Министерство образования и науки Российской Федерации

ФГБОУ ВО Ульяновский государственный технический университет

Кафедра «Вычислительная техника»

Лабораторная работа №3

«Функции»

Выполнил студент

группы ИВТАСбд-11

Абросимов К. С.

Ульяновск, 2022

**Цель работы**

Необходимо написать программу, которая будет определять – является ли введённое отношение функцией. Отношения вводятся по парам элементам. Также вводятся все элементы двух множеств. При компиляции программа должна проверить корректность ввода данных пользователя.

**Описание метода выполнения**

Функцией, определенной на множестве А и со значениями на множестве В, называется бинарное отношение, при котором каждый элемент из множества А связан с единственным элементом множества В. Другими словами, для каждого а из А существует ровно одна пара из отношения вида (а, Ь).

Код проводимых операций будет написан на файле PHP, но кодом с HTML. PHP - это язык программирования, специально разработанный для написания web-приложений (сценариев), исполняющихся на Web-сервере. PHP взаимодействует с html.

Воспользуемся уже созданным файлом с прошлой работы, чтобы установить ссылку на другую лабораторную работу.

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Лабораторные работы</title> - основной заголовок  
 <script type="text/javascript" src="/scripts/scripts.js"></script>  
</head>  
<body>  
 <h1> Мои лабораторные работы </h1>   
 <a href ="mldmLab1.php">Первая лабораторная работа по МЛиДМ</a><br>  
 <a href ="mldmLab2.php">Вторая лабораторная работа по МЛиДМ</a> - ссылки к другим лабораторным работам

<a href ="mldmLab3.php">Третья лабораторная работа по МЛиДМ</a>  
</body>  
</html>

Для того, чтобы пользователь мог вводить данные необходимо написать небольшую html-структуру, состоящую поля для ввода (input), полей для вывода и кнопки, по которой произойдут операции над множествами. Также для полного вывода операций нужно создать значение для вывода.

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Лабораторная работа 3</title>  
 <script type="text/javascript" src="/scripts/Scriptslab3.js"></script>  
</head>  
<body>  
<h1> Лабораторная работа №3 </h1>  
<form>  
 <table>  
 <tr>  
 <td> Множество A </td>  
 <td> <input type="text" id="mass1" value="" size="50"/></td>  
 </tr>  
 <tr>  
 <td> Множество B </td>  
 <td> <input type="text" id="mass2" value="" size="50"/></td>  
 </tr>  
 <tr>  
 <td> Пары элементов </td>  
 <td> <input type="text" id="mass3" value="" size="50"/></td>  
 </tr>  
 <tr>  
 <td colspan="2"> <input type="button" value="Сделать расчёт" onclick="rasschetfunction();"/></td>  
 </tr>  
 </table>  
  
</form>  
<div id ="Matrix"></div>  
<div id ="outResult"></div>  
  
</body>  
</html>

Сначала самая большая функция “rasschetfunction” создаёт переменные для считывания информации, введённое пользователем. Но они не являются массивом, а значением. После преобразования значения в массивы, проверяем массив на правильность ввода. После несколько циклов на создание двумерного массива - бинарного отношения из пары элементов.

function rasschetfunction()  
{  
 A = ***document***.getElementById('mass1');  
 B = ***document***.getElementById('mass2');  
 var couples = ***document***.getElementById('mass3'); - берём значения, введённые пользователем  
 A = normalarr(A.value);  
 B = normalarr(B.value);  
 const setA = new ***Set***(A); - создаём «Сеты» только для функции проверки ввода  
 const setB = new ***Set***(B);  
 couples = normalarr(couples.value); - преобразовываем значения в массивы  
 let full\_result = "";  
 if (validate(setA, setB, couples) == false) { - проверка ввода  
 alert(***error\_text***);  
 }  
 else {  
 var matrix = []; - будущее бинарное отношение  
 var flagA = true;  
 var flagB = true; - эти «флаги» будут отвечать за вывод информации о том, является ли отношение функцией  
 for (let i = 0; i < A.length; i++) {  
 matrix[i] = [];  
 for (let j = 0; j < B.length; j++) {  
 matrix[i][j] = 0; - массив пустой, его нужно заполнить нулями, чтобы с ним работать  
 }  
 }  
 var coarr = couples.slice(); - копируем массив пар элементов  
 for (let i = 0; i < couples.length; i++) {  
 for (let j = 0; j < couples.length; j++) {  
 if (couples[i][0] == A[j]) {  
 couples[i] = 0; - присваиваем первому символу элемента (кроме пробела), цифру (будущее положение в массиве)  
 couples[i] += j;  
 break;  
 }  
 }  
 for (let j = 0; j < couples.length; j++) {  
 if (coarr[i][1] == B[j]) {  
 couples[i] += '' + j; - теперь со вторым символом  
 break;  
 }  
 }  
 }  
 for (let i = 0; i < couples.length; i++) {  
 matrix[couples[i][0]][couples[i][1]] = 1; - присваиваем подготовленной матрице «положение» пар элементов в виде единиц  
 }  
 let outmatrix = ""; - копируем матрицу, чтобы просто вывести её на сайте  
 for (let i = 0; i < A.length; i++) {  
 for (let j = 0; j < B.length; j++) {  
 outmatrix += matrix[i][j] + ' ';  
 }  
 outmatrix += '\n';  
 }  
 ***document***.getElementById('Matrix').innerText = "Матрица отношения:\n" + outmatrix +  
 "A - строки, B - столбцы"; вывод полученной матрицы

Следующие несколько циклов будут определять, является ли функцией отношение. Но также стоит учитывать, что отношение может быть функцией как А к В, так и В к А.

let countrow = 0; - счёт строки  
 let countheight = 0; - счёт столбца  
 for (let i = 0; i < A.length; i++) {  
 for (let j = 0; j < B.length; j++) {  
 countrow += matrix[i][j];  
 }  
 if (countrow >= 2) { - элемент А должен иметь связь только с одним элементов В  
 flagA = false;  
 break;  
 } else if (countrow == 0) { - каждый элемент А должен иметь связь  
 flagA = false;  
 break;  
 } else {  
 countrow = 0;  
 }  
 }

Такой же принцип и с В  
 for (let i = 0; i < A.length; i++) {  
 for (let j = 0; j < B.length; j++) {  
 countheight += matrix[j][i];  
 }  
 if (countheight >= 2) {  
 flagB = false;  
 break;  
 } else if (countheight == 0) {  
 flagB = false;  
 break;  
 }  
 else {  
 countheight = 0;  
 }  
 }   
 if (flagA) { - вывод полученных результатов  
 full\_result += "Отношение является функцией А в B\n";  
 }else {  
 full\_result += "Отношение не является функцией А в B\n";  
 }  
 if (flagB) {  
 full\_result += "Отношение является функцией B в A";  
 }else {  
 full\_result += "Отношение не является функцией B в A";  
 }  
 ***document***.getElementById('outResult').innerText = "\n" + full\_result;  
 }  
}

Функция needmass просто убирает создаёт нормальный массив и убирает лишние пробелы, которые могут засчитываться как отдельный элемент.

function normalarr ( str )  
{  
 var mass = str.split(" ");  
 for (let i = 0; i < mass.length; i++) {  
 if (mass[i] == "") {  
 mass.splice(i, 1);  
 }  
 }  
 return mass;  
}

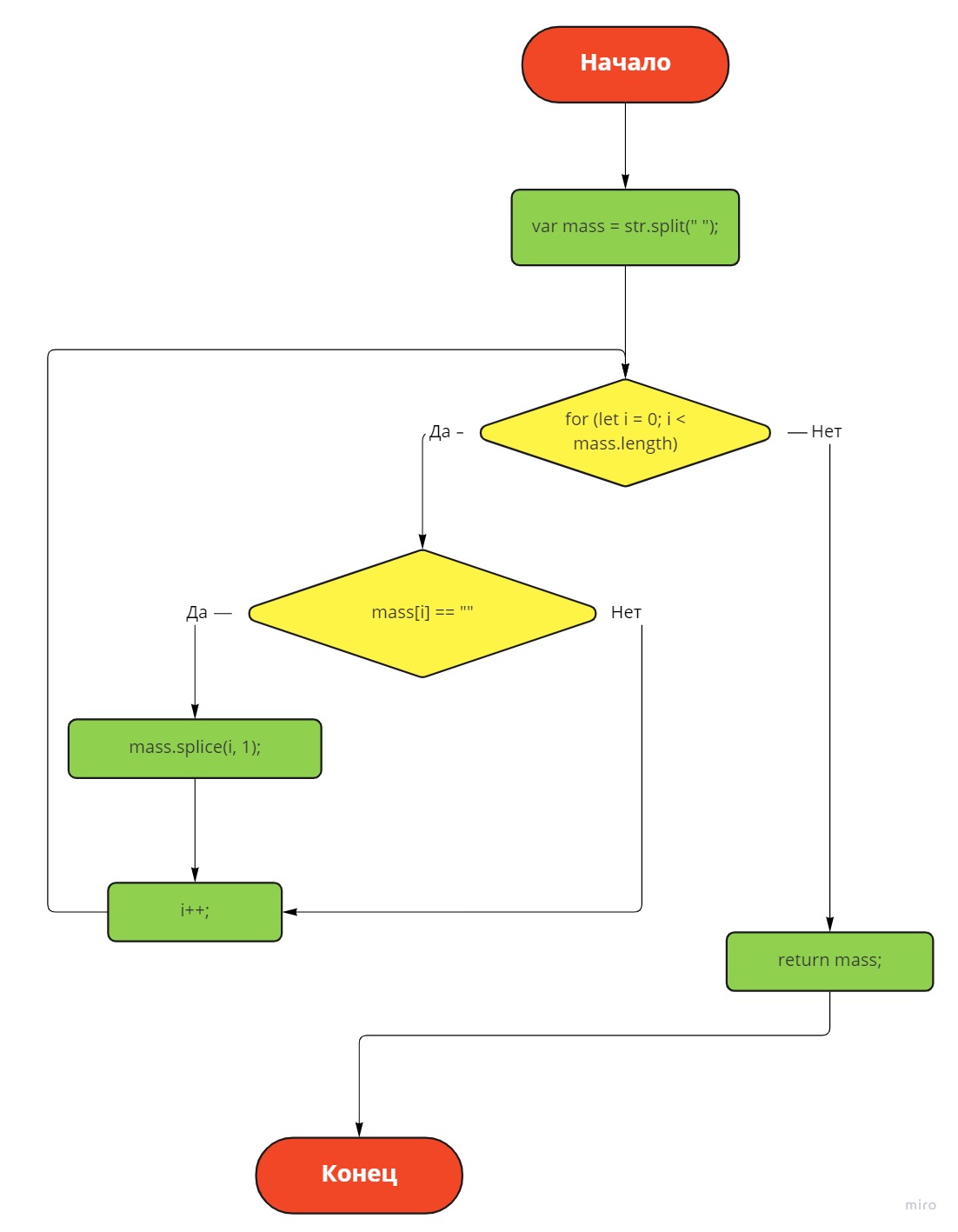


Рис. 1 - функция needmass

Функция validate проверяет несколько условий:

* Имеются ли у множеств А и В те элементы, которые есть у массива с парами элементов. Вместе с этим проверяется, записаны ли именно пары элементов с двумя символами.
* Имеют ли множества А, В и пар элементов более одного элемента.

function validate(A, B, couples)

{

for(elem of couples) {

if(!(A.has(elem[0]) && B.has(elem[1])) || elem.length != 2) {

error\_text = "Пары элементов должны быть вида Ai, Bi ... )";

}

}

if (A.size <= 1) {

error\_text = "Не менее 2 элементов в множестве A\n";

}

else if (B.size <= 1) {

error\_text = "Не менее 2 элементов в множестве B\n";

}

else if (couples.length <= 1) {

error\_text = "Не менее двух пар элементов";

}

if(error\_text) {

return false;

}else {

return true;

}

}

**Выводы**

В результате проведения работы была изучена теория об определении функции в бинарных отношениях.

# **Список литературы**

(б.д.).

*studfile*. (б.д.). Получено из 2. Бинарные отношения: https://studfile.net/preview/4229244/