**javascript**

**//var,constantand let--->print variables**

**/\*r a=10**

**var b=20**

**//vaR store a variable to value var---> global scope**

**console.log(a+b)\*/**

**/\* {**

**let a=10**

**console.log(a**

**}i// let --->private scope it can only print the result**

**{let a=10}**

**console.log(a) this can show a syntax error**

**const->constant**

**const a=20;**

**a=2;**

**console.log(a}**

**when const is mentioned a value cannot be changed**

**program 1**

**var price=1200**

**var product=iphone // when we given like this It is not defined it is considered as variable**

**var product=”iphone” when we store a string it want to be mention in double quotes**

**var tax=20**

**console.log(product)**

**var total=price+tax**

**console.log(total)**

**console.log(“total”) it print a statement what we mentioned in a double quotes**

**program 2**

**var fruit name =”grapes”**

**var count=10**

**var price=100**

**var total=count\*price**

**console.log(total)**

**keywords**

**var,let,for,const,function,for**

**for eg var if =20// we cannot mention keywords as a variable**

**comment**

**single line comment //**

**// var a =20**

**Multiline comments**

**/\*hi how are you**

**I am fine\*/**

**Operators**

**+,-,\*,/**

**Post increment**

**Var a=210**

**a++**

**console.log(a)**

**output**

**211**

**Post drecement**

**Var a=210**

**a--**

**console.log(a)**

**output**

**209**

**Post increment**

**Var a=210**

**Var b=++a**

**console.log(b)**

**console.log(a)**

**output**

**210**

**211**

**Post decrement**

**Var a=210**

**Var b=a--**

**console.log(b)**

**console.log(a)**

**output**

**210**

**209**

**Pre increment**

**Var a=210**

**Var b=++a**

**console.log(a)**

**output**

**211**

**Pre decrement**

**Var a=210**

**Var b=--a**

**console.log(a)**

**output**

**209**

**Data type**

**Console.log(typeof(“10”)**

**When we given in a double quotes it is a string**

**Output**

**String**

**Two types of data type**

**a)primitive datatype**

**number,string,Boolean,null,undefined**

**b)non primitive datatype**

**object,array**

**examples**

**var a**

**console.log(a)**

**output**

**undefined**

**function syntax**

**function<function name>() {}**

**eg**

**function hi()**

**{**

**Console.log(“hey there”)**

**}**

**hi()Onclick-->when we click a mouse**

**var faname="gowsi"**

**var faplace="sankari"**

**var fahero="vijay"**

**function favourite()**

**{**

**console.log("favourite name:"+faname)**

**console.log("favourite place:"+faplace)**

**console.log("favourite hero:"+fahero)// + is uesd for mention the variable namefor eg in python print("a=",b+c)here (,)-->operator work is done by +**

**}**

**favourite()**

**function with parameters**

**function add(a,b)**

**{**

**console.log(a+b)**

**}**

**add(30,50)**

**function mul(len,bre)**

**{**

**console.log(len\*bre)**

**}**

**mul (34,60)**

**function return type**

**add(30,50)**

**function mul(len,bre)**

**{**

**console.log(len\*bre)**

**}**

**mul (34,60)**

**function myname()**

**{**

**return "gowsi"//"var a" can call a function name and then it print a name**

**}**

**var c=myname()**

**console.log (c)**

**function addition(d,e)**

**{**

**return (d+e)**

**}**

**var total=add(30,50)**

**console.log(total)**

**conditional statement**

**1) if else**

**if (true)**

**{**

**Console.log(“if is working”) // code to be executed if the condition is true**

**}**

**else**

**{**

**Console.log(“else is working”) // code to be executed if the condition false**

**}**

**Eg**

**var rain=True**

**if (true) {**

**console.log("take the umberalla")**

**}**

**else**

**{**

**console.log("enjoy the sunshine")**

**}**

**Logical operators**

**// logical AND**

**Console.log(true && true);//true**

**Console.log(true && false);//false**

**//logical OR**

**Console.log(true || true);//true**

**// logical NOT**

**Console.log(!true);false -->inverse function**

**Eg**

**// logical  operators**

**console.log(false && false)// and**

**console.log(false && true)// and**

**console.log(true && false)// and**

**console.log(true && true)// and**

**console.log(false || false)// or**

**console.log(true || false)// or**

**console.log(false || true)// or**

**console.log(true || true)// or**

**console.log( !false)// not**

**console.log( !true)// not**

**else if program**

**var color = "yellow"**

**if (color == "red")**

**{**

**console.log("stop")**

**}**

**if (color =="yellow")**

**{**

**console.log(" get ready to go")**

**}**

**else if (color =="green")**

**{**

**console.log("go")**

**}**

**var score = 50**

**if (score <=50){**

**console.log("you need to improve")**

**}**

**else if (score>50 && score < 70)**

**{**

**console.log("gooidx job")**

**}**

**else{**

**console.log("excellent")**

**}**

**For loop**

**It is a control flow statement used to repeatedly execute a block of code**

**Syntax**

**For (initialization;condition;iteration)**

**{ // code to be executed in each iteration }**

**Eg**

**for (count=1;count<5;count=count+1)**

**{**

**console.log("gowsi")**

**} //output:gowsi,gowsi,gowsi,gowsi**

**for (a=1;a<=100;a=a+1)**

**{**

**console.log(a)**

**} // output 1-100**

**for (b=1;b<=100;b=b+2)**

**{**

**console.log(b)**

**}  // output 1,3,5,7,etc…………….**

**Print 10 to 1(which will be in reverse order)**

**for (c=10;c>=1;c=c-1)**

**{**

**console.log(c)**

**}**

**Alert message in web page with html**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

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

**<title>script</title>**

**<script>**

**alert("hi to every one");**

**</script>**

**</head>**

**<body>**

**<div id="output">gowsi</div>**

**<button type="button" onclick="alert('hello! how are you?')">ok</button>**

**<script>**

**// alert('hi to everyone');**

**console.dir(document);**

**console.log('hello');**

**</script>**

**</ body>**

**DOM manipulation IN JS WITH HTML**

**Onchange-html element has been changed**

**Onclick-user click an html element**

**Onmouseover-user moves the mouse over an html element**

**Onmouseout-user moves the away from the html element**

**Onkeydown-userpushes a keyboard key**

**Onload-browser has finished loading the page**

**DOM MODEL IN JS WITH HTML**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

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

**<title>script</title>**

**<script>**

**alert("hi to every one");**

**</script>**

**</head>**

**<body>**

**<p id="one">hi i am gowsigaa</p>**

**<h1 id="two">hello word</h1>**

**<button onclick="change()">change</button><!---on click is the event function it will occur only at that particular time this change the hi to bye when i click the change button-->**

**<script>**

**// select paragraph by using id**

**//console.log(document.getElementById("one"))**

**// store paragraph in variable**

**var para=document.getElementById("one")**

**// console.log(para) when i want only a text or two words for eg i am**

**console.log(para.textContent)**

**para.textContent="bye"**

**var head=document.getElementById("two")**

**console.log(head.textContent)**

**function change()// this will only occcur when will i click the particular button**

**{**

**head.textContent="bye world"**

**}**

**</script>**

**</body>**

**</html>**

**Eg for fixed number dom manipulation**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

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

**<title>dom model</title>**

**</head>**

**<body>**

**<input id="box1" value="43">**

**<input id="box2" value="54">**

**<button onclick="resultfunction()">add</button>**

**<p id="result">result</p>**

**<script>**

**//for manipulation**

**// selecting box1 and box2**

**var num1=document.getElementById("box1")**

**var num2=document.getElementById("box2")**

**var result=document.getElementById("result")**

**//store a values**

**var num1value=Number(num1.value)**

**var num2value=Number(num2.value)**

**var total=num1value+num2value**

**function resultfunction()**

**{**

**result.textContent=total**

**}**

**//console.log(num1.value);**

**// console.log(num2.value);**

**</script><!--this is given a fixed number-->**

**</body>**

**</html>**

**Another example**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

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

**<title>dom model</title>**

**</head>**

**<body>**

**<input id="box1" >**

**<input id="box2" >**

**<button onclick="resultfunction()">add</button>**

**<p id="result">result</p>**

**<script>**

**//for manipulation**

**// selecting box1 and box2**

**var num1=document.getElementById("box1")**

**var num2=document.getElementById("box2")**

**var result=document.getElementById("result")**

**//store a values**

**//var num1value=Number(num1.value)**

**// var num2value=Number(num2.value)**

**// var total=num1value+num2value**

**function resultfunction()**

**{**

**var num1value=Number(num1.value)**

**var num2value=Number(num2.value)**

**var total=num1value+num2value**

**result.textContent=total**

**}**

**//console.log(num1.value);**

**// console.log(num2.value);**

**</script><!--this is given a fixed number-->**

**</body>**

**</html>**

**Example for guess the number**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

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

**<title>dom manipulation</title><!--dom manipulation guess the number project-->**

**</head>**

**<body>**

**<h1>guess the number</h1>**

**<input id="guessnumber">**

**<button onclick="check()">check</button>**

**<p id="result">you are right/ wrong</p>**

**<p id="score">score:10</p>**

**<script>**

**var guessnumber=document.getElementById("guessnumber")**

**var score=document.getElementById("score")**

**var result=document.getElementById("result")**

**var randomnum=Math.floor(Math.random()\*10)+1**

**var totalscore=10**

**function check()**

**{**

**var enterednumber = guessnumber.value**

**if (randomnum==enterednumber)**

**{**

**console.log("right")**

**result.textContent="right"**

**alert("you are correct!!!!")**

**}**

**else**

**{**

**totalscore=totalscore-1**

**score.textContent="score:"+totalscore**

**console.log("wrong")**

**result.textContent="wrong"**

**}**

**}**

**</script>**

**</body>**

**</html>**

**Style manipulation**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

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

**<title>style manipulation with jS</title>**

**<style>**

**div{**

**background-color: brown;**

**height: 400px;**

**width: 400px;**

**}**

**.max{**

            transddddddddddddition: 3s;

**width:700px;**

**}**

**</style>**

**</head>**

**<body>**

**<div id="box"></div>**

**<button onclick="change()">change color</button>**

**<script>**

**var box=document.getElementById("box")**

**function change()**

**{**

**box.style.backgroundColor="blue"**

**box.setAttribute("class","max")**

**}**

**</script>**

**</body>**

**</html>**

**Task 1**

<input id="inputbox" onkeyup="update()">

<button onclick="update()">click</button>

<h1 id ="result"></h1>

<script>

var inputbox=document.getElementById("inputbox")

var result=document.getElementById("result")

function update()

    {

result.textContent=inputbox.value

    }

</script>

**Task 2**

<button onclick="update()">add</button>

<div id="result"></div>

<script>

    var div=document.getElementById("result")

    function update()

    {

        var listitem=document.createElement("h2")

        listitem.textContent="gowsi"

        div.append(listitem)//append is used to repeated the words

    }

</script>

<button id ="color" onclick="get()">ok</button>

<script>

   var ok=document.getElementById("color")

   function get()

    {

    ok.style.backgroundColor="red"

   }

</script>

**Todo list**

<div id="one">

    <h2>gowsi</h2>

</div>

<script>

    var div=document.getElementById("one")

    //div.innerHTML="<h2>bye</h2>"

    div.textContent="<h2>bye</h2>"

</script>

<div class="one">

    <h1 class="ok">gowsi</h1>

    <h1 class="ok">gow</h1>

</div>

<script>

    // for class

    var div=document.querySelector("div")

    //div.innerHTML="<h2>bye</h2>"

    //div.textContent="<h2>bye</h2>"

    for(count=0;count<divlength;count+=1)

    {

    console.log(div[count].textContent)

    }

</script>

**By using query selector we can tag,classand id.we want mention what this type**

<div class="one">

    <h2>gowsi</h2>

    <h1>gow</h1>

</div>

<script>

    var div=document.querySelector("h1")

    //div.innerHTML="<h2>bye</h2>"

    //div.textContent="<h2>bye</h2>"

    console.log(div)

</script>

**Array**

**Represent with in [apple,orange,mango]**

//todo list array

var fruit=["apple","orange","grape"]

//console.log(fruit)

//console.log(fruit[2])

for(count=0;count<=fruit.length;count+=1)

    {

console.log(fruit[count])

    }

console.log(fruit.length)

**append and preppend**

<div >

    <h2>gowsi</h2>

</div>

<script>

    var div=document.querySelector("div")

    div.append("hello")

    div.prepend("hello")

</script>

**Adding html element**

**Using insertadjacentelement**

**Beforebegin🡪startbefore like div tag**

**Beforeend🡪act as append**

**Afterbegin-🡪act as preppend when div tag start**

**Afterend--🡪after the end tag**

<div >

    <h2>gowsi</h2>

</div>

<script>

    var div=document.querySelector("div")

    var h1=document.createElement("h1")

    div.insertAdjacentElement("beforebegin","hello")

    div.insertAdjacentElement("afterbegin","hello")//work as preppend

</script>

**deleteing html element--🡪here we can use remove()**

div >

    <h2>gowsi</h2>

    </div>

<script>

    var div=document.querySelector("div")

    div.remove()

</script>

**Todo complete example**

<input id="input">

<button onclick="add()">add</button>

<ul id="list-container">

    <li>

        hello

        <button onclick="deleteitem(event)">delete</button>

    </li>

</ul >

<script>

    var ul=document.getElementById("list-container")

    var input=document.getElementById("input")

    function add()

    {

        var listitem=document.createElement("li")

        listitem.innerHTML=input.value+"<button onclick='deleteitem(event)'>delete</button>"

        ul.append(listitem)

        i

    }

    function deleteitem(event)

        {

            event.target.parentElement.remove();        }

</script>

**Objects**

**We define properties and methods**

**Var obj={name:’gowsi’,phno:87888989080989,age:23}like dirticonary in python**

var item={

    name:'phone',

    price:20000,

    quantity:2,

    dimensions:{

        length:'10cm',breadth:'20cm'

    },fruit:['apple','grape']

}//key :value

console.log(item)

// when we get a new object and other method

var item2=new Object()

item2.name='rose'

item2.price=20

item2.quantity=2

//we are using (.)-->notation

console.log(item.price)

item.price=23000

console.log(item.price)

console.log(item.dimensions)

//when we add a new item in an object

item.place="sankari"

console.log(item)

// square bracket notation

console.log(item['price'])

// when we change a value

item['place']="t.gode"

// when it is separate variable we can add it

// this method is not work out

item.key='t.gode'

console.log(item)

console.log(item.fruit[1])

item1={

    name:'red',

    color:"pink",

    buy:function () {

        console.log("item to be added")

    },

    phno(){

        console.log("item can be removed")

    }

    //function an be used in object and function key name  is not necessary be used in the object

}

item1.buy()

item1.phno()

**strings**

var a="i am a good "

var b ="this is a bag"

// concatination a string -->combine two sentence into one sentence

// first method using +

var c=a+b

console.log(c)

// second method a.concat(b)

a.concat(b)

console.log(b)

a.concat("!@",b)

//partitioning string which we want part part or characters

//i)slice(start,end) we can use negative index

//ii)substring(start,end) it is similar to slice

//substr(start,length) l;ength mention how many characters are there want to be

//replace ('which word is going to be replace','new word')

// touppercase()-->change all words in capital letter

// tolowercase()-->change all words in small letter

//length-->mention the length of the sentence

//trim()-->remove the extra space,trimstart()-->remove extra space at start of the sentence,trimend()-->remove space at end of the sentence

//padend(length, what want to be added),padstart()same to padend

//a[]--> we can give the index value of the specific character or a.charat(1)it is same to a[1]

//.charCodeAt()we can give the index value but it can give the unic code of the character

//.indexOf('e') we can character and then we can get the index value

//.lastIndexOf(e) if the character is present these word we can get it index value otherwise it give -1

//.search('not')if the word present or not it is present it give the word first letter index otherwise -1

//.includes('not')if the word present or not it output is true or false

//.startsWith(s) same as includes but it says character it is similar to .endswith()

b.slice(3,6)

a.substring(-5,-10)

a.substr(5,10)

a.replace('good','nice')

a.toUpperCase()

b.toLowerCase()

a.length

a.padEnd(12,'e')

**math functions**

Math.round(6.9)

// it can give the round value after .5

Math.floor(6.9)

// it can give only point before value

Math.ceil(6.9)

//it can give next number if yoiu give like this also math.ceil(8.1)

math.sqrt(67)

//it can give the square root of the value

math.abs(-10)

// it can give the positive value of the given number

math.pow(3,2)

// it can power the value of the given number first is base,second is power value

math.random ()

// it can give the random value from 0 to 1

// prperties it can give the constant value for given function for example

math.Pi

math.E

**NUMERIC FUNCTIONS**

var a=10

var b=a.toString();

//.toString //()-->convert a number into a String

console.log(typeof(b))

// typeof-->identifies the what its data type

var c=a.toExponential(3)

//to exponential() give the exponential format in bracket what are the number can be given like that it can be display after the point

var d=a.toFixed()

// .to fixed is used to how many numbers want to be displayed it is fixed after point

var e =a.toPrecision(4)

// .toprecision totally how many numbers displayed

//now we see the how to converrt primitive methods

var f=a.valueOf()

// .valueof() can be used convert primitive value

var g=Number("10.67")

console.log(g)

// number()can be used given a number as a string it can be converted into a number otherwise we given a words it can be said to be NaN

var h=parseInt(10.78)

// it can be convert into interger format

var i=parseFloat()

// parsefloat() it can be changed string or integer into float

var j=Number.MIN\_VALUE

// the above given shows possible min value this is same to Number.MAX\_VALUE

var k=1/0

console.log(k)

var l=Number.NEGATIVE\_INFINITY

// the above mentioned gives the -infinity it related to -1/0

var m=Number.NaN

// not a number (NaN)

**DATE AND ITS FORMATS**

var a =new Date()// current date and time

console.log(a)

/\*new Date()

new Date(date string)

new Date(year,month)

new Date(year,month,day)

new Date(year,month,day,hours)

new Date(year,month,day,hours,minutes)

new Date(year,month,day,hours,minutes,seconds)

new Date(year,month,day,hours,minutes,seconds,ms)

new Date(milliseconds)\*/

var b=new Date("november 12,2020 12:50:55")

console.log(b)

var c=new Date(2023,11)

console.log(c)

/\*// when we specifify a month JavaScript counts months from 0 to 11:

January = 0.

December = 11.\*/

var d=new Date(88,9,12)

console.log(d)

var e=new Date(-200000000000000)

console.log(e)

//Example (Complete date)

//const d = new Date("2015-03-25");

/\*ISO Dates (Year and Month)

ISO dates can be written without specifying the day (YYYY-MM):

Example

const d = new Date("2015-03");\*/

//short dates

var da= new Date("03/25/2015");

console.log(da)

// warning for short dates

/\*WARNINGS !

In some browsers, months or days with no leading zeroes may produce an error:

const d = new Date("2015-3-25");

The behavior of "YYYY/MM/DD" is undefined.

Some browsers will try to guess the format. Some will return NaN.

const d = new Date("2015/03/25");

The behavior of  "DD-MM-YYYY" is also undefined.

Some browsers will try to guess the format. Some will return NaN.

const d = new Date("25-03-2015");

\*/

// longh dates

/\*Long dates are most often written with a "MMM DD YYYY" syntax like this:

Example

const d = new Date("Mar 25 2015");

Month and day can be in any order:

Example

const d = new Date("25 Mar 2015");

And, month can be written in full (January), or abbreviated (Jan):

Example

const d = new Date("January 25 2015");

Example

const d = new Date("Jan 25 2015");

Commas are ignored. Names are case insensitive:

Example

const d = new Date("JANUARY, 25, 2015");

\*/

// date input

var time= Date.parse("march 13,2023")

console.log(time)

//you can use the Date.parse() method to convert it to milliseconds.

// when we mill second into date

var s=new Date(time)

console.log(s)

var f=s.getFullYear()

console.log(f)

// get date methods

/\* Date Get Methods

Method  Description

getFullYear()   Get year as a four digit number (yyyy)

getMonth()  Get month as a number (0-11)

getDate()   Get day as a number (1-31)

getDay()    Get weekday as a number (0-6)

getHours()  Get hour (0-23)

getMinutes()    Get minute (0-59)

getSeconds()    Get second (0-59)

getMilliseconds()   Get millisecond (0-999)

getTime()   Get time (milliseconds since January 1, 1970)\*

warning

Old JavaScript code might use the non-standard method getYear().

getYear() is supposed to return a 2-digit year.

getYear() is deprecated. Do not use it\*/

// set date method--> is used to change date or time or set it

/\*

Set Date methods are used for setting a part of a date:

Method  Description

setDate()   Set the day as a number (1-31)

setFullYear()   Set the year (optionally month and day)

setHours()  Set the hour (0-23)

setMilliseconds()   Set the milliseconds (0-999)

setMinutes()    Set the minutes (0-59)

setMonth()  Set the month (0-11)

setSeconds()    Set the seconds (0-59)

setTime()   Set the time (milliseconds since January 1, 1970)

\*/

var g =new Date(s.setMonth(9))

console.log(g)

**array**

let mark=[100,20,30,40,50]

console.log(mark.length)

// array index starts from 0 .in reverse it starts from -1

console.log(mark[3])

console.log(mark[mark.length-2])

// when we want a index

// we can give array im numbers and words

let ar=[[1,2,3],[4,5,6],[8,9.10]]// two dimensional

console.log(ar)

//push-add elements to array at the end

ar.push('e','f','a')

console.log(ar)

//pop-->removes last element  of the array from the end and returns removed value

ar.pop('f')

console.log(ar)

console.log(ar.pop('f'))

console.log(ar)

// shift -removes elememt from the begining of the array and returns removed value

console.log(ar.shift('e'))

console.log(ar)

//unshift-adds element ton the start of the array and returns now length

console.log(ar.unshift('c'))

console.log(ar)

//delete-->remove the element from the array

//delete ar[0]

//console.log(ar[0])// better not to use the array because this can leave the index empty

//instead of using delete we can use splice this can used to remove and replace the element

ar.splice(2,1)

console.log(ar)

//ar.splice(0,'c','e')

ar.splice(0,2,'e',[1,2,3])// replace-deletes element at index 0 and inserts e and[1,2,3]

console.log(ar)

ar.splice(1,0,"b")// we cannnot delete any element herebut we can add a element here

console.log(ar)

// slice(startingindex,ending index)

//ending index not included

console.log(ar.slice(0,2))

console.log(ar.slice(-4))

// .reverse by using these we can rerverse the array

ar.reverse()

console.log(ar)

// join- convert array to string

// split-convert string to array

let str=ar.join()

console.log(str)

let arr=str.split(',')

console.log(arr)

//concat and spread operator(we can use use both)

let b=[1,2,3]

let c=[4,5,6]

//let d=[b,c]

let d=b+c

d=b.concat(c)// joined the two array into one array

console.log(d)

// spread can be used [...b,...c]

let e=[...b,...c]

console.log(e)

**conditional statement**

**i)switch**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

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

    <title>City Stats</title>

    <style>

    html,body{

           margin: 0;

           font-family: 'Poppins', sans-serif;

           background-color:#f5e8ba;

        }

        #wrapper div,#wrapper h2{

            display:flex;

            justify-content: center;

        }

        h2{

            color:#ffeba7;

            letter-spacing: 1px;

        }

        select{

            border-radius: 5px;

            border: 2px solid black ;

            margin-top:30px;

            height:2.5em;

            width:250px;

            padding-left:10px;

        }

        input:active{

            border:none;

        }

        #wrapper{

            height:500px;

            width:400px;

            margin:200px auto;

            background-color:#1f2029;

            border-radius:10px;

            padding:30px;

        }

        button{

            margin-top:20px;

            background-color:#ffeba7;

            border-radius: 4px;

            height: 44px;

            font-weight: 600;

            padding: 0 30px;

            letter-spacing: 1px;

            border: none;

            background-color: #ffeba7;

            color: #102770;

        }

        button:hover{

            cursor:pointer;

        }

        #result{

            color:white;

            margin-top:20px;

            font-size:32px;

        }</style>

</head>

<body>

    <div id="wrapper">

        <h2>Select a City</h2>

        <div class="inputdiv">

           <select id="input">

                <option value="Bengaluru">Bengaluru</option>

                <option value="Chennai">Chennai</option>

                <option value="Delhi">Delhi</option>

                <option value="Mumbai">Mumbai</option>

           </select>

        </div>

        <div>

            <button type = "submit" >Show Stats</button>

        </div>

    </div>

    <script >

        /\*

'Chennai': population = 4646732

           literacyRate = 90.20

           language = 'Tamil'

'Bengaluru':population = 8443675

            literacyRate = 88.71

            language = 'Kannada'

'Mumbai':population = 12442373

        literacyRate = 89.73

        language = 'Marathi'

'Delhi': population = 16787941

         literacyRate = 86.20

         language = 'Hindi'

The Indian city of Bengaluru has a population of 8443675. Language spoken is Kannada and literacy rate is 88.71%.

\*/

const button=document.querySelector('button')

let resultdiv=document.createElement('div')

    resultdiv.id='result'

    document.getElementById('wrapper').appendChild(resultdiv)

    button.addEventListener('click',displaystats)

function displaystats() {

    const input=document.getElementById("input")

    const city=input.options[input.selectedIndex].value

    let population=0,literacyRate=0,language=''

    switch(city){

        case 'Bengaluru':

        population = 8443675

            literacyRate = 88.71

            language = 'Kannada'

            break

            case 'Chennai':

            population = 4646732

           literacyRate = 90.20

           language = 'Tamil'

           break

           case 'Mumbai':population = 12442373

        literacyRate = 89.73

        language = 'Marathi'

        break

        case'Delhi':

         population = 16787941

         literacyRate = 86.20

         language = 'Hindi'

        break

    }

    let text=`The Indian city of ${city} has a population of ${population}Language spoken is ${language} and literacy rate is ${literacyRate}%.`

    console.log(text)

    document.getElementById('result').innerHTML=text

}

    </script>

</body>

</html>

**Loops**

<script>

    //loops

// for,while,dowhile

//for for(initalisation;condition,increment or decrement)

let i

for( i=1;i<=5;i++){

    //console.log("hello")

    console.log(i)

}

console.log('outside loop'  +i)

console.log('outside loop',i)

// print 10 to 1

for (i=10;i>=1;i--)

    {

        console.log(i)

    }

    console.log('outside loop', i)

    // while()--> we can give only condition

    i=10// initialization

    while(i>=1)

        {

console.log(i)

i-=1// when we forget to give this it will go infinity

        }

        console.log('outside loop'+i)

       let j=0

       while(j>=1){

        console.log(j)

       }

console.log(i)

        //do while-->when the condition is fail it will work automaticallybut while don't

i=10

        do{

console.log(i)

i-=1

        }while(i>=1)// t is atleast one tme can be excute

        // break-stop the loop which can stop and leave the loop

        while(true){

            num=Number(prompt('enter a number'))// prompt

           if(!isNaN(num))

           break;

        }

       // let a=prompt("whast is your name")

        //onsole.log(a)

        // isNaN()--> not a number whether to say yes or no

        // continue-->skip the current iteration

        // print 1 to 10 but i don't want the  multiple of 3 iteration

        for(i=1;i<=10;i++){

            if(i%3==0)

            continue

        console.log(i)

        }

        // for..of

        let arr=['apple','orange','mango','grapes']

        for(i=0;i<arr.length;i++)

        {

console.log(arr[i].toUpperCase())

        }

        for(let fruit of arr)// this is help to store the variable from one to another variable

        console.log(fruit)

        //for ..in

        item={

            name:'phone',

            quantity:2,

            price:34000

        }

        for (let key in item)

        {

            console.log(key)// like this way we given means it will give only for eg we say name,price before colon only it will given

            console.log(item[key])// it will only given the value after the colon

        }

</script>

**Maps**

// map-->excutes call back for each array element and return new array

let priceusd=[20,35,13]// convert dollar into rupees

let priceINR=priceusd.map(x=>x\*83)// it works like a foreach() it give only give one by one value

console.log(priceINR)

// another method without using arrow function

priceINR=priceusd.map(convert)

function convert(val) {

    return val\*83

}

console.log(priceINR)

const input=[// array of objects

    {name:'gowsi',age:15},

    {name:'ram',age:13},

    {name:'geetha',age:24},

    {name:'raj',age:19},

    {name:'anu',age:12}

]

// i want a ages

const age=input.map(y=>y.age)

console.log(age)

const name=input.map(z=>z.name)

console.log(name)

// filter=>retuen new array by checking each value of original arr using call back function

// it can filter the value want we given in that

let cost=[34,234,12,34,54,123]

let lessthan100=cost.filter(x=>x<=100)

console.log(lessthan100)

// reduce-excutes reducer callback and rerturns accumulated result

// reduce-returns a single value

cost=[34,234,12,34,54,123] //cost.reduce(acummalator) atleast two parameters

let cart=cost.reduce((total,el)=>total+el)// total is accumulator ,el is used to take and given one by one element

//let cart=cost.reduce((total,el)=>total+el, 1000) //here we give 1000 means we add a values from thousand

/\*Syntax

array.reduce(function(total, currentValue, currentIndex, arr), initialValue)\*/

console.log(cart)

arr2d=[['a','b','c'],

['d','e','f'],

['d','g','c']

]

// result ={a;1,b;1;c=1} how many a,b,c are present

//console.log(arr2d.flat())//flat-> changes a two dimensional array into one dimensional array

let count=arr2d.flat().reduce((accumulator,currval)=>{

    if(accumulator[currval])

        accumulator[currval]++

    else

    accumulator[currval]=1

return accumulator

}

,{});// when an empty object is created  processed value is stored in the empty box

console.log(count)

// if i give a array i want to remove the duplicates of the array

let arr=[1,2,3,4,6,7,2,7,8,8]

let a=arr.filter((index,items)=>arr.indexOf(index)===items)// when you want a original value do not want the duplicate value

//let a=arr.filter((index,items)=>arr[index]!=items) when you want get only the duplicate value

console.log(a)

// when  i give a string   in this i only want a uppercase leters

let b="Gowsigaa Bala Murugan"

let c=b.split("").filter((x)=>x===x.toUpperCase())

console.log(c.join("").trim().trim())

// returning function and closure

let aa=100

function funct1(){

    console.log("a is",aa)

}

aa=200

funct1()

**returning and closure**

// returning function and closure

let aa=100

function funct1(){

    console.log("a is",aa)

}

aa=200

funct1()

// returning function-higherr order function

function outer(name) {

    let outerVariable='bread'

    function inner()

    {

        let innervariable='butter'

    console.log(innervariable)

    console.log(outerVariable)

    console.log('hello',name)

    }

    //inner()// we inner function able to access the outer function is called lexical scoping

   return inner// return a function

}

//outer()

//let call1=outer()

let call1=outer('gowsi')

call1()//return value is stored in variable

function makeadder(x)

{

    return function(y)

    {

        return x+y

    }

}

let add5=makeadder(5)

console.log(add5(10))

let add100=makeadder(100)

console.log(add100(20))

console.log(add100(45))// this is called closure

**memory allocation**

// memory allocation

//BigInt when we want large size of numbers we can stored in a bigint

// for eg const a =1234567899774n  => this is one type of method here we want to mention n atlast of the value

// another type is const a=BigInt(1234567890122)

//in memory there two types of memory that is i)stack ii)heap

//stack

let g=100 // it is stored in a stack

// reference types like array,function are atored in heap

let obj1={name:'gowsi',age:22} //this is stored in heap but stored heap has a variable that is in the stack

let obj2=obj1 // here heap stored only a value but value's variable stored in the stack heap doesn't change but stack create new storage for that variable here eg locating same variable

**Set and map**

let map1=new Map()

map1.set('a',1) //set() is ued to add values in the map

map1.set('b',2)

map.set('a',3)// here a value can be 3 not 1

// same as set()

map1.delete(a)

// using for() and foreach()

    for(let i of map1){

        console.log(i)

    }

    for(let[k,v] of map1)

{

    console.log(v)

}

for(let k of map1.keys())

    {

        console.log(k)

    }

   map1.forEach((v,k)=>{

    console.log('key',k,'value',v)

   })

   // we cant convert a value from one variable to another variable

   //2d arraY to map

   let kvarray=[['a',1],['b',2]]

   let map2= new Map(kvarray)

   console.log(map2)

   console.log(...map2)//...map2 convert into 2d array thisis called spread operator

**oops in javascript**

// oops=>object oriented programming we want to think all the one are ocjects and classes.except the primitive types all the one are objects

let user1={

    name:'gowsi',

    age:22,

    login(){

        console.log('hey',this. name)

        console.log('you are logged in')

    },

    logout(){

        console.log('hey',this. name)

        console.log('you are logged out')

    }

}

let user2={

    name:'anu',

    age:21,

    login()

    {

        console.log('hi',this.name,'and your age',this.age)

        console.log('you are logged in')

    },

    logout()

    {

        console.log('hi',this.name,'and your age',this.age)

        console.log('you are logged out')

    }

}

let user3={

    name:'raj',

    age:23,

    login()

    {

        console.log('Hi',this .name)//this .key used to specify the we are in which object

        console.log('you are logged in')

    },

    logout()

    {

        console.log('Hi',this .name)

        console.log('you are logged out')

    }

}

user2.logout()

user3.logout()

user1.login()

// by using class => class Name {} class name must start with capital letter.clas is a template of properties and method

class User{

    static numberofusers=0; // static is commonly used to track the total numbers we couldn't concentrate

    constructor(name,age) // constructor => is used to initiaize the data it can immediately we can give our datas inside the construction    {

    {

        // this is called the instance od class

        this.name=name

        this.age=age

    }

    login()

    {

        console.log('Hi',this .name)//this .key used to specify the we are in which object

        console.log('you are logged in')

    }

    logout()

    {

        console.log('Hi',this .age)

        console.log('you are logged out')

    }

}

let userone=new User('neha',24)

let usertwo=new User('vidhya',25)

let movie ='ps1'

console.log(movie.length)// this method has a temporary wraper class this is a primitive data type

// now we can  create an object like the string

let music=new String('arr') //this an object

userone.login() // classes normal variables is called instance

**static variable in constructor**

/ by using class => class Name {} class name must start with capital letter.clas is a template of properties and method

class User{

    static numberofusers=0; // static is commonly used to track the total numbers we couldn't concentrate and this is a common variable for class when we mention a static variable inside a constructor we want to use the

       constructor(name,age) // constructor => is used to initiaize the data it can immediately we can give our datas inside the construction it calls an object when it create a class

    {

        // this is called the instance od class

        this.name=name

        this.age=age

        User.numberofusers++  //when we mention a static variable inside a constructor we want to use the classname.staticvariablename

    }

    login()

    {

        console.log('Hi',this .name)//this .key used to specify the we are in which object

        console.log('you are logged in')

    }

    logout()

    {

        console.log('Hi',this .age)

        console.log('you are logged out')

    }

}

let userone=new User('neha',24)

let usertwo=new User('vidhya',25)

console.log('number of users',User.numberofusers)

let movie ='ps1'

console.log(movie.length)// this method has a temporary wraper class this is a primitive data type

// now we can  create an object like the string

let music=new String('arr') //this an object

userone.login() // classes normal variables is called instance

pital letter.clas is a template of properties and method

class User{

    static numberofusers=0; // static is commonly used to track the total numbers we couldn't concentrate and this is a common variable for class when we mention a static variable inside a constructor we want to use the

       constructor(name,age) // constructor => is used to initiaize the data it can immediately we can give our datas inside the construction it calls an object when it create a class

    {

        // this is called the instance od class

        this.name=name

        this.age=age

        User.numberofusers++  //when we mention a static variable inside a constructor we want to use the classname.staticvariablename

    }

    login()

    {

        console.log('Hi',this .name)//this .key used to specify the we are in which object

        console.log('you are logged in')

    }

    logout()

    {

        console.log('Hi',this .age)

        console.log('you are logged out')

    }

}

let userone=new User('neha',24)

let usertwo=new User('vidhya',25)

console.log('number of users',User.numberofusers)

let movie ='ps1'

console.log(movie.length)// this method has a temporary wraper class this is a primitive data type

// now we can  create an object like the string

let music=new String('arr') //this an object

userone.login() // classes normal variables is called instance

// inheritance ==> we can use the source of existing class to the new class

// acquiring properties of a base class or parent class or super class  for eg user()=> is a base class

// derived class or child class or sub class is a paiduser here  base class user() properrties can be used by derived class

 class paiduser extends user1{

    constructor (name,age){

       /\* this.name=name

        this.age=age\*/ // instead of using these we can user super() keyword

        SourceBuffer(name,age)

        this.storage =100

    }

    message()

    {

       console.log("you have 100 gb storage")

    }

    // overriding => here when we give a function name it is already mentioned in the base class

    login()

    {

        console.log("thanks for your support")// here i give a login function within a derived class it doesnot goes to base class

    }

 }

 let paiduser1= new paiduser('ram',28)

 paiduser1.login()

 paiduser1.message()

 // base class canot be able to derive the derived class sources

**Method chaining**

// base class canot be able to derive the derived class sources

 // method chaining => when there are two or more functions in a single varible

paiduser1.login().message()

// es6 can be introduced by using the class.but before es6  class can not be used

// prototype => can be attached to prototype we can acess the inheritance by using protype

// now we cannot use the class

function user(name,age)

{

    this.name=name

    this.age=age

    this.login=function(){

        console.log("you are logged in")

    }

user.prototype.login=function()

    {

        console.log('hi',this.name)

        console.log("you are logged in")

    }

let user1=new user('gowsi',34)

user1.login()

// get and set => we can see how to use in class

// get => said what is that value

// set=>when we want to change the value

class temperature{

    constructor(temp){

        this.\_temp=temp // the \_ is used to specify the variable especially we used in get and set and \_ is the private property we could not use this outside of class

    }

    get temp()

    {

        return `${this.\_temp} deg.celcius`

    }

    set temp(temp){

        if(temp>100)

            temp=100

        this.\_temp=temp

    }

}

let temp1=new temperature(25)

temp1.temp=150

console.log(temp1.temp)

// when we want to export a class from a one file to another file

// export deafult class name

// export {function name} // the functions which are mentioned in the class

// in new file we want to be mentioned where it is used

// import class name from pathname of the file

//import class name,{function name1,function name2} from pathname of the file

// to be accesed we want to use the <script type="module" src="script file name">

**Asynchronous js**

// synchronous ==> a task completed step by step it is a blocking

// asynchronous ==>a task completed simtaneously

// basically js is a synchronous

//with the help of the browser we act the js as asynchronous

console.log('step 1')

console.log('step 2')

console.log('step 3')

setTimeout(()=>console.log('step 1'),4000)//excute the functon after 4  seconds

setTimeout(()=>console.log('step2'),3000)

setTimeout(()=>console.log('step3'),1000)

console.log('good bye')

// synchronous ==> a task completed step by step it is a blocking

// asynchronous ==>a task completed simtaneously

// basically js is a synchronous

//with the help of the browser we act the js as asynchronous

console.log('hello') // f1

function sync()

{

console.log('step 1')//f3

console.log('step 2')//f4

console.log('step 3')//f5

}

sync()//f2

let a=100

let b=20

let c=a+b

// global scope can be excuted at first when new function it stored in the callstack but when function is printed  and completed successfully it is removed from the call stack

// next function can be added to the call stack same process can be done again and again it is common to all funcxtion

setTimeout(()=>console.log('step 1'),4000)//excute the functon after 4  seconds

setTimeout(()=>console.log('step2'),3000)

setTimeout(()=>console.log('step3'),1000)

console.log('good bye')

for (let i=1;i<10;i++)

    console.log(i)

**event loop,setInterval()**

// synchronous ==> a task completed step by step it is a blocking

// asynchronous ==>a task completed simtaneously

// basically js is a synchronous

//with the help of the browser we act the js as asynchronous

console.log('hello') // f1

function sync()

{

console.log('step 1')//f3

console.log('step 2')//f4

console.log('step 3')//f5

}

sync()//f2

let a=100

let b=20

let c=a+b

// what are the fu

// global scope can be excuted at first when new function it stored in the callstack but when function is printed  and completed successfully it is removed from the call stack

// next function can be added to the call stack same process can be done again and again it is common to all funcxtion

// seTimeout() is used to excute the function when we set a time it will excute after that time and it will go to another work

// when an another function excution is going on settimeout() is added to call back queue because these method is happen to the not to distrubing the the current processing function

setTimeout(()=>console.log('step 1'),4000)//excute the functon after 4  seconds //f6

setTimeout(()=>console.log('step2'),3000) // f7

setTimeout(()=>console.log('step3'),1000) // f8

console.log('good bye')// f9

for (let i=1;i<10;i++)

    console.log(i)

// event loop => check call stack is empty if it is queue data can be added to the stack it main process is to check the call stack is empty

// setInterval() =>this method is used to excute function or a print statement every 2 seconds

  setInterval(()=>console.log("hi...."),2000)

  // asynchronous is non blocking

**Digital clock**

**Basic:**

let dateTime=new Date() // newdate() gives current date and time

console.log(dateTime)

// .getDate() => give only the date for eg today date is 10

dateTime.getDate()

console.log(dateTime)

**example code**

**html code**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

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

    <link rel="stylesheet" href="clock.css">

    <title>Digital Clock</title>

</head>

<body>

    <div class="container">

        <div id="wrapper">

            <span id="hours">00 Hrs</span>

            <span>:</span>

            <span id="mins">00 Min</span>

            <span>:</span>

            <span id="seconds">00 Sec</span>

            <span id="ampm">AM</span>

        </div>

    </div>

    <script src="clock.js">

    </script>

</body>

</html>

**CSS code**

body{

    margin:0;

    padding:0;

    background: rgb(2,0,36);

    background: linear-gradient(90deg, rgba(2,0,36,1) 0%, rgba(9,98,121,1) 35%, rgba(0,212,255,1) 100%);

}

.container{

    font-family:'Gill Sans', 'Gill Sans MT', Calibri, 'Trebuchet MS', sans-serif;

    color:white;

    display:flex;

    width:100vw;

    height:100vh;

    align-items: center;

    justify-content: center;

    font-size: 5rem;

}

#wrapper{

    border: 3px solid white;

    padding:1rem;

}

.container span{

    margin-left:10px;

}

#ampm{

    font-size:3rem;

    align-items: end;

}

**Js code**

/\*let dateTime=new Date() // newdate() gives current date and time

console.log(dateTime)

// .getDate() => give only the date for eg today date is 10

dateTime.getDate()

console.log(dateTime)\*/

// i create a function because the time want to be for every seconds

let ampm=document.getElementById('ampm')

function displaytime()

{

    let dateTime=new Date()

    let hr=dateTime.getHours()

    let min=padzero(dateTime.getMinutes())

    let sec=padzero(dateTime.getSeconds())

    if(hr>12)

        {

            hr=hr-12

            ampm.innerHTML='PM'

        }

        else

        {

            ampm.innerHTML='AM'

        }

    // this below function is used to get to store the id value in the variable

    document.getElementById('hours').innerHTML=padzero(hr)

    document.getElementById('mins').innerHTML=min

    document.getElementById('seconds').innerHTML=sec

}

function padzero(num)

{

    return num<10?"0"+num:num

}

setInterval(displaytime,500) //here we give 2000s it will skip the one second value for eg when it show 12sec next second it skip 13 and go to to 14 sec

**promise**

// promise=> we will get a two outcome it is like a probability  for eg we can say today it may rain or not

// i) type-1 it rain today ->this is called resolve=> successful

//ii)type-2 it not rain today -> this is called reject=>failure

// promise-> is a object

const tatkalbook=new Promise((resolve, reject) => {

    let bookingsuccess=true

    if(bookingsuccess)

            resolve()

        else

        reject()

})

tatkalbook.then(success)//in then() we can give the resolve

.catch(failure) // in catch () we can give the reject

function success()

    {

        console.log("thanks buddy i will transfer the ammount")

    }

    function failure()

    {

        console.log("thanks for trying!.....")

    }

**Another method0**

// promise=> we will get a two outcome it is like a probability  for eg we can say today it may rain or not

// i) type-1 it rain today ->this is called resolve=> successful

//ii)type 2 it not rain today -> this is called reject=>failure

// promise-> is a object

const tatkalbook=new Promise((resolve, reject) => {

    let bookingsuccess=true

    if(bookingsuccess)

            resolve(1000)// it is stored in the then

        else

        reject()//it can stored in the catch()

})

//tatkalbook.then(success)//in then() we can give the resolve

//.catch(failure) // in catch () we can give the reject

tatkalbook.then((amt)=>console.log(`thanks buddy i will transferred Rs.${amt} to you`))

.catch(()=> console.log("thanks for trying!....."))

function success()

    {

        console.log("thanks buddy i will transfer the ammount")

    }

    function failure()

    {

        console.log("thanks for trying!.....")

    }

**Another method**

// promise=> we will get a two outcome it is like a probability  for eg we can say today it may rain or not

// i) type-1 it rain today ->this is called resolve=> successful

//ii)type 2 it not rain today -> this is called reject=>failure

// promise-> is a object and it is easy to handle the asynchronous object

/\*const tatkalbook=new Promise((resolve, reject) => {

    let bookingsuccess=true

    if(bookingsuccess)

            resolve(1000)// it is stored in the then

        else

        reject()//it can stored in the catch()

})\*/

//tatkalbook.then(success)//in then() we can give the resolve

//.catch(failure) // in catch () we can give the reject

function tatkalbook()

    {

    return new Promise((resolve, reject) => {

        let bookingsuccess=true

        if(bookingsuccess)

                resolve(1000)// it is stored in the then

            else

            reject()//it can stored in the catch()

    })

    }

tatkalbook().then((amt)=>console.log(`thanks buddy i will transferred Rs.${amt} to you`))

.catch(()=> console.log("thanks for trying!....."))

/\*function success()

    {

        console.log("thanks buddy i will transfer the ammount")

    }

    function failure()

    {

        console.log("thanks for trying!.....")

    }

\*/

function tosscoin(){

return new Promise((resolve, reject) => {

    // 0-head(success) 1-tail(failure)

   const rand= Math.floor(math.random()\*2)

   if(rand==0)

    resolve()

   else

    reject()

})

}

/\*tosscoin()

.then(()=>console.log("Congrats it's head"))

.then(()=>console.log("Congrats it's head"))

.then(()=>console.log("Congrats it's head"))

.then(()=>console.log("Congrats it's head"))

// we don't use the promise at that time we can use then but we can use the lot of functions this is called call back hell

.catch(()=>console.log("sorry!!!! it's tail"))\*/

let reachA=new Promise(()=>

{

    const reached=true

    if(reached)

        setTimeout(resolve,3000,"anu reached")

    else

    reject("anu not reached")

})

let reachB=new Promise(()=>

    {

        const reached=true

        if(reached)

            setTimeout(resolve,2000,"ramya reached")

        else

        reject("ramya not reached")

    })

    let reachC=new Promise(()=>

        {

            const reached=true

            if(reached)

                setTimeout(resolve,1000,"gita reached")

            else

            reject("gita not reached")

        })

        // promise can be in three states =>pendig,resolved,rejected

Promise.all(reachA,reachB,reachC)

.then((message)=>console.log(message) )

.catch((message)=>console.log(message))

//promise.any()any one condition can be resolved everything is resolved otherrwise everthing caan be rejected

//promise.race()=>if one function is solved it can give the result

// promise can be in three states =>pendig,resolved,rejected

Promise.all(reachA,reachB,reachC)

.then((message)=>console.log(message) )

.catch((message)=>console.log(message))

//promise.any()any one condition can be resolved everything is resolved otherrwise everthing caan be rejected

//promise.race()=>if one function is solved it can give the resultt

Promise.all([reachA,reachB,reachC])

.then((message)=>console.log(message))

.catch((message)=>console.log(message))

//promise - pending,resolved,rejected (settled)

Promise.allSettled([reachA,reachB,reachC])

.then((message)=>console.log(message))

.catch((message)=>console.log(message))

Promise.any([reachA,reachB,reachC])

.then((message)=>console.log(message))

.catch((message)=>console.log(message))

Promise.race([reachA,reachB,reachC])

.then((message)=>console.log(message))

.catch((message)=>console.log(message))

**Error handling or exception handling**

// error handling or exception handling ===> try,catching and throw =>first we want to know in which code of line it has the error incase these code may occur error we want to put it in the try block

// try - we know that this code is going to get the error that want to be in user understandable format so we use the try

//num=prompt("enter a number") //Prompt ==> is to used to get the input from user

//console.log(num\*\*2)// when we give this way in input i give b the output is nan

try{

    num=prompt("enter the number")

    if (num==='')

        throw 'cannot be empty'

    if(isNaN(num))

        throw 'enter a valid number' // throw=>if the statement is not true throw message will be available

        console.log(num\*\*2)

}

catch(error)

{

    console.log(error)

}

// when we use throw we want to use the catch

finally{

    console.log('bye')

}

// if the code has the error or not finally statement can print

**Async await**

let reachA=new Promise(()=>

{

    const reached=true

    if(reached)

        setTimeout(resolve,3000,"anu reached")

    else

    reject("anu not reached")

})

let reachB=new Promise(()=>

    {

        const reached=true

        if(reached)

            setTimeout(resolve,2000,"ramya reached")

        else

        reject("ramya not reached")

    })

    let reachC=new Promise(()=>

        {

            const reached=true

            if(reached)

                setTimeout(resolve,1000,"gita reached")

            else

            reject("gita not reached")

        })

        // promise can be in three states =>pendig,resolved,rejected

Promise.all(reachA,reachB,reachC)

.then((message)=>console.log(message) )

.catch((message)=>console.log(message))

//promise.any()any one condition can be resolved everything is resolved otherrwise everthing caan be rejected

//promise.race()=>if one function is solved it can give the resultt

Promise.all([reachA,reachB,reachC])

.then((message)=>console.log(message))

.catch((message)=>console.log(message))

//promise - pending,resolved,rejected (settled)

Promise.allSettled([reachA,reachB,reachC])

.then((message)=>console.log(message))

.catch((message)=>console.log(message))

Promise.any([reachA,reachB,reachC])

.then((message)=>console.log(message))

.catch((message)=>console.log(message))

Promise.race([reachA,reachB,reachC])

//async await

// when we async before the function it always return the promise

function fn(){

    return 'hello'

}

console.log(fn())

fn().then((msg)=>console.log(msg))// here we can use the then or wait when we are using async

// when we use the async to call the resolve we don't want to usen then()

async function status(){

    console.log('hi....')

    res=await reachA// await is used only when we use the async and promise function take time to excute await is used to wait a functiom

    console.log(res)

    console.log('bye')

}

//when the promise is not resolved we want to handle that scenario in try and caught

async function status(){

    try{

    console.log('hi....')

    res=await reachB// await is used only when we use the async and promise function take time to excute await is used to wait a functiom

    console.log(res)

    console.log('bye')

    }

    catch(err)

    {

        console.log(err)

    }

}

**JSON-Java Script Object Notation**

**It is used for data transmission before using json we are using the xml**

**Xml looks loke a html**

**For eg**

**Xml program**

**<note>**

**<to>Tove</to>**

**<from>Jani</from>**

**<heading>Reminder</heading>**

**<body>Don't forget me this weekend!</body>**

**</note>**

**This XML file does not appear to have any style information associated with it.**

let json1='dave'// we can store a string,num,boolean,array and we cans store js object in json

let json2=4

let json3=true

let json4=[4,5,6,7]

let json5={

    "stock":"tcs",

    "price":3500

}// here is the minor difference between js and json to storing a object in json for key we want to write in double quotes but js not like that

/\*let json6=[

    {

       "stock":"tcs",

    "price":3500

    },

    {

        "stock":"hul",

    "price":2000

    },

    {

        "stock":"tcs",

    "price":1300

    }

] \*/  // these is the array contain the objects

`let json6=[

    {

       "stock":"tcs",

    "price":3500

    },

    {

        "stock":"hul",

    "price":2000

    },

    {

        "stock":"tcs",

    "price":1300

    }

]`

//parse()convert thst string into json Object

// i want a statement totally a string so i using backtick

let parsed=JSON.parse(json6)// this method can give an separate array of objects

//JSON.parse() is a JavaScript function that converts a JSON string into a JavaScript object.

console.log(json6)

console.log(parsed)

console.log(parsed[1])

console.log(parsed[0].price)

//stringfy=> for transmiting the objects we want  to convert into the string

//JSON.stringify() is a JavaScript function that converts a JavaScript object or value to a JSON string

console.log(JSON.stringify(parsed))

**fetch API**

// fetch API

//we can use thr github "https://github.com/public-apis/public-apis?tab=readme-ov-file"

// the github api is used to get the api link

//console.log(fetch('https://dog.ceo/api/breeds/image/random'))

// when we use the fetch we get the promise

/fetch('https://official-joke-api.appspot.com/jokes/programming/random')

.then((res)=>res.json())

.then((msg)=>console.log(msg[0].setup,msg[0].punchline))

.catch((err)=>console.log(err))

// get and post

fetch('https://randomfox.ca/floof/')//get request

.then((result)=>{

    if(result.ok)

        console.log('success')

    else

    console.log('failed')

return result.json()

})

.then((message)=>console.log(message.image,message.link))

.catch((error)=>console.log(error))

// post

fetch('https://dog.ceo/api/breeds/image/random',{

    method:'POST',

    headers:{'content-type':'application.json'},

    body:JSON.stringify({

        userId:22,

        id:123,

        title:'kural',

        completed:true

    })

})

.then((re)=>re.json())

.then((ms)=>console.log(ms.message,ms.status))

// put=>is also used but it is basically  used for update