// .....**armstong number** .....
//#153

```
let x=153;
let tmp=x;
let sum=0;
while(tmp>0){
    y=tmp%10;
    sum +=y**3;
    tmp=parseInt(tmp/10);
}
if(x==sum){
    console.log('this is ArmStrong number ');
}else{
    console.log('this is not  ArmStrong number ');
}
```

// .....Anagram String.....

```
let st1="Anil";
let st2="nilA";

function isAnagram(str1,str2){
    let len1=str1.length;

    let len2=str2.length;
    if(len1 !=len2){
       return false;
    }
    let counter={};
    for(let i=0;i<str1.length;i++){
       console.log('len1',i);
       counter[str1[i]]=(counter[str1[i]]||0)+1;
    }
    for(let j=0;j<len2;j++){
       if(!counter[str2[j]]){
             return false;
       }else{
             counter[str2[j]]--;
       }
    }
    return true;
}
```

```
console.log(isAnagram(st1,st2));
```

//**********Debouncing and Throttle**********

```
let normal= document.querySelector("#normal");
let throuttle= document.querySelector("#throuttle");
let debounce= document.querySelector("#debounce");

let n_count=0;
const normal_fun=()=>{
   n_count++;
   normal.innerHTML=`<p>Normal count is: ${n_count}<p>`
}
let t_count=0;
let flag=true;
const throttle=()=>{
    if(flag){
       t_count++
       throuttle.innerHTML=`<p>Throttle count is: ${t_count}<p>`;
       flag=false;
       setTimeout(()=>{
             flag=true;
       },1000);

    }

}

let d_count=0;
const debouncing=()=>{
    let interval;
    clearTimeout(interval);
      interval = setTimeout(()=>{
       d_count++
       debounce.innerHTML=`<p>Debounce count is: ${t_count}<p>`;
    },1000);

}

const showCount=()=>{
    normal_fun();
    throttle();
    debouncing();
```

```
}
```

**********Convert Array into flat array **********

```
let newArra=[];
function objectFlat(arr){
    for(let val of arr){
       if(Array.isArray(val)){
             objectFlat(val);
       }else{
             newArra.push(val);
       }
    }
}
objectFlat(arr);
console.log(newArra);
```

.....flat an object in to single object.....

```
let obj={
    firstName:'Anil',
    lastName:'Singh',
    address:{
       country:'India',
       state:'Up',
       city:'Bulandshahr',
      company:{
             one:'Braintechnnosys',
             two:'CWS Technology',
             three:'Eastern Software System'
       }
    }
};

let obj2={};
function objectFlat(obj){
    for(let key in obj){
       if(typeof obj[key]=='object'){
             objectFlat(obj[key]);
       }else{
             obj2[key]=obj[key]
```

```
        }
    }
}
objectFlat(obj);
console.log(obj2);
```

// .....LCM of Two number.....

```
function Lcm(a,b){
    let i=a;
    while(true){
      if(i%a==0 && i%b==0){
            return i;
       }
       i++;
    }

}

let gcd=Lcm(21,7);
console.log(gcd);

function Lcm(a,b){
    for(let i=a;i<a*b;i++){
       if(i%a==0 && i%b==0){
            return i;
       }
    }
}
let gcd=Lcm(21,7);
console.log(gcd);


function Lcm(a,b){
    if(b==0){
       return a;
    }else if(a%b==0){
       return b;
    }
    else{
       return Lcm(b,a%b);
    }
```

```
}

let gcd=Lcm(a,b);
let lcm=a*b/gcd;
console.log(lcm);
```

.....Min And Max value.....

```
const arr = [1,2,3,4,11,4,5,5,6,7,7,8,6,10];
let min=arr[0];
```

```
let max=arr[0];
for(let i=0;i<arr.length;i++){
      if(arr[i]>max){
      max=arr[i];
    }else if(min>arr[i]){
       min=arr[i];

    }
}
console.log(min,max);
```

.....Remove Duplicate value.....

```
const arr = [1,2,3,4,11,4,5,5,6,7,7,8,6,10];
let newDup=[];
let counter={};
let increment=0;
for(let i=0;i<arr.length;i++){
   counter[arr[i]]=(counter[arr[i]]||0)+1;
   if(counter[arr[i]]==1){
     newDup[increment]=arr[i];
     increment++;
   }
}
console.log(newDup);
```

.....Find Missing number from Array.....

```
let arr= [1,2,3,5,6,7,8,9,10];
let num=10;

let miss;
```

```
for(let i=0;i<num-1;i++){
    if(arr[i]+1 !=arr[i+1]){
       miss=arr[i]+1;
       break;
    }

}
console.log(miss);
```

```
let arr= [1,2,3,5,6,7,8,9,10];
let num=10;
let sum = num*(num+1)/2;

for(let i=0;i<num-1;i++){
    sum =sum-arr[i];
}
console.log(sum);
```

**********Arrange string element of end array**********

```
function leftToRightArrage(arr){
    let counter=0;
    let str=[];
    for(let i=0;i<arr.length;i++){
       if(typeof arr[i] =='string'){
             str.push(arr[i]);
       }else{
             arr[counter]=arr[i];
             counter++;
       }

    }
    for(let j=0;j<str.length;j++){
       arr[counter]=str[j];
       counter++;
    }
    console.log(arr);
}
let arr=["Anil",2,4,5,6,"Amit","anuj",7,8];
let result=leftToRightArrage(arr);
```

**********Emply of an Array**********

```
let arr=[1,2,3,4,5,6,7,8,8,9,10];
let arr2=arr;
arr2=[];
console.log(arr);
arr2.length=0;
console.log(arr);
```

**********Fibonacci Series**********

```
function fibonacci(num){
    let arr=[0,1]
    for(let i=2;i<num;i++){
       console.log(i);
      arr[i]=(arr[i-1]+arr[i-2]);
    }
    console.log(arr);

}
console.log(fibonacci(20));
```

**********Reverse Number**********

```
let num=123456;
let reverse=0;
while(num !=0){
    reverse=(reverse*10)+(num%10);
    num = parseInt(num/10);
}
console.log(reverse);
```

********** Convert Binary To Decimal**********

```
function decimalToBinary(num){
      let bin=[];
    let final_bn=0;;
    while(num>0){
       bin.push((num%2));
       num=parseInt(num/2);

    }
      for(let i=bin.length-1;i>=0;i--){
```

```
      final_bn = final_bn*10+bin[i];
    }
    console.log(final_bn);
}
decimalToBinary(15);
```

Convert binary to decimal ***********************

```
function convertBinaryToDecimal(n){
    /*let number=1*2*2*2+0*2*2+1*2+0*2;
    console.log(number);*/
    let num = n;
      let dec_value = 0;
      let base = 1;
      let temp = num;
      while (temp) {
      let last_digit = temp % 10;
      temp = Math.floor(temp / 10);

      dec_value += last_digit * base;

        base = base * 2;
      }

   console.log(dec_value);

}
convertBinaryToDecimal(1010);
```

How do you find all pairs of an integer array whose sum is equal to a given number?

```
let twoSum = (array, sum) => {
    let hashMap = {},
      results = []

        for (let i = 0; i < array.length; i++){
            if (hashMap[array[i]]){
                results.push([hashMap[array[i]], array[i]])
            }else{
                hashMap[sum - array[i]] = array[i];
```

```
            }
          }
          return results;
    }
console.log(twoSum([10,20,10,40,50,60,70,30],50));
```

Explanation
- declare two varibale array and object type
- check condition if array element is exit in object then its ia pair so given sum
- else add new value in object as array element with key [sum-arr[elemt]]=element.

How do you find duplicate numbers in an array if it contains multiple duplicates

```
let duplicateCount = (arr) => {
      let dup=[];
    let obj={};
    for(let i=0;i<arr.length;i++){
       obj[arr[i]] = (obj[arr[i]]||0)+1;
       if(obj[arr[i]]>1){
             dup.push(arr[i]);
       }
    }
    return dup;
}
console.log(duplicateCount([1, 4, 8, 2, 4, 1, 6, 2, 9, 7]));
```

One line Exp

```
let duplicateCount = (arr) =>arr.filter((item,index)=>arr.indexOf(item)
!=index);
console.log(duplicateCount([1, 4, 8, 2, 4, 1, 6, 2, 9, 7]));
```

How to remove duplicates from a given array in javascript

```
let duplicateCount = (arr) => {
      let dup=[];
    let obj={};
    for(let i=0;i<arr.length;i++){
       obj[arr[i]]= (obj[arr[i]]||0)+1;
       if(obj[arr[i]]==1){
```

```
            dup.push(arr[i]);
        }
    }
    return dup;
}
console.log(duplicateCount([1, 4, 8, 2, 4, 1, 6, 2, 9, 7]));
```

// Single line Code
Exp.1

```
let duplicateCount = (arr) =>
arr.filter((item,index)=>arr.indexOf(item)==index);
console.log(duplicateCount([1, 4, 8, 2, 4, 1, 6, 2, 9, 7]));
```

Exp.2

```
let arr=[1, 4, 8, 2, 4, 1, 6, 2, 9, 7];
let uniqueArr=[... new Set(arr)];
console.log(uniqueArr);
```

Exp. 3

```
let arr=[1, 4, 8, 2, 4, 1, 6, 2, 9, 7];
let uniqueArray=[];
arr.forEach((item,index)=>{
    if(!uniqueArray.includes(item)){
       uniqueArray.push(item);
    }
});
console.log(uniqueArray);
```

Exp. 4

```
const members = [
  { id: 1, name: 'John' },
  { id: 2, name: 'Jane' },
  { id: 1, name: 'Johnny' },
  { id: 4, name: 'Alice' },
];


const unique = [...new Map(members.map((m) => [m.id, m])).values()];
console.log(unique);
```

Simplify

```
const members = [
  { id: 1, name: 'John' },
  { id: 2, name: 'Jane' },
  { id: 1, name: 'Johnny' },
  { id: 4, name: 'Alice' },
];
let newMem=members.map((item)=>[item.id,item]);
let mp=[...new Map(newMem).values()];
console.log(mp);
```

Search an element from an sorted array With Normal array .

```
let arr=[1,2,3,4,5,10,11,12,13,14,15,16,17,18,19,20];
let searchEle=14;
let start=0;
let end=arr.length-1;

while(start<=end){
    let mid= Math.floor(start+(end-start)/2);
    if(arr[mid]==searchEle){
      console.log(mid);
       break;
    }else if(arr[mid]>searchEle){
       end=mid-1;
    }else if(arr[mid]<searchEle){
       start=mid+1;
    }
```

}

With Rotated array

```
var search = function(nums, target) {
      if(nums.length == 0 || nums == null) return -1;

    let left = 0;
      let right = nums.length-1;
      //console.log('first=',left,right);
      while(left < right){
      let mid = Math.floor((left+right)/2);
```

```
      if(nums[mid]>nums[right]){
            left = mid+1;
      }else{
            right = mid;
      }
      }
      //console.log('Second=',left,right);

    let pivot = left;
      left = 0;
      right = nums.length-1;
      //console.log('Second=',left,right,left);
    console.log(nums[pivot],nums[right],target);
      if(nums[pivot]<=target && target <= nums[right]){
      left = pivot;
      }else{
      right = pivot;
      }
   console.log('Second=',left,right);
      while(left<=right){
      let mid = Math.floor((left+right)/2);
      //console.log(mid , nums[mid] , target);
      if(nums[mid] == target){
            return mid;
      }
      if(nums[mid]<target){
            left = mid+1;
      }else{
            right = mid-1;
      }
      }
      return -1;
};
search([40,50,60,5,10,20,30],10);
```

search element in infinite array

```
// using linear search
let searchEle=112;
let
arr=[10,20,27,24,25,2,6,23,45,6,7,7,8,8,8,6,44,3,3,3,4,4,5,5,55,112,113,114
,115,116,177];
```

```
let i=0;
let searchIndex=-1;
while(arr[i]<=searchEle){
    if(arr[i]==searchEle){
       searchIndex=i;
       break;
    }else{
       i++;
    }
}
console.log(searchIndex);


// using binary saerch
function searchInfinite(array,k){
    let l=0;
    let h=1;
    while(array[i]<=k){
       l=h;
       h=2*h;
    }
    return binarySearch(array,l,h,k);
}

function binarySearch(arr,start,end,searchEle){
    while(start<=end){
       let mid=Math.floor(start+(end-start)/2);
       if(arr[mid]==searchEle){
             return mid;
       }else if(arr[mid]>searchEle){
              start=mid+1;
       }else if(arr[mid]<searchEle){
             end=mid-1;
       }
    }

}
```

Find First and Last occurrence of an array using binary search.

```
function firstElement(arr,elem){
```

```
    let l=0;
    let r=arr.length-1;
    let firstIndex=-1;
    while(l <= r){
       let mid=Math.floor((l+r)/2);
       console.log(mid);
       if(arr[mid]==elem){
             firstIndex=mid;
             r=mid-1;
       }else if(arr[mid]>elem){
             r=mid-1;
       }else if(arr[mid]<elem){
             l=mid+1;
       }
       return firstIndex;
    }


}
 function lastElement(arr,elem){
    let l=0;
    let r=arr.length-1;
    let lastIndex=-1;
    while(l<=r){
       let mid=Math.floor(l+(r-l)/2);
       if(arr[mid]==elem){
             lastIndex=mid;
            start=mid+1;
       }else if(arr[mid]>elem){
             end=mid-1;
       }else if(arr[mid]<elem){
             start=mid+1;
       }
    }
    return lastIndex;
}

 let array=[1,4,4,10,10,10,10,15,20];


let startIndex=firstElement(array,10);
let endIndex=lastElement(array,10);
let count=endIndex-startIndex;
```

```
console.log(count);
```

Binary Search to Find the Rotation Count in a Rotated Sorted array.

Step:

1. If the middle element is smaller than its previous element, then it is the minimum element
2. If the middle is greater than its next element, then the next element is the minimum element
3. left array is sorted. So the pivot (min element) is on the right side
4. right array is sorted. So the pivot (min element) is on the left side
5.

First negative integer in every window of size k

```
function printFirstNegativeInteger(arr , n , k){
    let allNegative=[];
    let firstNegative=[];
    let i=1;
    for(let f=0;f<k;f++){
       if(arr[f]<0){
            allNegative.push(arr[f]);
       }
    }
      for(let j=k;j<n;j++){
       if(arr[j]<0){
             allNegative.push(arr[j]);
       }
       if(j-i+1==k){
             if(allNegative.length > 0){
                   firstNegative.push(allNegative[0]);
             }else {
                   firstNegative.push(0);
             }

             if(arr[i-1]==allNegative[0]){
```

```
                    allNegative.shift();
                }
                i++;

                console.log(allNegative.length);
            }
        }
        if(allNegative.length==0){
            firstNegative.push(0);
          }

        console.log(firstNegative);
}

var arr = [ 12, -1, -7, 8, -15, 3, 1, 7, 30, 16, -28,23,34,45 ];
var n = arr.length;
var k = 3;
printFirstNegativeInteger(arr, n, k);
```