JAVASCRIPT COADING NOTES DOCUMENT

#-JAVA+SCRIPT-# Contents:

4. comparision operator

5. greater than & lesser than

i. and

ii. or

iii. not

iv. Terinary operator

-Conditions

i. if

ii. if else

iii. if else if

iv. switch case conditional statement

-Functions

i. defining function

ii. calling function

3-types of function:

i. function with parameters

ii. function without parameters

iii. function with parameters and return else

-Events using js

OnClick()

OnChange()

-String Methods

1. length

2. touppercase()

3. tolowercase()

4. concat()

5. trim()

6. charAt()

7. indexOf()

8. slice()

-Array

1. push (add data end of an array)

2. pop (remove data end of an array)

3. shift (remove data start of an array)

4. unshift (add data start of an array)

i. Concat

ii. includes

iii. sort()

iv. reverse()

v. type of list

vi. tostring()

vii. join()

viii. splice() -permanent deletion.

-Array-

IsArray

for loop(increment & decrement)

for of (increment only)

-data inside the array is elements

-data inside the object is properties

-Object-

key-value: -pair

i. create

ii. update

iii. delete

array of objects using (For Of) method

To access (key or value) in object

Hoisting

DOM Input Values

Let concepts

Const

Spread Operator & (Array&Objects)

Destructing - It's used only in Arrays and Objects

Class Application form

Inheritance

Two types of storage in web: storing data on the client/browser

1. session storage -store data for one session

   (if browser tab closes then data is lost)

2. local storage -store data with no validty or expiry date.

JSON - Javascript object notation - data interchange format

Synchoronous and Asynchoronous:

Sync - code will execute step by step. Each instrcution waits for the previous instruction

to complete the execution

Asyn - it will allows to execute next instruction immediately and it does not block the flow.

set timeout() - to executes a block of code after some specified time. It will execute code only once

set interval() - to set a delay for function - repeated execution

---

[10-04-2023]

 # JAVASCRIPT_NOTES:

Sample:

```
<html>
   <head>
 </head>
   <body>
      <p id="demo"> </p>
<script>
document.getElementById("demo").innerHTML="WELCOME TO JAVASCRIPT";
document.getElementById("demo").style.color="red";
document.getElementById("demo").style.backgroundColor="indigo";
document.getElementById("demo").style.width="200px";
document.getElementById("demo").style.height="200px";
</script>
   </body>
</html>
```

 # Image Insert Part:

```
<html>
   <head>
   </head>
   <body>
      <img id="demo">
<script>
document.getElementById("demo").src="laptop.png(image file name)";
</script>
   </body>
</html>
```

++

DOM - Document Object Model

Document object represents the HTML documents to be displayed in web browser.

with the help of DOM, developers can create a dynamic/interactive websites

with DOM, you can access or modify the HTML content

alert("");// methods or function (pre defined function)

person - object

properties - height, weight, skin color, hair etc

methods - running(),walking(),talking()

[11-04-2023]

 # Variables :

Its a container for storing values

var is a keyword

a - identifier

var a = 10; holding number

var b = "akash"; holding string (sequence of characters)

 # Data types:

1. Primitive Data types

2. Non Primitive data types(Array, Object)

string - represents sequence of characters eg "hello" or 'javascript'

number - represents numeric values eg 1, 2 etc

boolean - value either true or value

undefined - unknown value (i have not initialized any value) eg: var a;

++

#Please ensure to check results in console : Right click inspect then console.

```
<script>
  var a = 15;
  var b = 45;
  var c = 15;
  var addition = a + b + c;
  console.log(" Addition of three number is "+addition);
  var substract = b-a;
  console.log(" substraction of two number is "+substract);
  var multiple = a*b;
  console.log(" Multiplication of two number is "+multiple);
  var division = a/b;
  console.log(" Division of two number is "+division);
  var modulus = a%b;
  console.log(" Reminder of two number is "+modulus);
</script>
```

[12-04-2023]

 # Comparison Operator:

= used for assigning values to a variable in Js

== used to compare 2 values irrespective of data type (number, string, boolean)

=== used to compare 2 values (data type should be same)

++

 # Using ===

```
<script>
var a = "1";
var b = 1;
var result = a===b;
console.log(result);
</script>
```

++

 # Using ==

```
<script>
var a = "1";
var b = 1;
var result = a==b;
console.log(result);
</script>
```

++

> - Greater than

< - Lesser than

>= Greater than or equal to

<= Lesser than or equal to

AND Operator && (*)

T & T = true

T & F = false

F & T = false

F & F = false

OR Operator || (+)

T | T = true

T | F = true

F | T = true

F | F = false

Not Operator !

True = false

False = True

++

 # AND Operator Eg :

```
<script>
  var a =10;
  var b = 20;
  var c= 50;
  var d = 30;
  var andresult = a<b  &&  c>d;
  console.log(andresult);
</script>
```

++

 # OR operator Eg :

```
<script>
  var a =10;
  var b = 20;
  var c= 50;
  var d = 30;
  var orresult = a<b  ||  c>d;
  console.log(orresult);
</script>
```

++

 # NOT Operator Eg :

```
<script>
  var a =10;
  var b = 20;
  var notresult = !(a>b);
  console.log(notresult);
</script>
```

[13-04-2023]

 # Ternary Conditions:

```
 <script>
var city = "bangalore";
var result = city=="chennai"? "You are from chennai" : "You are not from chennai";
console.log(result);
</script>
```

++

```
<script>
   var state ="Tamil Nadu";
   var age = 21;
   var result = (state=="Tamil Nadu" && (age>=18 && age <=21))? "You are eligible for free
laptop":"you are not eligible for free laptop";
   console.log(result);
</script>
```

++

 # If else statement:

```
<script>
   var city="chennai";
   var age = 25;
   if(city=="chennai" && (age>=18 && age<=24)){
console.log("You are eligible for free laptop");
   }
   else {
      console.log("You are not eligible for free laptop");
   }
</script>
```

++

 # If else if statement:

```
<script>
   var age = 8;
   if (age>=1 && age<=12){
      console.log("You fall under children category");
```

```
    } else if (age>=13 && age<=23){
        console.log("You fall under adult category");
    } else if(age>=24 && age<=45){
        console.log("You fall under adult category");
    }else{
        console.log("You fall under Senior citizen category");
    }
</script>
```

[17-04-2023]

 # Switch Statement:

Switch: Used to execute the different blocks of statment based on the value of given expression

```
<script>
var str ="sky" ;
switch(str){
case "city":
    alert(" This is Chennai");
    break;
case "color":
alert(" This is Red color");
break;
    default:
        alert(" No match found");
}
</script>
```

==========

```
<script>
var days =8 ;
switch(days){
case 0:
   console.log("Today is Sunday");
   break;
   case 1:
   console.log("Today is Monday");
   break;
   case 2:
   console.log("Today is Tuesday");
   break;
   case 3:
   console.log("Today is Wednesday");
   break;
   case 4:
   console.log("Today is Thursday");
   break;
   case 5:
   console.log("Today is Friday");
   break;
   case 6:
   console.log("Today is Saturday");
   break;
   default:
      console.log(" No match found");  }
</script>
```

 # Functions :

Function: used to perform some operation or specific task

Two main advantages : Code reuse and less coding

==========

 # Function without parameters:

```
<script>
function display(){
   console.log(" Welcome to Javascript function");
}
display();//calling function
display();
display();
display();
display();
</script>
```

++++++++++

 # Function with parameters:

```
<script>
function display(input){
   console.log(" Welcome to Javascript " +input);
}
display("akash");
display("selvam");
display("John");
</script>
```

 # Function with return value:

```
<script>
function addition(a,b){
return a + b;
}
console.log(addition(20,20));
console.log("Summation of 2 values is", +addition(34,24));
console.log("Summation of 2 values is", +addition(100,100));
</script>
```

[19-04-2023]

 # EVENTS – OnClick:

```
<html>
   <head>
   </head>
   <body>
<button onclick="onSubmit()">Submit </button>
   </body>
   <script>
function onSubmit(){
   alert("This is called Onclick event")
}
   </script>
</html>
```

++

 # Events - OnChange:

```
<html>
   <head>
   </head>
   <body>
<input type="text" onchange="Onchangehandler()">
   </body>
   <script>
      function Onchangehandler(){
         alert("This is called Onchange event");
      }
   </script>
</html>
```

==========

 # Concat Method()

```
<script>
var fullName = "AKASH";
var lastName = "Sharma";
var result = fullName.concat(lastName);
console.log(result);
</script>
```

 # Trim() - To remove blank spaces

```
<script>
var fullName = "AKASH   ";
```

```
var result = fullName.trim();

console.log(result.length);

</script>
```

 # CharAt() - method returns the character at a specified index (position) in a string

```
<script>

var fullName = "AKASH";

var result = fullName.charAt(4);

console.log(result);

</script>
```

 # indexof() - method returns the position of the first occurrence of a specified value in a string

```
<script>

   var fullName ="IMPORTANT";

   var result = fullName.indexOf('R');

   console.log(result);

   </script>
```

[20-04-2023]

 # Slice method:

```
<script>

var fullName = "akashkumar";

var result = fullName.slice(5,10);

//first parameter index (starts from 0)

//second parameter length( starts from 1)

console.log(result);

</script>
```

 # Concat:

```
<script>
var name1 ="jack";
var name2 = " sam";
var name3 = " ian";
var result = name1.concat(name2).concat(name3).concat(" aaa");
console.log(result);
</script>
```

==========

 # Array:

```
<script>
var list = [34,22,35,"jack","sam","true"];
console.log(list);
console.log(list[3]);
console.log(list[5]);
//storing data continuosly in a memory
</script>
```

==========

 # Adding new data inside the array

```
<script>
var list = [34,22,35,"jack","sam","true"];
console.log(list);
list.push(777); // will add data at the end of an array
console.log(list);
```

```
list.push(88,33,44);
console.log(list);
list.unshift(111,2222); // will add data at the start of an array
console.log(list);
</script>
```

==========

 # Removing data inside the array:

```
<script>
var list = [34,22,35,"jack","sam","true"];
console.log(list);
list.pop(); // to remove data at the end of an array
console.log(list);
list.shift(); // to remove data at the start of an array
console.log(list);
list.shift();
console.log(list);
</script>
```

==========

 # Slice method:

```
<script>
var list = [34,22,35,"jack","sam","true"];
console.log(list);
var result = list.slice(3,6);
console.log(result);
</script>
```

 # Indexof :

```
<script>
var list = [34,22,35,"jack","sam","true"];
console.log(list);
var result = list.indexOf("sam");
console.log(result);
</script>
```

==========

 # Length :

```
<script>
var list = [34,22,35,"jack","sam","true"];
console.log(list);
var result = list.length;
console.log(result);
</script>
```

==========

 # Reverse()

```
<script>
var list = [34,22,35,44,55,77];
console.log(list);
var result = list.reverse();
console.log(result);
</script>
```

#Please pay more attention to this array topic

#Please practice events and string methods like trim, length, index of

[21-04-2023]

 # Concat – Array

```
<script>
var list = [22,33,44,"salmon","jack",true];
var sublist =[65,34,77,88,99,111];
var result =list.concat(sublist);
console.log(result);
</script>
```

 # Includes : - to check if value is present inside the array If yes - answer true in case no false

```
<script>
var list = [22,33,44,55,66,77,88];
var result = list.includes(880);
console.log(result);
</script>
```

==========

 # Sorting : - either the result will be ascending or descending

```
<script>
var list = ["aaa","iii","ccc","eee","bbb","ddd"];
console.log(list);
var ascending = list.sort();
console.log(ascending);
var descending = ascending.reverse();
console.log(descending);
</script>
```

++++++++++

 # Sorting : - numbers

```
<script>
var list = [34,55,23,400,255,11];
console.log(list);
var ascending = list.sort(function(a,b){
   return a-b;
}
);
console.log(ascending);
var descending = list.sort(function(a,b){
   return b-a;
}
);
console.log(descending);
</script>
```

==========

 # tostring()

```
<script>
var list =[34,55,66,"jack","ian"];
console.log(typeof list);
console.log(list);
var result = list.toString(); // it will convert the array to string
console.log(typeof result);
console.log(result);
</script>
```

++

```
<script>
var list =[34,55,66,"jack","ian"];
console.log(list);
var result = list.join(''); //convert arrays to string
console.log(typeof result);
console.log(result);
result = list.join('#');
console.log(result);
console.log(result.split('#')); // convert string to array
</script>
```

==========

```
<script>
var list =[34,55,66,"jack","ian"]; //splice - permanent deletion
console.log(list);
list.splice(0,3);
console.log(list);
</script>
```

[24-04-2023]

 # Isarray()

to check if given element or variable is array,

it will give output as true (In case of Array) or vice versa.

```
<script>
   var list =[23,34,556,34,434];
   var result = Array.isArray(list);
   console.log(result);
   var list1 =34;
   var result1 = Array.isArray(list1);
   console.log(result1);
</script>
```

++

 # For loop:

```
<script>
var list =[23,34,55,67,888,999];
// for(var i =0;i<list.length;i++)
// console.log(list[i]);
for(var i =list.length-1;i>=0;i--)
console.log(list[i]);
</script>
```

==========

 # For of:

```
<script>
var list =[23,34,55,67,888,999];
for(var obj of list)
console.log(obj);
</script>
```

==========

 # Object:

```
<script>
var address = {
doorNo:"2nd Street",
location:"chennai",
pincode:676545,
company:"Wipro"
};
console.log(address);

//new data addition
address.state="Tamil Nadu";
console.log(address);

//update
address.company="TCS";
console.log(address);
</script>
```

[25-04-2023]

 # Object:

Object: key value pair, holds group of similar data

access the value in object then you have to use '.' or ['']

==========

 # Update the value in object:

```
<script>
   var employee = {
fullName:"sunil",
age: 44,
company:"CTS",
address:{
   doorno: 33,
   location:"chennai",
   phone:9999888899
}
   };
 console.log(employee);
 employee.age= 66; // Update the value in object
 console.log(employee);
 employee.fullName="akash";
 console.log(employee);
</script>
```

==========

 # To update key value pair inside the object:

```
<script>
   var employee = {
fullName:"sunil",
age: 44,
company:"CTS",
```

```
address:{
   doorno: 33,
   location:"chennai",
   phone:9999888899
}
   };
 console.log(employee);
 employee.salary =50000;
 console.log(employee);
</script>
```

==========

 # To delete Key value pair in object:

```
<script>
   var employee = {
fullName:"sunil",
age: 44,
company:"CTS",
address:{
   doorno: 33,
   location:"chennai",
   phone:9999888899
} };
 console.log(employee);
 delete employee.age;
 console.log(employee);
</script>
```

==========

 # Array of object:

```
<script> // array of object
var list = [
{
fullName:"sunil",
age:43,
company:"wipro"
},
{
fullName:"akash",
age:41,
company:"Facebook"
},
{
fullName:"anil",
age:33,
company:"CTS"
},
{
fullName:"ashwin",
age:43,
company:"TCS"
},
{
fullName:"John",
```

```
age:46,
company:"Barclays"
},
];
for(var obj of list){
   console.log(obj.fullName,obj.age,obj.company);
}
</script>
```

==========

 # To access Key or value in object :

```
<script>
var employee={
fullName:"ashwin",
age:22,
company:"CTS",
salary:45000
};

var keys = Object.keys(employee); // this will help us to display only keys
console.log(keys);
var values = Object.values(employee); // this will help us to display only values
console.log(values);
</script>
```

[27-04-2023]

 # Hoisting:

```
<script>
console.log("Result of 2 values",+addition(203,55));
function addition(a,b){
   return a + b;
}
</script>
```

==========

 # Array of object:

```
<script>
var students = [
{
fullName:"sam",
age:23,
DOB:"22/1/2000"
},
{
fullName:"kiran",
age:24,
DOB:"22/1/2001"
},
{
fullName:"peter",
age:25,
DOB:"22/1/2004"
},
```

```
{
fullName:"samual",
age:18,
DOB:"22/1/2010"
},
{
fullName:"jack",
age:23,
DOB:"22/1/2004"
},
];
//console.log(students);
//console.log(students[4]);
console.log(students[4].fullName);
</script>
```

==========

 # Array of object using for of()

```
<script>
var students = [
{
fullName:"sam",
age:23,
DOB:"22/1/2000"
},
```

```
{
fullName:"kiran",
age:24,
DOB:"22/1/2001"
},
{
fullName:"peter",
age:25,
DOB:"22/1/2004"
},
{
fullName:"samual",
age:18,
DOB:"22/1/2010"
},
{
fullName:"jack",
age:23,
DOB:"22/1/2004"
},
];
for(var list of students){
   console.log(list.fullName,list.age,list.DOB);
}
</script>
```

==========

 # DOM Input values:

```
<html>
<head>
</head>
<body>
<input type="text" id="one"/> <br><br>
<input type="text" id="two"/> <br><br>
<button onclick="onSubmit()">Submit</button>
<div id="demo"></div>
</body>

<script>
var a,b;
function onSubmit(){
   a = document.getElementById("one").value;
   b = document.getElementById("two").value;
   var result = Number(a) + Number(b);
   document.getElementById("demo").innerHTML="Addition of 2 values"+result;
    document.getElementById("one").value="";
    document.getElementById("two").value="";
}
</script>
</html>
```

[28-04-2023]

 # Let:

```
<!-- <script>
   var fullname = "akash";
   var fullname = "suresh";
   console.log(fullname);
</script> -->
```

==========

```
<script>
let fullname ="akash";
let fullname = "suresh";
console.log(fullname);
</script>
```

==========

```
<script>
   {
var address = "chennai";
console.log(address);
   }
</script>
```

==========

```
<!-- <script>
{
let address = "chennai";
console.log(address);
}
</script> -->
```

 # Const:

```
<script>
 const a = 34;
 a = 33;
 console.log(a);
</script>
```

==========

```
<!-- <script>
   function addition(a,b){
      return a+b;
   }
   console.log(addition(3,4));
</script> -->
```

==========

```
<script> // Arrow function
addition =(a,b)=>{
return a+b;
}
console.log(addition(30,40));
</script>
```

==========

 # Spread operator:

```
<!-- <script>
   let list = [2,3,5];
   let listtwo = list;
   listtwo.push(44);
```

```
    console.log(list);
    console.log(listtwo);
</script> -->
```

==========

```
<script>
    let list = [2,3,5];
    let listtwo = [...list]; //spread operator - use to copy array or object
    listtwo.push(44);
    console.log(list);
    console.log(listtwo);
</script>
```

==========

 # Spread operator (Arrays and Objects):

```
<script>
let student = {
fullName:"akash",
age:23
}
let student1 = {...student}
student1.address = "chennai";
console.log(student);
console.log(student1);
</script>
```

==========

 # Destructuring :

```
<!-- <script>
let list = [33,22,44];
let a= list[0];
let b= list[1];
let c= list[2];
console.log(a,b,c);
</script> -->
```

==========

```
<script>
let list = [332,221,44];
let [a,b,c] = list; //Destructuring - used only in Arrays and objects
console.log(a,b,c);
</script>
```

==========

```
<script>
   let list={
      name:"akash",
      age:33
   }
   let obj={
      address:"chennai"
   }
  let {age,name,address}= {...list,...obj};
  console.log(name,age,address);
</script>
```

[02-05-2023]

 # Class - Application form:

Name:

Father Name:

DOB:

Mobile Number:

Email ID:

```
<script>
class ApplicationForm{ // used to create the pattern   // class/constrcutor/this/new - in built functions
constructor(fullName,age,gender,address){ // used to initialize the data
this.fullName = fullName; // it refers to current object
this.age = age;
this.gender = gender;
this.address = address;
}
getEditAge(input){ // user defined function/method
this.age = input;
}
}
let akashobj = new ApplicationForm("akashkumar",23,"Male","chennai");
console.log(akashobj);
let suresh1obj = new ApplicationForm("Sureshkumar",24,"Male","Madurai");
suresh1obj.getEditAge(55);
console.log(suresh1obj);
</script>
```

==========

 # Inheritance:

```
<script>
class ApplicationId{ // used to create the pattern   // class/constrcutor/this/new - in built functions
constructor(fullName,bloodgroup,designation){ // used to initialize the data
this.fullName = fullName; // it refers to current object
this.bloodgroup = bloodgroup;
this.designation = designation;
}
getfullname(input){
   this.fullName=input;
}
}
class FoodToken extends ApplicationId{ // inheritance
   constructor(fullName,bloodgroup,designation,amount){
      super(fullName,bloodgroup,designation);
      this.amount = amount;
   }
}
let akashFood = new FoodToken ("akash","B Negative","Team Lead",3000);
console.log(akashFood);
let sureshFood = new FoodToken ("suresh","B positive","Deputy Manager",4000);
console.log(sureshFood);
</script>
```

[02-05-2023]

 # Inheritance:

```
<script>
class ApplicationId{ // used to create the pattern   // class/constrcutor/this/new - in built functions
constructor(fullName,bloodgroup,designation){ // used to initialize the data
this.fullName = fullName; // it refers to current object
this.bloodgroup = bloodgroup;
this.designation = designation;
}
getfullname(input){
   this.fullName=input;
}
}
class FoodToken extends ApplicationId{ // inheritance
   constructor(fullName,bloodgroup,designation,amount){
      super(fullName,bloodgroup,designation);
      this.amount = amount;
   }
}
let akashFood = new FoodToken ("akash","B Negative","Team Lead",3000);
console.log(akashFood);
let sureshFood = new FoodToken ("suresh","B positive","Deputy Manager",4000);
console.log(sureshFood);
sureshFood.getfullname("Rakesh");
console.log(sureshFood);
</script>
```

[04-05-2023]

 # Storage Types:

two types of storage in web: storing data on the client/browser

1. session storage - store data for one session (if browser tab closes then data is lost)

2. local storage -store data with no validty or expiry date.

==========

```
<script>
//sessionStorage.setItem("fullName","kumar");
let name= sessionStorage.getItem("fullName");
console.log(name);
</script>
```

==========

```
<script>
   localStorage.setItem("fullName","akash");
   let names = localStorage.getItem("fullName");
   console.log(names);
   localStorage.removeItem("fullName");
</script>
```

==========

 # JSON:

JSON - Javascript object notation - data interchange format

for storing and transmitting data. To send data from server to web page .

```
<script>
let obj ={
   fullName : "albert",
```

```
   age : 23,
};
console.log(typeof obj);
console.log(obj.age);
let list = JSON.stringify(obj); // convert JS object to JSON string
console.log(typeof list);
console.log(list.age);
list = JSON.parse(list); // convert JSON string to JS object
console.log(typeof list);
console.log(list.age);
</script>
```

==========

 # Synchoronous and Asynchoronous:

Sync - code will execute step by step. Each instrcution waits for the previous instruction to complete the execution

Asyn - it will allows to execute next instruction immediately and it does not block the flow.

```
<script>
setTimeout(()=>{
console.log("it will execute in 10 seconds");
},10000)
let firstName = "kumar";
console.log(firstName);
function addition(a,b){
   console.log("inside funtion");
```

```
   return a+b;
}
console.log("outside the function");
addition(3,4);
</script>
```

==========

set timeout() - to executes a block of code after some specified time. It will execute code only once

set interval() - to set a delay for function - repeated execution

 # Set timeout()

```
<script>
setTimeout(()=>{
alert("function will be called in 2 seconds");
},2000)
</script>
```

 # Set interval()

```
<script>
setInterval(()=>{
alert("every 4 seconds it will be called");
},4000)
</script>
```
