① What Is The purpose of CSS Media quires:

→ CSS Media Queries Is a powerfull tool used to affly different style for different devices. They Enable responsive web design by adjusting the layout & styling based on the characterstics of the user device.

2. write a Basic Media query In CSS.

→ @media Screen and (max-width: 600px) {
" Style for Screen with a Maximum width of 600px"

3. Explain the difference between Max-width & Min-width In media queries.

* Min-width Sets a minimum width for the Specified value Style to affly. This width will be affied when viewport width Is Equal to or larger than the Specified value.

Scanned with CamScanner

* Max-width: Sets the maximum width for the specified style to apply. Max-width will be applied when the width of the viewport is equal to or lesser than the specified value.

4. Purpose of the viewport meta tag in responsive web design.

-> Viewport metatag in HTML is used to control how a webpage is displayed on different devices & screen sizes. It enable developers to set the initial viewport width, scale and other properties.

5] How can you apply styles for landscape & potrait orientations using media queries?

-> We can apply style for landscape & potrait orientations by using media queries and orientation

* for landscape:

```
@media (orientation: landscape) {
    #style { };
```

* for potrait:

```
@media (orientation: potrait) {
    #style { };
```

Scanned with CamScanner

6. Explain the concept of a Mobile-first approach in responsive design.

→ It involves designing a web-page for mobile devices first and then progressively enhancing the design for larger screens. The approach ensures better user experience on the smaller screens.

7] what are the common breakpoints in responsive design.

→ common breakpoints in responsive design are used to adapt the layout and styles of a webpage to different screen sizes

different breakpoints:

1] Mobile (potrait):- below 480 px.
2] Mobile (landscapes): around 480 px to 767 px.
3] tablets (P+L) :- between 768px to 1023 px.
4] Small desktops and laptops :- Between 1024px & 1279 px widths.
5] Medium to large desktop: Above 1280 px.

Scanned with CamScanner

8] What is the purpose of ~~mode~~ Rem unit in media queries

→ Rem stands for unit is relative to the root element's font size. It helps consistent & scalable design across different screen sizes.

9]. How can you combine multiple media queries in CSS?

→ We can combine multiple media queries by using logical operators 'and", "or", 'not'

Eg:

@media screen and (min-width: 768px) and (max-width: 1024px) { #Style }.

10] What is the significance of 'all' keyword in media queries.

→ It is used to target all media type for example @media all {...} applies style to all devices.

Scanned with CamScanner

11] How do you use media queries to apply styles only for print stylesheets?

→ To apply styles specifically for print stylesheet in CSS, we can use media query with print keyword.

```
@media print {
    /* print style */
}
```

12] What is the difference between screen & print in media queries?

→ Screen media type is a default media type and is used to style the actual website for different screen sizes

```
@media screen and (min-width: 600px) {
```

→ print media type is used when the user want to print something the website. Then this particular styling will be applied.

```
@media print {
```

Scanned with CamScanner

13] How can you hide an Element on a Specific Screen using media queries?

→ We can use media query to hide an Elements at a Specific Screen by using display property and then set it to none. within the media query. So when the width of the particular viewport is changed it will be hidden.

```
@media Screen (max-width: 1024px) {
    .body {
        display: none;
    }
}
```

14] Explain the role of orientation property in media-queries.

→ The orientation property in a media query is used to target specific device orientation such as landscape or portrait. This allows users to apply styling based on whether the device is in vertical or horizontal orientation.

Portrait: vertical orientation
landscape: horizontal orientation

Scanned with CamScanner

15. How do you target Specific devices using media queries.

→ We can style for specific devices by targetting the particular devices width, height.

Eg: @media (max-width: 768px);
This will target the device with the width of 768px.

16. what is the purpose of the 'NOT' keyword in media queries?

→ The 'not' keyword in media queries is used to negled a specific condition. It allows user to apply styles when certain condition is not met.

Eg: @media (max-width: 600px) {
#Style for Smaller Screens #
@media not (max-width: 600px) {
#Style for Screens larger than 600px.

Scanned with CamScanner

17) How can you media queries to adjust font sizes for different Screen sizes?

⇒ Media queries in CSS allows user to apply Style based on certain ~~the~~ conditions. Such as characteristic of the device or the Screen Size. we can use media queries to adjust font size for different Screen sizes.

Eg:

```
body {
    font-size: 16px;
    } # default font size

@media Screen and (max-width: 600px){

    font-Size: 14px;
    } # This will be applied when the Screen
    Size drops below 600px.
```

Scanned with CamScanner

18] box-sizing property in CSS is used to control how the total widths & height are calculated. basically it determine whether to the specified height & widths property should include border, margin and padding or exclude them.

19] i] content box:

It is a default value. It calculate the width and height of the element by excluding padding, margin, and border.

box-sizing : born content-box;

ii] border-box:

→ It calculate the height & width of an element by including border and padding but excluding margin.

border-box can particularly usefull for responsive design & layout.

Scanned with CamScanner

20] How does the box-sizing property affects the css calculation of an Element's width & height in CSS?

→ Box-sizing determines whether the widths & height property of the Element should include padding, border and margin or not. So that it will look appealing as per the properties. We can use content box & border box to choose how it has to be calculated.

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

→ Choosing ~~box~~ border-box as a default box-model has several advantages. Mainly:

1. Simplified layout: The specified width, height include content box, padding, border. This makes easier to create layout, without doing any complex calculations

Scanned with CamScanner

Responsive design:- border-box is the first choice for the people who are working on responsive design. It simplifies the process. user can easily set percentage based widths without worrying about padding, border to the overall dimensions

* It is a default behaviour of bootstrap framework.

22] different between normalizing & resetting

→ Normalizing: Normalizing stylesheet aims to make default styles more consistent across different browsers. rather than Eleminating altogether. It does'nt remove all the styles but rather tries to Ensure that they are consistent.

→ Resetting: Resetting a stylesheet is designed to remove or reset all the default styles provided by the browser. The goal is to create a clean state.

Scanned with CamScanner

Eg: Normalizing:

```
html {
  line-height: 1.5;
}
article, aside, footer {
  display: block;
}
```

Eg: Resetting:

```
html, body {
  Margin: 0;
  Padding: 0;
  Border: 0;
}
```

23] What is a CSS Combinator, and How is It used In a Selector

→ In CSS, a combinator is a character that Specifies the relationship between the Selectors In a compound Selector. It determine How the Elements matching one Selector should be Related to Element matching another Selector.

Scanned with CamScanner

Eg: ul li{ → This Selects all <li> that are descendent of <ul> Element.
#style:
}

div > p { Selects all <p> Elements that are Immediate Childeren of a Element
}

24. Differentiate between descend and childeren combinators In CSS Selectors.

→ Descendant combinator (whiteSpace).

This combinator Selects all the Elements that are descendants of a Specific Element regardless of How deeply nested they are..

→ child combinator: The child combinator ('>') Selects all that are direct child of a Specified Element. It only Considers Immediate Childeren. Not Elements nested further down the hierarchy.

Scanned with CamScanner

25] Explain the purpose of adjacent sibling combinator (+) in CSS. provide a use case.

→ The adjacent sibling combinator ('+') is used to select and style an element that is immediately preceded by a specific element. It targets an element that is sibling and comes directly after another specified element. sharing the same parent.

Eg: <h2> Heading </h2>
<P> This is the content </P>

Now let's say we want to apply specific style to the paragraph that directly follows h2.

```
h2+p {
  font-style : Italic ;
}
```

In this scenario, 'h2+p' will select all the <p> element that directly follows an <h2>.

Scanned with CamScanner

2 8] Provide an Example of using descendant combinator to Style nested Elements.

→ HTML:

```
<div>
  <p> This is a paragraph Inside div </p>
  <Span> This Is a Span 1 </Span>
  <div>
    <Span> This Is a Span 2 </Span>
  </div>
</div>
```

CSS:

~~div Style~~ div Span {
    Color : green;
    font-weight : bold;
}

This will Style all the Span Elements Including the nested Elements as well.

Scanned with CamScanner

32] Explain the concept of CSS pseudo-Selectors with Example.

→ CSS pseudo-Selectors are keywords that are added to Selectors to style Elements based on their State, Position or other criteria.

Pseudo-Selectors begin with a colon (':'). and are used to target specific Elements without needing to add additional classes or attribute to the HTML Mark up.

Commonly used pseudo-Selectors:

1] :hover ⇒ It target the Element when user hover over it with the Mouse.

```
a:hover {
    background-color: #FFF;
}
```

2] :active → targets an Element when it is being activated (e.g:- clicked) by the user.

```
button:active {
    background-color: #00F;
}
```

Scanned with CamScanner

3. :focus → targets an Element That currently has keyword focus.

```
Input:focus {
    border: 2px solid green;
    }.
```

34. Different between Pseudo-classes and Pseudo-Elements In CSS. Give Example of Each.

→ Pseudo-classes :- used for Styling Elements based on their State or relationship with the user (Eg: ':hover', ':active', ':focus').

Pseudo-Elements :- used for styling Specific part of an Element rather than a Specific State or relationship. They are denoted by a double colon ('::')

Eg: html

```
<h1> Heading </h1>
<P> This is a paragraph </P>
```

CSS:

```
h1::before {
    content: "**"
    }
```

This will add two Stars before heading.

Scanned with CamScanner

35. How can you use the :nth-child Pseudo-class to select specific elements in a list? Provide Example.

→ The ":nth-child" pseudo-class in CSS allows you to select elements based on their position with the parent container. It particularly useful for styling every nth child element.

Eg: html

```
<ul>
  <li> Item 1 </li>
  <li> Item 2 </li>
  <li> Item 3 </li>
  <li> Item 4 </li>
</ul>
```

CSS

```
ul li : nth-child (odd) {
  background-color : #f0f0f0;
  color : blue;
  }
ul li : nth-child (even) {
  background-color : #e0e0e0;
  }
```

Javascript:

1] what are the primitive data types in javascript.

* String → characters defined within single quote or double quote.

"hello, word".

* Number → represents numeric value

42

* Boolean: represents a logical Entity and can have two values.

'true' or 'false'.

* undefined: represents a variable that has been declared but not been assigned a value.
* Null: represents a Intentional absence of any object value.

let x = null;

Scanned with CamScanner

2] Explain the difference between null and undefined in javascript.

→ undefined is a datatype in javascript where the variable has been declared but not yet defined. It is like a unintentional absence of any object value.

```
let x;
```

null → represents the intentional absence of any object value.

```
let y = null;
```

3. How do you check the datatype of a variable in javascript?

→ we can check a datatype of any variable in javascript using the typeof operator.

Eg:
```
let x = 42;
let y = "hello";
let z = true;
let w;
```

Scanned with CamScanner

```
Console.log(type of x);   → number
Console.log(type of y);   → String
Console.log(type of z);   → boolean
Console.log(type of w);   → undefined
```

4] Explain the conseft of truthy and falsy values In javaScript. provide Example.

⇒ In javascript, values are Inherently truthy or false In boolean contexts.
The value Is Considered to be truthy when If It ~~to~~ coerces to 'true' when Evaluated In a boolean context.

```
If (true) {
  console.log("This Statement");
```

This will be Executed Since the condition Is true.

* number other than '0' & 'NaN'
* The boolean 'true'.

Scanned with CamScanner

falsy values.

A value is considered falsy if it coerces to 'false' when evaluated in a boolean context.

* The boolean 'false'
* The number '0'.
* The special value 'NaN'.
* 'null'
* 'undefined.

Eg:
```
If (false) {
  console.log("This will NOT be Executed");
}
If (0) {
  console.log("This will not be Executed");
}
```

5] What is the difference between == and === operators in javascript, and how do they relate to data types?

Scanned with CamScanner

6. How do you convert a string to a number in javascript?

→ In javascript, you can convert a string to a number using various Methods.

1. parseInt or parseFloat:

```
let x = "456";
let y = parseInt(x,10);
console.log(y) → 456.

let x = '3.14'
let y = parseFloat('3.14');
console.log(y) → 3.14 Number.
```

2. Number constructor.

```
let x = "123"
let y = Number(x).
console.log(y) → 123
```

Scanned with CamScanner

-> "==" refers to Loose Equality

The '==' operator performs type coercion if the operands are of different types before making the comparision

'===' refers to Strict Equality

The '===' operator on the other hand does not perform type coercion. It checks both the values and their type and the comparision is only true when both are same.

Eg: console.log('5'==5) - true
console.log(false == 0) - true
console.log(null = undefined) - true

Eg: console.log('5'===5) - false diff type
console.log(false===0) - false diff type

Scanned with CamScanner

7] Explain the difference between the ++x & x++ Incremental Operators In javascript.

⇒ In Java script both ++x & x++ are Incremental Operators used to Increase the value by 1. However, the differs In their behaviour regarding when the Increment actually occurs.

Prefix Increment Operator ++x :

→ This will first Increments the value of 'x' and then update the value.

```
let x = 5;
let y = ++x;
console.log(x); 6 (x is Incremented before the assignment).
console.log(y); 6 (y is assigned the Incremented value of x)
```

Postfix Increment Operator x++;

→ Returns the current value of 'x' and then Increments the value of 'x'.

→ It is also known as Post-Incremental Operator.

```
let a = 10;
let b = a++;
console.log(a); // a is Incremented after the assign
console.log(b). 10. (b is assign a value of a before the Increment)
```

Scanned with CamScanner