Минестерство образования и науки Российской Федерации

ФГБОУ ВО Ульяновский государственный технический университет

Кафедра «вычислительная техника»

Лабораторная работа №1

«Операция над множествами»

Выполнил студент

Группы ИВТАСбд-11

Долгов Д. П.

Ульяновск, 2022

**Работа №1. Операции над множествами**

**Цель работы.**

Требуется написать программу, которая будет выполнять операции над множествами, а именно операции объединения, пересечения, дополнения, симметрическую разность, отрицание и <индивидуальное задание>. Обязательно должна присутствовать проверка на корректность введённых данных. Также входные данные должны соответсвовать следующему формату: первый символ – буква, второй символ – цифра, третий символ – чётная цифра, четвёртый символ – буква. Пользователь обладает возможностью ввода данных.

**Описание метода выоплнения**

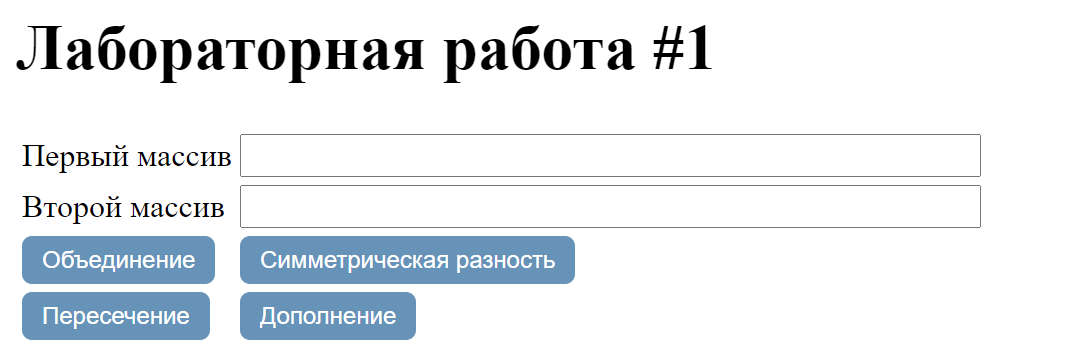
Операции над множествами представляют собой алгебраические операции над массивами на каком-либо языке программирования. Существуют такие операции над множествами как пересечение – берутся только элементы, которые присутствуют в двух множествах одновременно, симметрическая разность – берутся все элементы множеств без тех из них, которые одинаковы в обоих множествах, объединение – берутся элементы обоих множеств и убираются все те, которые повторяются, разность – берутся элементы одного множества без всех элементов, которые также есть в другом множестве.

Чтобы реализовать данные операции используется язык программирования javascript вместе с языком разметки html. При этом реализация происходит через среду разработки PHPStorm, и для дальнейших возможностей html код пишется прямо в .php файле.

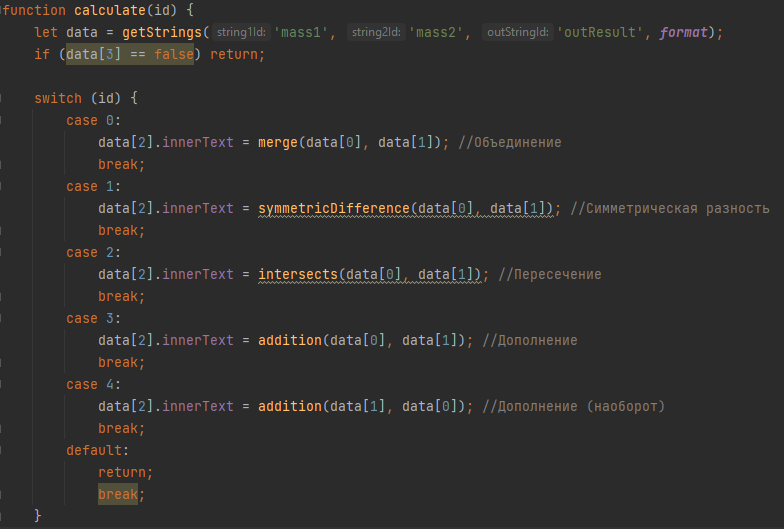
Для запуска сервера используется программа OpenServer.

На html странице присуствуют такие элементы как два поля ввода, куда вводятся массивы, а также кнопки, благодаря которым можно выбрать требуемую алгебраическую операцию над введёнными массивами. Результат данных выводится ниже – в виде обычного текста см. *Рисунок 1. html страница работы*

Рисунок 1. html страница работы

После того, как пользователь ввёл данные и нажал на одну из кнопок, запускается метод calculate(id) на языке javascript, где id – номер выполняемой операции (каждой из перечисленных алгебраических операций присвоен свой номер).

Метод calculate() представлят собой отправную точку, с которой уже происходят расчёты в соотвествии с выбраной алгебраической опирацией.



Как видно из кода, за каждую из операций отвечает своя функция: merge() – объединение, symmetricDifferece() – симметрическая разность, intersects() – пересечение, addition() – дополнение (под id = 4 есть дополнонение наоборот, - то же самое, но разность идёт не от data[0] – data[1], а data[1] – data[0]). Выбор требуемой функции осуществляется по id с помощью оператора switch.

Перед началом выполнения операции, сначала нужно разделить строки на слова и проверить на соответствие формату. Это осуществляется в методе getStrings().

function getStrings(string1Id, string2Id, outStringId, format) {  
 let a = ***document***.getElementById(string1Id);  
 let b = ***document***.getElementById(string2Id);  
 let outResult = ***document***.getElementById(outStringId);  
 let result = true;  
  
 let result1 = checkMessage(a, 1, format);  
 if (result1[1] == false) result = false;  
 let result2 = checkMessage(b, 2, format);  
 if (result2[1] == false) result = false;  
  
 if (result == false)  
 alert(result1[2] + result2[2]);  
  
 return [result1[0], result2[0], outResult, result];  
}

Здесь строчки считываются из элементов html страницы, а затем в методе checkMessage() разбиваются на слова, а также проверяются на соответствие формату. После этого, если нет соответвия формату или строка пустая, выводится сообщение об ошибке. Вот как выглядит функция checkMessage():

unction checkMessage(str, id, format) {  
 let result = false;  
 let mass = [];  
 let errorMessage = '';  
 if (validateString(str)) {  
 mass = str.value.split(' ');  
 if (checkWordsFormat(mass, format)) {  
 result = true;  
 } else {  
 errorMessage += 'Массив ' + id + ' имеет неподходящий формат. Формат должен быть вида ' + format +  
 '. Здесь c - цифра, b - буква, i - чётная цифра, j - нечётная цифра\n\n';  
 }  
 } else  
 errorMessage += 'Массив ' + id + ' не должен быть\n\n';  
  
 return [mass, result, errorMessage];  
}

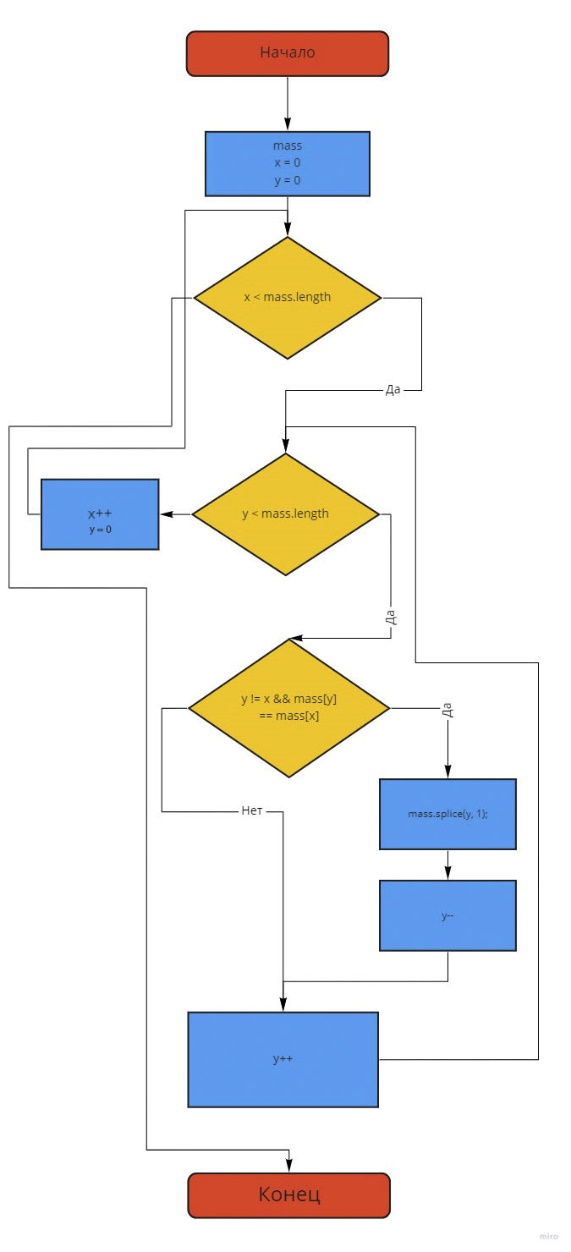
Здесь проверяется, является ли строка пустой или нет, а затем проверяется формат вводимой строки, путём обычного сравнивания значений слова с требуемым значением в строке формата.

Теперь пройдёмся по каждой из функций алгебраических операций.

*Merge() - Объединение*

Эта функция объединяет два массива с помощью встроенной функции concat(), а затем удаляет в получишвимся новом массиве одинаковые элементы.

function merge(mass1, mass2) {  
 return removeRepetitions(mass1.concat(mass2));  
}



Кода здесь всего одна строчка, ведь для расчёта используется функция removeRepetitons(), которая удаляет повторяющиеся элементы массива, просто проходя циклом по каждому элемента массива и проверяя, не нашёлся ли элемент с таким же значением см. *Рисунок 2. Блок-схема метода removeRepetitions*

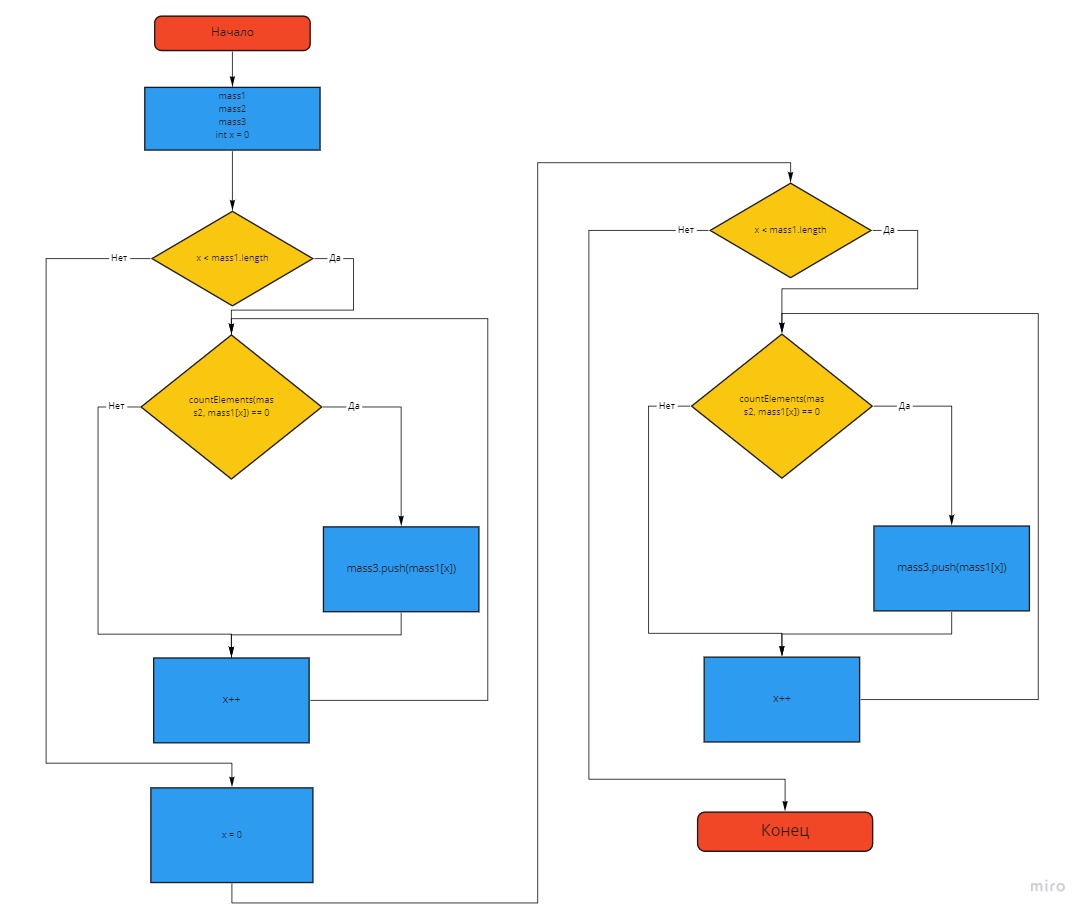
function removeRepetitions(mass) {  
 for (let x = 0; x < mass.length; x++)  
 for (let y = 0; y < mass.length; y++)  
 if (y != x && mass[y] == mass[x]) {  
 mass.splice(y, 1);  
 y--;  
 }  
 return mass;  
}

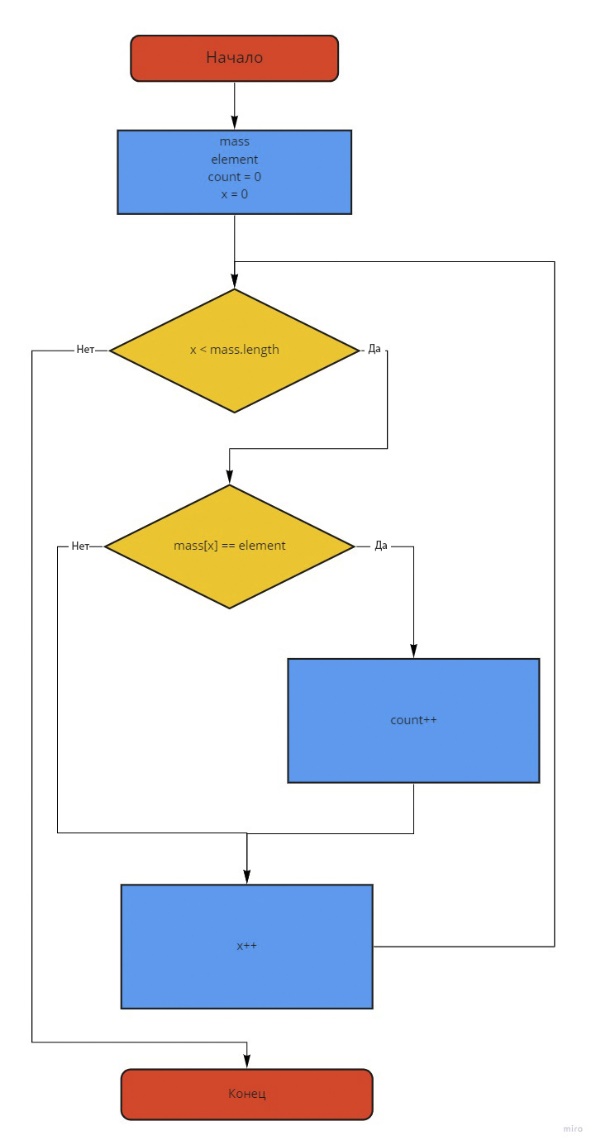
*symmetricDifferece() - Симметричная разность.*

Этот метод сначала добавляет в выходной массив mass3 элементы первого массива, которых нет во втором массиве, а затем – наоборот см. *Рисунок 3. Блок-схема функции symmetricDifference().*

Рисунок 2. Блок-схема метода removeRepetitions

function symmetricDifference(mass1, mass2)  
{  
 let mass3 = [];  
 for(let x = 0; x < mass1.length; x++)  
 if(countElements(mass2, mass1[x]) == 0)  
 mass3.push(mass1[x]);  
  
 for(let x = 0; x < mass2.length; x++)  
 if(countElements(mass1, mass2[x]) == 0)  
 mass3.push(mass2[x]);  
 return mass3;  
}



Рисунок 3. Блок-схема функции symmetricDifference()

Здесь используется функция countElements(mass, element), которая обычным перебором массива mass узнаёт, сколько в массиве находится элементов, равных значению element см. *Рисунок 4. Блок-схема метода countElements().*

Рисунок 4. Блок-схема метода countElements()

function countElements(mass, element) {  
 let count = 0;  
 for (let x = 0; x < mass.length; x++)  
 if (mass[x] == element) count++;  
 return count;  
}

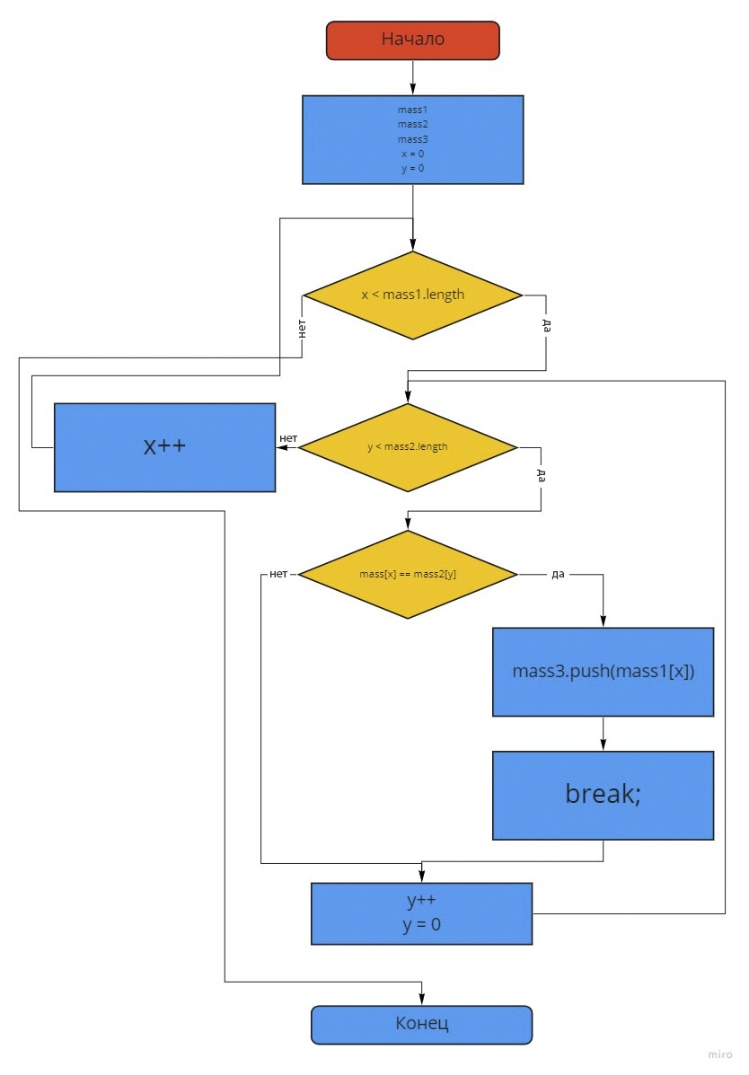
*intersects() - Пересечение*

Рисунок 5. Блок-схема для метода intersects()

Этот метод принимает на вход два массива и записывает в выходной массив mass3 элементы, которые есть и в том, и в том множестве, путём перебирания элементов первого массива, и перебирания двойным циклом элементов второго массива, пока не найдётся одинаковый элемент. Когда он нашёлся, он добавляется в массив mass3, а затем цикл перебора второго массива завершается и продолжается перебор первого массива см. *Рисунок 5. Блок-схема для метода intersects().*

function intersects(mass1, mass2)  
{  
 let mass3 = [];  
 for(let x = 0; x < mass1.length; x++)  
 for(let y = 0; y < mass2.length; y++)  
 if(mass1[x] == mass2[y])  
 {  
 mass3.push(mass1[x]);  
 break;  
 }  
 return mass3;  
}

*addition() - Дополнение.*

Эта функция принимает в качестве входных значений два массива – mass1 и mass2. Результат записывается в массив mass3. mass3 изначально равняется mass1, а затем из него путём перебора удаляются те элементы, которые присутствуют во втором массиве. Для этого используется вышеупомянутая функция countElements() см. Рисунок 6. Блок-схема функции addition()

function addition(mass1, mass2) {  
 let mass3 = mass1;  
 for (let x = 0; x < mass3.length; x++)  
 if (countElements(mass2, mass3[x]) > 0) {  
 mass3.splice(x, 1);  
 x--;  
 }  
 return mass3;  
}

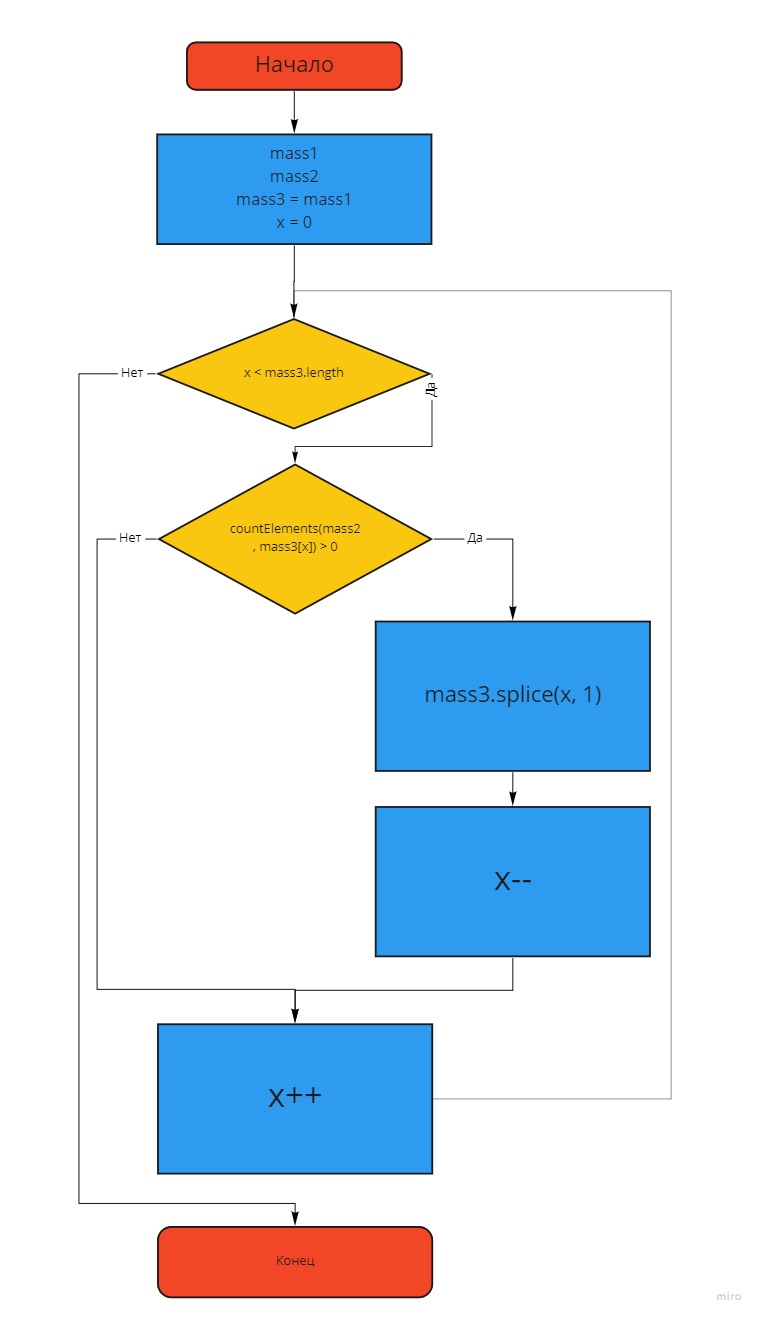


Рисунок 6. Блок-схема функции addition()

**Выводы**

Программа протестированна и успешно работает. После написания программы я научился работать с html и javascript, а также использовать новую среду разработки PHPStorm, научился выполнять алгебраические операции над множествами.

**Список литературы**

Владилен Минин. JavaScript Основы для Начинающих – Полный Курс за 6 часов [2020]. [Электронный ресурс] URL: <https://www.youtube.com/watch?v=Bluxbh9CaQ0&t=1650s>

Хауди Хо. Учим HTML за 1 час! #От Профессионала | HD Remake. [Электронный ресурс] URL: <https://www.youtube.com/watch?v=bWNmJqgri4Q&t=2192s>