Validate data in javascript

validatedropdown	In dropdown box it stores property by name selectedIndex, it stores the index position of the selected option
	
	It also stores a array by name options, hence to find the value of selected dropdown box in javascript
	
	//get reference of dropdown box centers
	var c=document.getElementById(id);
	//retrieve the value of index of the selected option
	var pos=c.selectedIndex;
	//alert(c.selectedIndex)
	//retrieve the value of selected option
	var val=c.options[pos].value;

validateradio	to find which radio button is checked we use checked property, In radio button we can select only one option, so as soon as we get the checked radio button we may break the loop
	const validateradio=(nm,errid)=>{
	var arr=document.getElementsByName("gender");
	for(var i=0;i<arr.length;i++){
	if(arr[i].checked){
	document.getElementById(errid).innerHTML="";
	return true;
	}
	}
	document.getElementById(errid).innerHTML="pls select one choice"
	return false;
	}

validatecheckbox	to find which check is checked, we use checked property, checkboxes are used to select multiple options, we use the checked property to find which checkboxes are checked
	const validatecheckbox=(nm,errid)=>{
	var subarr=document.getElementsByName(nm);
	var cnt=0;
	var str="";
	for(var i=0;i<subarr.length;i++){
	if(subarr[i].checked){
	cnt++;
	str=str+subarr[i].value+",";
	}
	/*if(cnt>=2){
	break;
	}*/
	}
	if(cnt>=2){
	//clear the message
	document.getElementById(errid).innerHTML="";
	alert(str);
	return true;
	}

	//display error message
	document.getElementById(errid).innerHTML="pls select minimum 2 subjects"
	return false;
	}

validatefile	to validate file size, we may use size property
	
	function validateSize(){
	alert("in validatesize")
	var f=document.getElementById("files");
	//f.files[0].size it will give size in bytes
	alert(f.files[0].size/1024+"KB");
	if(f.files[0].size/1024 < 10){
	alert("in if")
	document.getElementById("errfile").innerHTML="";
	//return true;
	}
	document.getElementById("errfile").innerHTML="file size exceeds the limit 10KB"
	//return false;
	}

operators

Arithmetic operators	+, -, *, / , % , **(exponential) 3**3=27 (3 raise to 3)

relational operator	>, <, <=,>=, == , ===, !=, !==
	
	=== → strict checking
	10===”10” false //it check value as well as data type
	10===10 true
	10==”10” true
	10==10 true

logical operators	&& , || , !

Functions

a.toFixed(num)	to display the number upto 2 decimal places a=3.3333333 a.toFixed(2)= 3.33
Math.round(num)	it will round the number Math.round(3.456778)= 3 Math.round(3.6789)=4

String functions

s.trim()	to remove leading and trailing spaces	s=” aaaaa “
		s.trim()
		aaaaa

s.length	to find length of the string	s.length
s.charAt(i)	to find the character at specific index position	s=”testing” s.charAt(4)=i

s.concat(str1,str2,…..)	to concat multiple strings	s=”taste”
		s.concat(“is”, “good”)
		taste is good

s.indexOf(str)	to find the position of first occurrence of the given str, indexing starts with 0	s=”Hello World!!”
		s.indeXOf(“Hello”) //0
		s.indexOf(“World) //6

s.split(delimiter)	to break the string into array of strings at delimiter position	s=”rain in spain, is pain” s.split(“ “) [“rain”, “in”,”spain,”,”is”,”pain”] s.split(“,”) [“rain in spain”,” is pain”]

s.toUpperCase()	convert the string into uppercase	s=”abcd”
		s.toUpperCase()
		ABCD

s.toLowerCase()	convert the string into lowercase	s=”ABcd”
		s.toLowerCase()
		abcd

s.slice(startIndex[,endIndex])	it retrieves the portion of the string from startIndex to endIndex, endIndex is excluded	s=”Hello world!!”
		s.slice(0,1) //H
		
		s.slice(4) //o world!!

s.includes(substr)	it returns true, if substring exists, false otherwise	s=”Hello world!!”
		s.includes(“ell”) //true
		
		s.includes(“xxx”) //false

using regular expression

s.match(regular expression)	it checks the specified value in the string, and return all occurences	s=”rain in SPAIN is pain” pattern=/a.*?n/ s.match(pattern) //2 pattern=/a.*?n/i -→ values enclosed in / / is treated as regular expression i is flag for ignore index g is a flag for global s.match(pattern) //2
s.replace(regexpression, newstring)	It will search the given pattern and replaces it with the newstring	s=”To be or not to be” s.replace(/be/,”exists”) –>replace first occurence s.replace(/be/g,”exists”)-> flag g, will replace all occurences
s.search(regexpression)	it checks the specified pattern in the string, and	var test=”testing: 1, 2, 3” var pattern= /\d+/g

	return the position of first occurrence	s.search(pattern) //9
		s.match(pattern) // [“1”,” 2”,”3”]
		s.replace(pattern,”#”) // testing: #, #,#

Arrays in Javascript

In javascript size of array grows or shrinks dynamically.

Array allows to store heterogeneous data

To create a array

var a=new Array(2,3,4,”xxxxx”)

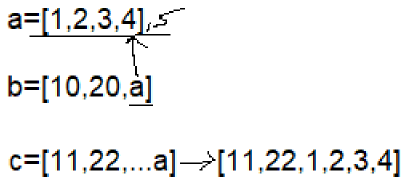
var b=[1,2,3,”xxxxx”]

to create a copy of the array

var c=[1,2,3,...a]   //... is a spread operator of array, length of c is 7

To copy the reference

var d=[1,2,3,a]   //length of d is 4



a=[1,2,3,4],5

b=[10,20,a]

c=[11,22,...a]→[11,22,1,2,3,4]

arr.push(val)	to add the value at the end of the array	
arr.pop()	to delete the value from the end	
arr.unshift(val)	to add the value at the beginning	
arr.shift()	to delete the value from the beginning	
arr.splice(position, number)	delete the number of elements starting from the given position	arr=[1,2,3,4,5,6] arr.splice(3,2) //4 and 5 will be deleted [1,2,3,6]
arr.splice(position, number,list of values)	delete the number of elements starting from the given position, and replace it with the list of values	arr=[1,2,3,4,5,6] arr.splice(3,2,100,200,300,400,500) //4 and 5 will be replaced by the list of values [1,2,3, 100,200,300,400,500,6]

arr.splice(position, 0,list of values)	to add the values at the given position	arr=[1,2,3,4,5,6]
		arr.splice(3,0,100,200,300,400,500) //all the values will be added at the position
		[1,2,3, 100,200,300,400,500,4,5,6]

arr.indexOf(value)	find the position of the first occurrence of the given value
	This function is useful when the value is known
		arr=[12,13,12,15,13]
		arr.indexOf(13)
		1

arr.findIndex(predicate function)	predicate function -→ accepts one parameter and returns true / false
	
	findIndex function will find the position of the value for which predicate function returns true
		arr=[12,13,12,15,13,20]
		
		//to find index of first value which is divisible by 5
		
		//findindex will give the index of 15,
		
		arr.findIndex((val,index,arr)=>val%5==0)
		//use return keyword inside {}
		arr.findIndex((val,index,arr)=>{return val%5==0})
		
		arr.findIndex(val=>val%5==0)

arr.find (predicate function)	predicate function -→ accepts one parameter and returns true / false
	
	find function will find the value in the array for which predicate function returns true
		arr=[12,13,12,15,13,20]
		
		//to find the first value which is divisible by 5
		
		//find will give the value 15,
		
		arr.find ((val,index,arr)=>val%5==0)
		//use return keyword inside {}
		arr.find ((val,index,arr)=>{return val%5==0})
		
		arr.find (val=>val%5==0)

arr.filter(predicate function)	predicate function -→ accepts one parameter and returns true / false
	
	filter function will find the all the values for which predicate function returns true
		arr=[12,13,12,15,13,20]
		
		//to find all values which is divisible by 5
		
		//filter will give the array of all values which are divisible by 5
		
		arr.filter((val,index,arr)=>val%5==0)
		//use return keyword inside {}
		arr.filter((val,index,arr)=>{return val%5==0})
		
		arr.filter(val=>val%5==0)

arr.map(coverter function)	map function will apply the given expression on every value in the array and return a new value
	filter function will find the all the values for which predicate function returns true
		arr=[12,13,12,15,13,20]
		
		//to find squares of all the numbers
		
		arr.map((val,index,arr)=>val*val)
		//use return keyword inside {}
		arr.map((val,index,arr)=>{return val*val})
		
		arr.filter(val=>val*val)