1) what is the Purpose of css media queries?
→ is to create responsive design for website
& allow you to create different layouts
depending on the size of the viewport. &
a part of a style sheet can apply in CS

```
@media (max-length: 50Px;
        display: none
        background-colour: Blue;
}
```

2) How do you write a basic media query in CSS?
→ it consists of a media type & can contain one or more media features, which resolve to either true or false

```
@media (width <= 600Px) {
    body {
        background-color: green;
    }
}
```

it checks like a true & false condition

3) Explain the difference b/w max-width and min-width in media query?
→ if we want to define
max-width → Sets the maximum width of an element
min-width → Sets a minimum width of an element

4) What is the purpose of the viewport meta tag in responsive web design?

→ The Meta tag viewport gives the browser instructions on how to control the Pages dimensions & Scaling.

* The device should tell the how our website is looking, wiodevice width, zoom in - zoom out

<meta name = "viewport" content = " width device_width, initial-Scale 1.0 " />

5) How can you apply different Styles for landscape and portrait orientations using media queries

→ Based on the orientation of the device wether its in landscape or portrait mode.

```
body {
    back color : #fofofo;
    Color : #333;
    font family : Arial
}
@ media Screen and (orientation : portrait) {
    body {
        ont-size : 16 px
        Padding : 20 px
    }
@ media Screen and (orientation : landscape) {
    body {
        ont-Size : 18 Px ;
        Padding : 30 Px
    }
```

Orientation & landscape Arjust the Properties according to design Performance

6] explain the concept of a mobile - first approach in responsive design.

→ start small : Design the website with smaller screen in mind typically those of mobile phones critical element established in small screens

* Progressive enhancement :- once the mobile version is established, design for larger screens like tablets & laptops, desktops adjust more complex features

* Responsive design :- use responsive design such as fluid grids & media queries adjust based on the screen size.

7] what are common break points used in responsive design?

→ * major & minor

* increase the hight of the element,
  font size of the element
  max width of the element
* increasing the large image

8)

→ rem (root em) unit, root of our document

* the purpose of regular CSS the way to define relative to the root elements font size layout and styles are responsive for the font size.
* Scaling the document, adjust the font size changes & height

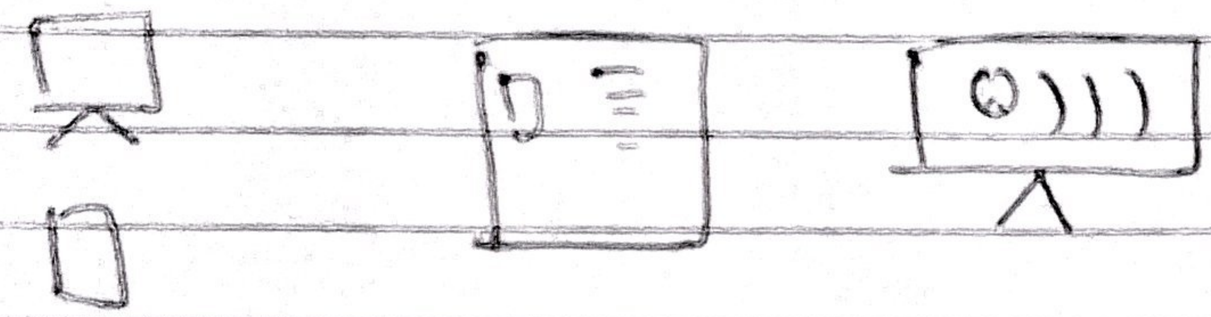
@ media screen and (min-width: 30rem)

→ Style for screen wider than 30 times the base font size

9) How can you combine media query in CSS
→ multiple media query by separating them with comma

10) what is the significance of all keyword in media query
→ All media tags

Screen Print speech All



if we din't write it take default All

@ media screen ( )
otherwise default.
@ media screen and (min-width: 760px) {
}

11) How do you use media query to apply styles only for print stylesheets?
→ Print media type in your media query
@ media print {
font-size: 12pt
color: black;
}
styles that should only print inside the curly braces

12) what is difference b/w screen & print in media query ?

→ Screen media
it is used for styles that are intended for display on a computer screen, tablet, phone or similar devices

@media screen {
body {
background color: lightBlue;
}
}

Print media :-
when document being printed

@media print {
font-size: 12pt;
color: black;
}

13) How can you hide an element on a specific screen size using media query ?

→ By using display none

14) explain the role of the orientation property in media query ?

→ to change layout of Page depending on the orientation of the browser
Portrait to landscape
to change Portrait to landscape

@media (orientation : value)
{
}

↳ landscape Portrait

15) How do you target specific devices using media queries

→ we can use combination of media features such as screen width device height and resolution

16) what is the Purpose of the not keyword in media query ?

→ it always applying allow you to apply styles if a certain condition is not true

```
@ media not only screen & (max - width 600px
        body {
            background -color : Blue ;
        }
}
```

The not keyword negates the condition. making the styles apply when the specific condition is not met.

(17) How can you media quries to adjust the font sizes for different screen sizes

→ making text more readable & appropriately sized across various devices

18] what is the box-sizing property in CSS & what does it control ?

→ box sizing control the total width & height of an element is calculated

i] Content-box :- Padding & border are added to the default element

Content box :- width & height are calculated using the content only excluding Padding and border

* border box :- Content specifies width & height of the border.

19] difference b/w Box-sizing, Content box, box sizing border-box

→ box-sizing :- default width & height calculation
height
width
border

border box :- its include the Padding & border in element
box sizing : border box ;

Content box :- include Padding & border in the elements total width & height
box-sizing : Content box (based on the content excluding

20] How does the box-sizing affect the calculation of an elements width & height in CSS ?

→ Box-sizing taking into account its content, Padding & border.

Content

box-sizing :-

* The width & height of the element are calculated based on the content alone, excluding Padding & border
* border-box :
when you set a specific width & height for an element any Pading or border it will not increase the overall dimensions of the element

Content box :- width & height are calculated based on the content alone, and Padding & border increase the overall size.

21] why might you choose box-sizing, border box as the default box model for your project

→ * Easy Sizing & Layout
* Responsive Design
* Compatibility

22] Difference b/w normalizing & resetting

→ normalizing & resetting two approach for address default styling

Resetting :-

The goal of a reset to remove or reset all default styles provided by browser so that developers start with a clean slate
A CSS set styling a various HTML element to remove default margin, Padding, font sizes.

Normalizing :-

To make a default styles more consistent across browsers by preserving default styles while resetting inconsistent ones.
default styles while normalizing inconsistencies

23] what is css Combinator and how it used in selector?

→ descendant combinator (' ')
selector with specific element
div p {
}

* child Combinator
Select all the direct children of a specfied element
div > p {
}

* Adjacent sibling Combinator ('+')
selector is element that is immediately proceded by specified element
h2 + p {
}

* universal selector

24] Difference b/w descendant & child Combinator in css selector

→ Descendant Combinator (' ')
* it is a white space character
* it takes all the elements that are descendants (children, grandchildren, & so on)

Ex:→ div p {
}

Child Combinator ('>')
* children of the specified element, excluding nested descendants that are not direct Children

div p {

}

25] adjacent sibling (combinator(+) in CSS.

→ the adjacent sibling combinator is used to select an element that is immediately preceded by a specified element

element 1 + element 2 {

}

use case :-

* if we want to specify Paragraph text follow by heading we use use case

26] How does the general sibling combinator (~) differ from the adjacent sibling combinator (+)

→ Adjacent sibling Combinator ('+)'

Select the element that is immediately preceded by a specified element. both element have same parent

h2+p {

}

General sibling selector

it selects all sibling elements that come after the specified element elements have same parent

h2~p {

}

27] Child combinator in CSS?

→ it specified the children of a specified Parent element it create a more specific relationship b/w the parent & child elements

28] How can we select all paragraphs that are direct descendants of a div using a CSS Selector?

→ we are child combinator (">")

29] How descendant combinator to style nested elements.

→ The selector P element

```
<div class =" contains">
  <P> This is a Paragraph inside the
        contains , </P>
  <span> This is a span inside the contains</p
</div>
  <P> This is a nested Paragraph inside
    the contains </P>.
  <Span> This is a nested inside the contain
</div>
```

30) explain How to space b/w two selectors represents a descedant Combinator

→ we use selector1, Sector2 {

}

31) How would you select an element that is the immediate next sibling of another element in CSS?

→ we use adjacent sibling Combinator ('+)'

element 1 + element 2 {

}

32) In what

→ Descendant combinator (' ')

* when we want to select elements that are descendants (children, grandchildren) we use.
* Consideration :- caution about potential unintended matches if the structure is complex.

→ Child Combinator (>)

* select only direct children of the specified element.

→ Adjacent Sibling (' + ')

* immediately proceded by another specified element.
* Styling elements based on their immediate relationship

→ General sibling combinator (' ~ ')

* select all sibling that share same Parent as specified element.

33] Pseudo selector :-

→ these are selected based on state, Position. with double colon (::) or a single colon (:) allows us to style a specific parts of the element.

```
button : hover {
    background Color : #3495b
    Color : #fff ;
}

active : selects and styles an element when being activated
button : active {
    transform : Scale (0,9) ;
}
```

: empty : selects that have no children.

```
p : empty {
        display : none ;
    }
```

3B) Pseudo class & Pseudo element example

→ Pseudo classes are used to style the elements like hover, empty

→ Pseudo elements :- style

they are used to specify part of an element

```
• quote :: before {
        content : open-quote ;
        color : Blak
    }

p :: first-line {
        font-weight : bold ;
        font-size : 120% ;
    }
```

3C)

→ :nth child based on index or orginal position they will start styling it allows. specific elements in a list or containes using formulic expression.

```
selector : nth-child (an+b) {

}
```

Javascript

1] what are primitive data types in Javascript?

→ undefined :- variable declared but not assigned
Null :- Represents absence of any object value
Boolean :- it have been true & false
Number :- number in terms of numeric value
String :- sequence of a characters represented
Symbol :- Symbols are unique & immutable

2] difference b/w null & undefined ?

null	undefined
* null value represents absence of any object value * no value or does not point to any object in memory let car = null; console.log (car);	* variable is declared but not assigned. * when we access an object Property does not exist result is undefined let x; console.log (x);

3] How do you check the data type of a variable in Javascript

→ Type of operator we use

let var = 42;
console.log (type of myvariable);

4) Explain the concept of truthy & falsy values in Javascript:

→

```
if (true) {
    console.log ("true is truthy");
}
if (42) {
    console.log ("42 is truthy");
}
if ("Hello") {
    console.log ("No-empty string is truthy");
}
if ({key: "value"}) {
    console.log ("object is truthy");
}

if (false) {
    console.log ("false is falsy")
}
if (0) {
    console.log ("0 is falsy");
}
if (NaN) {
    console.log ("nan is falsy");
}
if (null) {
    console.log ("null is falsy");
}
if (undefined) {
    console.log ("undefined is falsy");
}
```

5) difference b/w == & === & how it relates

== it checks the value

5 == "5" ; True

=== it checks the both values & datatypes

5 === "5" false

6) How do you convert string to number in J

→ ParseInt :→

function converts string to an integer

ParseFloat :- converts string to floating-point n

```
let strNumber = "50";
let intNumber = ParseInt(strNumber);
let floatNumber = parseFloat (strNumber)
Console.log (intNumber); o/p → 50
```

using + operator

```
let strNumber = "12";
let convertNumber = + strNumber;
Console.log (convertNumber);
```

7) difference between ++x & x++ increment operato in Javascript :

→ ++x :→ increase the value of a variable by 1

```
let x = 5;
let result = ++x;
Console.log (x);
```

x++ Post increment
→ increase the value of x after current value
let x=15;
let result = x++;
console.log(x)

6] How does javascript handle NaN values, & how can you check the value is NaN?
→ operator with no meaning full numeric result
Ex:-
let num = "Hi" / 2
→ checking for 'NaN'
we can use isNaN() function to check value of 'NaN'

Ex:- isNaN(NaN);
→ Number.isNaN()
this method returns true only if the provided value is NaN otherwise it returns false
Ex:- Number.isNaN(NaN)

7] Explain the concept of type coercion in Javascript provide example
→ variables can hold values of any type, & type coercion allows the language to be flexible in working with different data.

implicit type
→ automatically when values of different types are used together in an operator
let value = 3
let str = "4";
let result = value + str;
Console.log(result)

explicit type
→ initially value converts from one type to another type using operators.
Ex:- let value = 52
let res = Num(value)
Console.log(res);

10)

→ undefined datatype has one value which named undefined the value is not been assigned.
default value:-
let x;
console.log(x)

11) How do you create template literals in Js?
→ They are enclosed with(` `) backticks allowing to embded variable expressions within the string

12) what is IIFE?
→ IIFE → immediatly invoked function expression this pattern is often used to create a private scope of variables & functions

(function() {
})();

Ex:- (function() {
let x = 10;
console.log(x);
})();

using Parameters in IIFE

```
(function(message) {
    console.log(message);
})("Hi", Name")
```

12] what is mean by Default Parameter Passing
→ To specify default values for function parameter By assigning the default values.

```
function name(name, Condition = "Index") {
    console.log('name');
}
name("Josh")
```

Passing default values.

```
function value(num = 5) {
    console.log(value)
}
value();
```

13] whats the default return value
→ the function does not have a return statement or has an empty return statement it will implicity return undefined

16) How To Pass unlimited Parameters in Js ?
→ By using arguments

```
function add(...value) {
    return value.reduce((total, num) => total + num
};
console.log(add(1,2,3));
```