====================== ТЕОРІЯ ========================

Метод sort() сортує масив. Він сортує в порядку зростання, перетворюючи елементи до рядкового типу. Це дає неправильні результати для числових масивів.

var arr = [1, 2, 15,36,305];

arr.sort ();

alert (arr); / / 1, 15, 2, 305, 36

Методу sort (функція порівняння) вміє сортувати будь масиви, якщо вказати функцію fn від двох елементів, яка вміє порівнювати їх. Вона повинна повертати:

Додатне значення, якщо перший елемент більший,

Від'ємне значення, якщо другий більший,

Якщо рівні – то 0.

function compareNumber (a, b) {

if (a>b) {return 1};

if (a<b) {return -1};

return 0;

//а можна просто return a-b

}

var arr = [1,2,15,36,305];

arr.sort (compareNumber);

alert (arr); / / 1, 2, 15, 36, 305

Приклад. Скласти скрипт для сортування масиву arr порядку спадання

arr.sort(function(a,b){return b-a});

arr.sort( (a,b) => b-a);

Приклад. Скласти скрипт для сортування масиву arr в порядку, коли спочатку йдуть всі парні чила, потім – непарні.

function evenCompare (a, b) {

  // Числа однієї парності - сортуються звичайним чином

if (a%2 == b%2) return a - b;

// Інакше, якщо a - парне, то воно менше

if (a% 2 == 0) return -1;

// Лишився один варіант: a - непарне, і b – парне

return 1;

}

var arr = [5, 2, 1, -10, 8];

arr.sort (evenCompare);

alert (arr); / / -10, 2, 8, 1, 5

Використання випадкових чисел для функції порівння дозволить перемішати елементи масиву випадковим чином. Math.random() повертає результат що належить [0,1).

var arr = [1, 2, 3, 4, 5];

arr.sort (function () { return Math.random () - 0.5;});

alert (arr); / / елементи у випадковому порядку, наприклад [3,5,1,2,4]

**//=================================================================**

**//=================================================================**

**//=================================================================**

forEach (функція) викликає функцію для кожного елемента масиву. Функція викликається з параметрами (item, i, arr): item – значення поточного елементу масиву; i - його номер; arr - масив, що перебирається. Зміна значення item не приводить до зміни відповідного елементу, для цього слід звертатись до arr[i].

Приклад. Скласти скрипт, що виводить кожен елемент масиву в окремий рядок у форматі a[i]=…

function print(x,i,arr){

document.writeln('a['+i+']='+x+'<br>');

}

function print2(x,i,arr){

document.writeln(x\* x);

}

var a=[2,3,5];

a.forEach(print2);

for(var i=0;i<a.length;i++)

{

let x=a[i];

print2(x,i,a);

}

alert(a);

**//=================================================================**

**//=================================================================**

**//=================================================================**

filter (callback) створює новий масив, в який увійдуть тільки ті елементи вихідного масиву, для яких виклик callback (item, i, arr) поверне true

Приклад. Створит скрипт, який знаходить усі парні числа

//Функція порівння (визначення чи парне)

function isEven(number, i, arr) {

return number % 2 == 0;

}

//=========================

var a = [2, 11, 56, 43, 23, 890];

var a\_even = a.filter(isEven);

var a\_even = a.filter(x => x%2==0 ) ;

document.write(a\_even);

Приклад. Скласти скрипт, що знаходить всі парні елементи на непарних місцях.

function condition(x,i,arr){

return (i%2!=0)&&(x%2==0);

}

var b=a.filter(condition);

**//=================================================================**

**//=================================================================**

**//=================================================================**

map (callback) створює новий масив, який складатиметься з результатів виклику callback (item, i, arr) для кожного елемента arr.

Приклад. Скласти скрипт, що формує новий масив збільшуючи кожен елемент масиву на його індекс;

function change(x,i,arr){

return x+i;

}

alert(a.map(change));

var t=a.map((x,i)=>x+i);

**//=================================================================**

**//=================================================================**

**//=================================================================**

every (callback) повертає true, якщо виклик callback поверне true для кожного елемента масива.

some (callback) повертає true, якщо виклик callback поверне true для якогось елемента масива.

Приклад. Чи всі елементи массиву додатні, чи є хоча б один додатній.

var arr = [1, -1, 2, -2, 3];

function isPositive (number) {

return number> 0

}

alert (arr.every (isPositive)); / / false, не всі додатні

alert (arr.some (isPositive)); / / true, є хоч одне додатне

**Приклад. Дано оцінки учня з»ясувати чи є учень двійочником. Якщо не є, то чи є він трійочиником(12 система).**

function isNotBad(mark) { //З»ясовуємо чи двійка

return mark>3;

}

function is\_3(mark) { //З»ясовуємо чи трійка

return mark < 7;

}

//=========================

var a = [8, 11, 9, 10, 9];

if (a.every(isNotBad)) { //Перевіряємо чи не двійочник (усі оцінки більші за 3)

if (a.some(is\_3)) { //Перевіряємо чи трійочник (хоча би одна оцінка <3)

//a.some((x)=>x<7)

alert("3333");

}

else {

alert(">3");

}

}

else {

alert("2222");

}

**//=================================================================**

**//=================================================================**

**//=================================================================**

reduce (reduceCallback [, initialValue]) застосовує функцію reduceCallback по черзі до кожного елементу масиву зліва направо, зберігаючи при цьому проміжний результат.  
Аргументи функції reduceCallback (previousValue, currentItem, index, arr):  
     previousValue - останній результат виклику функції, він же проміжний результат.  
     currentItem - поточний елемент масиву, елементи перебираються по черзі зліва-направо.  
     index - номер поточного елемента.  
     arr - масив.  
Значення previousValue при першому виклику одно initialValue. Якщо initialValue немає, то воно дорівнює першому елементу масиву, а перебір починається з другого.

Приклад. Скласти скрипт для знаходження суми всіх елементів та кількості додатних елементів в масиві.

function kilk(result,x,i,arr){

if (x>0) {result++};

return result;

}

function suma(result,x,i,arr){

result+=x;

return result;

}

var a=[1,2,-3,4,-5];

alert(a.reduce(kilk,0));

alert(a.reduce(suma));

====================== ПРИКЛАДИ =====================

------------------------------------------------------------------------------

Задача. Упорядкувати елементи масиву:

1)за зростанням;

var arr = [1, 2, 15,36,305];

arr.sort( )

2)за спаданням;

3)порівнюючи модулі;

4)порівнюючи квадрати;

var arr = [1, 20, 15, 36, 305];

arr.sort((a, b) => a\*a - b\*b);

5)порівнюючи за останньою цифрою.

arr.sort((a, b) => a % 10 - b % 10);

var names = ["John", "Kate", "Mikle"];

name.sort((a, b) => { a.charCodeAt(1) - b.charCodeAt(1)});

name.sort((a, b) => { a.charCodeAt(a.length - 1) - b.charCodeAt(b.length - 1) });

-----------------------------------------------------------------------

Задача. Кожен від’ємний елемент масиву піднести до квадрату

var a = [1, -4, 8, -3, 5];

function toSqure(x, i, arr)

{

if(x<0)

arr[i] = x \* x;

}

a.forEach(toSqure);

a.forEach((x, i, arr) => { if (x < 0) a[i] = x \* x; })

-----------------------------------------------------------------------

Задача. Кожне парне число збільшити на 5.

-----------------------------------------------------------------------

Задача. Дано масив цін. Сформувати масив цін, які більші за 100.

var t=a.filter((x,i,arr)=>x>100)

//a.filter((x) => x > 100)

Вибрати елементи, які менші за свій індекс

var t =a.filter((x,i)=>x<i);

-----------------------------------------------------------------------

Задача. Дано масив років народження працівників. Вивести тих, які народилися між 1980 і 2003 роках.

var t = a.filter((x) => x>=1980 && x<=2003);

-----------------------------------------------------------------------

Задача. Дано масив імен працівників. Сформувати новий масив, який складається з перших літер імен.

var names = ["John", "Kate", "Ivan"];

var t = names.map((x) => x[0]);

-----------------------------------------------------------------------

Задача. Дано масив чисел. Сформувати новий масив, у якому усі числа збільшено на 20.

-----------------------------------------------------------------------

Задача. Дано масив показників термометра протягом місяця. З’ясувати, чи є серед них дні, коли температура була меншою за 5 градусів.

-----------------------------------------------------------------------

Задача. Дано масив показників термометра протягом місяця. З’ясувати, чи усі показники є більшими за 10 градусів.

-----------------------------------------------------------------------

Задача. Знайти добуток від’ємних елементів масиву.

var product= a.reduce( (result,x) => x<0? result\*x : result , 1)

-----------------------------------------------------------------------

Задача. Знайти суму від’ємних елементів масиву.

-----------------------------------------------------------------------

Задача. Створити функцію, яка б могла знаходити суму для довільної кількості переданих чисел

---------------------------------------------------------

Задача. Знайти або суму, або добуток чисел.

------------------------------------------------------------------------------

Задача. Дано масив цілих чисел. Вибрати одну із дій при перегляді елементів.

Якщо елемент більше за заданий то:

1)помножити на 2;

2)поділити на 7;

Якщо елемент менше рівне за заданий то:

1)помножити на 3;

2)помножити на 5.

------------------------------------------------------------------------------

Задача 1. Дано одновимірний масив. Кожен елемент

1)піднести до квадрату;

2)помножити на 7;

3)поділити на 10.

Використати функції.

//------------------------- З використанням передачі функії як параметру --------

//---------------- З використанням стандартного підходу ------------