Вывод**

В ходе данной лабораторной работы была разработана программа определения свойств бинарного отношения, изучены сами свойства бинарного отношения: транзитивность, симметричность, кососимметричность и рефлексивность.

# **Листинг кода:**

let error\_message = " ";

var matrix, string, rows, columns, arr,

stringsm, arrlength;

function isvalid(*strok*) {

    let arr = false;

    var long = *strok*.length;

    if (long > 0) {

        arr = *strok*.split(" ");

        var symb = arr.length;

        for (let i = 0; i < symb; i++) {

            if (symb % 2 != 0) {

                error\_message = "Введены не пары чисел!";

                arr = false;

                break;

            }

        }

    } else {

        error\_message = "Введите пары чисел!";

    }

    return arr;

}

function count(*arr*, *el*) {

    let k = 0;

    for (let i = 0; i < *arr*.length; i++)

        if (*arr*[i] == *el*)

            k++;

    return k;

}

function MM(*arr*) {

    arrlength = *arr*.length;

    for (let i = 0; i < arrlength; i++) {

        if (count(*arr*, *arr*[i]) > 1) {

*arr*.splice(i, 1);

            i--;

            arrlength--;

        }

    }

    for (let i = 0; i < arrlength; i++) {

        if (*arr*[i] > *arr*[i + 1]) {

            let tmp = *arr*[i]

*arr*[i] = *arr*[i + 1];

*arr*[i + 1] = tmp;

        }

    }

    stringsm = [];

    let k = 0;

    var couplerows = string.length / 2;

    for (let i = 0; i < couplerows; i++) {

        stringsm[i] = [];

        for (let j = 0; j < 2; j++) {

            stringsm[i][j] = string[k];

            k++;

        }

    }

    let r = 0;

    let l = 0;

    var to\_matrix = [];

    rows = *arr*.length + 1;

    columns = *arr*.length + 1;

    for (let i = 0; i < rows; i++) {

        to\_matrix[i] = [];

        for (let j = 0; j < columns; j++) {

            to\_matrix[0][0] = '0';

            if (i == 0 || j == 0) {

                if (i == 0 && j != 0) {

                    to\_matrix[i][j] = *arr*[r];

                    r++;

                }

                if (j == 0 && i != 0) {

                    to\_matrix[i][j] = *arr*[l];

                    l++;

                }

            } else {

                for (k = 0; k < couplerows; k++) {

                    if ((to\_matrix[0][j] == stringsm[k][1] && to\_matrix[i][0] == stringsm[k][0])) {

                        to\_matrix[i][j] = '1';

                        break;

                    } else {

                        to\_matrix[i][j] = '0';

                    }

                }

            }

        }

    }

    return to\_matrix;

}

function reflexivity(*matrix*) {

    var refl = "Нет";

    let refl\_ = 0;

    for (let i = 1; i < rows; i++) {

        for (let j = 1; j < columns; j++) {

            if (i == j && *matrix*[i][j] == '1') {

                refl\_++;

            }

        }

    }

    if (refl\_ == arrlength) {

        refl = "Есть";

    }

    return refl;

}

function symmetric(*matrix*) {

    var symm = "Нет";

    let symm\_ = 0;

    let matrixsize = (rows - 1) \* (columns - 1);

    for (let i = 1; i < rows; i++) {

        for (let j = 1; j < columns; j++) {

            if (*matrix*[i][j] == *matrix*[j][i]) {

                symm\_++;

            }

        }

    }

    if (symm\_ == matrixsize) {

        symm = "Есть";

    }

    return symm;

}

var coss\_;

var matrixsize;

function cossymmetric(*matrix*) {

    var coss = "Нет";

    coss\_ = 0;

    let matrixsize = (rows - 2) \* (columns - 1);

    for (let i = 1; i < rows; i++) {

        for (let j = 1; j < columns; j++) {

            if (*matrix*[i][j] != *matrix*[j][i] && i != j) {

                coss\_++;

            }

        }

    }

    if (coss\_ == matrixsize) {

        coss = "Есть";

    }

    return coss;

}

function transitivity(*matrix*) {

    var tran = "Нет";

    let tran\_ = 0;

    for (let i = 1; i < rows; i++) {

        for (let j = 1; j < columns; j++) {

            if (i != j && *matrix*[i][j] == '1') {

                tran\_++;

            }

        }

    }

    if (tran\_ >= rows) {

        tran = "Есть";

    }

    return tran;

}

function result() {

    let res = "";

    var strings = document.getElementById('strings');

    string = isvalid(strings.value);

    arr = isvalid(strings.value);

    if (string == false) {

        alert(error\_message);

    } else {

        matrix = MM(arr);

        for (let i = 0; i < rows; i++) {

        }

        res += "Рефлексивность: " + reflexivity(matrix) + "\n";

        res += "Симметричность: " + symmetric(matrix) + "\n";

        res += "Кососимметричность: " + cossymmetric(matrix) + "\n";

        res += "Транзитивность: " + transitivity(matrix) + "\n";

        document.getElementById("result").innerText = res;

    }

}

# **Список использованной Литературы**

* Яблонский С.В. Введение в дискретную математику. – М.: Высш. шк., 2003. – 384 с.
* Даккетт, Джон. JavaScript и jQuery. Интерактивная веб-разработка / Джон Дакетт; [пер. с англ. М.А.Райтмана]. - Москва: Эксмо, 2019. - 640 с.: ил. - (Мировой компьютерный бестселлер).