Министерство образования и науки Российской Федерации ФБГОУ ВО Ульяновский государственный технически университет

Кафедра «вычислительная техника»

**Лабораторная работа №2**

«Свойства отношений»

Выполнила студентка

группы ИВТАСбд-11

Денисова Д.В.

Ульяновск, 2022

**Оглавление.**

1. Цель работы…………………………………………………….…………..2
2. Описание метода выполнения………………………………...…………..2
3. Блок-схема алгоритма………………………………………………….......2
4. Вывод………………………………………………………………………12
5. Список использованной литературы……...……………………………..12
6. **Цель работы.**

Необходимо написать программу, которая будет определять свойства отношения. Формат ввода: пары элементов. Необходимо выполнить проверку корректности ввода отношения. Определяемые свойства: рефлексивность, симметричность, кососимметричность, транзитивность

1. **Описание метода выполнения.**

**Отношение** — бинарное отношение между элементами данного множества, свойства которого сходны со свойствами отношения равенства. **Рефлексивное отношение** — бинарное отношение на множестве, при котором всякий элемент этого множества находится в отношении с самим собой. В математике бинарное отношение R на множестве X называется **симметричным**, если для каждой пары элементов множества (a, b) выполнение отношения aRb влечёт выполнение отношения bRa. Матрица **кососимметричного отношения** несимметрична. В графе антисимметричного отношения могут быть петли, но связь между вершинами отображается только одной дугой. Бинарное отношение R на множестве X называется **транзитивным**, если для любых трёх элементов множества a, b, c выполнение отношений aRb и bRc влечёт выполнение отношения aRc. Для того, чтобы пользователь мог вводить данные необходимо написать небольшую html-структуру, состоящую из одного поля для ввода, полей для вывода и кнопку, по которой произойдут операции над множествами. Код проводимых операций будет написан на JavaScript. Мы получаем наши массивы в html-структуру на PHP и по id передаём в JavaScript.

1. **Блок-схема алгоритма.**

Для проверки на корректность ввода данных пользователем, я пробегаюсь по нашим массивам с определенными условиями и если ввод элементов массива некорректен, то у пользователя высвечивается предупреждение, в котором написано в каком массиве и в каком элементе был допущена ошибка ввода.

1. function checkmass(str){

mass=false;

if(str.length){

mass=str.toString().split(", ");

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

if(mass[i][1]!= " " || mass[i][0]>'9' || mass[i][0]<'0'|| mass[i][2]>'9' || mass[i][2]<'0' || mass[i].length!==3){

error\_text='Ошибка при вводе массива: ' + str + ' в элементе: ' + mass[i];

mass=false;

break;

}

}

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

if(uniqueness(mass, mass[i])>1){

mass.splice(i, 1);

i--;

}

}

}

else {

error\_text="Вы не ввели массив"

}

return mass

}

В первом цикле for мы проверяем является ли первый символ элемента – любой цифрой, второй – пробелом, третий – любой цифрой. Если условие не выполняется, то мы указываем пользователю на ошибку (если массив не введен или если не соблюден формат ввода)

Во втором цикле for мы удаляем повторяющиеся элементы с помощью функции splice.

1. function MaxEl(mass){

let max = 0;

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

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

if(mass[i][0]>max){

max=mass[i][0];

}

if(mass[i][2]>max){

max=mass[i][2];

}

}

}

return max;

}

Эта функция находит максимальный элемент массива и возвращает его значение.

1. function reflexivity(mass){

let res='Не рефлексивно';

let count = 0;

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

if(mass[i][0]==mass[i][2]){

count++;

}

}

if(count==MaxEl(mass)){

res='Рефлексивно';

}

return res;

}

Эта функция определяет свойство рефлексивности отношения. В цикле for она проверяет находятся ли элементы множества в отношениях с самими собой. Затем через условие проверяется все ли элементы множества находятся в отношениях с самим собой, если да, то отношение рефлексивно, и функция возвращает результат.

4. function symmetry(mass){

let el\_mass\_0=0;

let el\_mass\_2=0;

let res='Не симметрично';

let count = 0;

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

el\_mass\_0=mass[i][0];

el\_mass\_2=mass[i][2];

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

if(mass[j][0]==el\_mass\_2 && mass[j][2]==el\_mass\_0){

count++;

}

}

}

if(count==mass.length){

res='Симметрично';

}

return res;

}

Эта функция определяет свойство симметричности отношения. В цикле for она проверяет для каждой ли пары элементов множества (a, b) выполнение отношения aRb влечёт выполнение отношения bRa. Затем через условие проверяется все ли пары прошли это условие, если да, то отношение симметрично, и функция возвращает результат.

5. function antisymmetry(mass){

let el\_mass\_0=0;

let el\_mass\_2=0;

let res='Кососимметрично';

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

el\_mass\_0=mass[i][0];

el\_mass\_2=mass[i][2];

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

if(mass[j][0]==el\_mass\_2 && mass[j][2]==el\_mass\_0 && mass[j][0]!=mass[j][2]){

res='Не кососимметрично';

break;

}

}

}

return res;

}

Эта функция определяет свойство кососимметричности отношения. В цикле for она проверяет для каждой ли пары элементов множества a,b выполнение отношений aRb и bRa влечёт a=b, или, то же самое, выполнение отношений aRb и bRa возможно только для равных a и b. Затем если условие не пройдено, то функция не кососимметрична и цикл заканчивается. Далее функция возвращает результат.

6. function transitivity(mass){

let res='';

let el\_mass0=0;

let el\_mass2=0;

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

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

if(mass[i][2]==mass[j][0] && mass[i][0]!=mass[j][2]){

el\_mass0=mass[i][0];

el\_mass2=mass[j][2];

for(let a=0; a<mass.length; a++){

if(mass[a][0]==el\_mass0 && mass[a][2]==el\_mass2){

res='Транзитивно';

break;

}

}

if(res!='Транзитивно'){

res='Не транзитивно';

break;

}

}

}

if (res=='Не транзитивно'){

break;

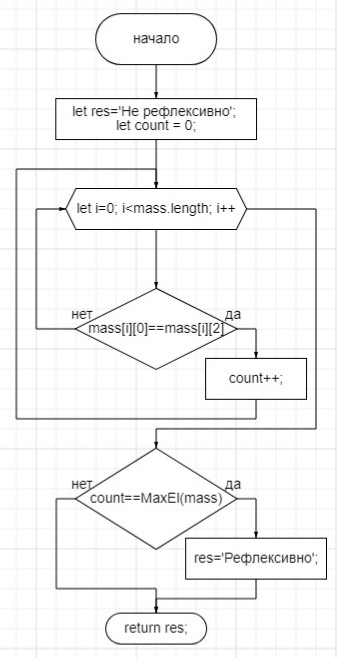
}

}

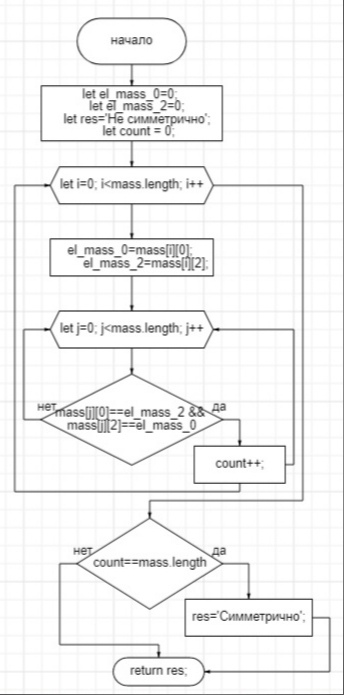
return res;

}

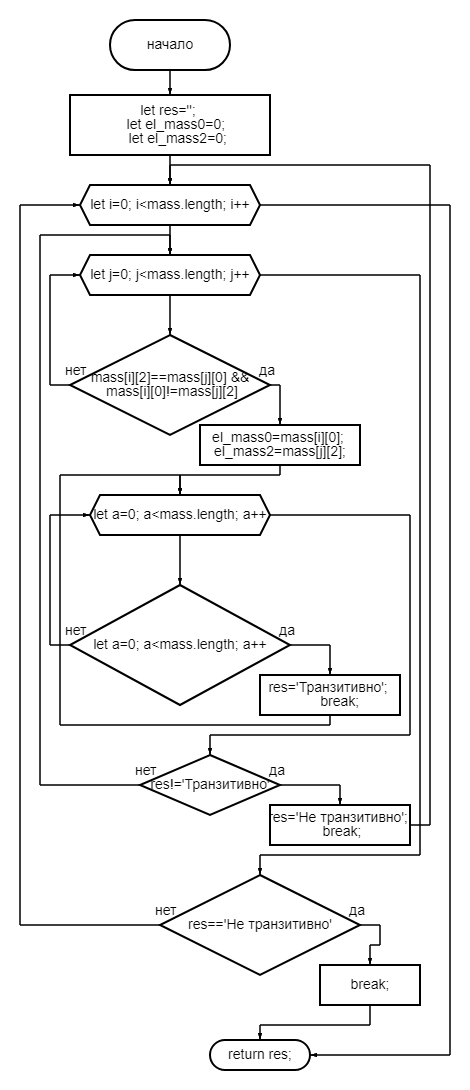
Эта функция определяет свойство транзитивности отношения. В цикле for она проверяет, если для любых трёх элементов множества a, b, c выполнение отношений aRb и bRc влечёт выполнение отношения aRc, то отношение транзитивно. Далее функция возвращает результат.

7. Блок-схема функции reflexivity()-рефлексивность.

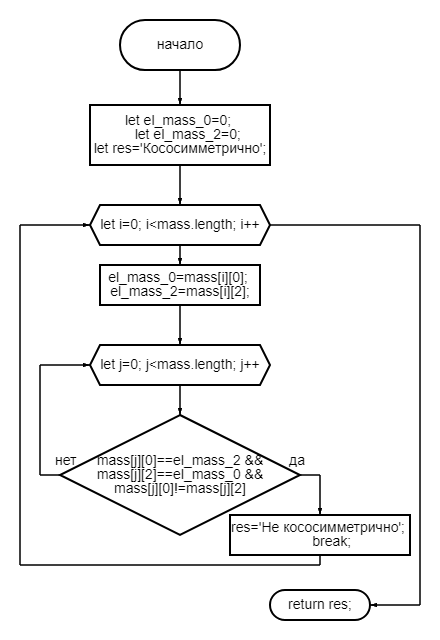
8. Блок-схема функции symmetry()-симметричность:



9. Блок-схема функции transitivity() - транзитивность:



10. Блок-схема функции antisymmetry() - кососимметричность:



11. function calculation\_l2() {

let result="";

var m1=document.getElementById('mass')

if((mass\_1=checkmass(m1.value))==false){

alert(error\_text);

}

if((mass\_1=checkmass(m1.value))!=false){

result = 'Результат: ' + '\n' +reflexivity(mass\_1) + ', ' + symmetry(mass\_1) + ', ' + antisymmetry(mass\_1)+', ' + transitivity(mass\_1) +'\n';

}

if((mass\_1=checkmass(m1.value))!=false) {

document.getElementById('outresult\_lab2').innerText = result;

}

}

В этой функции мы сначала вызываем функцию для проверки на корректность ввода массивов, если он выведен некорректно, то выводим на экран пользователю нашу переменную, в которой хранится информация об ошибке. Затем вызываем наши функции, собираем весь результат операций в одну строку и выводим ее.

1. **Вывод.**

Программа протестирована, работает успешно. Программа определяет свойства отношений, а именно рефлексивность, симметричность, кососимметричность и транзитивность

1. **Список использованной литературы.**

<https://www.osp.ru/winitpro/2010/06/13004157> - свойства отношений

<https://learn.javascript.ru/> - Современный учебник JavaScript