Министерство образования и науки Российской Федерации

ФГБОУ ВО Ульяновский государственный технический университет

Кафедра «вычислительная техника»

Лабораторная работа №1

«Операции над множествами»

Выполнил студент

Группы ИВТАСбд-11

Долгов А. П.

Ульяновск, 2021

**Цель работы.**

Требуется написать программу, которая выполняет операции над множествами, а именно пересечение, объединение, симметрическая разница, дополнение. Кроме того необходимо сделать валидацию данных по заданному формату: первый символ – буква, второй – цифра, третий и четвёртый – чётные цифры. Пользователь имеет возможность ввода данных

**Описание метода выполнения.**

Множество является совокупность объектов. В программе нужно реализовать несколько операций над множествами: объединение, пересечение, симметрическая разница, дополнение первого до второго и наоборот.

Объединение – операция над множествами, возвращающая множество, содержащее все элементы двух других множеств без повторений.

Пересечение – операция над множествами, возвращающая множество, содержащее элементы, которые принадлежат и к первому, и ко второму множеству без повторений.

Симметрическая разница – операция над множествами, возвращающая множество, содержащее элементы и первого и второго множества, не совпадающие между собой.

Дополнение первого до второго – операция над множествами, возвращающая множество, содержащее элементы, которые принадлежат к первому множеству, но не принадлежат ко второму.

Ввод множеств осуществляется с помощью двух полей input в html файле. Чтобы произвести операции над двумя введёнными множествами, нужно нажимать на соответствующие кнопки. Итоговое множество выводится под кнопками. Код проводимых операций будет написан на JavaScript. JavaScript – это язык, с помощью которого можно писать скрипты для сайтов. JavaScript взаимодействует с html.

При нажатии на кнопку выполняется соответствующая ей операция путём вызова нужной функции из скрипта.

Перед разбором всех кнопок, сначала посмотрим на метод, который выполняет валидацию строки по формату. Называется он «validate» и принимает на вход строку и строку-формат. При совпадении строки и формата, функция возвращает true, в противном случае false. Вот её код:

function validate(str, format)  
{  
 let isValid = true;  
 if(str.length == format.length)  
 {  
 for(let i = 0; i < format.length; i++)  
 {  
 switch(format[i])  
 {  
 case 'c':  
 {  
 if(!isDigit(str[i]))  
 {  
 i = format.length;  
 isValid = false;  
 }  
 break;  
 }  
 case 'b':  
 {  
 if(!isAlpha(str[i]))  
 {  
 i = format.length;  
 isValid = false;  
 }  
 break;  
 }  
 case 'i':  
 {  
 if(!isDigit(str[i]))  
 {  
 i = format.length;  
 isValid = false;  
 }else  
 {  
 if((str[i] % 2) != 0)  
 {  
 i = format.length;  
 isValid = false;  
 }  
 }  
 break;  
 }  
 case 'j':  
 {  
 if(!isDigit(str[i]))  
 {  
 i = format.length;  
 isValid = false;  
 }else  
 {  
 if((str[i] % 2) == 0)  
 {  
 i = format.length;  
 isValid = false;  
 }  
 }  
 break;  
 }  
 }  
 }  
 }else  
 isValid = false;  
 return isValid;  
}

Ещё есть функция, которая делает то же самое, но уже с массивом строк:

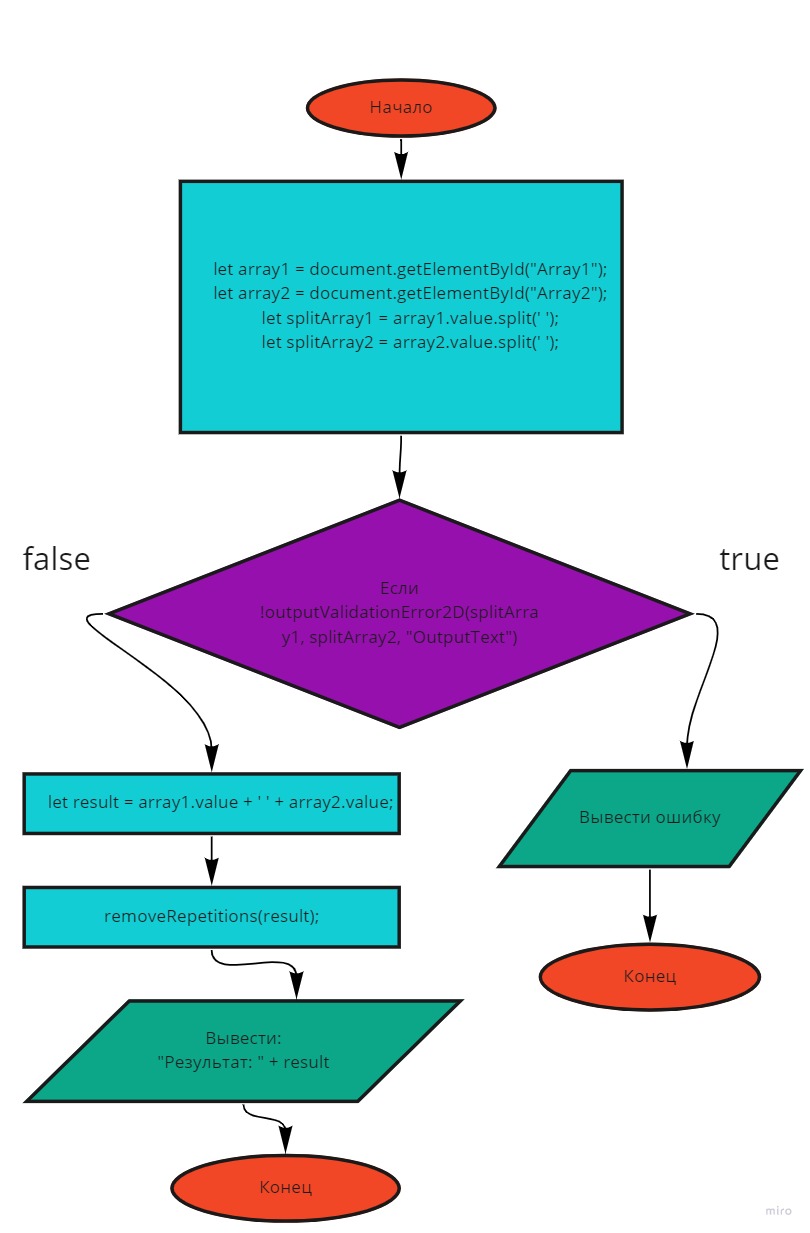
function validate2D(str2D, format)  
{  
 for(let i = 0; i < str2D.length; i++)  
 if(!validate(str2D[i], format))  
 return false;  
 return true;  
}

«validate2D» используется в методе «outputValidationError2D», который проверяет на правильность формата два массива строк, и в случае неудачи выводит об этом извещение, указывая, в каком именно массиве ошибка. Код функции:

function outputValidationError2D(splitArray1, splitArray2, outputTextFieldID)  
{  
 let valid1 = validate2D(splitArray1, ***INPUT\_FORMAT***);  
 let valid2 = validate2D(splitArray2, ***INPUT\_FORMAT***);  
  
 if(!valid1 && !valid2)  
 {  
 //document.getElementById(outputTextFieldID).innerText = "Incorrect format in both arrays, correct format is " + "\"" + INPUT\_FORMAT + "\"";  
 ***document***.getElementById(outputTextFieldID).innerText = "Не корректный формат в обоих массивах, корректный формат это " + "\"" + ***INPUT\_FORMAT*** + "\"";  
 return false;  
 }  
 else if(!valid1)  
 {  
 //document.getElementById(outputTextFieldID).innerText = "Incorrect format in array 1, correct format is " + "\"" + INPUT\_FORMAT + "\"";  
 ***document***.getElementById(outputTextFieldID).innerText = "Не корректный формат в обоих массивах, корректный формат это " + "\"" + ***INPUT\_FORMAT*** + "\"";  
 return false;  
 }else if(!valid2)  
 {  
 //document.getElementById(outputTextFieldID).innerText = "Incorrect format in array 2, correct format is " + "\"" + INPUT\_FORMAT + "\"";  
 ***document***.getElementById(outputTextFieldID).innerText = "Не корректный формат в обоих массивах, корректный формат это " + "\"" + ***INPUT\_FORMAT*** + "\"";  
 return false;  
 }  
 return true;  
}

Теперь перейдём к кнопкам. Рассмотрим кнопку «Объединение». При нажатии на неё вызывается функция «merge». Суть её состоит в том, чтобы объединить два массива в один и удалить повторяющиеся элементы. Вот её код:

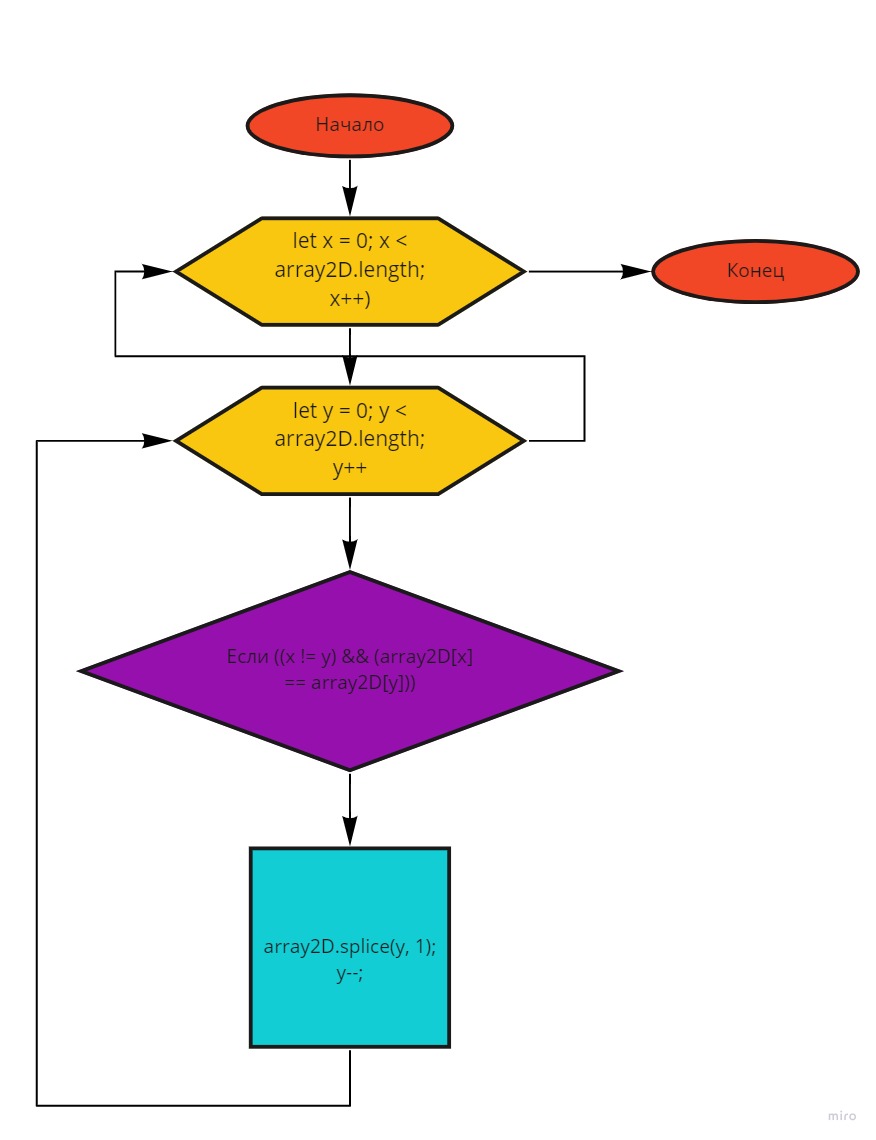
function merge()  
{  
 let array1 = ***document***.getElementById("Array1");  
 let array2 = ***document***.getElementById("Array2");  
  
 let splitArray1 = array1.value.split(' ');  
 let splitArray2 = array2.value.split(' ');  
  
 if(!outputValidationError2D(splitArray1, splitArray2, "OutputText"))  
 return;  
  
 let result = array1.value + ' ' + array2.value;  
  
 result = result.split(' ');  
 removeRepetitions(result);  
  
 //document.getElementById("OutputText").innerText = "Result: " + result;  
 ***document***.getElementById("OutputText").innerText = "Результат: " + result;  
}



*Блок-схема метода merge*

Основная работа происходит в функции «removeRepetitions»:

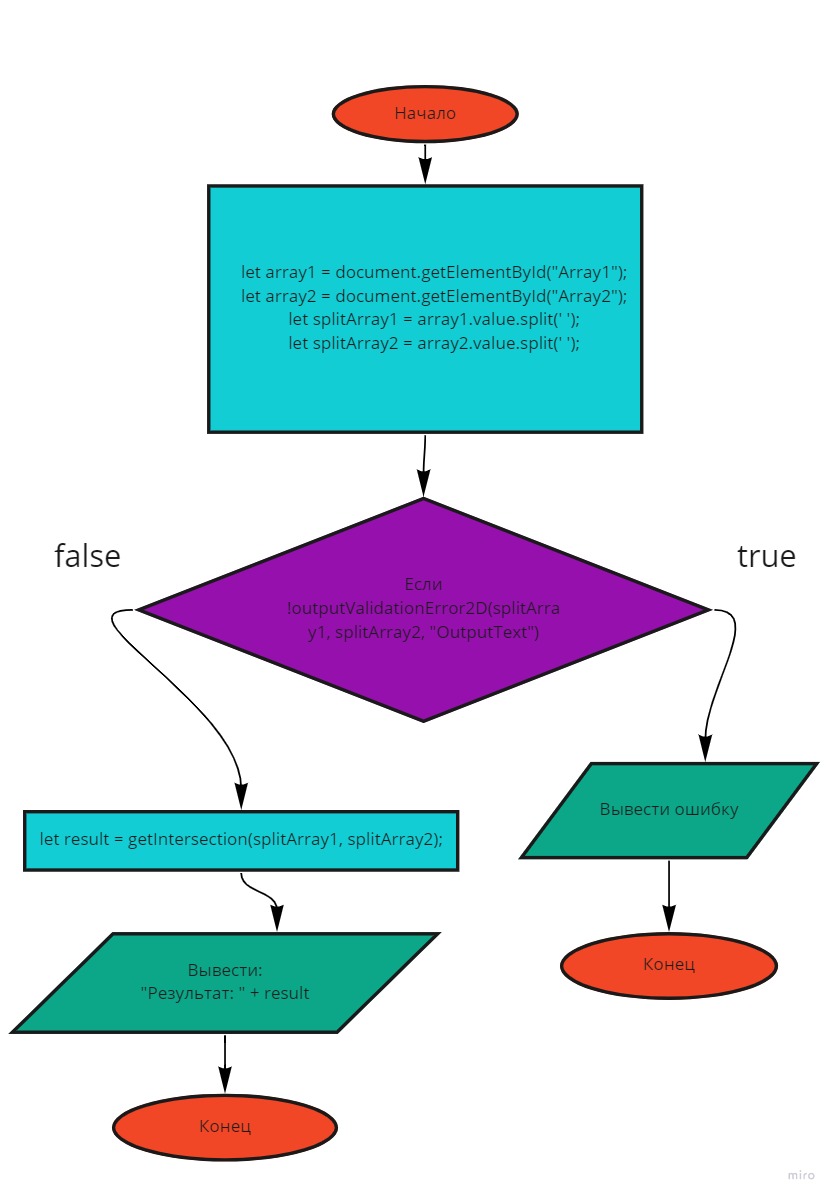
function removeRepetitions(array2D)  
{  
 for(let x = 0; x < array2D.length; x++)  
 for(let y = 0; y < array2D.length; y++)  
 {  
 if((x != y) && (array2D[x] == array2D[y]))  
 {  
 array2D.splice(y, 1);  
 y--;  
 }  
 }  
}



*Блок-схема метода removeRepetitions*

Теперь разберём кнопку «Пересечение». При её нажатии вызывается метод «intersect». Он находит одинаковые элементы в двух входных массивах и выводит их. Код функции:

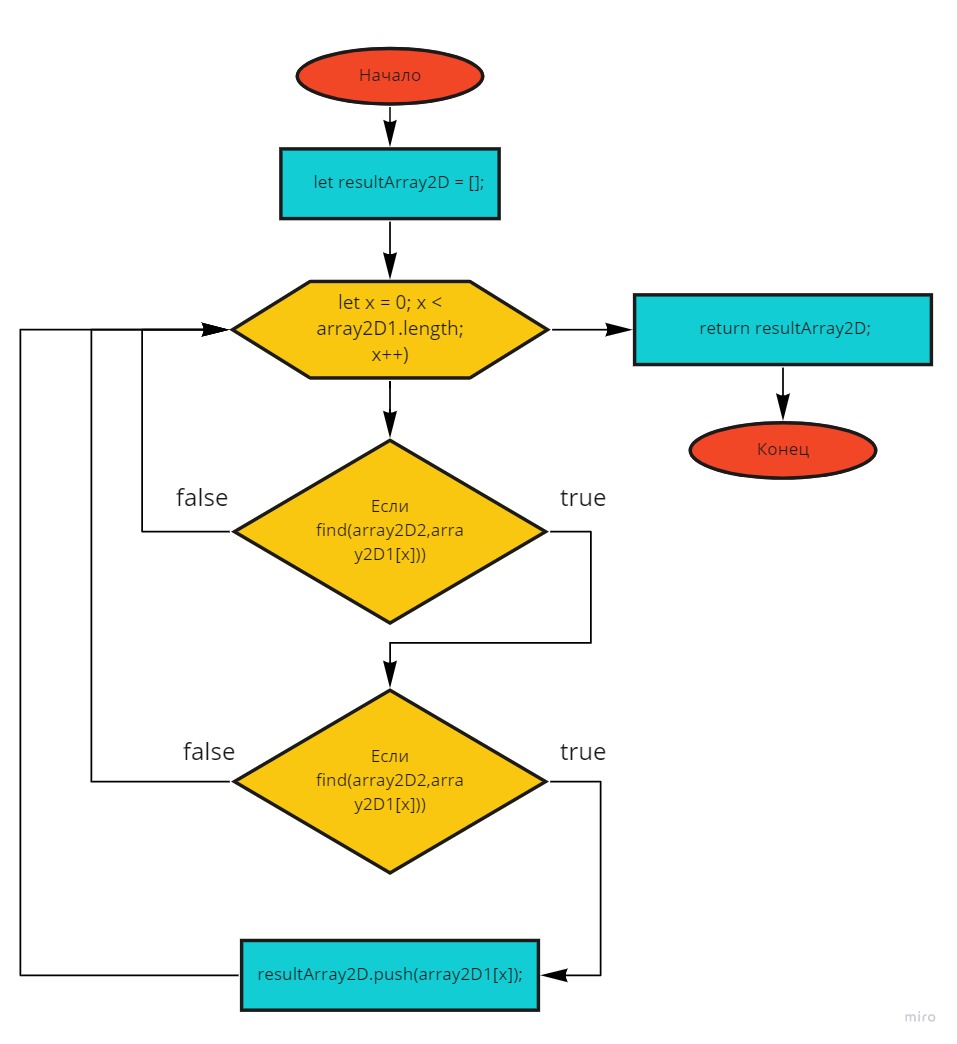
function intersect()  
{  
 let array1 = ***document***.getElementById("Array1");  
 let array2 = ***document***.getElementById("Array2");  
  
 let splitArray1 = array1.value.split(' ');  
 let splitArray2 = array2.value.split(' ');  
  
 if(!outputValidationError2D(splitArray1, splitArray2, "OutputText"))  
 return;  
  
 let result = getIntersection(splitArray1, splitArray2);  
 //document.getElementById("OutputText").innerText = "Result: " + result;  
 ***document***.getElementById("OutputText").innerText = "Результат: " + result;  
}



*Блок-схема метода intersect*

Основная работа по пересечению выполняется в функции «getIntersection»:

function getIntersection(array2D1, array2D2)  
{  
 let resultArray2D = [];  
 for(let x = 0; x < array2D1.length; x++)  
 {  
 if(find(array2D2,array2D1[x]))  
 if(!find(resultArray2D, array2D1[x]))  
 {  
 resultArray2D.push(array2D1[x]);  
 }  
 }  
 return resultArray2D;  
}



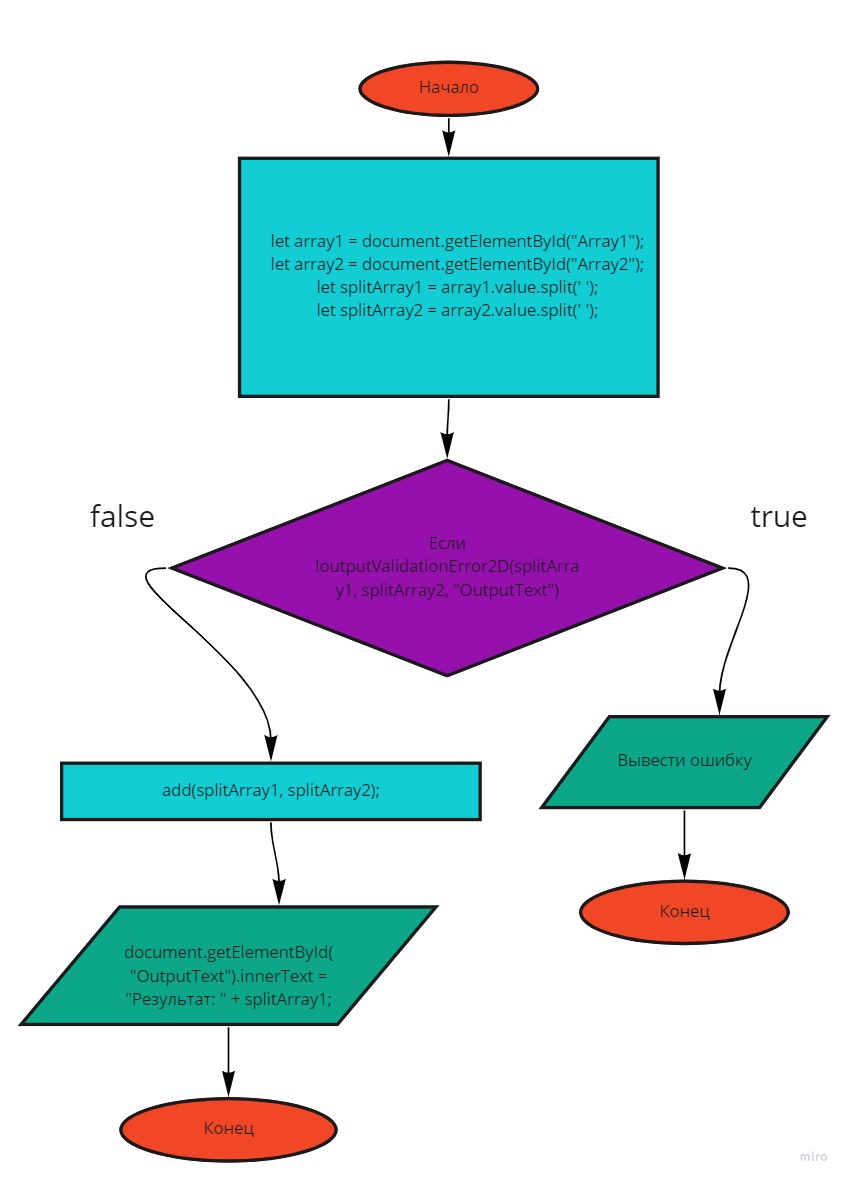
*Блок-схема метода getIntersection*

В этом методе используется функция «find», которая ищет в массиве array элемент element. При нахождении данного элемента, функция возвращает true, иначе false. Вот её код:

function find(array, element)  
{  
 for(let i = 0; i < array.length; i++)  
 {  
 if(array[i] == element)  
 return true;  
 }  
 return false;  
}

Дополнение первого множества ко второму реализуется с помощью удаления из первого массива всех элементов, которые есть и во втором массиве. Функция, которая вызывается при нажатии на кнопку, называется «addition1to2». Вот её код:

function addition1to2()  
{  
 let array1 = ***document***.getElementById("Array1");  
 let array2 = ***document***.getElementById("Array2");  
  
 let splitArray1 = array1.value.split(' ');  
 let splitArray2 = array2.value.split(' ');  
  
 if (!outputValidationError2D(splitArray1, splitArray2, "OutputText"))  
 return;  
  
 add(splitArray1, splitArray2);  
 //document.getElementById("OutputText").innerText = "Result: " + splitArray1;  
 ***document***.getElementById("OutputText").innerText = "Результат: " + splitArray1;  
}

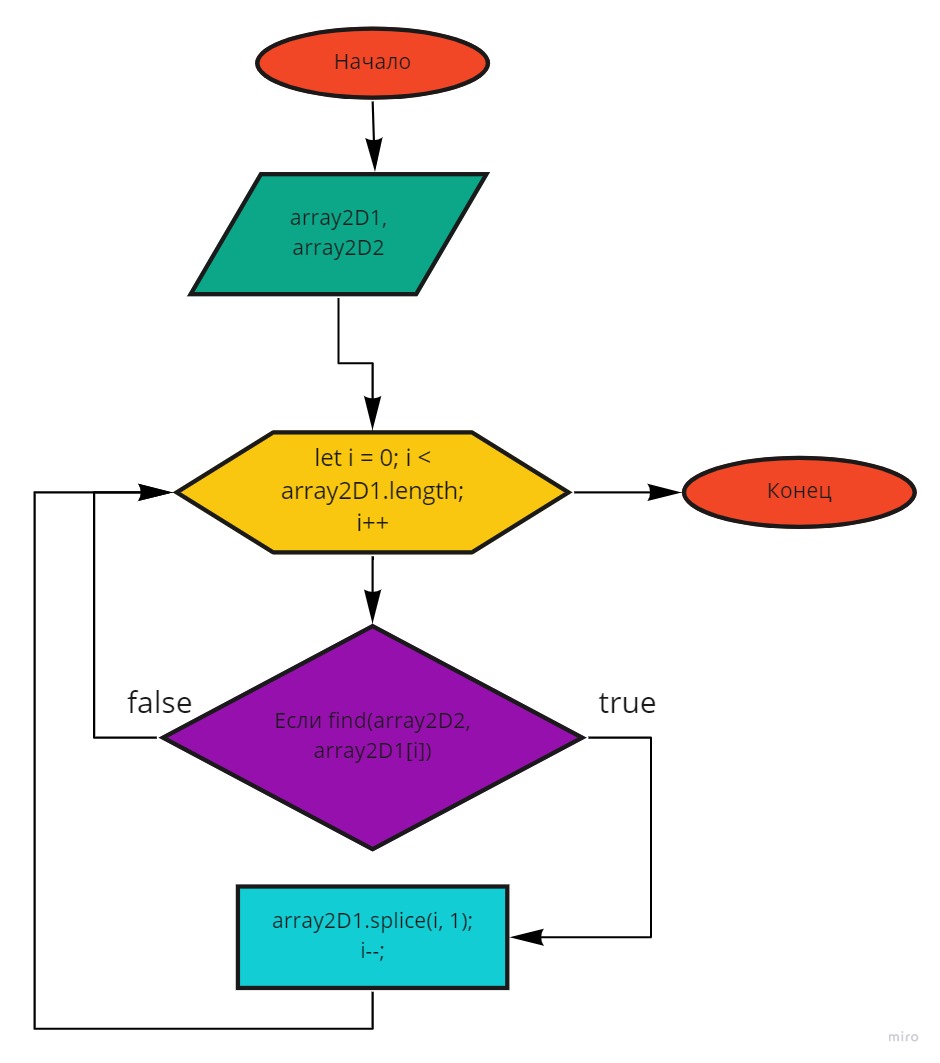


*Блок-схема метода addition1to2*

Дополнение второго множества до первого реализуется аналогичным образом в функции *addition2to1*, лишь меняются местами массивы splitArray1 и splitArray2.

Основная работа этих методов выполняется в функции «add»:

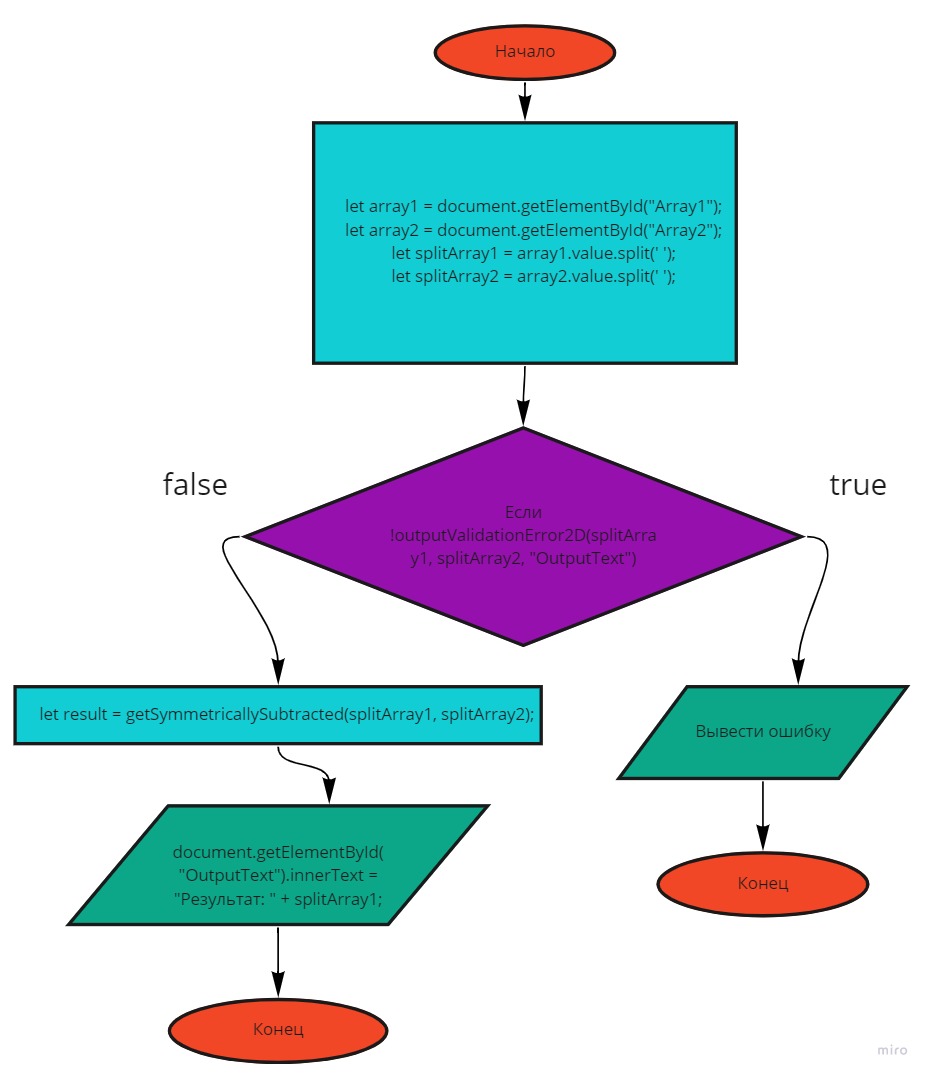
function add(array2D1, array2D2)  
{  
 for(let i = 0; i < array2D1.length; i++)  
 {  
 if(find(array2D2, array2D1[i]))  
 {  
 array2D1.splice(i, 1);  
 i--;  
 }  
 }  
}



*Блок-схема метода add*

И последняя кнопка – это симметричная разность. При её нажатии вызывается метод «symmetricallySubtract», который оставляет только те элементы, которые есть в первом массиве, но нету во втором и те, которые есть во втором и которых нету в первом. Вот его код:

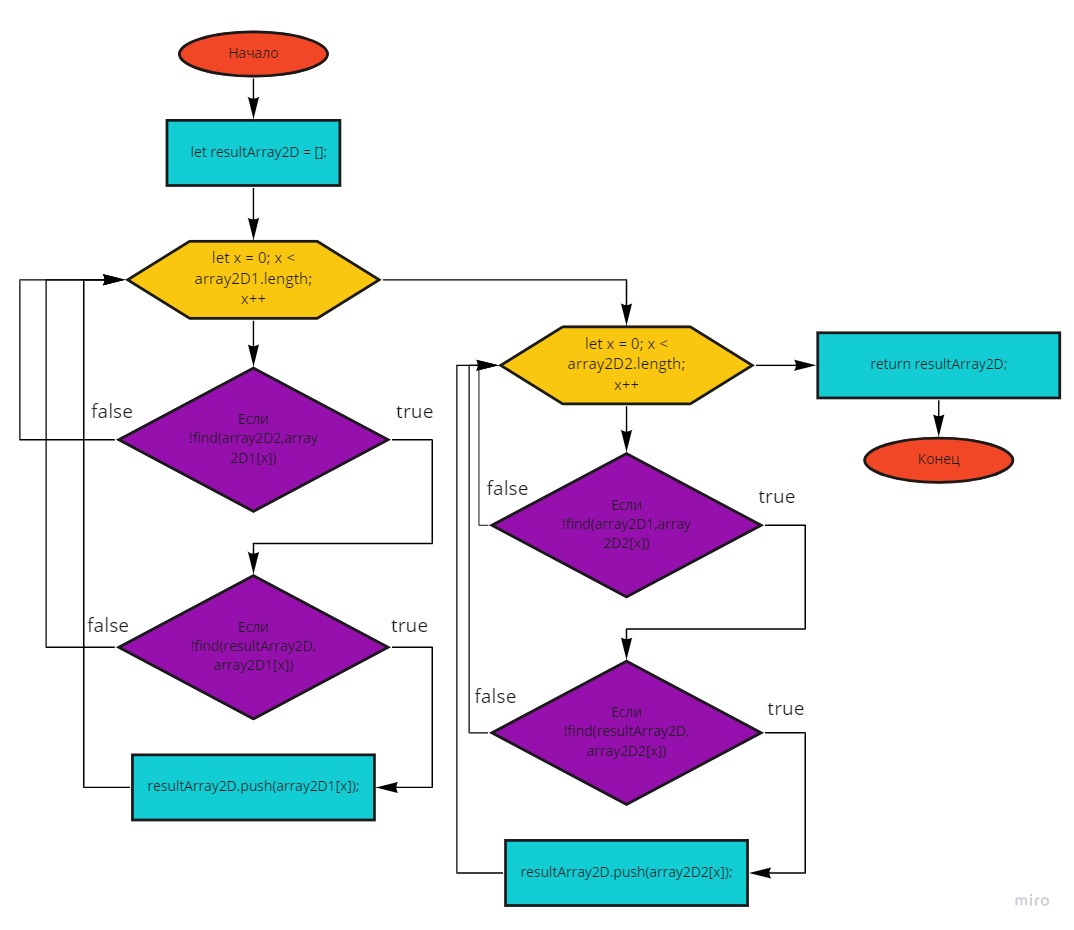
function symmetricallySubtract()  
{  
 let array1 = ***document***.getElementById("Array1");  
 let array2 = ***document***.getElementById("Array2");  
  
 let splitArray1 = array1.value.split(' ');  
 let splitArray2 = array2.value.split(' ');  
  
 if(!outputValidationError2D(splitArray1, splitArray2, "OutputText"))  
 return;  
  
 let result = getSymmetricallySubtracted(splitArray1, splitArray2);  
 //document.getElementById("OutputText").innerText = "Result: " + result;  
 ***document***.getElementById("OutputText").innerText = "Результат: " + result;  
}



*Блок-схема метода symmetricallySubtract*

Основная работа выполняется в функции «getSymmetricallySubstracted». Код:

function getSymmetricallySubtracted(array2D1, array2D2)  
{  
 let resultArray2D = [];  
 for(let x = 0; x < array2D1.length; x++)  
 {  
 if(!find(array2D2,array2D1[x]))  
 if(!find(resultArray2D, array2D1[x]))  
 {  
 resultArray2D.push(array2D1[x]);  
 }  
 }  
  
 for(let x = 0; x < array2D2.length; x++)  
 {  
 if(!find(array2D1,array2D2[x]))  
 if(!find(resultArray2D, array2D2[x]))  
 {  
 resultArray2D.push(array2D2[x]);  
 }  
 }  
 return resultArray2D;  
}

****

*Блок-схема метода getSymmetricallySubstracted*

**Вывод**

Выполнив данную лабараторную работу я научился программировать на языке JavaScript и использовать такие языки разметки как html и css. С всего этого я создал сайт, который может выполнять некоторые операции над двумя множествами и делать проверку вводимых данных по определённому формату.

**Список использованных источников:**

* Видео с youtube канала «Владилен Минин». [Электронный ресурс] URL: <https://www.youtube.com/watch?v=Bluxbh9CaQ0&t=1650s>
* Видео с youtube канала «[Хауди Хо™ - Просто о мире IT!](https://www.youtube.com/channel/UC7f5bVxWsm3jlZIPDzOMcAg)». [Электронный ресурс] URL: <https://www.youtube.com/watch?v=bWNmJqgri4Q>