Министерство образования и науки 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 ИВТАСбд-12

Давыдов Д. Д.

Проверил

Игонин А. Г.

Ульяновск, 2022

# Оглавление

[Оглавление 2](#_Toc103937212)

[Цель работы 3](#_Toc103937213)

[Описание метода выполнения 3](#_Toc103937214)

[Вывод 7](#_Toc103937215)

# Цель работы

Требуется написать программу, которая определяет свойства отношения, введенного в виде матрицы nxn. Должны определяться следующие свойства: рефлективность, симметричность, кососимметричность и транзитивность.

# Описание метода выполнения

Отношение рефлексивно, если для любых х из множества А существуют отношение xRx. Отношение симметрично, если для любых х из множества А существуют отношение xRy и yRx. Отношение симметрично, если для любых х и y из множества А существует xRy, из которого следует yRx. Отношение кососимметрично, если для любых х и у из множества А существует xRy и yRx, из которых следует, что x=y. Отношение транзитивно, если для любой тройки элементов x, y, z из А и таких бинарных отношений, как xRy и yRx следует xRz.

Для того чтобы пользователь мог вводить данные необходимо создать html страницу, состоящую из одной мультистрочной формы и кнопок взаимодействия. Код алгоритма был написан на JS.

Пользователь вводит матрицу отношения. После нажатия кнопки «Добавить» происходит алгоритм добавления элементов из формы в массив, имеющая встроенную систему проверки.

function addElements(){

let lines = relationInput.value.split(/\r?\n/);

let testLength = lines.length;

let isOkay = true;

lines = lines.map((el, i) => {

let t = el.split(/(\s+)/).filter( e => e.trim().length > 0).map(e => {

if (+e != 0 && +e != 1)

isOkay = false;

return +e;

});

if (testLength != t.length){

isOkay = false;

}

return t;

});

if(!isOkay){

alert("Была допущена ошибка при вводе, попробуйте ещё раз.");

return;

}

relationMatrix = JSON.parse(JSON.stringify(lines));

console.log(relationMatrix);

}

После этого пользователь может нажать кнопку «Отчистить», чтобы убрать все данные или «Анализ», для начала поиска свойств.

При анализе алгоритм проходит через весь массив с помощью цикла и проверяет условия соблюдения свойств. Если было найдено несовпадение, то флаг этого свойства становится ложным.

function analizeElements() {

let isReflective = true;

let isSymmetric = true;

let isSkewSymmetric = true;

let isTransitive = true;

multipliedMatrix = multiplyMatrix(relationMatrix);

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

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

if (i == j && relationMatrix[i][j] == 0) isReflective = false;

if (relationMatrix[i][j] != relationMatrix[j][i]) isSymmetric = false;

if (i != j){

if (relationMatrix[i][j] == relationMatrix[j][i]) isSkewSymmetric = false;

}

if (relationMatrix[i][j] == 0 && multipliedMatrix[i][j] == 1) isTransitive = false;

}

}

console.log(relationMatrix);

workZone.style.display = "block";

relationReflective.innerHTML = isReflective ? "Отношение рефлексивно" : "Отношение не рефлексивно";

relationSymmetric.innerHTML = isSymmetric ? "Отношение симметрично" : "Отношение не симметрично";

relationSkewSymmetric.innerHTML = isSkewSymmetric ? "Отношение кососимметрично" : "Отношение не кососимметрично";

relationTransitive.innerHTML = isTransitive ? "Отношение транзитивно" : "Отношение не транзитивно";

}

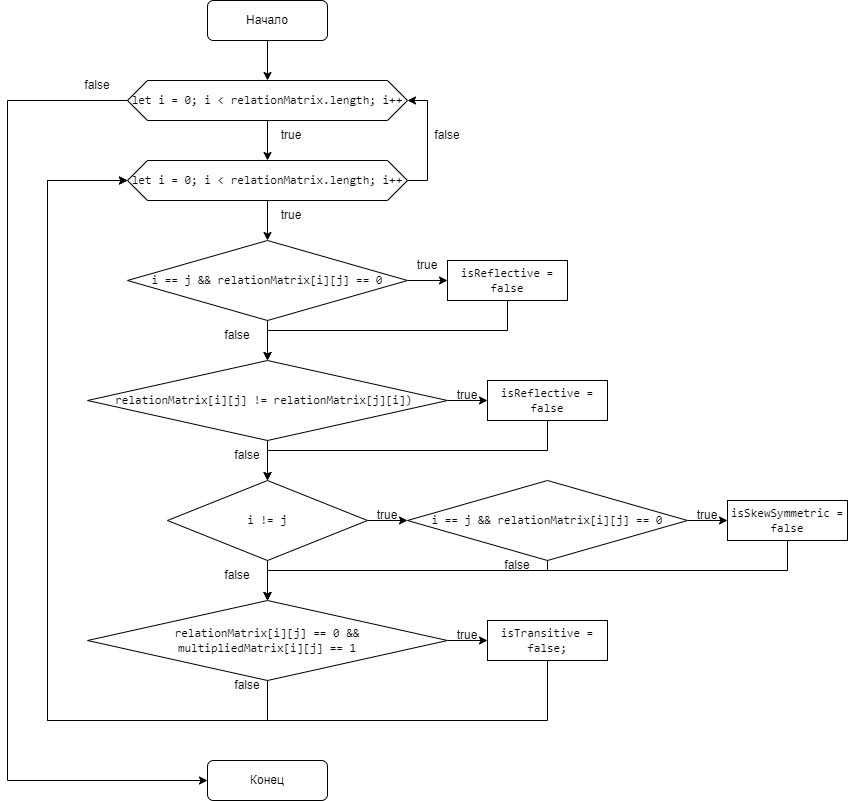


Рисунок 1. Блок-схема анализа свойств.

Таким образом если все элементы диагонали равны 1, то отношение рефлексивно. Если симметрично расположенные элементы относительно главной оси равны, то отношение симметрично. Если все элементы, расположенные симметрично относительно главной оси не равны, то отношение кососимметрично. Если элемент помноженной на себя матрицы на под таким же индексом больше элемента не помноженной, то отношение не транзитивно.

Для определения транзитивности нам необходимо умножить матрицу саму на себя. Для этого нам необходим специальный алгоритм.

function multiplyMatrix(arr){

let cloneArr = JSON.parse(JSON.stringify(arr));

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

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

let t = 0;

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

t += arr[i][j] \* arr[j][k];

}

cloneArr[i][k] = t % 2;

}

}

console.log(cloneArr);

return cloneArr;

}

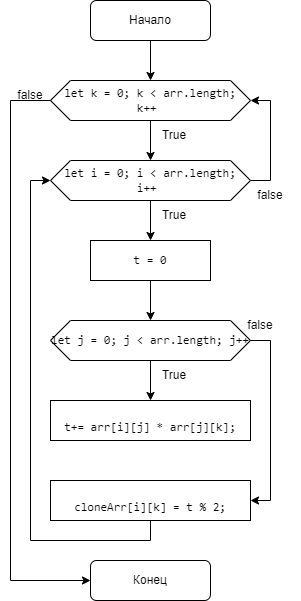


Рисунок 2. Блок-схема умножения.

Графический интерфейс сайта выглядит так:

****

Рисунок 3. Интерфейс сайта.

# Вывод

Выполнив данную лабораторную работу, я создал сайт, который может проверять свойства отношений и также проверять ввод на ошибки.