**J A V A S C R I P T**

**ARRAY**

const arr = [0,1,2,3,4,5];

console.log(arr);

const arr2 = ['a',1 ,'b', 2 ,'c' ,3];

console.log(arr2);

// result :- array  supprot both numeric and string character

//new way to declare array

// array give shallow copy . shallow :-refrence & deep pass copy

const arr3 =new Array(5,8,6,9,7);

console.log(arr3);

console.log("-----------------------------------------------------------");

// methods of array....

console.log("methods of array");

// push() method ... it will add element at last index..

 arr.push(6);

 console.log (arr);

 console.log("-----------------------------------------------------------");

 //pop() :- remove last element of array...

 arr.pop();// don't pass any element or index..

 console.log(arr);// here it removes the 6 th value

 arr.pop(3);// it ignore the parameter.

 console.log(arr); // del the last value of array which is 5.

 console.log("-----------------------------------------------------------");

 // unshift():- this method adds value in starting index of array....

 /\*unshift is not good practice each element has to

  be when element added to starting index then each element has to be shift 1 step

  from there place by which load comes on computer , time increases if the size

  of array is greater like 10000

   \*/

  arr.unshift(9);// 9 added starting index

  console.log(arr);

  console.log("-----------------------------------------------------------");

  // shift() method :- IT removes the starting index value .No argument is passed.

  arr.shift();// remove 9

  console.log(arr);

  console.log("-----------------------------------------------------------");

  // include ():- used to check weather the particular element present in array or not .

  console.log(arr.includes(9));//boolean ans comes

  console.log("-----------------------------------------------------------");

  //indexOf():- used to index of particular element ..

  console.log(arr.indexOf(9));// if element not found then it return -1 index

  console.log(arr.indexOf(3));

  console.log("-----------------------------------------------------------");

  // join():-adding all the element of array into a string ...

  const newarry = arr.join();

  console.log(newarry);

  console.log(typeof(newarry));// it bind the array and coverted into the string ...

  console.log("-----------------------------------------------------------");

  // slice ();- return a copy of a section of an array..

  console.log("original array before splice and slice method =" , arr);

  const myarr = arr.slice(1,3);// 3 is not included ....

  console.log("original array on using slice=" , arr);

  console.log(myarr);

  console.log("-----------------------------------------------------------");

//splice ():- takes out the value from array... as shown in output ...

  const myarr2 = arr.splice(1,3);// 3 is included ....

  console.log("original array on using splice=" , arr);

  console.log(myarr2);

  /\* what is the difference between slice and splice....

  slice :- don't manipulate original array

        :- don't include last index..

  splice:- monipulate original array gives output by out the range of element from the array..

        :- include the last index also..

\*/

console.log("-----------------------------------------------------------");

const marvel = ["iron man " , 'hulk' , " thor " ,"spiderman"];

const dc = ["flash" , "superman" , "batman" , "wonderwomen"];

marvel.push(dc);// here it take dc array as data, dc array behaves like single element in marvel..

console.log(marvel);// array can take any kind of entry data.

console.log("------------------------------------------------------");

console.log("value at last index of marvel=" ,marvel[4]);

console.log("------------------------------------------------------");

// concate():- combines two or more array and returns new array...

const allheros=marvel.concat(dc);

console.log(allheros);

console.log("------------------------------------------------------");

//push():-pushes data in an array whereas concatenate returns new array.....

// there is one more easy method to add up the array...

// spred method here each element get spread and becomes an indiviual element (NOT AN ARRAY)

const newarray1= [...marvel , ...dc];

console.log(newarray1);// in marvel we add dc therefore it treated as a indinvidual element as shown in o/p.

console.log("------------------------------------------------------");

//rare situtation...

const anotherarr =[1,2,3, [4,5,6] ,7 , [6 , 7 , [4 , 5]] ,8,];

//flat():-another array can be flated ...  in argument depth is pass if infinity pass then whole array flst.

const realarr = anotherarr .flat(Infinity);

console.log(realarr);

//another way of spread...

const check = marvel.flat(Infinity);

console.log(check);

console.log("------------------------------------------------------");

//Array.isArray(){Array.is , Array.from , Array.of }

/\* when accesing a data or select data from webpage then it comes in another form

like node , list , object but we need array so in such situation use Array ..

we can check also weather it is array or not and also convert it also ..

\*/

// EXAMPLE..............

console.log(Array.isArray("eshan"));

// FOR COVERTING IN ARRAY.......it can covert object , node , list etc into array

console.log(Array.from("tiwari"));

//interesting case for interview

console.log(Array.from({name: "eshan"}));// it will empty array as it not coverting it directly...

// here we have to tell weather we want to make key as array or value as array,if it cant make then it give empty arr.

//covering multiple variable in array....

let score1=100;

let score2=200;

let score3=300;

console.log(Array.of(score1 ,score2 ,score3));

console.log("------------------------------------------------------");

// singleton

/\* it is created by the constructor

jab literal se declare karte hai tab ni

bunta haai

 \*/

//object literal

const jsuserdemo = {};//KITERAL

// another of creating object is

console.log("------------------------------------------------------");

Object.create// it is made through constructor also called singleton

// its notes is in dairy .

const jsuser = {

    name: "eshan",

    age :15,// already consider as no.

    location: "barabanki",

    email: "eshan123@gmail.com",

    isloggedIn:false,

    lastLoginDays:["monday" , "saturday"],//array(mutiple value in a single var)

}

/\* here name is treated as string "name"

if write name as "name" then u can't use (.) operator

they are with square notation \*/

// access of object....................

console.log(jsuser);

console.log(jsuser.email);// not a good way to access

//console.log(jsuser[email]), here error comes bcoz email is treated as string(string ki terha rakha ja raha) so it access through string

console.log(jsuser["email"]);//you can acces throug square notation(good way of accessing )

console.log("------------------------------------------------------");

const jsuser2={

     name:"harsh",

    "full name":"eshan tiwari",

}

// now we cant acces this value with dot

//console.log(jsuser2.full name);that's why dot is not good way to access obj

console.log(jsuser2["full name"]);

//Q1) take symbol key in obj and print it...

/\* we cant use symbol directly as key in object  \*/

console.log("------------------------------------------------------");

const mysyn = Symbol("mykey1")

const jsuser3 = {

    name: "eshan",

    age :15,// already consider as no.

    //mysym:"mykey1",// accessing symbol as key in obj , TO USE IT AS SYMBOL WRITE IT UNDER SQUARE BRACKET

    [mysyn]:"mykey1",// symbol declare in object...

    location: "barabanki",

    email: "eshan123@gmail.com",

    isloggedIn:false,

    lastLoginDays:["monday" , "saturday"]

}

//console.log(jsuser3.mysym); for accessing it use it with square notation

//console.log(typeof(jsuser3.mysym));// here you can see that symbol is behaves as string ,not uses as symbol

console.log(jsuser3[mysyn]);

//freeze (f : Function):

/\*object on which to lock the attributes from the modifications \*/

Object.freeze(jsuser3);

jsuser3.email="harsh123@gmail.com";  //this value  will not propagate as it frezess before

console.log(jsuser3);  // here in output it show [mysyn] as symbol not string...

console.log("------------------------------------------------------");

//FUNCTION................................................................................................

jsuser.greeting = function(){

    console.log('hello js user');

}

// accessing greeting in jsuser3

console.log(jsuser.greeting);      //[Function (anonymous)] comes which means function not executed just its reference comes

console.log(jsuser.greeting());

// making another function , making refrence as name..........................

console.log("------------------------------------------------------");

//FUNCTION :- (2) ............................................................

jsuser.greetingtwo = function(){

    console.log(`hello js user, ${this.name} `);

}

console.log(jsuser.greetingtwo);

console.log(jsuser.greetingtwo());

/\* NOTE:-

acces the key of obj through dot (.) operator

but we can acces with square notation

there are some special condition where we need to use the square notation

like case of symbol \*/

//here dicuss how onject singleton and by constructor.......

const tinderUser = new Object();// singleton object , o/p = {}

const tinderuser2 = {} ;// non singleton object , o/p = {}

tinderuser2.id ="123abc";

tinderuser2.name = "sammy",

tinderuser2.isLoggedIn = false;

console.log(tinderuser2);

//object k ander object . . . . . .

const regularuser = {

    email: "harsh123@gmail.com",

    fullname :{

        userfullname:{

            firstname : "e s h a n",

            lastname : "t i w a r i"

        }

    }

}

console.log(regularuser);

console.log("------------------------------------------------------");

console.log(regularuser.fullname);// opening the nesting...............

console.log("------------------------------------------------------");

console.log(regularuser.fullname.userfullname.firstname);// opening more nesting.

//if fullname is not exist and we try to access it then in that case we use (?).

// ? ternary operator ...

console.log(regularuser.fullname?.userfullname.firstname);

console.log("------------------------------------------------------");

// combine two different object....

const obj1 = { 1: "a" , 2: "b"};

const obj2 = { 3: "a" , 4: "b"};

const obj3 = { obj1 , obj2}; //prblm:- object k under he object ho jayega

const obj4 = Object.assign({} ,obj1 , obj2);//(target , source)

/\*{} -> ye target h jisme sare source ko merge ker diya jata hai yeah starting me

emty lete hai taki ager multiple array ho to merge ker de

ager obj1 lenge starting me to obj1 me he sare value daal k return ker dega\*/

//obj1 & obj3 => source...

console.log(obj3);

console.log("------------------------------------------------------");

console.log(obj4);

console.log("------------------------------------------------------");

// IN GENERAL WE USE SPREAD OPERATOR .....

const obj6= { 7: "a", 8: "b" };

const obj7= { 9: "a", 10: "b" };

// Merging objects using spread operator

const obj5 = { ...obj1, ...obj2 };

console.log(obj3);

console.log("------------------------------------------------------");

// this way the value comes from the database then they comes in array form.

//A R R A Y    O B J E C T

const user =[

    {

        id: 1,

        email:"h@gmail.com"

    },

    {

        id: 1,

        email:"h@gmail.com"

    },

    {

    },

]

user[1].email

console.log(user);

console.log(tinderuser2);

// accessing keys

console.log(Object.keys(tinderuser2));//o/p datatype is array taking all o/p and put it into the array

console.log(Object.values(tinderuser2));

console.log(Object.entries(tinderuser2));

/\* some time loop through the object for any value retrival

some time value may not exist due to which crash chance may occurrs

so u ask it either that value is present or not  \*/

console.log(tinderuser2.hasOwnProperty('isLoggedIn '));

const course = {

    coursename: "js in hindi",

    price :"999",

    courseInstructor: "hitesh"

}

//course.courseInstructor (bar bar(.)laga k use kerna clear or simple ni hota h)

 const {courseInstructor} = course;

 console.log(courseInstructor);

// for changing the name of courseInstructor

const {courseInstructor:instructor} = course;

 console.log(instructor);

 // method of function

 //{} means to doing de structing

 //    A P I

 /\* google login we don't have headache that google will verify that account

  some values are comes from the backend and how will u write it

   earlier value comes through xml that were are very complex

   now all the values comes in json\*/

   //json api

  /\* {

    "name": "harsh",

    "course": "btech",

    "price" : "free"

   }

[

    {},

    {},

    {},

]\*/

function saymyname(){

    console.log("R");

    console.log("A");

    console.log("M");

}

// saymyname();

function addtwonum( num1 , num2){//parameters:- passed in function definition

    console.log(num1+num2);

}//function call

addtwonum( 5 , 46);//arguments:- passed in function call

addtwonum( 5 , "46");

addtwonum( 5 , "a");

addtwonum( 5 , null);

function subtwonum (numa ,numb){

    let result =numa-numb;

    return result;

    console.log("eshan"); // return k baad function kaam ni karega

}

const result =subtwonum (10 ,5);

 console.log("RESULT :" , result);

 function multwonum (numa1 ,numb2){

    return numa1 \* numb2;

 }

 const result2 =multwonum (10 ,5);

 console.log("RESULT :" , result2);

 function loginUserMessage (username){// can pass default value like username = "eshan" if any value pass in fun call then it overide default val

    if(username == undefined){// you can also write !username

        console.log("please enter a username")

        return;

        //fir baki code ni chalega like return wala code na chale

    }

    return `${username} just logged in `;

 }

 /\*loginUserMessage ("harsh"); here function executed and the value is get retuned

 but don't said to that to fuction to print the value \*/

console.log( loginUserMessage ("harsh"));// here the value which is return get a print instruction.

// empty string -----------

console.log( loginUserMessage ("")); // just logged in print ------

//if no value is passed------------

console.log( loginUserMessage ()); // UNDEFINED comes when if condition is not used

function calculateCartPrice(num1){

    return num1;

}

console.log(calculateCartPrice(200 , 400 , 500));// there are certain many value in cart but it give first value

// to solve this problem use rest opertor or spread operator (...)-> sabko ek bundle me pack kerta h or de deta h

function calculateCartPrice2(...num2){//... in function it is rest operator

    return num2;

}

console.log(calculateCartPrice2(200 , 400 , 500));//here we get the array,or array me loop laga k add kr do

function calculateCartPrice3(val1 , val2, ...num3){

    return num3;

}

console.log(calculateCartPrice2(200 , 400 , 500 , 2000));// here val1 = 200 , val2 = 400 , and rest operator

// how to pass object in function and how does it usee .........

const user = {

    username:"eshan",

    price: 199

}

function handleObject(anyobject){

    console.log(` user name is ${anyobject.username} and price is ${anyobject.price}`);

}

handleObject(user);

// don't need to create the object every time you can pass it as in function call

function handleObject2(anyobject){

    console.log(` user name is ${anyobject.user} and price is ${anyobject.rollnum}`);

}

handleObject2({

    user:"sab",

    rollnum:31,

});

// how to pass ARRAY in function and how does it usee .........

const mynewarr=[200 , 400 , 100 , 600];

function returnSecondValue(getArray){

    return getArray[1];

}

console.log( returnSecondValue(mynewarr));

// don't need to create the array every time you can pass it as in function call

function returnSecondValue2(getArray2){

    return getArray2[1];

}

console.log( returnSecondValue2([200 , 400 , 500 , 1000]));

let a = 10;

const b = 20;

var c = 30;

//{}=SCOPE

if(true){

    let d = 40;

    const e=50;

    var f = 60;

// these variables are valid within the scope

}

console.log(a);

console.log(b);

console.log(c);

//console.log(d); // o/p comes as not defined

//console.log(e);

console.log(f);// problem :-but this can be printed

/\* If all variable are working properly then what need to brought the

const var  it brought bcoz they not work as block\*/

var f= 500;//global scope

if(true){// block scope

    var f = 30;

}

console.log(f);

// if here let use then this problem not occurs

let g= 500;//global scope

if(true){// block scope

    let g = 65;

}

console.log(g);

/\* brower inspect scope is different and

in code environment with node js is different \*/

//nested scope:- here child function can access parent function

function one(){// here one is greater

    const username = "eshan"

    function two(){ // here two can use the one variables

        const website = "youtube"

        console.log(username);

    }

    //console.log(website);(error occur ) jobe varaibale two k ander declare huai hai wo wahi katum ho gaye hai

    two();

}

one();

if (true){

    const name ="harsh";

    if(name == "harsh"){

        const website = "youtube";

        console.log(name + website);

    }

   // console.log(website);(error as it use out of scope)

}

//console.log(name);(error as it use out of scope)

//++++++++++++ intresting +++++++++++++++++++++

console.log(addone(5));// no error on call before function decleration

function addone (num){

    return num +1;

}

//another way of declaring the function

//console.log(addtwo(5)): here error comes bcoz here we hold the function in variable

const addtwo= function(num){

    return num +2;

}

console.log(addtwo(5));

const user = {

    username: "hitesh",

    price: 999 ,

    welcomeMessage : function(){

        // { username}:- here we can't directly pass it over here insted of that use this key word

        console.log(`${this.username} , welcome to website`);

        // in arrow function no this key word is present

        console.log(this);//gives the current context

    }

}

user.welcomeMessage();

user.username = "sam";

user.welcomeMessage();

//if we here prit the current context in node environment then

// this key word refers to empty object , abhi global me koi context he ni h

console.log(this);

function chai(){

    console.log(this);

}

//chai();

function chai2(){

    let username = "harsh2"

    console.log(this.username);

}

chai2();// output is undefined , here we can see that this is works under the function

const chai3 = ()=> {

    let username = "harsh3"

    console.log(this);

}

chai3();

/\* what is difference between normal function and arrow function

 \*/

//ARROW Function

const addTwo = (num1 , num2 )=>{

    return num1 + num2 ;

}

console.log(addTwo(3 , 4));

// emplicit return :- return key word likne ki jarurat ni h bcoz ek he line ka statement h

//const mulTwo = (num1 , num2 ) =>  num1 \* num2 ; this also work

// '{' curly braket likha mtlb ki return use kerna hai or agr "(" parenthesis likha to ni krna

const mulTwo = (num1 , num2 ) => ( num1 \* num2);

const obj =() => ( { username:"eshan",});

console.log(mulTwo(3 , 4));

console.log(obj());

function chai (){

    console.log(`DB CONNECTED`);

}

chai();

/\*IMMEDIATELY INVOKE FUNCTION EXPRESSION (IIFE)

 ->GLOBAL scope se cheeze polute hote hai or wo polution

 particular function ko effect na kere use liye IIFE likte hai

 -> global scope k polution ko hatata hai

 ->function immediatly execute kerta hai (this line is not for interview)

\*/

(function chai2 (){

    //named iife

    console.log(`DB CONNECTED`);

})();// function ko rap kiya parenthesis me or ececute

//firt ():- here function definition is writen

//second():- it for the execution call

// using arrow function

( () => {

    //un named iife

    console.log(`DB CONNECTED ho gaya`);

})();//do iife sath me to semi colon lagao

// how we passs parameter

( (name) => {

    console.log(`DB COnect kero ${name}`);

})('eshan');

//loop is also called iteration.........

console.log("FOR LOOP");

for(let index = 0 ; index <10; index++){

    const element = index;

    console.log(element);

}

console.log("----------------------------------------------");

for(let i =0; i<=10 ; i++){

   // console.log(`outer loop value : ${i}`);

    for(let j=0 ;j<=10 ; j++){

        //console.log(`inner loop value:${j}`);

        //console.log(i+ '\*' + j + '=' + i\*j); print table form 1 to 10

    }

}

console.log("----------------------------------------------");

let myarray =[ 'flash' , 'batman', 'superman'];

for(let index =0 ; index < myarray.length ; index++){

    const element = myarray[index];

    console.log(element);

    console.log(myarray.length);

}

console.log("----------------------------------------------");

console.log("break & continue ")

// BREAK

for (let index = 1 ; index<=10 ; index++){

    if(index == 5)

    {

        console.log(`Detected 5`);

        break;

    }

    console.log(`value of i is ${index}`);

}

console.log("----------------------------------------------");

//CONTINUE :- EK MAR MAAF KER DO OR IGNORED OR SKIPED

for (let index = 1 ; index<=10 ; index++){

    if(index == 5)

    {

        console.log(`Detected 5`);

        continue;

    }

    console.log(`value of i is ${index}`);

}

console.log("----------------------------------------------");

console.log("while loop");

//WHILE LOOP..........

let index = 0;

while(index <= 10 ){

    console.log(`value of index is ${index}`);

    index = index +2;

}

//by ARRAY

let myarr =[ 'flash' , 'batman', 'superman'];

let i = 0;

while (i < myarr.length){

    console.log(`value is ${myarr[i]}`);

    i=i+1;

}

console.log("----------------------------------------------");

console.log("do while loop");

//do while

let score = 1;

do {

    console.log(`score is ${score}`);

    score++;

}while(score<=10);

console.log("----------------------------------------------");

// for of

["" , "" ,""];// array k ander bhut se string

[ {} , {} , {}] ;// array k ander bhut se object

/\*

her iteration me object se koi value nikalne hai like in this case

for of loop are much better

\*/

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

for(const num of arr){

    console.log(num);

}

const greeting = "hello world";

for(const greet of greeting){

    console.log(`Each char is ${greet}`);

}

// ....... M A P ........

// map is the object that hold an key value pair , THEY ARE UNIQUE

const map = new Map();

map.set ('IN' , " India " );

map.set ('USA' , " United state of america " );

map.set ('FR' , "france" );

map.set ('IN' , "India" );//it will not repeat as map are unique

console.log(map);

//using for of loop in map

 for (const key of map ){

    console.log(key);//in output we get array but we don't want this

 }

// to avoid above problem

 for (const [key2,value] of map ){

    console.log(key2,"=" ,value);

 }

 // can we use for loop on object ?

 const myobj ={

    "game 1":"bgmi",

    "game 2":"cod",

    "game 3":"spiderman",

 }

//for(const [key3 , value3] of myobj){

   // console.log(key3 , ' :- ' ,value3 );//this object is not iterable

// so here we use for in loop}

const myobj2 = {

    js : 'javascript',

    cpp : 'c++',

    rb : 'ruby',

    swift : 'swift by apple'

}

//for in loop

for (const key4 in myobj2){

    console.log(key4);

    console.log(myobj2[key4]);

    console.log(`${key4} shortcut is for ${myobj2[key4]}`)

}

//using for in loop in array

const programming = ['js' , 'rb' , 'py' , 'java' , 'cpp' ];

for (const k in programming){

    console.log(k);

    console.log(programming[k]);// geting value from key kkjk

}

/\*

array ki key num he hote hai  joki 0 se start hote hai

\*/

// can we use for in loop in map

for (const key in map){

    console.log(key);// not iterable

}

// for each loop

const coding = ['js' , 'python' , 'java' , 'ruby' , 'cpp' ] ;

//for (const key in coding){}

// function ander declare hoga without name

coding.forEach( function (item) {

    console.log(item);

})

//USING ARRAY FUNCTION

coding.forEach( (val) => {

    console.log(val);

})

function printMe(item){

    console.log(item);

}

coding.forEach(printMe);

coding.forEach((item , index , arr)  => {

    console.log(item , index , arr);

})

const mycoding = [

    {

        lang:"js",

        file:"java"

    },

    {

        lang:"js",

        file:"java"

    },

    {

        lang:"python",

        file:"py"

    },

]

mycoding.forEach((item) => {

    console.log(item.file);

})

const coding =['js' , 'ruby' , 'java' , 'python' , 'c++'];

const values = coding.forEach((item) => {

    console.log(item);

})

console.log(values);

//as this function don't returns any value therefore in output we get undefined

// but in case of for each loop we always get undefined bcoz for each don't returns

// que retun only if java but in this case we don't prefer for each loop

const mynums =[1,2,3,4,5,6,7,8,9,10];

//filter be apne ander call back leta hai

const newnums =mynums.filter((num)=> num >4);// if scope not open then directly write num>4

const newnums2 =mynums.filter((num)=> {// here scope start

    //if scope start then write the return key word

    return num >6

});

console.log(newnums);

console.log(newnums2);

const newarr=[];

mynums.forEach((num)=> {

    if(num>2){

        newarr.push(num)

    }

})

console.log(newarr);

const books = [

    { title: 'Book One', genre: 'Fiction', publish: 1981, edition: 2004 },

    { title: 'Book Two', genre: 'Non-Fiction', publish: 1992, edition: 2008 },

    { title: 'Book Three', genre: 'History', publish: 1999, edition: 2007 },

    { title: 'Book Four', genre: 'Non-Fiction', publish: 1989, edition: 2010 },

    { title: 'Book Five', genre: 'Science', publish: 2009, edition: 2014 },

    { title: 'Book Six', genre: 'Fiction', publish: 1987, edition: 2010 },

    { title: 'Book Seven', genre: 'History', publish: 1986, edition: 1996 },

    { title: 'Book Eight', genre: 'Science', publish: 2011, edition: 2016 },

    { title: 'Book Nine', genre: 'Non-Fiction', publish: 1981, edition: 1989 },

];

// Filtering books with genre 'History'

const userBooks = books.filter((bk) => bk.genre === 'History');

console.log(userBooks);

console.log("---------------------------------------------");

// Filtering books published after 2000

const userBooks2 = books.filter((bk) => bk.publish >= 2000);

console.log(userBooks2);

console.log("----------------------------------------------");

// Filtering books of genre 'History' published after 1995

const userBooks3 = books.filter((bk) => {

    return bk.publish > 1995 && bk.genre === 'History';

});

console.log(userBooks3);

const mynum = [1, 2, 3, 4, 5, 6,7, 8,9 ,10];

//it automatically return the value

const newnum = mynum.map((num) => num + 10)

console.log(newnum);

console.log("-------------------------------------------");

// if we use scope then get undeined not empty array

const newnum2 = mynum.map((num) =>{ return num + 10})

console.log(newnum2);

console.log("-------------------------------------------");

// chaining method :- here we can use multiple method at a time

//like map lagaya fir dubara map lagaya fir filter laga diya

const newnum3=mynum

.map((num)=> num\*10)

.map((num)=> num +1)

.filter((num)=>num>=40)

console.log(newnum3);

console.log("-------------------------------------------");

//reduce method.........................

const myn = [1,2,3];

const myt = myn.reduce(function (accumlator , currentval){

    console.log(`acc: ${accumlator} and currval: ${currentval}`);

    return accumlator + currentval;

})

console.log("-------------------------------------------");

const myt2 = myn.reduce(function (accumlator , currentval){

    console.log(`acc: ${accumlator} and currval: ${currentval}`);

    return accumlator + currentval;

},0)

console.log(myt2);

console.log("-------------------------------------------");

const myt3 = myn.reduce(function (accumlator , currentval){

    console.log(`acc: ${accumlator} and currval: ${currentval}`);

    return accumlator + currentval;

},3)

console.log(myt3);

console.log("-------------------------------------------");

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Dom learning </title>

    <style>

        .bg-black{

            background-color: #212121;

            color : rgb(255, 128, 0);

        }

    </style>

</head>

<body>

    <div class = "bg-black">

        <h1 id = "title" class="hello"  > DOM (document object model)learning on chai aur code

            <span style = "display: none;">test text</span>

        </h1>

        <h2> hello world </h2>

        <h2> hey computer  </h2>

        <h2> java script is life  </h2>

        <p>

            <P> dom :- document onject model means jo ye apka document bana hua hai

                wo kis model ka bana hua hai , kya uska structure hai , page kaisa dekta

            </P>

            <p>

                google -> inspect ->console -> console.log (window) or ->

                console .dir(document);

                html collection seems like an array but they are not a array

            </p>

            <ul>

                <li class = "list" >one</li>

                <li class = "list" >two</li>

                <li class = "list" >three</li>

                <li class = "list" >four</li>

            </ul>

            <input type = "password" name="" id="">

            <input type = "text" name="" id="">

            <input type = "submit" name="" id="">

    </div>

</body>

</html>

<!--

    INSPECT CONSOLE OPERATIONS .....

i/p = document.getElementById('title')

o/p = <h1 id=​"title" class>​ DOM (document object model)learning on chai aur code​</h1>​

i/p = document.getElementById('title').id

o/p = 'title'

i/p = document.getElementById('title').class

o/p = undefined

here we don't write class directly instead of that we write class name

i/p = document.getElementById('title').getAttribute

o/p = ƒ getAttribute() { [native code] }

i/p = document.getElementById('title').getAttribute('class')

o/p = 'heading'

i/p = document.getElementById('title').getAttribute('id')

o/p = 'title'

note = document.getElementById('title').setAttribute('class' , 'test') , yaha hello se test ho gaya

i/p = document.getElementById('title').setAttribute('class' , 'test')

o/p = undefined (output comes this but actually the value get change)

#storing in variables

i/p = const title = document.getElementById('title')

o/p = undefined

i/p = title.style.backgroundColor = 'green '

o/p = 'green '

i/p = title.style.padding = "15px"

i/p = title.innerText:- jo dek raha h ushi k hisaab se change karoonga

o/p = 'DOM (document object model)learning on chai aur code'

i/p = title.textContent:- sara text content par

0/p = ' DOM (document object model)learning on chai aur code\n test text\n'

Q)what is difference between inner text and content ?

in h1 we add span test text , then we stlye in sapan to display none

test text shown in text content where as it will not shown in innerText

means inner text shows only visible text where as textContent show all text

i/p = title.innerHTML  // gives whole value of html

o/p = ' DOM (document object model)learning on chai aur code\n<span style="display: none;">test text</span>\n

i/p = document.querySelector('h2')

o/p = <h2>​ hello world ​</h2>​

i/p = document.querySelector('#title')

i/p = document.querySelector('.hello')

o/p = <h1 id=​"title" class=​"hello">​…​</h1>​

i/p = document.querySelector('input[type="Password"]')

o/p = <input type=​"password" name id>​

const myul = document.querySelector('ul')

undefined

myul.querySelector('li')

<li>​…​</li>​

const turnGreen = myul.querySelector('li')

undefined

turnGreen.style.backgroundColor = "green"

'green'

turnGreen.innerText = "five"

'five'

document.querySelectorAll('li')

NodeList(3) [li, li, li]

0

:

li

1

:

li

2

:

li

length

:

3

[[Prototype]]

:

NodeList

const tempLiList = document.querySelectorAll('li')

undefined

tempLiList

NodeList(3) [li, li, li]

tempLiList.style.color='green'(iska color green ni kar skte bcoz ye ek node list h)

tempLiList[0].style.color='green'

'green'

node element me batana padta hai ki kon sa element with index\

const myH1 = document.querySelectorAll('h1')

undefined

myH1.style.color ='greeen'

VM311:1 Uncaught TypeError: Cannot set properties of undefined (setting 'color')

    at <anonymous>:1:18

(anonymous) @ VM311:1

tempLiList

NodeList(3) [li, li, li]

tempLiList.forEach(function(l){

    l.style.backgroundColor = "yellow"

})

undefined

node list ko array me be convert kr skte hai

but jada tar cases me node list me for each loop use kerte hai bcoz wo array ni h waha map ni laga skte hai

document.getElementsByClassName("list")

HTMLCollection []

length

:

0

[[Prototype]]

:

HTMLCollection

item

:

ƒ item()

length

:

(...)

namedItem

:

ƒ namedItem()

constructor

:

ƒ HTMLCollection()

Symbol(Symbol.iterator)

:

ƒ values()

Symbol(Symbol.toStringTag)

:

"HTMLCollection"

get length

:

ƒ length()

[[Prototype]]

:

Object

yaha proto typing me koi looping ka option he ni dek raha h

to yha agr for each use kiya to kaam he ni karega

const tempClassList = document.getElementsByClassName("list")

undefined

tempClassList

HTMLCollection []

tempClassList.forEach(function(li){

    console.log(li)

})

VM890:1 Uncaught TypeError: tempClassList.forEach is not a function

    at <anonymous>:1:15

(anonymous) @ VM890:1

yaha temp list me loop karne k liye usee phele convert kerna padega

or ham yaha isko array me convert karenge

-->

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>DOM</title>

</head>

<body style="background-color: #212121; color: #fff;">

    <div class="parent">

        <div class="day">monday</div>

        <div class="day">tuesday</div>

        <div class="day">wednesday</div>

        <div class="day">thursday</div>

    </div>

</body>

<script>

    const parent = document.querySelector('.parent');

    console.log(parent);

    console.log(parent.children); // here we get html collection which is array like property

    console.log(parent.children[1]);

    console.log(parent.children[1].innerHTML);

    // CLASSIC FOR LOOP TO CHAL JAYEGA AGER ARRAY JAISA YA HTML JAISE PER means we don't need to convert it into array

    // we can't use on parent bcoz parent child gives the HTML collection, ye direct array par na chal k parent par chalega

    console.log("-----------------------------------------------");

    for (let i = 0; i < parent.children.length; i++) {

        console.log(parent.children[i].innerHTML);

    }

    parent.children[1].style.color = "orange";

    console.log(parent.firstElementChild);

    console.log(parent.lastElementChild);

</script>

</html>