Java script

1) what is Java script?
→ Java script is programming language used to create and control dynamic website content.
like: animated graphics, photo slideshow

2) Features of JavaScript?
→ it is case sensitive
→ object orientation → like OOP's object create both data & function that manipulate data.
→ cross platform compatibility
→ it can be used in HTML & CSS to create websites
→ control statements → if else, nested if, for, while

3) Why we use javascript
→ Bcz it is more webbased. - to adding behaviour to the document
→ empty lines allowed
→ it executes in synchronous way. -

4) Difference b/w Let and var.

Let	var
→ is a keyword used to declare variable	→ is a keyword used to create variable
→ The var have function scope	
→ the let have block scope	→ var Statement have function scope
→ We cannot declare a let variable more than once	→ we can declare a variable again
→ let not support hoisting	→ variables are start with - / $ / alphabets it doesn't start with no.
• Const:- Can't be redeclare within same block. let & const supports the ES6. (ECMA)	→ var supports hoisting
→ Const for read only	

5) what is data type?
→ Data type describe the different type of data that we're gonna working with & storing in variable

6) Type data
i) primitive data type - we can store only one data
→ are immutable

ii) non-primitive data type → we can store multiple data type

7) Difference b/w undefined and undeclare
undefined
→ it occurs when a variable has been declare but not has been assigned value
→ ~~it is not a keyword~~
→ if there is no error it shows as undefine

undeclare
→ A variable is undeclared when it is used without being previously declared using 'var' 'let' 'const'

DATE: PAGE:

Q) Difference b/w Java & JavaScript

Java	JavaScript
Java is a object oriented programming language	Javascript is a Scripting language
used for developing complex enterprise apk	used for creating interactive & dynamic webpages
Java is mainly used for backend	used for both frontend & backend
> use more memory	> use less memory
> it requires a JDK to run code	> it requires any text editor / browser compiler to run code

Declare : the creation of variable & function

Initialization: assign initial values to variable

Java script is single threaded because it is interpreted language it doesn't

Q) Features of Javascript
> Scripting language
> Cross platform compatibility
> object oriented

DATE: PAGE:

10) V8 Engine is a open-source JavaScript Engine developed by google. it is primarily known for its use in google chrome web browser where it serves the engine that execute JScode within webpage

11) what is operators?
operators are symbols b/w the values that allow different operations like add, sub.

12) Types of operator
- Arthematic operator → + - %.

- Comparision operators:- The comparision operator operator compares the operands/values & returns a logical value like true or false, ==, === >,<=

- logical operator:-
&& = combining both condition
|| = it returns the even if one of the operand is true
! = Truthy or falsy values

- Increment operator → a++ → post increment
++a → pre increment

- Conditional Ternary operator → it is similar to if else operation.
Syntax condi ? expres is true : is false.

odd
multiple
even

Condi
if →
Syntax
→ if
condi
exe
→ if

odd
Multiplication
even

DATE: PAGE:

Conditional Statements:- are used to control a flow of program based on certain conditions.

→ if :→ use if statement to specify a block of code to be executed if a condition is true.

Syntax:- if (condition) { // . . . }

→ if-else :→ ~~use the if else block~~ in the if else if the condition is true it executes the statement otherwise it execute false statement. Syntax:- if(cond){ ... } else{ ... }

→ if else if:- use if else if statement:- used if else if statement to specify new condition if 1st condition is false

Syntax if (cond) { ... }
else if(cond) { ... }
else {
// ...
}

→ Switch:- it is used to perform ~~perfor~~ multiple condition

Syntax switch (cond) {
case x;
. . . .
break
~~case~~ y:
default:
// code block
}

DATE: PAGE:

non-primitive data types:- it is type of data stores multiple data type & data can be modified [changable]

- array
- objects

i) Array:- it is a ordered collection of heterogeneous elements. heterogeneous means different data types
→ non primitive data

declare array let arr = [ ]

let arr = ['abcd', 'efg', 'hij', 'klm']

methods:-
- push() → it add an element to end of array
- pop() → it remove — (— — (—
- shift() → it — (— starting of array
- unshift() → it add an element to start of array
- splice → it used to add & remove the array
- concat → — (— concat the array
- reverse → reverse
- Join → Join all ~~elements~~ into a string (' ')
- ~~palindrome~~ →

splice → let a = ['ab', 'cd', 'ef', 'gh', 'ij']
a.splice(1, 2, 'cde')
output) let a ['ab', 'gh', 'ij']

DATE: PAGE:

Hoisting:- it is default behaviour of Javascript moving all declarations to top of current scope.

Enable TO extract all the values of variable & function even before initializing values without getting error & happens during 1st phase.

Ex function a(b, d){ , var & function can do in hoisting.
hoisting { } a(1, 2)

Looping statement:- looping statement allows you to repeatedly execute a block of code as long as condition satisfied

for loop:- it repeatedly executes a block of code until condition became false.

→ classical for , of , in

for [initialization ; condition; increment/dec){ // }.

i] for of :- it loops through the value of an iterable object.

let nm = ['abc', 'def', 'ghi']
for(pa of nm){
console.log(' $nm is a alphabet']

i→ for in:- it loops throug the position index of an element.

let nm = ['abc', 'def', 'ghi']
for(pa in nm){
console.log ('$nm' is in the position')
}

DATE: PAGE:

→ while loop: executes a block of code only if condition is true. and continue as long as specified condition is true.

Syntax: while (condition) {
// execute only if condition is true.
// controlling parameter (i++ or i--)
}

→ do-while loop: it is similar to while loop but it first execute the statement after the it will check the condition

do {
// statement
// control parameter
} while (condition)

for loop

while loop

do-while

PAGE: DATE: PAGE:

→ function
* A Java Script function is a block of code or do. perform a particular task.
* Function is executed when something calls it.

→ why we use functions
* with function you can reuse code
* once you write the code we can use many times.

syntax: function abc() {
console.log(' ')
}
abc()

~~pal bap~~

→ max & min no
let x = [12, 10, 9, 30, 64]
let y = x[0]
for (i=0; i<x.length; i++) {
if (x[i] > y) {
y = x[i]
}
}
console.log(y)

→ palindrome
let name = 'deepak'
let rev = ''
for (i = name.length-1; i>=0; i--) {
rev += name[i]
}
if (rev == name) {
console.log('it is palindrome')}
else {
console.log('it is not a pal')
}

DATE: PAGE:

→ add all no

```
let sum=0, function plus( ){
for(let add of arguments){
  sum= sum+add
}
console.log(sum)
}plus(5,3,2,6)
```

→ Anonymous function → function without name

→ Expression function → are the anonymous function after we create a function without a name & we assign it to be variable / Expression function to evaluate a anonymous function.

→ Arrow function → Arrow function is alternative to write a function. however function declaration & arrow function have some minor difference
Arrow function uses arrow to ~~omit~~ instead of the keyword function to declare a function

Eg

```
const square = n => {
  return n*n
}
```

→ Self invoking function → also called IIFE
immediately invoked function execution
→ function call itself function
→ This is only one time callable
→ We cannot use any variable & function name to invoke the function.

Javascript

1) what are the primitive data type in Javascript?
→ The primitive data type store only single type of data.
data types are.
- number
- boolean
- String
- undefined
- Symbol
- null

2) Explain difference between null and undefined in Javascript.
→ null:- is a value can explicitly set to indicate the absence of any meaningful value.
- it explicitly set by programmer.

→ undefined:- means a variable has been declared but has not been assigned value.
- it is built-in primitive value in Javascript.

3) How do you check the data type of variable in Javascript?
→ we can check the datatype of variable using the 'typeOf' operator.

4) Explain the concept of truthy & falsy values in Javascript. provide examples.
→ in Javascript categorized truthy & falsy value in Boolean data type.

Ex: let x = true
console.log(x)

5) What is the difference b/w == and ===
operators in JavaScript, & how do they relate to data types.

→ == is This operator comparing values
=== This operator comparing the both values & type of operands.

Example:
console.log(5 == "5") comparing value
→ it returns true
console.log(10 === "10") comparing the types
→ it returns false

7) Explain the difference b/w the ++x and x++ increment operators in JavaScript.

→ ++x and x++ → Both are increment operators
++x → this is pre increment operator
x++ → this is post increment operator

8) How does JavaScript handle NaN (not a number) values, & how can you check if value is NaN?

→ In JavaScript 'NAN' is a special value representing the result of an operation that cannot produce a meaningful numeric result.

Ex

```
let x = 5
console.log(x * "abc")
```

9) ~~what is~~ Explain the concept of type coercion in Javascript. Provide examples.

→ Type coercion in Javascript is the automatic conversion of value from one data type to another during runtime of a program.

There are two types of ~~cont~~ coercion

- Explicit Type conversion
- implicit Type conversion.

Ex:

Implicit coercion:

This occurs automatically during operation and the Javascript tries to convert one or both operand to a common data type

```
let y = 50;
    z = "20"
console.log (y+z)
```

Explicit coercion:

Developers can also explicitly coerce value from one type to other using function

```
let x = 489
    y = String(x)
console.log (typeof (x))
console.log (typeof (y))
```

10) what is the purpose of the undefined data type & when might it be explicitly assigned to a variable?

→ undefined is a primitive data type, it occurs when we declare a variable but we don't assign the value.

Ex

```
let z;
console.log (z)
```

DATE: PAGE:

11) How do you create and use template literals (String interpolation) in Javascript

→ We use template literals enclosed within the backticks (` `) instead of single or double quotes. & we use in $(dollar) symbol & we use curly braces.

Ex let x = 10
y = 20
console.log (`the sum of ${x} and ${y} is ${x+y}`)

12) What is hoisting?

→ Hoisting is default behaviour of Javascript it moving declaration to top of current scope.

13) What is IIFE?

→ IIFE is stands for "Immediately invoking function expression" also called as self invoking function and function call itself function

Ex

14) What is meant by default parameter Parsing.

→ ~~default~~ default parameter mean we are allow to specify default values for function parameters. when the function is called, if value is not provided for parameter, then deafault value is used.

Ex function info(name, age = 16) {
console.log (`My self ${name} and
My age is ${age}`)
} info('AJay')

15)

16)

15) what is default return value?
→ The default return value is undefined.

```
function a(x){
    return
}a(8)
console.log(a(2))
```

16) How to pass unlimited number of parameter to a function
→ We pass unlimited number of parameter by using Arguments keyword. and we in the Arrow function we use rest parameter.(...)

DATE: PAGE:

Javascript Scope

→ The scope refers to the context of variables. function declared & objects are accessed

2 types of scope
- local scope
- global scope

→ local scope:- Local variables are declared within the function @ block level of where they declared.
- variable declared inside a function

→ global scope:- global variable can be accessed from anywhere of the code, both inside and outside function.
- variables declared outside of any function

In Javascript function has block scope & functionScope that introduced in ES6 version, ECMA [European computer manufacturing Association., with let & const declaration any block enclosed by { }

Hoisting:-

Hoisting is a behaviour of Javascript where variables & function declaration moving to the top of the containing scope. during compilation phase before the code executed.

→ What is higher order function?

} Any HOF's is a function that takes ~~one~~ ~~or more~~ another function as arguments @ return a function as its result

→ Array higher order function

→ what is foreach?
→ The ForEach method is used for iterating over a each element in array.
→ The purpose of foreach is to perform an action foreach ~~element~~ array element without creating a new array.
— array.forEach(<callback(element, index, inp_array)>)

→ what is map?
→ The map method creates a new array by applying a provided function to each element of original array, modify array elements.

→ what is filter?
→ The filter method creates a new array containing only elements that satisfy a provided condition
· it returns a filtered version of original array

or

→ what is reduce?
· The reduce method is used to accumulate values of an array into a single value.
syntax: array.reduce(<callback>, accu, current, init>)
accumulator:- initialize the output value)

→ what is every()?
→ ~~ever~~ it check every element, if in one element condition false it returns false. [every element condition is true]

→ what is some()?
→ it also check every element if one element has true it returns true

callback function is a function that is passed an argument to another function

DATE: PAGE:

① what is Destructing
→ Distructing in Javascript is way to extract values of arrays & objects & assign them to variable in concise & convinient manner

② what is callback?
→ JS callback is a function which is to be executed after another function has finished execution.

③ what is call() method?
→ we can use call method to pass the object, other object to different object methods.
· one object can use other's object method.
· we use comma seprated values using arg

④ what is apply() method?
→ we use apply method when we pass the args to function we give args in array []
apply() & call is used for same purpose.

⑤ what is bind() method
→ we use bind() method "holds" the function in a variable

⇒ These 3 methods or objects borrow a functionality from object.

DATE: PAGE:

→ Difference b/w Arrow function and Normal function

Arrow function

Binding of this
- Arrow function inherit 'this' from sorrounding code.

Argument object
- Regular function have an 'arguments' object that holds the arguments passed to the function

- Arrow function cannot be used with ~~new keyword~~ constructor new keyword

1. → Introduced in ES6 as more concise way to define function
→ Don't have their own this, args, bindings, instead they inherit these from the enclosing scope
2. → cannot be used as constructor (i.e. we cannot use new with arrow functions) ∴ ~~type error~~ runtime error.
→ Automatically return the result of the expression if there is no curly braces {} around body
3. → Arrow function are not hoisted They must be define before they are used
4. → Arrow function don't have their own this. bcz it is not binding
b. → it is not well suited for designing object methods

Normal function

- Regular having their own 'this' context which can change
- in arrow function dont have their own 'arguments' object but can access it from their sorrounding scope
- Regular function can be used with 'new' keyword for constructor function.
- this is traditional way to defining function in JS
- Have their own this, args bindings
- can be used as constructors with the new keyword to create instances
→ more flexible & suitable for complex function, especially those that require custom binding of this or use of args
- Hoisting applies to variable declarations & the function declaration. Because of this JS functions can be called before they declared
- JS function can be used to describe the objects

closure is created when function defined in another function, allowing their inner function to access variable from outer function scope

2 → what is closure
→ closure is a function having access to the parent scope, even after the parent function has closed

today
① what is OOPs?
→ in oop we create a template for an object

② constructor → it is initialization
→ it is a special function that will be called when we create object. class, that creates & initializes an object instance of a class.

③ what is new keyword?
→ new keyword creates a new block of memory, & we use new keyword to access/call the class name method. create a object.

④ what is instantiation
→ The process of creating object is called instantiation

→ what is object?
→ collect of properties is called object

⑤ what is static method?
→ The static is a keyword it is, in this method cannot be access through instantiated objects. but could be accessed through classname.

∴ inheritance evenly applies to static methods

→ ~~To what~~ How to use inheritance
→ To create a class inheritance, using extend keyword
A class is created with a class inheritance inherits all the methods from another class.

Inheritance is a feature for code reusability.
reuse the properties & methods of existing class when you create a new class

Difference b/w class & object

class	object
• class is used as a template for declaring & creating objects	• An object is a instance of class
• when class is created no memory allocated	• objects are allocate the memory space in the memory use new keyword
class has been declared first & only once	• An object is created many times as per requirement
• it is declared with class keyword	• it is declared with new keywords in Java.

DATE: PAGE:

2 ⇒ this. keyword
⇒ this is a keyword that refers the object
⇒ which object depends on how 'this' is being invoked or called.

⇒ Spread operator:- allows us to expand an array/object into its individual elements
⇒ Rest operator:- condense multiple element into single array

⇒ ~~JSON.parse~~ ⇒
⇒ JSON is Javascript object Notation. it is text based format for representing structured data based on Javascript object syntax
Commonly used for transmitting data in web application
e.g. Sending some data from server to client,
JSON.parse - TO convert
JSON.stringify - TO convert
JSON.parse:- A common use of JSON is to exchange data to/from a web server.
When reciving data from a web server the data is always string. parse the data with JSON.parse(), & the data becomes a Javascript

JSON.stringify: it takes a object & returns the JSON string.

object destructuring:

→ let Person = { name: "John", age: 30 }
let { name, age } = Person
console.log(name) // output: John
console.log(age) // output: 30

⇒ closure:-
closure is a created when function defined in another function allowing their inner function to access variable from outer function.

⇒ function outerFunction (outerVariable) {
return function innerFunction (innerVariable) {
console.log ("outerVariable is:", outerVariable)
("innerVariable is:", innerVariable)
}
}

⇒ Object methods:- These 3 methods borrow some functionality.

① call()
it is a method that allow you to call a function with a specified 'this' value and arguments provided individually.

⇒ function greet() {
return "Hello" + this.name
}
let person = { name: "John", age: 30 }
console.log (greet.call (person))

Why, What, How, How /

DATE PAGE:

② apply():-
→ apply is similar to call. but it accepts args are an array.
→ This can be useful when you want to use an array like object as the arguments for a function.

```
function meet(greeting){
    return greeting + ", " + this.name.
let person = {name: 'John'
              age: 30
};
console.log(meet.apply(person, ['Hello']))
```

③ Bind():-
→ it allows you to create a new function with a specified 'this' value & initial args.
→ it doesn't call the function immediately, instead, it returns a new function with specified parameter.

⇒ Errors:-
are the statements that don't let the program to run properly
- Syntax errors
- Runtime —"—
- logical —"—

→ Exceptions:- An Exception signifies the presence of an abnormal condition(s) which requires special operable techniques

New

DATE: PAGE:

⇒ Exception handling:- it is a programming technique, used to detect, respond and manage the exceptional conditions that may occur during execution of a program.

* try:- it allow you to define a block of code to be tested for error while it is being executed.

* catch:- it allow you to define a block of code to be executed if any error occur in the try block.

* finally:- it is a statement which you execute code after try and catch, regardless of result.

⇒ oops:- it is programming paradigm, centred around objects rather than function.

⇒ classes:- class is a keyword acts as a template / blueprint for creating object.

⇒ object:- it is a collection of properties

⇒ instance:- Instance is a variable & methods that object have.

⇒ Constructor:- it is a special function that create & initialize the object [instance of class]
properties

2 ⇒ Instantiation: The process of creating object

→ Constructor

Ex for class, object & constructor

```
class House {
    constructor (color, floor) {
        this.clr = color
        this.floor = floor
    }
    Temple() {
        return "Ganesh"
    }
}
let Hob = new House (red, 2)

console.log(Hob)
```

Temple

Static methods & static variables

→ In classes provide a way to define function ality and data.

→ that is associated with class itself rather than with instances of the class.

⇒ Static method: These are defined using the static keyword within the class definition

→ these cannot access instance properties and methods using this keyword as they are not bound to any instance

we initialize another method in the class we use object
" dont create another variable & object to return that method"

eg. class abc {
static xyz() {
console.log('...');
}
}
abc.xyz();

→ static variable:- in Javascript does not have built in support for static variable in classes so we can achive same effect by defining static property using the static keyword.

Inheritance:-

In Javascript, class inheritance allows one class to inherit properties and methods from another class.

→ Syntax for class inheritance
- we can use extends keyword to create a subclass that inherit from superclass.

→ super keyword used to call constructor of the superclass & access methods and properties of the superclass.

Eg:- class abc {
constructor() {
}
}
class bcd extends abc {
constructor() {
super();
}
}

overriding is a method replaces the code of the method in the superclass with that of subclass

2 ⇒ Polymorphism
In Javascript, polymorphism is primarily achieved through the concept method overloading, & Method overriding.

Method-overloading :- it is feature of object oriented programming where two & more function have same name but different parameter

Method-overriding :- when child class method overrides the parent class method of the same name, parameter and return type.

⇒ Does JS support multiple inheritance
No. Javascript does not support multiple inheritance in the classical sense where a class directly inherits from multiple classes, unlike some other programing laggage like c++ & python.

JSON - Javascript object notation.

JSON.parse() :- it is used to exchange data to/from a web server.
** when recieving a data from a website, the data is always a string.
We use parse the data become Javascript object.

JSON.Stringify() :- it Serialize a Javascript object into a JSON.string out of an object & array.

callstack :- it tracing currently execution function

callback queue :- it is queue it push the function when callstack is empty . it allows the function from library

① Event loop :-

→ The event loop is a cycle that ensures a JavaScript can handle multiple tasks concurrently without blocking the main thread.

In JavaScript, event loop is a crucial part of its runtime environment. allowing asynchronous operation to be managed efficiently. It enable a non-blocking I/O operations such as handling user interaction, timer, & network input

→ it continuously execute the callstack & callback queue if the callstack is empty it take the first function from callback queue & push into call stack for execution

② Promises :- A promise is an object that may produce a single value sometime in the future. either a resolved value (or) rejected value.

Promises is good way to handle asynchronous operation > fetching data from server, reading files

→ it is used to find out the asynchronous operation is successfully completed (or) not., settled only once.

⇒ promises can be in 3 states

pending

. fulfilled

. Rejected

once the promise is settled - then we cannot change

Static methods :-

1) Promise.all() :- fulfill when all the promises fulfill reject any of promises Reject

2) Promise.allSettled() :- fulfill when all promises settle. when there is rejected it shows as reject

DATE: PAGE:

promise. anyc fulfill when any of the promises fulfill. reject when any the promises reject

•Then → it is called in promise object and a callback function that will be executed when the promise is resolved

•Catch → method is used for error handling in promise composition. this will be executed when the promise is rejected.

•Await & Async →

→ Async → in Javascript asynchronous function denoted by the 'async' keyword, it automatically returns a promise allow you to write a code that performs asynchronous operation more easily & with a cleaner syntax. it allow you to code to

→ Await → in JavaScript 'await' keyword is used inside the asynchronous function to pause the execution of the function until a promise is resolved (or) rejected.

→ Callback hell :- it is also known as "Pyramid of Doom" it used to describe a situation in Javascript where the multiple nested callback functions make the code hard to read it occurs when dealing with asynchronous operation To handle the callback hell by using Promises (or) Async and await

DOM → Document object model
When the web page is loaded, browser creates a DOM of the page
- The HTML DOM is a standard for how to get, add, change, delete the HTML element
- it is a programming interface for web document. it represent the page so that program can change the document structure, style and content

Events:-
Dom events are action that happen in the browser as result of user interaction

① Mouse events:- These events occur when a user interact with mouse
Ex click, mouseover, mouseout

② Keyboard event:- Ex keydown, keyup, keypress

③ Form events:- Ex Submit, change, focus, blur

④ window events:- These events occur when certain actions are performed on browser window
Ex load, resize, scroll

⑤ Document events:- Ex DomContentLoaded, unload

⑥ Media event:- These events are related to multimedia elements such as audio, video
Ex play, pause, ended etc

DATE: PAGE:

2 ⇒ Event Listner:- An event listner is a piece of code that waits for a specific event to occur & then responds to that event by executing predefined actions or functions. [it "listens" for the event to happen and then triggers a response]
- addeventlistner

⇒ Event handling refers to the process of defining & executing actions in response to event that occur in a webpage events can be operations like (clicking a button)

⇒ callstack:- JS uses callstack to track the behaviour function in a program
The callstack works on the LIFO principle. This means that the most recently called function will be the first to be completed
→ when function is called a new frame is ~~pushed onto call stack~~ added to top of stack

⇒ callback Queue:- is pass args to another function, it is a function which is to be executed after another function has finished its execution [QUE] is act like a FIFO principle

→ Inside the taskque task are classified 2 categories
Micro task & macrotask
→ Micro task :- it is a small job that runs after the current script finishes & when there's no other code running, but before Javascript starts on

callstack - LIFO
- Queue - FIFO

Storages

DATE: PAGE:

① Navigator:- in the navigator object is a part of window object and provides information about the users browser.
- it is used to detect various properties and capabilities of the browser.
- which can be useful for developing web application & website.

```
const userAgent = navigator.userAgent;
console.log(`userAgent: ${userAgent}`)
```

check cookies is enabled or not

```
const cookieEnabled = navigator.cookieEnabled;
if (cookieEnabled) {
  console.log('is enabled in this browser')
} else {
  console.log('not enabled')
}
```

web storage

② what is web storage?
→ webstorage can store and access from one origin. web storage API mechanism to store key-value pairs into the browser.

③ Types of storages
→ local storage
→ session storage

② ① What is local storage?

→ This storage allow you to store key-value pairs in a persistant manner.

› The advantage of local storage is it does not clear itself even when a user closes the webbrowser or browser tab. ② shutdown the system & come back some browsing ③ switch off the data is still persisted in the browser.

› Storage limit is the maximum amongst the two browsers we get at least 5mb of memory.

③ How to use local storage in our browser?

Store data in localstorage

localStorage.setItem('username', 'John')

localStorage.setItem('age', 31)

④ Retriving data from localStorage

const username = localStorage.getItem('username')

const age = localStorage.getItem('age')

⑤ Remove data from localstorage

localStorage.removeItem('age')

⑥ clear all data from local storage

localstorage.clear()

⑦ ⑧ What is Session Storage?

→ The data stored in the storage is persisted only till the user is on the webbrowser window. As soon as the user closes the window ⑨ tab of browser the data is lost.

› Session storage is never transferred to the server, while making the network call.

→ it is very useful for authentication cookies

→ ~~we store the~~ maximum we can store 5mb data.

(8) what is cookie?
→ cookie can store 4000 bytes data.
- it is a small piece of data that are sent from the server to the browser & stored in text file

10) What is Debouncing?
→ debouncing delays the execution of our code untill the user stop the performing a certain action for specified amount time.

11) what is throttling?
→ Throttling limits the execution of your code to once in every specified Time interval

→ There two techniques are used in J's to optimize the performance when handling event that occur frequently, such as scrolling, resizing @ keyboard

Why? These two techniques are used to improve the performance when handling the events like mouse movements in Java-script.

(6) Throttling:- limits the rate at which a function can be called

(1) what is cleartimeout?
→ to clear & reset the timer created by SetTimeout

Clearinterval → Stop the execution of interval

assign

Ok I will learn

DATE: PAGE:

2 ① Explain shallow copy & deep copy?

→ shallow copy - it a copy ~~a that only goes~~ object and all it's properties, but any objects (or) array will still reference the same memory location as the original object.

→ it mean that if you make changes to the nested object it will also affect the original object as well as copied object

Ex let a = {a:5; b:{c:4}}

let b = {...obj}

a.a = 6

b.b.c = 7

console.log(a) → a:6, b:{c:7}

console.log(b) → a:5, b:{c:7}

② Explain deep copy?

→ A deep copy is copy that creates a new object with new memory location for all of it's properties & nested objects (or) array, it mean if we make change in the copied object (or) any of it's nested object (or) array it will not affect the original object.

Ex let oobj = {a:1, b:{c:2}}

let ddc = JSON.parse(JSON.Stringify(originalObject))

oobj.a = 3

ddc.b.c = 4

console.log(oobj) →

console.log(ddc) →

Prototype:- & Prototype chaining it is crucial for understanding how objects and Inheritance work in the language

→ All Javascript objects inherit properties and methods from a Protype

Q. What is prototype?

→ in JS every object has a property called Prototype

Prototype chaining:- is the process that Javascript engine use to look up properties and methods of an object. it mechanism that allows the object to inherit the properties and methods from other object

Prototype

Why we need Prototype & Prototype chaining?

→ Prototype chaining is process that JS engine use to look up properties and methods of an object. if property & method does not found in object it will, then it took up the prototype chain. it check the object prototype. the prototype's prototype & so on. until it finds the property or reaches the end of chain in which it will.

→ This mechanism allows the inheritance feature in JS, allowing object to inherit properties & methods from their prototype

○ what is macrotask :- set interval, thread — diff b/w call stack & macrotask
— " — microtask :-

DATE: PAGE:

2 ① what is event bubbling?
→ bubbling travels from target to root (or) Bottom to top
- while developing the in a webpage or website via Javascript the concept of event bubbling is used where the event handler invoked when one element is nested to the another element and part of same event.
This technique (or) method is known as Event Bubbling
→ performing the eventflow for a webpage at the time we use. The propagation of event is done.

② what is event capturing?
→ Event capturing is opposite to bubbling, where in event capturing, an event moves from the outermost element to target

modules:-
what → A module in Javascript is just a file containing related code
how → we use export & require keywords to receive functionality from different modules
why → modules are allow us to break up code into seperate files. This make easier to maintain code base

Export statement is used in modules to expose the variables, function, (or) Export the values from a Javascript module

HTTP module
Event module

Require :- is used to import modules into our code
(when we call require() with path to a module, NodeJs will search for the module in specified location & load it in our program. the function return the export object of module, it contain any variable & object that module wants to make available to other parts of program).

what Node JS :- is a open source, cross-platform JavaScript runtime environment built on chrome's V8 JS engine.
→ it is used to software used to execute the JS code

why :- NodeJS use an event-driven, non-blocking I/O model, particularly for applications that involve a lot of I/O operation.

what Local Host :→ in computer Networking host mean a "Server". Just like you can put a website on the internet by hosting it on a Server. We can make your own computer that server.
and we give IP address like 127.0.0.1

local host is the computer & hostname currently making a request to itself, accessed through the loopback IP address 127.0.0.1

what Server :- A Server is a computer that provides services & resources to other computers, known as clients, over a network.

why Servers are designed to handle requests from clients & deliver responses accordingly.

Http methods :- get, put, post

dependencies

HTTP Methods:-

GET:- Request a Representation of specified Resource

POST:- Used to Submit an entity to the specified resource

PUT:- Replaces all current representations of the target resource with the request payload

PATCH:- Applies partial modifications to a resource

~~console~~ ~~Data~~ ~~send~~ ~~else~~

~~write File~~

~~cli tools~~

(2) what is HTTP?

communication between client computer & web servers is done by sending HTTP Requests and receiving HTTP Responses.

(3) what is npm?

npm stands for "Node package manager"

it is a package manager for the Javascript programing language

why → it is used for managing dependencies in Node.JS projects & it allows developers to easy install, & manage packages of code written in Javascript

npm → install packages npx → executing npm packages

(4) what is yarn?
→ yarn is a another package like npm to manage for javascript. it was developed by Facebook in collaboration with other companies.

(5) what is nodemon?
→ Nodemon is a utility tool for NodeJS application that monitors for changes in your source code and automatically restarts the nodeJS apk when changes are detected.
it is used in testing & development & also helps to execute without restarting.

How works: it works by continiously monitoring the files in your NodeJS apk's directories for any changes. when it detect that file has been modified, added or deleted nodemon automatically restarts the NodeJS apk.

(6) what is express?
→ express is a minimalist web application framework for nodeJS.

Why: it simplifies the process of handling HTTP requests and responses making it easier for developers to create server side application.