HTML & CSS Roadmap

1. Basics of HTML

Skills to Learn:

- What HTML is and its role in web development.
- Structure of an HTML document (`<!DOCTYPE html>`, `<html>`, `<head>`, `<body>`).
- Basic tags:
  - Text content (`<h1>`, `<p>`, `<em>`, `<strong>`, etc.)
  - Links (`<a>`)
  - Lists (`<ul>`, `<ol>`, `<li>`)
  - Images (`<img>`)
  - Forms (`<form>`, `<input>`, `<textarea>`, `<select>`, etc.)
  - Tables (`<table>`, `<tr>`, `<td>`)
- Elements vs. attributes
- Nesting and structure (e.g., `<div>`, `<span>`, etc.)

Project Practice:

- Build a basic **personal webpage** with headings, paragraphs, links, and images.
- Create a **simple portfolio page** with multiple sections (About Me, Projects, Contact).

---

2. Basics of CSS

Skills to Learn:

- Introduction to CSS (styling HTML).
- Inline vs. internal vs. external CSS.
- CSS syntax: selectors, properties, and values.
- Colors, fonts, and text styles.
- The Box Model: content, padding, border, margin.
- Positioning elements: `static`, `relative`, `absolute`, `fixed`.
- Display: `block`, `inline`, `inline-block`.
- Flexbox basics (alignment, distribution).
- CSS Borders and Shadows.

Project Practice:

- Style your personal webpage to make it visually appealing (colors, fonts, layout).
- Create a **simple landing page** for a fictional product with styling for text, buttons, and links.

---

3. Intermediate HTML & CSS

Skills to Learn:

- Advanced selectors: `:hover`, `:focus`, `:nth-child()`, `:not()`.
- Styling forms and inputs (e.g., custom buttons, validation styles).
- CSS Grid (for advanced layout techniques).

AlgoBoost

- Responsive design (media queries).
- Web typography: Google Fonts, font sizes, and line heights.
- Pseudo-elements (`::before`, `::after`) and pseudo-classes.
- CSS Transitions and Animations.

Project Practice:

- Create a **responsive portfolio** page using CSS Grid and Flexbox.
- Build a **multi-page website** with navigation, form validation, and a mobile-friendly layout.
- Develop a **product page** with grid-based layout and hover effects for buttons and images.

---

4. Advanced HTML & CSS

Skills to Learn:

- Custom properties (CSS variables).
- Advanced responsive design techniques (mobile-first design, breakpoints).
- Custom Fonts (Web fonts, embedding custom fonts).
- Advanced positioning with Flexbox and CSS Grid (using both together).
- CSS frameworks (Bootstrap, Tailwind CSS, etc.).
- CSS Preprocessors (Sass, Less).
- Advanced animations (keyframes, complex transitions).

Project Practice:

- Build a **complex landing page** using a CSS framework (e.g., Bootstrap or Tailwind).
- Recreate a **professional company website** with complex layouts and animations.
- Build a **dashboard design** (admin panel) with various widgets, using Flexbox and Grid.

---

5. Best Practices & Optimization

Skills to Learn:

- Semantic HTML (using proper tags for accessibility).
- Accessibility (ARIA roles, keyboard navigation).
- SEO Basics for HTML and CSS.
- Optimizing performance (minification, image optimization).
- Writing clean, maintainable, and scalable CSS (BEM, OOCSS).
- Understanding browser compatibility and CSS prefixes.
- Version control basics with Git/GitHub.

Project Practice:

- Refactor an existing project to be more **accessible** (use semantic tags and ARIA attributes).
- Build a **performance-optimized website** by minimizing CSS, using SVGs, and compressing images.
- Work on a **collaborative project** on GitHub (use Git to manage code, track changes, and collaborate with others).

AlgoBoost

6. Real-World Projects (Portfolio Building)

As you approach the advanced stages of HTML and CSS, you should begin working on real-world projects that showcase your skills. Consider building:

- **Personal Blog**: A blog with a content management system (e.g., WordPress, static site generator).
- **E-commerce Website**: A product listing, product details, and shopping cart page.
- **News Website**: A dynamic, article-based website with responsive design and complex layouts.
- **Clone a Popular Website**: Try replicating websites like Apple, Netflix, or a news site. This helps you learn responsive design, advanced CSS, and pixel-perfect layouts.

Additional Resources for Practice

- **CodePen**: Share your work and get feedback from other developers.
- **Frontend Mentor**: Offers real-world projects with designs that you can practice.
- **CSS-Tricks**: An excellent resource for learning all things CSS.
- **MDN Web Docs**: The best place to learn and reference HTML/CSS.

By following this roadmap and constantly practicing, you'll develop a solid foundation in HTML and CSS and be well-prepared for building websites that are both functional and visually appealing.

AlgoBoost

HTML Technical Interview Questions

1. What is HTML?
2. What are the basic building blocks of HTML?
3. What is the DOCTYPE declaration in HTML?
4. What is the difference between HTML elements, tags, and attributes?
5. What are some common HTML tags?
6. What is the purpose of the <head> tag in HTML?
7. What is the purpose of the <body> tag in HTML?
8. What is the difference between block-level elements and inline elements?
9. What is the difference between HTML and XHTML?
10. What is the purpose of the tag in HTML?
11. What is the purpose of the tag in HTML?
12. What is the purpose of the<meta> tag?
13. What is semantic HTML?
14. What are HTML entities?
15. What is the purpose of the alt attribute in <img> tags?
16. What is the purpose of the <img> tag in HTML?
17. What is the purpose of the src attribute in the <img> tag?
18. What are the various formatting tags in HTML?
19. Describe HTML layout Structure?
20. Are the HTML tags and elements the same thing?
21. What is the purpose of the href attribute in the tag?
22. What is the purpose of the target attribute in the tag?
23. What is the difference between a local and a session storage in HTML5?
24. What is the purpose of the tabindex attribute?
25. What is a self-closing tag? Give an example.
26. Explain the purpose of the <iframe> tag.
27. What is the purpose of the <noscript> tag?
28. What is the purpose of the <table> tag in HTML?
29. What are the <thead>, <tbody>, and <tfoot> tags used for?
30. What is the purpose of the <tr> tagin HTML?
31. What is the purpose of the <th> and <td> tags in HTML?
32. What is the purpose of the colspan and rowspan attributes in the <td> and <th>tags?
33. What is the purpose of the <audio> and <video> tags in HTML?
34. What is the purpose of the <canvas> tag in HTML?
35. What is the purpose of the <figure> and <figcaption> tags in HTML?
36. What are different types of lists in HTML?
37. What is the ‘class’ attribute in HTML?
38. What is the difference between the ‘id’ attribute and the ‘class’ attribute of HTML elements?
39. In how many ways can we position an HTML element? Or what are the permissible values of the position attribute?
40. What are forms and how to create forms in HTML?
41. Explain new input types provided by HTML5 for forms?
42. Can we display a web page inside a web page or Is nesting of web pages possible?

AlgoBoost

43. How is Cell Padding different from Cell Spacing?
44. How can we club two or more rows or columns into a single row or column in an HTML table?
45. Is it possible to change an inline element into a block level element?
46. Explain the difference between GET and POST methods in HTML forms.
47. What are the new features in HTML5 compared to HTML4?
48. How can you add comments in HTML?
49. What is the purpose of the HTML <article> tag?
50. How can you create a line break in HTML?
51. How do you embed video in html?
52. What is the purpose of the <time> tag in HTML?
53. What is the purpose of the <pre> tag in HTML?
54. What is the purpose of the download attribute in the <a> tag?
55. What is the purpose of the <script> tag in HTML?
56. What is the difference between inline and external JavaScript?
57. What is the purpose of the title attribute in HTML?
58. What is the purpose of the <fieldset> and < legend> tags in HTML?
59. What is the purpose of the required attribute in form elements?
60. What is the purpose of the autocomplete attribute in form elements?
61. What is the purpose of the disabled attribute in form elements?
62. What is the purpose of the readonly attribute in form elements?
63. What is the purpose of the <progress> tag in HTML?
64. What is the purpose of the <details> and <summary> tags in HTML?
65. What is the purpose of the spellcheck attribute in form elements?
66. What is the purpose of the <meter> tag in HTML?
67. how can you create an external link that opens in a new tab?
68. Explain the difference between absolute and relative URLs.
69. How to include javascript code in HTML?
70. When to use scripts in the head and when to use scripts in the body?
71. How Do You Create A Table In Html?
72. What are the New tags in Media Elements in HTML5?
73. What are the significant goals of the HTML5 specification?
74. How to handle events in HTML?
75. What are void elements in HTML?
76. What is the difference between “display: none” and “visibility: hidden”, when used as attributes to the HTML element.

AlgoBoost

CSS Technical Interview Questions

1. What do you understand about the universal * selector ?
2. What is Cascading?
3. What is the ruleset in CSS ?
4. What is the CSS Box Model & its elements ?
5. What is CSS ? Explain with its versions
6. Difference between CSS3 & CSS2 ?
7. What do you mean by RGB stream ?
8. What was the purpose of developing CSS?
9. Name some CSS Frameworks
10. Difference between a " class " & an "Id" ?
11. What are CSS Sprites ? What are the benefits
12. How can you use CSS to control image repetition
13. Define contextual selectors
14. Explain responsive web design
15. How to optimize the website for prints ?
16. What is CSS “working under the hood” ?
17. Differentiate B/W logical & physical tags
18. Explain CSS specificity? & how to calculate?
19. Define gradient in CSS ?
20. Explain CSS Positioning | In Detail
21. How can CSS be integrated into an HTML?
22. What are the advantages and disadvantages of using external style sheets?
23. Different types of selectors in CSS?
24. What are the different ways to hide elements using CSS ?
25. What does “Cascading” in CSS mean ?
26. Explain the advantages of CSS?
27. List out the components of CSS style ?
28. Explain "type selectors " in CSS
29. Explain "descendant selector " in CSS
30. Explain “attribute selectors” in CSS
31. Explain " child selectors " in CSS
32. How to use external style sheets?
33. Is the CSS case insensitive?
34. How to use grouping in CSS ?
35. What is “Float” property? & disadvantages
36. List out the media types in CSS
37. What is the use of z-index in CSS?
38. Which property is used to set the cursor pointer?
39. List out properties added in CSS3 ?
40. Difference B/W “display:none;” and “visibility:hidden” in CSS ?
41. What is shadow DOM ?
42. What are pseudo elements & classes?
43. What are Data Attributes ?

AnoBoost

44. Difference B/W display:inline, display:block & display:inline-block;
45. What are different breakpoints for different devices ?
46. What is difference B/W em & rem
47. What is difference B/W em & rem
48. How to make Your CSS cross browser compatible ?
49. Difference b/w mobile first & desktop first
50. Difference b/w bold & strong ?
51. What is opacity & why is it used?
52. What is Viewport Height (vh) & Viewport Width (vw)
53. Explain about General sibling selector & adjacent sibling selectors
54. What is BEM Method
55. What is a CSS preprocessor?
56. What is Bootstrap?
57. What is a CSS reset?
58. Difference B/W padding & margin
59. How to comment in css ? single & multiple line
60. Ways to define color in CSS ?
61. Explain about “font-variant” property
62. What are the common CSS properties related to styling tables
63. Use of :not selector ?
64. What are the css combinators ?
65. Explain “object-fit” & “object-position’ CSS properties
66. Explain CSS calc( ) Function
67. What are CSS counters?
68. What is a CSS grid system?
69. Write down a selector that will match any links ending in .zip, .ZIP, .Zip etc..
70. How select every element whose href attribute value begins with “https”, .pdf & CSS:
71. Is there any reason you'd want to use translate() instead of absolute positioning, or vice-versa? And why?
72. Explain ‘Tweening’
73. What are the differences between ‘reset’ and ‘normalize’ in CSS?
74. What is the property that is used to control image scroll?
75. Explain the concept of CSS Flexbox & its main advantages in web layout design.
76. How do you vertically center an element within a flex container using Flexbox?Provide the necessary CSS properties and values.
77. Describe the purpose of the flex-grow, flex-shrink, and flex-basis properties in the flex shorthand property. Provide an example
78. In what situations would you use the align-items and align-content properties in a flex container? Provide examples
79. Explain the difference between the flex-direction values row and column, and how they influence the arrangement of flex items. Provide a use case for each value.
80. Explain the concept of CSS Grid layout and how it differs from other layout systems like Flexbox.
81. How do you define a grid container and grid items using CSS Grid? Provide an example of a basic grid layout.
82. What are the differences between implicit and explicit grids in CSS Grid? Give an example of when each type might be used.
83. What are the differences between implicit and explicit grids in CSS Grid? Give an example of when each type might be used.

AlgoBoost

84. In what scenarios would you use the properties grid-gap, grid-row, and grid-column? Provide examples
85. How do you create CSS animations? Explain the key components involved in defining animations in CSS
86. What is the difference between CSS transitions and CSS animations? Provide examples
87. Explain the concept of keyframes in CSS animations. How do you use them to create complex animations?
88. What is the significance of the animation property in CSS? Describe its components and how they influence the behavior of an animation.
89. How can you optimize CSS animations for performance? What techniques or best practices would you consider to ensure smooth animations without causing performance bottlenecks?
90. What are CSS media queries, and how do they work? Provide an example
91. Explain the concept of breakpoints in the context of media queries. How do you decide on appropriate breakpoints for different devices?
92. What is the difference between min-width and max-width in media queries?
93. How can you use media queries to target different devices, such as smartphones, tablets, and desktops?
94. What is the purpose of the meta viewport tag in responsive web design? How does it relate to media queries, and why is it important for mobile devices?
95. What are CSS transitions and how do they work? Provide an example
96. How can you transition multiple properties simultaneously using CSS? Provide an example of transitioning both opacity and transform properties.
97. Explain the concept of event-triggered transitions. How do you use pseudo-classes like :hover to create interactive transitions?
98. What are CSS variables (custom properties)? How do they differ from traditional variables in programming languages?
99. What font-related CSS properties are used to control the appearance of text?
100. What is the @font-face rule in CSS? How do you use it to include custom fonts in your web pages?

AlgoBoost

HTML

Interview Questions and Answers

1. What is HTML?

a. HTML stands for **Hypertext Markup Language**. It is the standard markup language used for creating web pages.

2. What are the basic building blocks of HTML?

a. The basic building blocks of HTML are tags, which are used to structure and define the content of a web page.

3. What is the DOCTYPE declaration in HTML?

a. The DOCTYPE declaration is used to specify the version of HTML that the web page is written in. It helps the browser render the page correctly.

4. What is the difference between HTML elements, tags, and attributes?

a. HTML elements are the individual components that make up a web page, such as headings, paragraphs, and images. Tags are used to mark the beginning and end of HTML elements. Attributes provide additional information or modify the behavior of HTML elements.

5. What are some common HTML tags?

a. Some common HTML tags include <h1> to <h6> for headings, <p> for paragraphs,

<a> for links, <img> for images, <ul> and <li> for unordered lists, and <table> for

tables.

6. What is the purpose of the <head> tag in HTML?

a. The <head> tag is used to contain meta-information about the HTML document,

such as the title, character encoding, and linked stylesheets or scripts.

7. What is the purpose of the <body> tag in HTML?

a. The <body> tag is used to define the main content of the HTML document that is

displayed in the browser.

AlgoBoost

8. What is the difference between block-level elements and inline elements?

a. Block-level elements start on a new line and take up the full width available, whileinline elements do not start on a new line and only take up the necessary width todisplay the content.

9. What is the difference between HTML and XHTML?

a. XHTML is stricter than HTML and follows XML syntax rules. XHTML documents must be well-formed, meaning all elements must be properly nested and closed.

10. What is the purpose of the tag in HTML?

a. The tag is a container used to group and style HTML elements. It is commonly used for layout and organization purposes.

11. What is the purpose of the tag in HTML?

a. The tag is an inline container used to apply styles or manipulate specific portions of text within a larger block of content.

12. What is the purpose of the<meta> tag?

a. The <meta> tag provides metadata about the HTML document, such as character set, author, description, and keywords.

13. What is semantic HTML?

a. Semantic HTML is using HTML tags that convey meaning beyond just formatting to both the browser and developer, making the structure of the webpage more understandable.

14. What are HTML entities?

a. HTML entities are special codes used to display characters that have a special meaning in HTML, such as , &, ©, etc.

AlgoBoost

15. What is the purpose of the alt attribute in <img> tags?

a. The alt attribute specifies alternative text for an image if the image cannot be displayed. It is important for accessibility and SEO.

16. What is the purpose of the <img> tag in HTML?

a. The **<img>** tag is used to display images on a web page.

17. What is the purpose of the src attribute in the <img> tag?

a. The src attribute specifies the source file or URL of the image.

18. What are the various formatting tags in HTML?

a. HTML has various formatting tags:

- **<b>** -makes text bold
- **<i>** -makes text italic
- **** -makes text italic but with added semantics importance
- **<big>** -increases the font size of the text by one unit
- **<small>** -decreases the font size of the text by one unit
- **<sub>** -makes the text a subscript
- **<sup>** -makes the text a superscript
- **<del>** -displays as strikeout text
- **** -marks the text as important
- **<mark>** -highlights the text
- **<ins>** -displays as added text

19. Describe HTML layout Structure?

a. Every web page has different components to display the intended content and a specific UI. But still, there are few things which are templated and are globally accepted way to structure the web page, such as:

- **<header>:**Stores the starting information about the webpage.
- **<footer>:**Represents the last section of the page.
- **<nav>:**The navigation menu of the HTML page.
- **<article>:**It is a set of information.

AlgoBoost

- **<section>:**It is used inside the article block to define the basic structure of a page.
- **<aside>:**Side bar content of the page

20. Are the HTML tags and elements the same thing?

a. No. HTML elements are defined by a starting tag, may contain some content and a closing tag.

For example,<h1> Heading 1</h1> is a HTML element but just <h1> is a starting tag and </h1> is a closing tag.

21. What is the purpose of the href attribute in the tag?

a. The href attribute specifies the URL of the page the link goes to. It is used to create hyperlinks.

22. What is the purpose of the target attribute in the tag?

a. The target attribute specifies where to open the linked document. For example, "_blank" opens the link in a new window or tab.

23. What is the difference between a local and a session storage in HTML5?

a. Local storage stores data with no expiration date, while session storage stores data only for the duration of the page session.

24. What is the purpose of the tabindex attribute?

a. The tabindex attribute specifies the order in which elements are focused when using the keyboard to navigate through a webpage.

25. What is a self-closing tag? Give an example.

a. A self-closing tag is an HTML tag that doesn't require a separate closing tag.

Example:

26. Explain the purpose of the <iframe> tag.

a. The <iframe> tag is used to embed another HTML document within the current document. It is commonly used to display content from another website.

27. What is the purpose of the <noscript> tag?

a. The <noscript> tag provides alternative content for users who have disabled scripts in their browser or whose browser does not support scripting.

AlgoBoost

28. What is the purpose of the <table> tag in HTML?

a. The <table> tag is used to create tabular data with rows and columns.

29. What are the <thead>, <tbody>, and <tfoot> tags used for?

a. The <thead> tag is used to group the header content in a table. The <tbody> tag is used to group the body content, and the <tfoot> tag is used to group the footer content.

30. What is the purpose of the <tr> tagin HTML?

a. The <tr> tag is used to define a row in a table.

31. What is the purpose of the <th> and <td> tags in HTML?

a. The <th> tag is used to define a header cell in a table, while the <td> tag is used to define a data cell.

32. What is the purpose of the colspan and rowspan attributes in the <td> and <th>tags?

a. The colspan attribute specifies the number of columns a cell should span, and the rowspan attribute specifies the number of rows a cell should span.

33. What is the purpose of the <audio> and <video> tags in HTML?

a. The <audio> tag is used to embed audio content on a web page, and the <video> tag is used to embed video content. They provide built-in controls for playing andpausing the media.

34. What is the purpose of the <canvas> tag in HTML?

a. The <canvas> tag is used to draw graphics, animations, and other visualizations ona web page using JavaScript.

35. What is the purpose of the <figure> and <figcaption> tags in HTML?

a. The <figure> tag is used to encapsulate self-contained content, such as images, diagrams, or videos, along with an optional caption defined using the <figcaption>tag.

36. What are different types of lists in HTML?

a. In HTML, there are mainly three types of lists:

AlgoBoost

1. **Ordered List (`<ol>`):** This type of list is used when the items need to be numbered or ordered sequentially. Each item in an ordered list is marked with a number by default, but this can be customized using CSS if needed.

Example:

```
<ol>
  <li>First item</li>
  <li>Second item</li>
  <li>Third item</li>
</ol>
```

2. **Unordered List (`<ul>`):** This type of list is used when the order of items is not important. Each item in an unordered list is marked with a bullet point by default, but this can also be customized using CSS.

Example:

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

AlgoBoost

3. **Definition List (`<dl>`)**: This type of list is used to define terms and their corresponding descriptions. It consists of a series of term-definition pairs. Each term is enclosed in a <dt> tag, and each definition is enclosed in a <dd> tag.

Example

```
<dl>
  <dt>HTML</dt>
  <dd>HyperTextMarkupLanguage</dd>
  <dt>CSS</dt>
  <dd>CascadingStyleSheets</dd>
  <dt>JavaScript</dt>
  <dd>Aprogramminglanguageusedtomakeweb
pagesinteractive</dd>
</dl>
```

These are the primary types of lists in HTML. Additionally, there are nested lists, where one type of list is contained within another, which can be achieved by placing one list inside another list element (`<ol>`, `<ul>`, or`<dl>`).

37. What is the 'class' attribute in HTML?

a. The class attribute is used to specify the class name for an HTML element. Multiple elements in HTML can have the same class value. Also, it is mainly used to associate the styles written in the stylesheet with the HTML elements.

38. What is the difference between the 'id' attribute and the 'class' attribute of HTML elements?

a. Multiple elements in HTML can have the same class value, whereas a value of id attribute of one element cannot be associated with another HTML element.

39. In how many ways can we position an HTML element? Or what are the permissible values of the position attribute?

a. There are mainly 6 values of position attribute that can be used to position an HTML element:

1. static: Default value. Here the element is positioned according to the normal flow of the document.

AlgoBoost

2. Absolute : Here the element is positioned relative to its parent element. The final position is determined by the values of left, right, top, bottom.

3. Fixed : This is similar to absolute except here the elements are positioned relative to the element

4. Relative : Here the element is positioned according to the normal flow of the document and positioned relative to its original/ normal position.

5. Initial: This resets the property to its default value.

6. Inherit : Here the element inherits or takes the property of its parent

40. What are forms and how to create forms in HTML?

a. The HTML form is used to collect the user inputs. HTML provides a <form> tag to create forms. To take input from the user we use the <input> tag inside the form so that all collected user data can be sent to the server for processing. There are different input types like ‘ button ’ , ‘checkbox’, ‘number’, ‘text’, ‘password’, ‘submit’ etc.

```
<form action="/submit_data.php">
  <label>Enter your name: </label>
  <input type="text" name="name" />
  <label>Enter Mobile number </label>
  <input type="number" name="mobile_no"/>
  <input type="submit" value="Submit">
</form>
```

AlgoBoost

41. Explain new input types provided by HTML5 for forms?

a. Following are the significant new data types offered by HTML5:

- **Date -** Only select date by using type = "date"
- **Week -** Pick a week by using type = "week"
- **Month -** Only select month by using type = "month"
- **Time -** Only select time by using type = "time".
- **Datetime -** Combination of date and time by using type = "datetime"
- **Datetime-local -** Combination of date and time by using type = "datetime-local." but ignoring the timezone
- **Color -** Accepts multiple colors using type = "color"
- **Email -** Accepts one or more email addresses using type = "email"
- Number - Accepts a numerical value with additional checks like min and max using type = "number"
- **Search -** Allows searching queries by inputting text using type = "search"
- **Tel -** Allows different phone numbers by using type = "tel"

42. Can we display a web page inside a web page or Is nesting of web pages possible?

a. Yes, we can display a web page inside another HTML web page. HTML provides a tag using which we can achieve this functionality.

```
<iframe src="url of the web page to embed" />
```

AlgoBoost

43. How is Cell Padding different from Cell Spacing?

a. Cell Spacing is the space or gap between two consecutive cells. Whereas, Cell Padding is the space or gap between the text/ content of the cell and the edge/ border of the cell. Please refer to the above figure example to find the difference.

44. How can we club two or more rows or columns into a single row or column in an HTML table?

a. HTML provides two table attributes “rowspan” and “colspan” to make a cell span to multiple rows and columns respectively.

45. Is it possible to change an inline element into a block level element?

a. Yes, it is possible using the “display” property with its value as “block”, to change the inline element into a block-level element.

46. Explain the difference between GET and POST methods in HTML forms.

a. GET method sends data through the URL, while POST method sends data in the HTTP request body. GET is used for retrieving data, while POST is used for submitting data.

47. What are the new features in HTML5 compared to HTML4?

a. HTML5 introduces new semantic elements, audio and video support, local storage, canvas for drawing graphics, and enhanced form controls.

48. How can you add comments in HTML?

a. Using <!-- Comment text here --> to add comments in HTML.

49. What is the purpose of the HTML <article> tag?

a. <article> is used to define a self-contained piece of content, such as a news article, blog post, or forum post.

50. How can you create a line break in HTML?

a. Using the
 tag,

for example: Line 1
Line 2 .

AlgoBoost

51. How do you embed video in html?

a. Using the <video> tag, for example

<video src=”example.mp4” controls> </video>

52. What is the purpose of the <time> tag in HTML?

a. The <time> tag is used to represent a specific time or date. It can be used for machine-readable dates, event schedules, or time-related content.

53. What is the purpose of the <pre> tag in HTML?

a. The <pre> tag is used to display preformatted text, preserving both spaces and line breaks as they appear in the HTML code.

54. What is the purpose of the download attribute in the <a> tag?

a. The download attribute is used to specify that a hyperlink should be downloaded instead of navigated to when clicked. It specifies the filename of the downloaded file.

55. What is the purpose of the <script> tag in HTML?

a. The <script> tag is used to embed or reference JavaScript code within an HTML document.

56. What is the difference between inline and external JavaScript?

Inline JavaScript is directly embedded within the HTML document using the <script> tag, while external JavaScript is saved in a separate .js file and linked to the HTML document using the src attribute of the <script> tag.

57. What is the purpose of the title attribute in HTML?

a. The title attribute is used to provide additional information or a tooltip text for an element. It is displayed when the user hovers over the element.

58. What is the purpose of the <fieldset> and < legend> tags in HTML?

a. The <fieldset> tag is used to group related form elements together, and the < legend> tag is used to provide a caption or description for the <fieldset> .

59. What is the purpose of the required attribute in form elements?

a. The required attribute is used to specify that a form input field must be filled out before submitting the form.

AlgoBoost

60. What is the purpose of the autocomplete attribute in form elements?

a. The autocomplete attribute is used to control whether a form input field should have autocomplete suggestions or not.

61. What is the purpose of the disabled attribute in form elements?

a. The disabled attribute is used to make a form input field or button non-editable or non-clickable. It prevents user interaction with the element.

62. What is the purpose of the readonly attribute in form elements?

a. The readonly attribute is used to make a form input field non-editable. It allows the user to view the value but not modify it.

63. What is the purpose of the <progress> tag in HTML?

a. The <progress> tag is used to represent the progress of a task or the completion of a process, such as a file upload or a download.

64. What is the purpose of the <details> and <summary> tags in HTML?

a. The <details> tag is used to create a collapsible section that can be toggled open or closed. The <summary> tag is used to provide a summary or heading for the collapsible section.

65. What is the purpose of the spellcheck attribute in form elements?

a. The spellcheck attribute is used to enable or disable spell checking for a form input field.

66. What is the purpose of the <meter> tag in HTML?

a. The <meter> tag is used to represent a scalar measurement within a known range, such as a progress bar, disk usage, or temperature.

67. how can you create an external link that opens in a new tab?

a. adding target="_blank" attribiute to the <a> tag,like

** link text**

68. Explain the difference between absolute and relative URLs.

a. **Absolute URLs** specify the full web address of a resource, including the protocol (http/https). **Relative URLs** specify the path to a resource relative to the current page.

AlgoBoost

69. How to include javascript code in HTML?

a. **HTML** provides a **<script> tag** using which we can run the javascript code and make our HTML page more dynamic

AlgoBoost

70. When to use scripts in the head and when to use scripts in the body?

a. If the scripts contain some event-triggered functions or jquery library then we should use them in the head section. If the script writes the content on the page or is not inside a function then it should be placed inside the body section at the bottom. **In short, follow below three points:**

1. Place library scripts or event scripts in the head section.

2. Place normal scripts that do not write anything on the page, in the head section until there is any performance issue.

3. Place scripts that render something on the web page at the bottom of the body section.

71. How Do You Create A Table In Html?

a. Using the **<table>** tag with **<tr>** for rows**, <td>** for data cells and **<th>** for header cells.

Example

A simple HTML table:

```
<table>
  <tr>
    <th>Company</th>
    <th>Contact</th>
    <th>Country</th>
  </tr>
  <tr>
    <td>Alfreds Futterkiste</td>
    <td>Maria Anders</td>
    <td>Germany</td>
  </tr>
  <tr>
    <td>Centro comercial Moctezuma</td>
    <td>Francisco Chang</td>
    <td>Mexico</td>
  </tr>
</table>
```

AlgoBoost

72. What are the New tags in Media Elements in HTML5?

a. ➢ <audio> - Used for sounds, audio streams, or music, embed audio content without any additional plug-in.

➢ <video> - Used for video streams, embed video content etc.

➢<source> - Used for multiple media resources in media elements, such as audio, video, etc.

➢<embed> - Used for an external application or embedded content.

➢ <track> - Used for subtitles in the media elements such as video or audio.

73. What are the significant goals of the HTML5 specification?

a. These were the target area of the HTML5 specs: Introduction of new element tags to better structure the web page such″ as tag. Forming a standard in cross-browser behavior and support for different″ devices and platforms Backward compatible with the older version HTML web pages″ Introduction of basic interactive elements without the dependency of″ plugins such as tag instead of the flash plugin.

74. How to handle events in HTML?

a. HTML allows event trigger actions in browsers using javascript or JQuery. There are a lot of events like ‘ onclick ’ , ‘ ondrag ’ , ‘onchange’, etc.

AlgoBoost

75. What are void elements in HTML?

a. HTML elements which do not have closing tags or do not need to be closed are Void elements. For Example:
 ,<hr/> and <img/> etc…,

76. What is the difference between "display: none" and "visibility: hidden", when used as attributes to the HTML element.

a. When we use the attribute **"visibility: hidden"** for an HTML element then that element will be hidden from the webpage but still takes up space. Whereas, if we use the "**display: none**" attribute for an HTML element then the element will be hidden, and also it won't take up any space on the webpage.

AlgoBoost

CSS

Interview Questions and Answers

1. **What do you understand about the universal * selector ?**

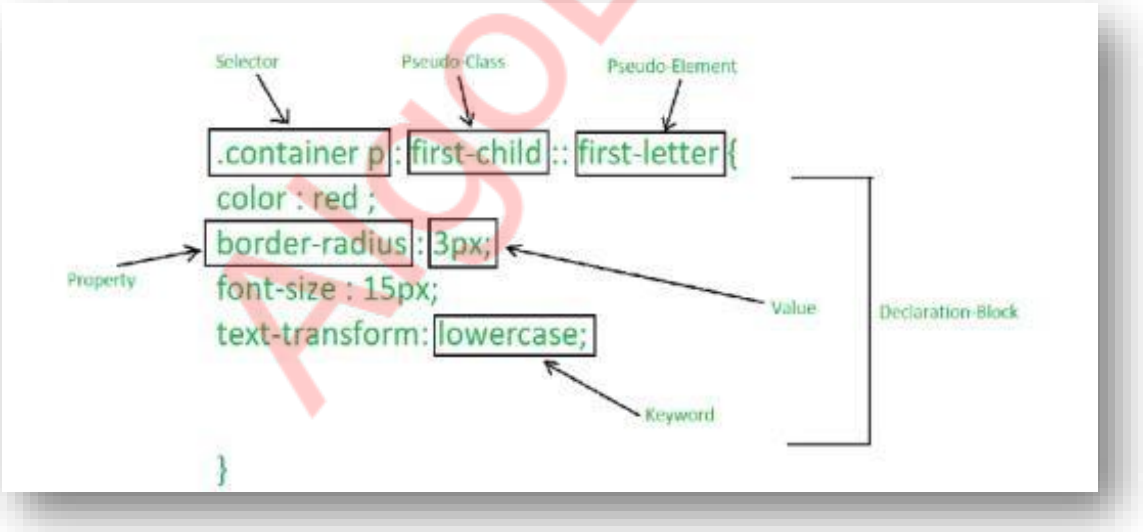
a. The **universal * selector** in CSS matches and applies styles to all elements on a webpage. It's a way to target everything at once.

2. **What is Cascading?**

a. **Cascading** in CSS refers to how styles from various sources combine and interact to determine the final appearance of elements. This process involves prioritizing styles based on specificity, inheritance, and order of declaration

3. **What is the ruleset in CSS ?**

a. A **ruleset** in CSS consists of a selector and a set of declarations. The selector determines which HTML elements the rule applies to, and the declarations specify the styling properties and values for those elements. It defines how elements should be styled on a webpage



4. **What is the CSS Box Model & its elements ?**

a. The **CSS Box Model** is a concept that defines how elements on a webpage are structured in terms of size and spacing.
It consists of four main elements:

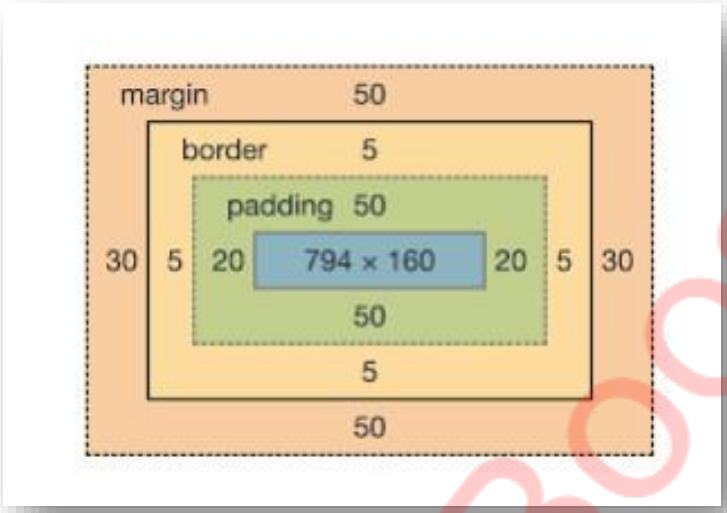
Content : The actual element's content, like text or images.

AlgoBoost

Padding : Space between the content and the border.

Border : The element's boundary, surrounding the padding and content.

Margin : Space outside the element, providing separation from other elements



5. **What is CSS ? Explain with its versions**

a. CSS (Cascading Style Sheets) is a language used to style and format HTML documents, controlling their appearance on the web. It separates the visual design from the document's structure.

There are different versions:

CSS1 : Introduced basic styling properties.

CSS2 : Added more advanced features like positioning and typography.

CSS2.1 : A revision that addressed issues in CSS2.

CSS3 : Modular update with various modules for advanced features like transitions, flexbox, grid and more

CSS4 : Not an official version but represents ongoing module-based updates for advanced styling capabilities.

6. **Difference between CSS3 & CSS2 ?**

a. **CSS3** is an upgraded version of **CSS2** that brings new and advanced styling capabilities to web design. It introduces modules for

AlgoBoost

- animations,

- transitions,

- flexible layouts (Flexbox),

- grid layouts (Grid), and more.

It offers improved typography, rounded corners, shadows, and better control over styling effects. **CSS3** allows for responsive design through media queries. Its modular approach and expanded features make it more versatile and powerful compared to **CSS2.**

7. What do you mean by RGB stream ?

a. An "**RGB stream"** refers to a continuous flow of color data in the form of Red, Green, and Blue components. Each parameter (red, green, and blue) defines the intensity of the color with a value between 0 and 255. rgb(red, green, blue) | rgb(255, 255, 255) = Generate white color rgba(red, green, blue, alpha) alpha for transparent

8. What was the purpose of developing CSS?

a. The purpose of developing **CSS (Cascading Style Sheets)** was to separate the presentation or styling of web documents from their underlying structure. Prior to the introduction of CSS, web designers had to include styling directly within HTML documents, which made it difficult to maintain and modify the design consistently across multiple pages.

9. Name some CSS Frameworks

- Bootstrap
- Foundation
- Materialize
- Bulma
- Semantic UI
- Tailwind CSS
- UIKit
- Pure.css
- Material-UI

10. Difference between a " class " & an "Id" ?

a. **Class**: Used to group multiple elements together for applying the same styling. Can be applied to multiple elements, and you target it in CSS using a dot (.)

ID: Used to uniquely identify a single element. Each ID must be unique in the document. You target it in CSS using a hash (#).

AlgoBoost

11. What are CSS Sprites ? What are the benefits

a. **CSS sprites** involve combining multiple images into one and using CSS background positioning to display specific parts of the combined image.

Fewer Requests : Reduced HTTP requests.

Faster Loading : Faster page loading times.

Better Performance : Improved user experience.

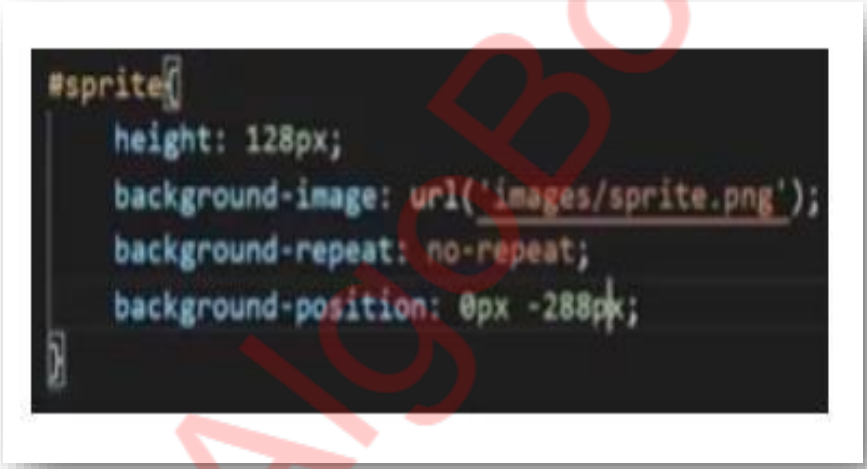
Efficient Caching : Browser caches a single image.

Simplified CSS : Manage image positions in CSS.

Reduced Latency : Minimized request delays.

Mobile Optimization : Improved mobile performance.

Responsive Friendly: Supports responsive designs.



Algo Boost

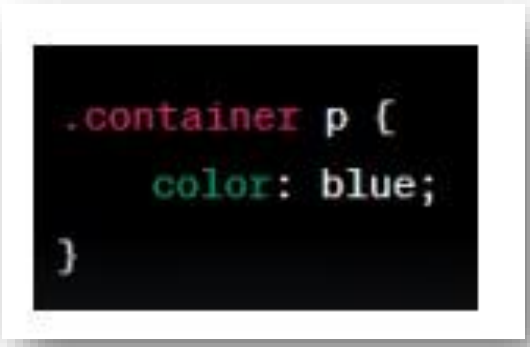
12. How can you use CSS to control image repetition

a. **CSS's** background-repeat property controls image repetition in an element's background: repeat: Default, image repeats both horizontally and vertically.

repeat-x: Image repeats only horizontally.

repeat-y: Image repeats only vertically.

no-repeat : Image is not repeated.



13. **Define contextual selectors**

a. Contextual selectors in CSS target elements based on their hierarchical relationship within the HTML structure. They are written as parentElement descendantElement and apply styles to the descendant elements that are within the specified parent element.

14. **Explain responsive web design**

a. **Responsive web design (RWD)** is an approach to web design & development that aims to create websites that automatically adapt and provide an optimal viewing experience on various devices & screen sizes, including desktops, laptops, tablets, and smartphones.

Key principles of responsive web design

- Fluid Grids..
- Flexible Images
- Media Queries
- Viewport Meta Tag
- Content Prioritization
- Breakpoints

15. **How to optimize the website for prints ?**

○ **Create Print Stylesheet:** Make a separate CSS stylesheet for printing. @media print { ... }.
○ **Remove Unnecessary Elements:** Hide irrelevant items like menus.
○ **Adjust Fonts and Colors:** Opt for legible fonts and colors.
○ **Remove Background Images:** Avoid printing background images.
○ **Control Page Breaks**: Use CSS to prevent awkward content splits.
○ **Increase Margins and Padding**: Add white space for readability.
○ **Test Printing:** Print and adjust to ensure content fits well.
○ **Offer Printer**-Friendly Format: Provide simplified content version.
○ **PDF Option**: Offer downloadable PDFs for printing.
○ **Headers and Footers:** Use CSS for page info.

AlgoBoost

○ Check Images: Ensure graphics are print-friendly

16. What is CSS “working under the hood” ?

a. CSS applies styles to **HTML** elements, controlling their appearance. Browsers load HTML and CSS, create a render tree(**DOM)** of elements, compute styles, calculate layout, paint, and render the page. This process ensures proper visual presentation of content.

● **Parse HTML:** Browser forms a DOM representing content structure.
● **Load CSS:** Encounters <link> or <style> tags, loads CSS.
● **Create Render Tree**: Combines DOM and CSS into Render Tree, excluding hidden elements.
● **Apply Styles:** Match elements to CSS rules, calculate styles.
● **Layout (Reflow):** Compute element positions and sizes.
● **Painting:** Apply computed styles, paint elements on screen.
● **Rendering**: Combine painted elements for final user view.

17. Differentiate B/W logical & physical tags

a. Logical Tags:

● Semantic, structural.

● Define content's meaning and role.

● Examples: , , .

Physical Tags:

● Presentational.

● Focus on visual appearance.

● **Examples:** <font>, <b>,<i>

18. Explain CSS specificity? & how to calculate?

a. **CSS** specificity is a rule used by web browsers to determine which style rules should be applied to an element when multiple conflicting styles are present. It's a way of resolving conflicts when multiple selectors target the same element with different styles.

CSS Specificity:

● Inline Styles (highest).

● ID Selectors (more specific than classes/elements).

AlgoBoost

- Class, Attribute, Pseudo-classes.
- Element, Pseudo-elements (least specific).

To calculate specificity:

- Inline styles: 1000 points
- ID selectors: 100 points
- Class, attribute, pseudo-class selectors: 10 points each
- Element, pseudo-element selectors: 1 point each

19. Define gradient in CSS ?

a. In CSS, a gradient is a smooth transition between two or more colours, creating a gradual blend from one color to another. Gradients can be applied as backgrounds, borders, or text colours. They offer design flexibility and depth to elements on a webpage.

Types of gradients in CSS:

- Linear Gradient
- Radial Gradient

```
background: linear-gradient(
    direction,
    color-stop1,
    color-stop2, ...);

background: radial-gradient(
    shape size at
    position, color-stop1, color-stop2, ...);
```

AlgoBoost

20. Explain CSS Positioning | In Detail

a. **CSS** provides several position properties that allow you to control the placement and positioning of elements on a webpage. Here are the different position properties and their possible values:

Position:

static (default): Elements follow the normal document flow.
relative: Positioned relative to its normal position.

absolute: Positioned relative to its closest positioned ancestor or to the containing blocks. It's removed from the normal document flow.

fixed: Positioned relative to the viewport. It remains fixed even when Scrolling.

sticky: It sticks an element to a specific position on the page based on the user's scroll, creating a "sticky" effect
top, right, bottom, left:

Used with position: relative or position: absolute to adjust element's position from its normal or initial position.

z-index:

Specifies the stacking order of positioned elements. Elements with higher values appear on top of elements with lower values.

overflow:
Specifies what happens if content overflows an element's box. Values include visible, hidden, scroll, and auto.

display:
Controls how an element is displayed. Values include block, inline, inline-block, none, and more. Float & clear is not recommended

21. How can CSS be integrated into an HTML?

a. **Integrate CSS into HTML:**

- **Inline Styles**: Use style attributes on elements.
- **Internal Stylesheet:** Include CSS within .
- **External Stylesheet**: Link external CSS with in .
- **@import Rule:** In external stylesheet, @import another CSS.

22. What are the advantages and disadvantages of using external style sheets?

a. **Advantages of External Style Sheets:**

- **Modularity:** Styles can be reused across multiple pages, ensuring consistency.
- **Centralized Control:** Changes in one place affect all linked pages.
- **Efficiency:** Browser caching reduces page loading time for subsequent visits.
- **Separation of Concerns**: Separates content from presentation, improving maintainability.
- **Faster Updates:** Alter styles without altering HTML content.

Disadvantages of External Style Sheets:

- **External Dependency**: Requires an additional HTTP request, which can marginally impact page load time.

AnoBoost

● **Rendering Delay:** External styles must load before content is styled, possibly causing a brief flash of unstyled content (FOUC).
● **Multiple Files:** More files to manage can complicate project structure.
● **Less Suitable for Small Styles:** For tiny projects, inline or internal styles might be more efficient.
● **Limited Offline Access:** May lead to unstyled pages if the external stylesheet isn't cached]

23. Different types of selectors in CSS?

● **Universal Selector (*):** Targets all elements.
● **Type Selector**: Targets elements by tag names (e.g., h1, p).
● **Class Selector (.):** Targets elements by class (e.g., .btn).
● **ID Selector (#):** Targets elements by unique ID (e.g., #header).
● **Attribute Selector ([]):** Targets elements by attributes (e.g., [type="text"]).
● **Pseudo-Class (:):** Targets special states (e.g., :hover, :nth-child(n)).
● **Pseudo-Element (:: or :):** Targets specific parts of an element (e.g., ::before, ::after).
● **Descendant Selector ( ):** Targets nested elements (e.g., ul li).
● **Child Selector (>):** Targets direct children (e.g., ul > li).
● **Adjacent Sibling (+):** Targets element after another (e.g., h2 + p).
● **General Sibling (~):** Targets elements with the same parent (e.g., h2 ~ p).

24. What are the different ways to hide elements using CSS ?

a. Here are the different ways to hide elements using CSS:
● **display: none;**: Removes element from layout.
● **visibility: hidden;:** Keeps space, but element is invisible.
● **opacity: 0;:** Element becomes transparent.
● **Positioning:** Move off-screen using position and top/right/bottom/left.
● **clip-path:** polygon(0 0, 0 0, 0 0, 0 0);: Clips element to hide.
● z-**index: -1;:** Puts element behind others to hide.

25. What does “Cascading” in CSS mean ?

a. **"Cascading"** in CSS refers to the process of combining and applying multiple styles to an element, where the styles can come from different sources like the browser default styles, user-defined styles, and external stylesheets.

26. Explain the advantages of CSS?

● Separates Style and Content
● Consistent Design
● Easy Updates
● Faster Loading
● Enhances Accessibility
● Responsive Design
● Reusability

AlgoBoost

● Efficient Maintenance
● SEO Benefits
● Print Styling

27. List out the components of CSS style ?

a. Here's a concise list of the components of a CSS style:

● **Selectors**: Choose which elements to style.
● **Properties**: Specify visual attributes.
● **Values:** Assign settings to properties.
● **Declarations:** Property-value pairs within curly braces.
● **Rules**: Combine selectors and declarations.

28. Explain "type selectors " in CSS

29. Explain "descendant selector " in CSS

30. Explain “attribute selectors” in CSS

31. Explain " child selectors " in CSS

(Above questions are covered in the question 21)

32. How to use external style sheets?

● **Create CSS File:** Make a separate .css file for your styles.
● **Write Styles:** Define styles in the CSS file.
● **Link CSS to HTML:** In the HTML <head> add<link> to your CSS file:

<link rel="stylesheet"type="text/css"href="styles.css">

● Write HTML: Add your HTML content in the <body>

Separates your styles from your HTML for better organization and reusability.

33. Is the CSS case insensitive?

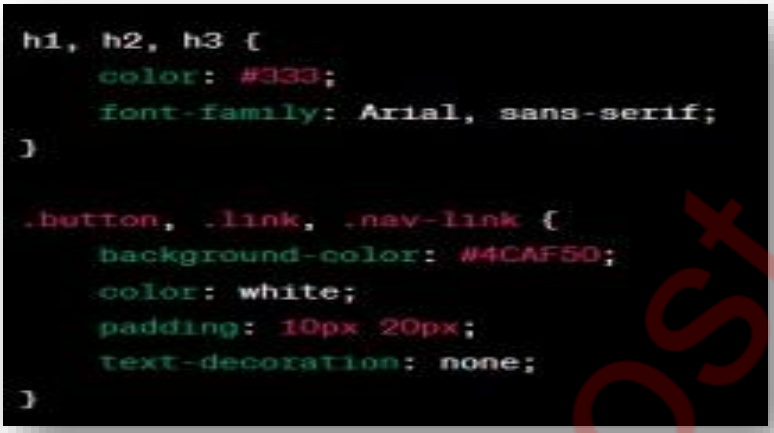
a. Yes, by default, CSS is case-insensitive.

AlgoBoost

34. How to use grouping in CSS ?

a. **How to use grouping in CSS:**

- **List Selectors:** Separate selectors with commas.
- **Define Styles:** Put shared styles in curly braces.



```
h1, h2, h3 {
    color: #333;
    font-family: Arial, sans-serif;
}

.button, .link, .nav-link {
    background-color: #4CAF50;
    color: white;
    padding: 10px 20px;
    text-decoration: none;
}
```

35. What is "Float" property? & disadvantages

a. "**Float**" is a CSS property that aligns elements horizontally. However, it can lead to layout instability, clearing problems, collapsed parent containers, complexity, challenges in responsive design, and compromised HTML structure. Modern layout techniques like Flexbox and Grid are often preferred over "float" due to these disadvantages. 36. List out the media types in CSS

36. List out the media types in CSS

a. **List of CSS media types:**

- **all:** All media types (default).
- **braille:** Braille devices.
- **embossed**: Paged braille printers.
- **handheld:** Handheld devices.
- **print:** Printed documents.
- **projection**: Projectors.
- **screen:** Screens (computers, tablets, smartphones).
- **speech**: Speech synthesizers.
- **tty:** Fixed-pitch character grid (teletypes, terminals).
- **tv:** Television-type devices.

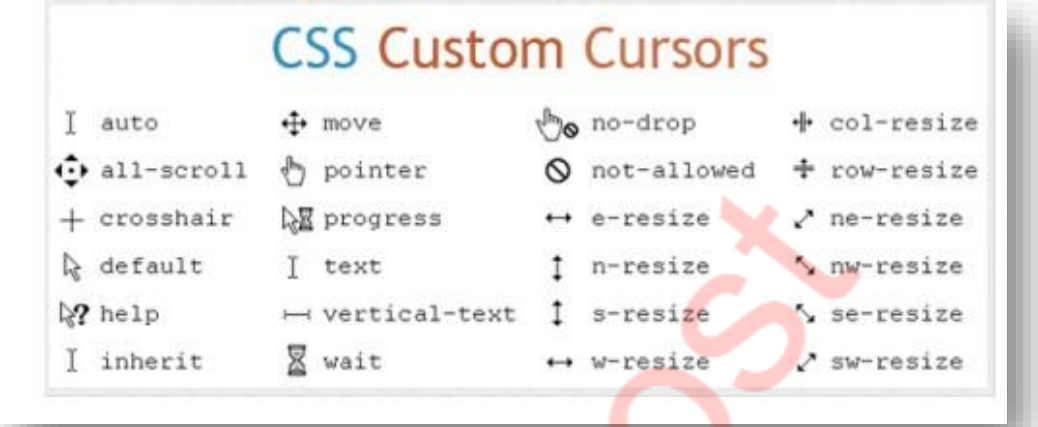
37. What is the use of z-index in CSS?

a. The z-index in CSS controls the layering of elements. A higher value places an element above others. It's useful for managing overlapping content and creating complex layouts.

AlgoBoost

38. Which property is used to set the cursor pointer?

a. The cursor property in CSS is used to set the appearance of the mouse cursor when it's hovering over an element. Different Values :



39. List out properties added in CSS3 ?

a. More specific use cases/added features in CSS3

● **Layouts:** Flexbox and Grid Layout for responsive designs
● **Effects:** Apply filters (blur, grayscale) using filter.
● **Animations**: Create animations with @keyframes and animation.
● **Transitions:** Smooth transitions with transition.
● **Backgrounds:** Design gradients with linear-gradient() & radial-gradient().
Typography: Fine-tune text with text-shadow, font-feature-settings.
● **Responsive:** Adapt styles with media queries.
● **Custom Fonts:** Embed fonts with @font-face.
● **Rounded Corners**: Rounded elements with border-radius.
● **Shadows:** Add depth using box-shadow.

40. Difference B/W "display:none;" and "visibility:hidden" in CSS ?

● **display: none; :**
  ○ Completely removes the element from the layout, including space it occupied.
● **visibility: hidden; :**
  ○ Hides the element but preserves its space in the layout.

41. What is shadow DOM ?

a. In simpler terms, shadow **DOM** lets you create self-contained components with their own private "**sandboxed**" DOM and styles, reducing conflicts and making it easier to create reusable,

AceBoost

customizable, and encapsulated elements on a web page. It's commonly used when building custom elements or widgets that need to behave independently and avoid interfering with other parts of the page Learn in depth…

42. What are pseudo elements & classes?

a. **Pseudo Element :** Pseudo elements are virtual elements in CSS that allow you to style specific parts of an element's content without adding extra HTML. They are indicated by double colons (::). Here's a list of common pseudo elements:

- **::before:** Inserts content before an element's content.
- **::after:** Inserts content after an element's content.
- **::first-line:** Styles the first line of text within an element.
- **::first-letter**: Styles the first letter of text within an element.
- **::selection:** Styles the selected text in the browser.
- **::placeholder:** Styles the placeholder text in input elements.
- **::marker:** Styles the marker of a list item.
- **::backdrop**: Styles the backdrop behind a modal or dialog.
- **::cue:** Styles cues in audio and video content.

Pseudo Classes : Pseudo classes are special keywords in CSS that target specific states or behaviors of elements that can't be selected with regular selectors. They are indicated by a single colon (:). Here's a list of common pseudo classes:

- **:hover:** Applies when an element is hovered over.
- **:active:** Applies when an element is clicked or activated.
- **:focus:** Applies when an element gains focus (e.g., through tab navigation).
- **:visited**: Applies to visited links.
- **:nth-child()**: Applies to elements based on their position within a parent.
- **:first-child:** Applies to the first child of a parent element.
- **:last-child:** Applies to the last child of a parent element.
- **:not():** Applies when an element does not match a certain selector.

AlgoBoost

● **:enabled:** Applies to form elements that are enabled.

● **:disabled:** Applies to form elements that are disabled.

● **:checked:** To checked input elements (e.g., checkboxes, radio buttons).

● :**empty:** Applies to elements with no children or text content.

● **:target:** When an anchor link's target matches the current URL fragment.

43. What are Data Attributes ?

a. Data attributes, in short, are custom attributes that can be added to HTML elements to store additional information. They start with the prefix "data-" followed by a descriptive name, allowing JavaScript and CSS to access and manipulate this information without affecting the core functionality

John Doe

In this case, the data-user-id attribute stores the value "123", which could be used by JavaScript to perform actions related to that user.

44. Difference B/W display:inline, display:block & display:inline-block;

● **display: inline:** Flows within content don't start a new line, no width/height control.

● **display: block:** Starts a new line, takes full width.

● **display: inline-block:** Inline flow, accepts width/height, respects spacing.

45. What are different breakpoints for different devices ?

● **Mobile Phones:** Typically around 320px to 480px width.

● **Tablets:** Generally from 481px to 1024px width.

● **Laptops/Desktops**: Often 1025px and above.

● **Large Screens:** Above 1440px width (e.g., 4K monitors). 46. Difference B/W ":last-child" and "list-of-type" Selector

● **:last-child Selector**: Targets the last child element within its parent.

AlgoBoost

● **:nth-of-type Selector:** Targets elements of a specific type based on their position within their parent.

```css
/* Targeting the last child */
li:last-child {
  color: red;
}

/* Targeting the second item of type li */
li:nth-of-type(2) {
  font-weight: bold;
}
```

47. What is difference B/W em & rem

● **em:** Relative to the font size of the nearest parent element with a defined font size.

● **rem:** Relative to the root (html) element's font size, providing a consistent base value.

48. How to make Your CSS cross browser compatible ?

● **Follow Standards:** Adhere to W3C standards for consistency.

● **Test on Many Browsers:** Test across various browsers.

● **Use Prefixes:** Add prefixes for experimental features.

● **Reset or Normalize**: Include a reset or normalize stylesheet.

● **Use Media Queries**: Adapt with responsive media queries.

● **Fallbacks**: Offer fallbacks for unsupported features.

● **Be Flexible**: Use relative units for flexibility.

● **Browser-Specific Styles**: Employ conditional styles if needed.

● **Avoid Hacks**: Minimize browser-specific hacks.

● **Stay Updated:** Keep up with evolving standards.

AlgoBoost

49. Difference b/w mobile first & desktop first

● **Mobile First:** Design approach starts with mobile screens and progressively enhances for larger screens. Initially minimal design, then adds complexity.

● **Desktop First**: Design starts with desktop screens and adapts downward for smaller screens. Initially complex, then simplifies for smaller screens.

50. Difference b/w bold & strong ?

●**** : Carries strong semantic meaning, often displayed as bold.

● **<b>:** Purely visual, makes text bold without emphasizing importance.

51. What is opacity & why is it used?

a. Opacity makes elements transparent. It's used for fading effects, overlays, and blending with backgrounds in CSS.

52. What is Viewport Height (vh) & Viewport Width (vw)

● **Viewport Height (vh):** Represents a percentage of the height of the device's viewport. 1vh is 1% of the viewport height. Useful for responsive vertical sizing.

● **Viewport Width (vw):** Represents a percentage of the width of the device's viewport. 1vw is 1% of the viewport width. Useful for responsive horizontal sizing.

53. Explain about General sibling selector & adjacent sibling selectors

a. General Sibling Selector (~): It selects all elements that are siblings of a specified element. It targets elements that share the same parent and appear after the specified element, regardless of their hierarchy level.

Adjacent Sibling Selector (+): It selects an element that is immediately preceded by a specified element. It targets the first element that directly follows the specified element, sharing the same parent.

AlgoBoost

Example :

```html
<!-- HTML -->

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

```css
/* CSS */

/* General Sibling Selector */
li ~ li {
  color: blue;
}

/* Adjacent Sibling Selector */
li + li {
  font-weight: bold;
}
```

AlgoBoost

The li ~ li selector targets all li elements that are siblings of another li (excluding the first one). It turns their text color blue.

The li + li selector targets the second li element, which directly follows another li element. It makes the second li element bold.

54. What is BEM Method

a. BEM (Block Element Modifier) is a method for organizing CSS code. It breaks down components into three parts:

● **Block:** The main component or module.(.button)

● **Element:** Parts of the block.(.button_ _text)

● **Modifier:** Variations or states of the block or element.(.button--large)

55. What is a CSS preprocessor?

a. A CSS preprocessor is a tool that extends the capabilities of standard CSS by introducing variables, functions, nesting, and more. It allows developers to write cleaner, more organized stylesheets and then compiles them into standard CSS that browsers can understand.

Examples :

● Sass/Scss

● Less

● Stylus

● PostCSS

56. What is Bootstrap?

a. **CSS Bootstrap** is a front-end framework with ready-made CSS styles, components, and layouts. It speeds up web development by offering a consistent, responsive design system for creating modern websites and apps without starting from scratch.

57. What is a CSS reset?

a. A CSS reset is a set of CSS rules that aim to neutralize or reset default browser styles applied to HTML elements. It's used to establish a consistent baseline for styling across different browsers, ensuring that your styles start from a clean slate without any unexpected variations caused by default browser styles.

58. Difference B/W padding & margin

● Padding is the space inside an element, affecting its content area.

● Margin is the space outside an element, influencing the spacing between elements in a layout.

59. How to comment in css ? single & multiple line

● **Single-line Comment:** /* Your comment */ anywhere in CSS.

● **Multi-line Comment**: /* Start and end */ for spanning lines.

60. Ways to define color in CSS ?

● **Keywords:** Names like red, blue.

● **Hex Codes:** #RRGGBB format, e.g., #FFA500 for orange.

● **RGB:** rgb(R, G, B) format, each value 0-255.

● **RGBA**: rgba(R, G, B, A) format, with alpha (transparency) 0-1.

● **HSL:** hsl(H, S%, L%) format, Hue-Saturation-Lightness.

● **HSLA:** hsla(H, S%, L%, A) with alpha for transparency.

AlgoBoost

61. Explain about “font-variant” property

a. The font-variant CSS property modifies how text looks by altering the font style. It's used to make specific typographic adjustments within text elements.

For instance:

font-variant: normal; (default) displays text with standard font.

font-variant: small-caps; shows uppercase as smaller capitals.

It's useful for highlighting text, like titles, using small caps. Keep in mind, not all fonts support this style, and results vary based on the chosen font.

62. What are the common CSS properties related to styling tables

● **border-collapse**: Defines whether table borders should be collapsed into a single border or separated.

  ○ **border-collapse**: separate; (default) - Borders are separate.

  ○ **border-collapse:** collapse; - Borders are collapsed into a single border.

● **border-spacing:** Specifies the space between adjacent cell borders when border-collapse is set to separate.

  ○ **border-spacing:** 5px 10px; (horizontal and vertical spacing).

● **table-layout:** Determines the algorithm used to layout table cells, affecting the distribution of column widths. ○ table-layout: auto; (default) - Cells size based on content.

  ○ **table-layout: fixed;** - Column widths are set by the first row or CSS.

● **caption-side:** Specifies where the table caption (if present) should be positioned.

  ○ **caption-side:** top; (default) - Caption above the table.

  ○ **caption-side:** bottom; - Caption below the table.

● **empty-cells:** Controls whether or not to display borders and background on empty table cells.

  ○ **empty-cells:** show; (default) - Show borders and background.

  ○ **empty-cells:** hide; - Hide borders and background.

● **border:** Sets the border properties for table elements.

  ○ **border:** 1px solid black;

AlgoBoost

63. Use of :not selector ?

a. The :not selector in CSS selects elements that don't match a specific criterion. It's used to exclude certain elements from a styling rule. For instance,

```css
/* Select all paragraphs except those with a class of "special" */
p:not(.special) {
    color: blue;
}
```

64. What are the css combinators ?

a. CSS combinators are symbols used to define relationships between HTML elements when applying styles:

● **Descendant (space):** Selects elements inside another.

● **Child (>):** Selects direct children of an element.

● **Adjacent sibling (+):** Selects elements immediately after another.

● **General sibling (~):** Selects elements after a specified element with the same parent.

65. Explain "object-fit" & "object-position' CSS properties

● **object-fit:** Specifies how an image/video should fit within its container while preserving its aspect ratio.

○ **object-fit:** fill; - Stretches to fill container.

○ **object-fit:** contain; - Fits inside container while maintaining aspect ratio.

○ **object-fit:** cover; - Covers container while maintaining aspect ratio.

○ **object-fit:** none; - Keeps original size, possibly overflowing.

○ **object-fit:** scale-down; - Resizes if needed, without exceeding original dimensions.

● **object-position**: Defines the position of an object (image/video) within its container.

Example: object-position: center top; places object at the top center.

AlgoBoost

These properties provide control over how media content is displayed and positioned within its container.

66. Explain CSS calc( ) Function

a. The calc() function in CSS allows you to perform mathematical calculations within property values. It's particularly useful for dynamic sizing and positioning in responsive designs.

For example:

● **width:** calc(100% - 20px); subtracts 20 pixels from 100% width.

● **font-size:** calc(14px + 2vmin); calculates font size based on viewport size. calc() helps create adaptable and precise styles without resorting to external calculations.

67. What are CSS counters?

a. CSS counters enable automatic numbering or labeling of HTML elements without manual numbering:

● **Creation:** Use counter-reset to initialize a counter.

● **Incrementing:** Employ counter-increment to increase the counter when specific elements appear.

● **Displaying:** Use content with ::before or ::after to show the counter value

● **Usage:** Apply counters to various HTML elements (headings, lists) for automatic numbering or labeling.

```
ol {
  counter-reset: item;
  list-style-type: none;
}

li::before {
  content: counter(item) ". ";
  counter-increment: item;
}
```

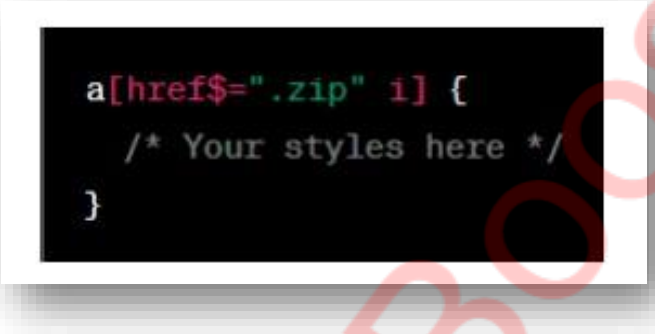
AlgoBoost

68. What is a CSS grid system?

a. A **CSS grid system** is a layout technique that uses a grid of rows and columns to arrange and align elements on a web page. It simplifies design by providing a structured framework for creating responsive layouts with consistent spacing and positioning.

69. Write down a selector that will match any links ending in .zip, .ZIP, .Zip etc..

a. To match links that end in .zip, .ZIP, .Zip, etc., you can use the following case-insensitive attribute selector in CSS



The $= selector checks if the attribute value ends with a specific string. The i flag makes the selector case-insensitive, so it will match .zip, .ZIP, .Zip, and other variations.

70. How select every element whose href attribute value begins with “https”, .pdf & CSS:

71. Is there any reason you'd want to use translate() instead of absolute positioning, or vice-versa? And why?

a. > Use translate() over absolute positioning when:

● **Performance:** translate() is often more efficient for animations because it triggers hardware acceleration, resulting in smoother motion.

AlgoBoost

● **Simplicity**: It's simpler to use and understand, especially when dealing with responsive layouts.

> Use absolute positioning over translate() when:

● **Precise Placement**: Absolute positioning lets you position elements precisely, regardless of their original position.

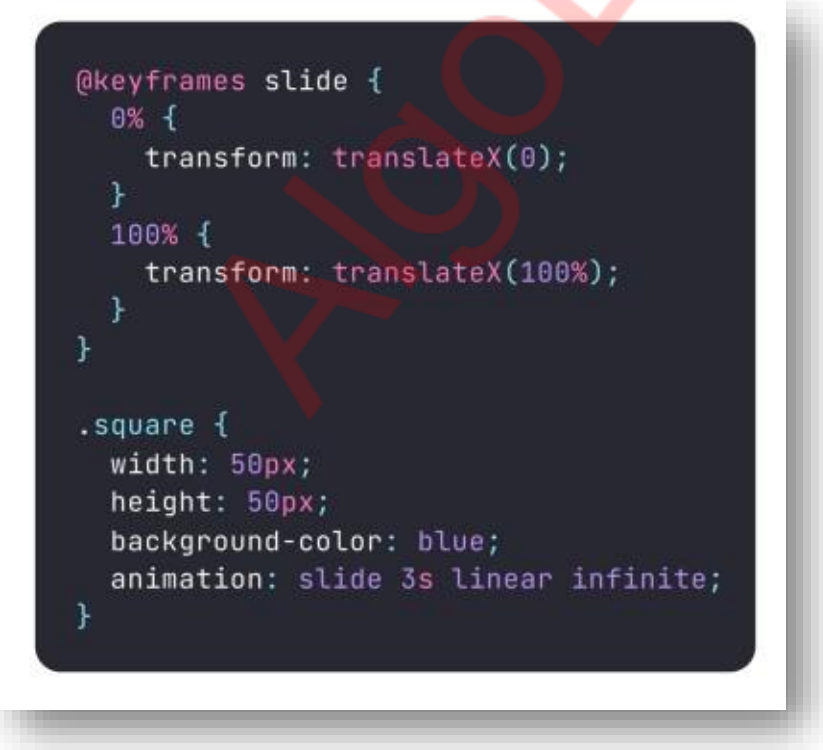
● **Z-Index: It's** easier to control the stacking order of elements with absolute positioning and the z-index property.

● **Complex Layouts**: For intricate layouts, absolute positioning provides better control over element placement.

72. Explain 'Tweening'

a. In CSS, "**tweening"** involves creating smooth animations by calculating intermediate states between **keyframes.** It ensures gradual transitions for elements, resulting in visually appealing and natural animations.

Example : Let's say you have a simple animation where you want a square element to smoothly move from the left side of the screen to the right side.



AlgoBoost

In this example, the @keyframes rule defines the animation steps. The .square class is animated using the slide animation, which moves the square horizontally from 0% (left) to 100% (right) over 3 seconds. Tweening calculates the in-between positions, making the movement smooth and continuous.

73. What are the differences between 'reset' and 'normalize' in CSS?

● **CSS Reset**: Removes all default styles, requiring complete restyling.

● **Normalize CSS:** Preserves useful browser defaults while ensuring consistent styling.

74. What is the property that is used to control image scroll?

a. The overflow property controls image scroll.

75. Explain the concept of CSS Flexbox & its main advantages in web layout design.

a. **CSS Flexbox** enables flexible and responsive layouts. It arranges elements in containers along axes, offering:

● **Easy Alignment**: Automatic alignment minimizes complex CSS rules.

● **Responsive:** Adapts smoothly to various screens.

● **Dynamic Sizing:** Elements adjust while maintaining proportions.

● **Order Control**: Easily reorder elements visually.

● **No Hacks:** Eliminates float-based hacks.

● **Efficient Spacing:** Controls space distribution.

● **Nested Layouts:** Ideal for complex structures.

● **Vertical Alignment**: Simplifies vertical centering.

In short, **Flexbox o**ptimizes layout, enhancing efficiency and adaptability.

76. How do you vertically center an element within a flex container using Flexbox?Provide the necessary CSS properties and values.

a. To vertically center an element within a flex container using Flexbox, you can use the align-items property with the value center. Here's the necessary CSS:

AlgoBoost

```
.flex-container {
  display: flex;
  align-items: center; /* Vertically centers items */
}
```

77. Describe the purpose of the flex-grow, flex-shrink, and flex-basis properties in the flex shorthand property. Provide an example

a. The flex shorthand property in CSS Flexbox combines three properties: flex-grow, flex-shrink, & flex-basis.

● **flex-grow:** It defines how much an item can grow compared to other flex items. It accepts a unitless value representing a proportion. An item with a higher value grows more than others.

● **flex-shrink:** It controls the ability of an item to shrink if the container is too small. Similar to flex-grow, it's unitless and indicates the ratio of shrinking for each item.

● **flex-basis:** This sets the initial size of an item before flex-grow and flex-shrink

```
.flex-item {
  flex: 1 0 200px; /* flex-grow: 1, flex-shrink: 0, flex-basis: 200px */
}
```

AlgoBoost

In this example, .flex-item has the following settings:

● **flex-grow**: 1: The item can grow proportionally if there's extra space.

● **flex-shrink**: 0: The item won't shrink to less than its initial size.

● **flex-basis**: 200px: The item starts with an initial size of 200 pixels.

This setup ensures that the item grows to fill available space but doesn't shrink, maintaining a minimum width of 200 pixels. These properties are especially useful for creating flexible layouts where items should distribute space differently while maintaining specific sizing behaviors.

78. In what situations would you use the align-items and align-content properties in a flex container? Provide examples

a. **Align-items:**

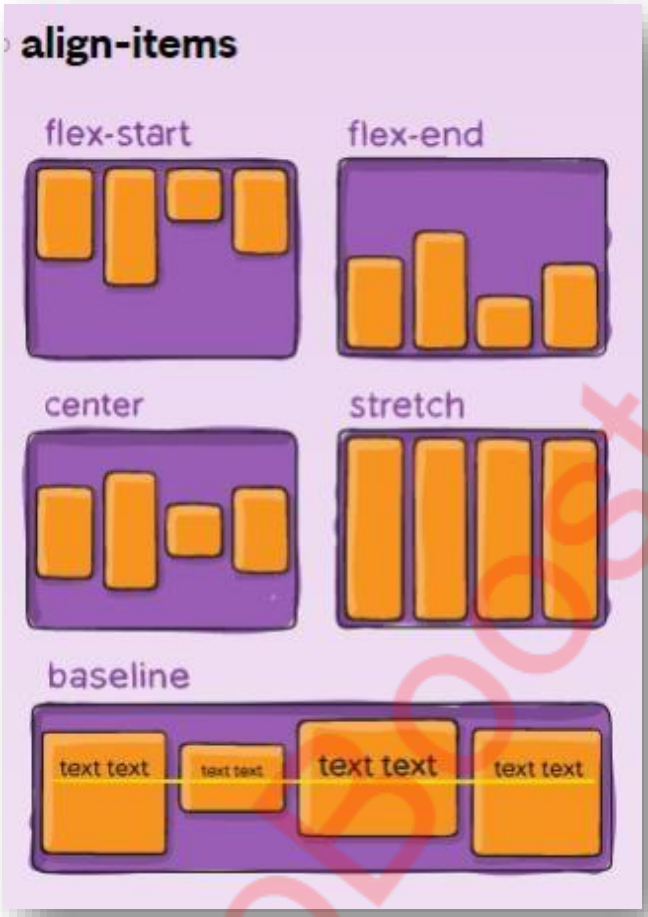
● This property is used to align flex items individually along the cross-axis.

● It's suitable for scenarios where you want consistent alignment for each individual item.

● Example: Vertically centering all items within a flex container

```
.flex-container {
  display: flex;
  align-items: center;
}
```

AlgoBoost

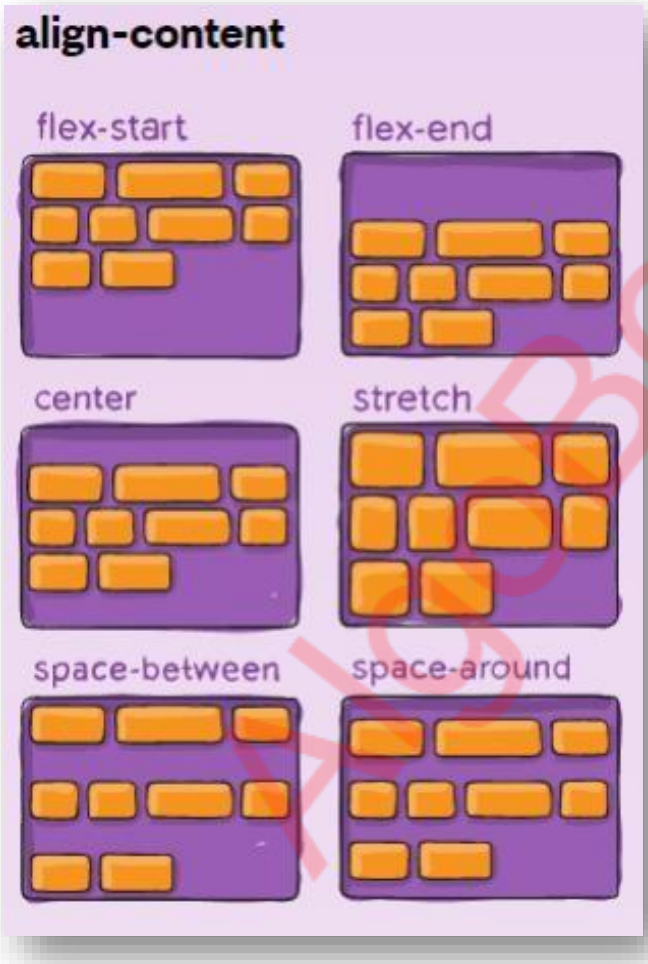


Align-content:

- This property is used to align lines of flex items when there's extra space in the cross-axis.
- It's useful when you have multiple lines of items in a multi-line flex container.
- **Example:** Distributing space between lines of items with space between them.

AlgoBoost

```css
.flex-container {
  display: flex;
  flex-wrap: wrap;
  align-content: space-between;
}
```



XeroBoost

79. Explain the difference between the flex-direction values row and column, and how they influence the arrangement of flex items. Provide a use case for each value.

a**. flex-direction: row**: Arranges items horizontally (left to right by default).

- **Use Case**: Navigation menu.

flex-direction: column: Stacks items vertically (top to bottom).

- **Use Case:** Vertical comment feed.

80. Explain the concept of CSS Grid layout and how it differs from other layout systems like Flexbox.

a**. CSS Grid layout** is a powerful two-dimensional layout system in CSS that allows you to create complex grid-based layouts with rows and columns. It offers precise control over both the horizontal and vertical alignment of items.

Flexbox vs. Grid:

- **One-Dimensional vs. Two-Dimensional**: Flexbox handles single-axis arrangements, while Grid manages both rows and columns for intricate layouts.

- **Layout Complexity**: Flexbox suits simple layouts, Grid excels in complex structures with rows and columns.

- **Alignment:** Grid provides advanced alignment control, allowing complex alignments and content justification across the grid.

81. How do you define a grid container and grid items using CSS Grid? Provide an example of a basic grid layout.

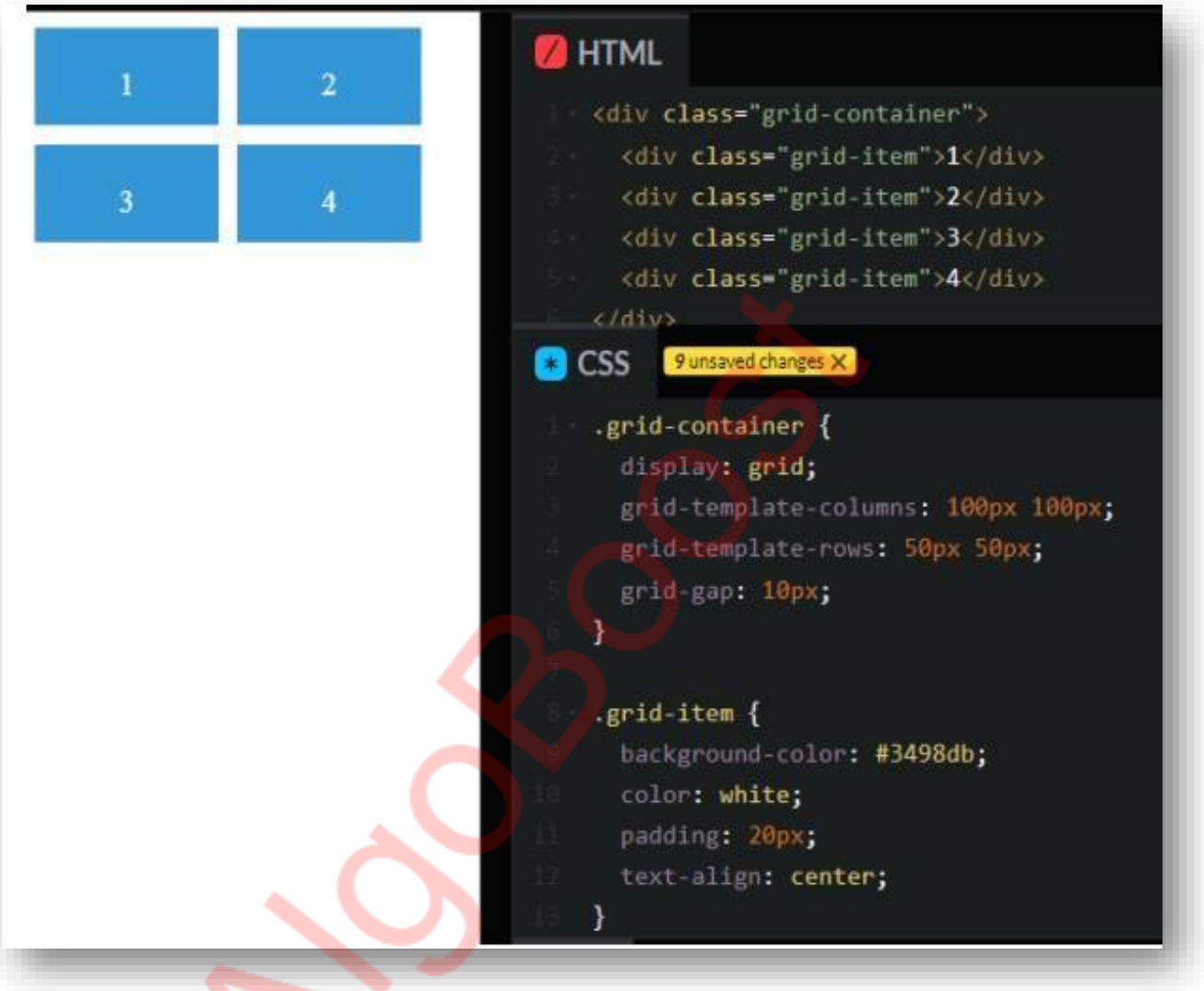
a**. 1. Define the Grid Container:** Apply display: grid; to the parent element to create a grid container.

2. Define Grid Columns and Rows: Use properties like grid-template-columns and grid-template-rows to set the size of columns and rows.

3. Place Grid Items: Use grid-column and grid-row to specify item positions in the grid.

AlgoBoost

Example of a basic grid layout



AlgoBoost

In this example, the .grid-container becomes a grid container with two columns and two rows. .grid-item elements are placed in the grid using these columns and rows. The grid-gap adds spacing between items, resulting in a basic grid layout.

82. What are the differences between implicit and explicit grids in CSS Grid? Give an example of when each type might be used.

a. **Implicit Grid**:

● Implicit grid tracks are created automatically when you place grid items outside the explicitly defined grid using properties like grid-row or grid-column.

● This occurs when items exceed the number of tracks defined in the explicit grid.

● Use Case: When items need to overflow and wrap in a grid without having to manually define every track

```
.grid-container {
  display: grid;
  grid-template-columns: 100px 100px;
  grid-auto-rows: 50px;
}
```

.

Explicit Grid:

● Explicit grid tracks are defined using properties like grid-template-columns and grid-template-rows.

● These tracks are specified and planned in advance.

● Use Case: When you want precise control over the number and size of tracks in your grid.

```
.grid-container {
  display: grid;
  grid-template-columns: repeat(3, 100px);
  grid-template-rows: 50px 100px;
}
```

AlgoBoost

84. In what scenarios would you use the properties grid-gap, grid-row, and grid-column? Provide examples

a. ● **grid-gap:** Adds spacing between grid items. | grid-gap: 10px;

● **grid-row:** Defines vertical span of an item. | grid-row: 2 / span 2;

● **grid-column**: Sets horizontal span of an item. | grid-column: 1 / -1;

85. How do you create CSS animations? Explain the key components involved in defining animations in CSS

a. ● **@keyframes Rule:** Define animation steps using @keyframes, setting styles at specific points.

- **Animation Property**: Apply animations with the animation property.

Name: Referenced animation name from @keyframes.

Duration: Time taken for the animation to complete.

Timing Function: Animation pace (ease, linear, etc.).

Delay: Time before animation starts. Iteration Count: Number of animation cycles.

Direction: Forward, backward, alternate, etc.

Fill Mode: Retaining animation styles before/after.

Play State: Running or paused.

Example:

```
@keyframes slide {
  0% { transform: translateX(0); }
  100% { transform: translateX(100px); }
}

.element {
  animation: slide 2s ease-in-out infinite alternate;
}
```

AlgoBoost

86. What is the difference between CSS transitions and CSS animations? Provide examples

a. **CSS Transitions:**

- Smoothly change property values on user interaction.
- Start and end states, triggered by events like hover.
- **Example:** Changing width on hover.

```
.element {
  transition: width 0.3s ease-in-out;
}

.element:hover {
  width: 200px;
}
```

CSS Animations:

- Use keyframes for complex, automatic animations.
- Multiple states, easing, delays, and auto-play.
- **Example**: Sliding element back and forth automatically.

```
@keyframes slide {
  0% { transform: translateX(0); }
  100% { transform: translateX(100px); }
}

.element {
  animation: slide 2s ease-in-out infinite alternate;
}
```

AlgoBoost

87. Explain the concept of keyframes in CSS animations. How do you use them to create complex animations?

a. **Keyframes** in CSS animations define specific moments during an animation's progression. They let you set styles for elements at different points in time, making animations controlled and smooth.

Using keyframes for complex animations involves:

- **Defining Keyframes:**
  - Use @keyframes followed by a name and animation progress (0% to 100%).
  - Inside each keyframe, set styles for the element.
- **Intermediate States:**
  - Keyframes create intermediate states between start and end, adding fluidity.
- **Multiple Keyframes:**
  - Combine keyframes at different percentages for intricate animations

```
@keyframes bounce {
  0%, 100% { transform: translateY(0); }
  50% { transform: translateY(-50px); }
}

.element {
  animation: bounce 2s ease-in-out infinite;
}
```

AlgoBoost

88. What is the significance of the animation property in CSS? Describe its components and how they influence the behavior of an animation.

a. The animation property in CSS is a shorthand property that combines several individual animation-related properties to define the behavior of an animation applied to an Element.

Property	Description	Influence
Animation Name	Specifies the name of defined @keyframes animation.	Determines animation to apply.
Animation Duration	Sets time for animation to complete one cycle.	Influences animation speed.
Animation Timing Function	Defines pace of animation's progress (ease, linear, etc.).	Controls acceleration and deceleration.
Animation Delay	Determines time before animation starts.	Influences animation start time.
Animation Iteration Count	Specifies how many times animation should repeat.	Sets number of animation cycles.
Animation Direction	Sets animation direction (normal, reverse, alternate).	Controls playback direction.
Animation Fill Mode	Defines element's appearance before/after animation.	Manages pre/post-animation styles.
Animation Play State	Determines if animation is running or paused.	Influences animation's active state.

AlgoBoost

89. How can you optimize CSS animations for performance? What techniques or best practices would you consider to ensure smooth animations without causing performance bottlenecks?

a. **Optimize CSS animations for performance:**

● **Use Hardware Acceleration:** Apply animations to properties like transform and opacity for smoother rendering using the GPU.

● **Minimize DOM Manipulation**: Limit changes to animated elements to reduce layout recalculations.

● **Reduce Animation Complexity**: Avoid heavy animations involving many elements or properties.

● **Use Debouncing and Throttling:** Control animation triggers to prevent excessive updates.

● **Avoid Long Durations:** Shorter durations provide more responsive animations.

● **Use will-change**: Declare animation-intensive elements for browser optimization.

● **Optimize Images:** Use appropriate image formats and sizes to reduce load times.

● **Use CSS Transitions**: Use transitions for simpler animations, as they often perform better.

90. What are CSS media queries, and how do they work? Provide an example

a. CSS media queries are a feature that allows you to apply styles based on the characteristics of the user's device or screen. They enable you to create responsive designs that adapt to different screen sizes, orientations, and other attributes.

Media queries work by evaluating the conditions you specify and applying the associated styles if those conditions are met.

AlgoBoost

```css
/* Default styles for all screen sizes */
.element {
  font-size: 16px;
}

/* Media query for screens narrower than 600px */
@media (max-width: 600px) {
  .element {
    font-size: 14px;
  }
}

/* Media query for screens wider than 1200px */
@media (min-width: 1200px) {
  .element {
    font-size: 18px;
  }
}
```

AlgoBoost

91. Explain the concept of breakpoints in the context of media queries. How do you decide on appropriate breakpoints for different devices?

a. **Breakpoints** in the context of media queries refer to specific screen widths at which you adjust your styles to create a responsive design. They are the points where your layout needs to adapt to fit different devices or screen sizes.

Deciding on breakpoints:

- **Device Research**: Know common sizes and trends for devices.
- **Content and Design:** Adapt for readability and usability.
- **User Behavior:** Analyze popular screen sizes.
- **Testing:** Identify content issues through testing.

● **Responsive Frameworks**: Adjust based on framework's breakpoints.

92. What is the difference between min-width and max-width in media queries?

a. **min-width:**

● Specifies the minimum width at which the styles should apply.

● Styles will be applied when the viewport width is equal to or greater than the specified value.

● Example: @media (min-width: 768px) { /* Styles here */ }

max-width:

● Specifies the maximum width at which the styles should apply.

● Styles will be applied when the viewport width is equal to or smaller than the specified value.

● Example: @media (max-width: 1024px) { /* Styles here */ }

In essence, min-width targets larger screens, while max-width targets smaller screens.

93. How can you use media queries to target different devices, such as smartphones, tablets, and desktops?

a. ● Smartphones (up to 767px)

● Tablets (768px to 1023px)

● Desktops (1024px and above)

94. What is the purpose of the meta viewport tag in responsive web design? How does it relate to media queries, and why is it important for mobile devices?

a. The meta viewport tag in responsive web design is a crucial element that controls how a web page is displayed on different devices, particularly mobile devices. It provides instructions to the browser regarding the initial zoom level and how content should be scaled to fit the viewpo

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

AlgoBoost

● width=device-width ensures the website's width matches the device's width,

● initial-scale=1.0 prevents the page from being zoomed in initially. This helps ensure the website is responsive and readable on mobile devices.

95. What are CSS transitions and how do they work? Provide an example

a. **CSS transitions** allow smooth property changes over time. They work by defining a property, duration, timing, and event. When triggered (e.g., on hover), the property gradually changes to the new value.

```
/* Initial state */
.element {
  width: 100px;
  background-color: blue;
  transition: width 0.5s ease-in-out;
}

/* Hover state */
.element:hover {
  width: 200px;
  background-color: red;
}
```

AlgoBoost

In this example, when you hover over the .element:

● The width property transitions smoothly from 100px to 200px over 0.5 seconds.

● The background-color property also changes from blue to red.

● The ease-in-out timing function ensures a smooth acceleration and deceleration effect during the transition.

96. How can you transition multiple properties simultaneously using CSS? Provide an example of transitioning both opacity and transform properties.

a. You can transition multiple properties simultaneously by listing them within the transition property, separated by commas. Here's an example of transitioning both the opacity and transform properties:

```css
.element {
  opacity: 0.5;
  transform: scale(1);
  transition: opacity 0.3s ease-in-out, transform 0.3s ease-in-out;
}

.element:hover {
  opacity: 1;
  transform: scale(1.2);
}
```

In this example:

- The opacity property changes smoothly from 0.5 to 1.
- The transform property transitions smoothly from its initial state to a scaled state (1 to 1.2) on :hover.
- Both transitions have a duration of 0.3 seconds and use the ease-in-out timing function.

AlgoBoost

97. Explain the concept of event-triggered transitions. How do you use pseudo-classes like :hover to create interactive transitions?

a. Event-triggered transitions are CSS transitions that are activated by specific events, such as hovering over an element or clicking it. These transitions enable you to create interactive effects that enhance user experience without the need for JavaScript.

```
/* Initial state */
.element {
  opacity: 0.7;
  transition: opacity 0.3s ease-in-out;
}

/* Hover state */
.element:hover {
  opacity: 1;
}
```

AlgoBoost

In this example, when you hover over the .element:

● The property transitions smoothly from 0.7 to 1 over 0.3 seconds.

● This creates a fade-in effect when the user interacts with the element. By combining event-triggered pseudo-classes like :hover with transitions, you can add interactivity and engagement to your designs.

98. What are CSS variables (custom properties)? How do they differ from traditional variables in programming languages?

a. CSS variables, also known as custom properties, are a way to store and reuse values in CSS. They allow you to define a value once and reuse it throughout your stylesheet

CSS variables are defined using the -- prefix followed by a name, like --main-color or --font-size.

CSS Variables vs. Traditional Programming Variables:

Aspect	CSS Variables	Traditional Programming Variables
Scope	Scoped to elements or globally using :root	Often block or function scope
Dynamic Nature	Dynamic, can change with JavaScript	Assigned constant values
Property Usage	Style property values	Various data types
Calculation and Transformation	Used in property calculations	Operate based on data type
Interpolation	Direct interpolation using var()	Concatenation or formatting may be needed
Fallbacks	Define fallbacks for unsupported browsers	Fallbacks may vary by language

AlgoBoost

99. What font-related CSS properties are used to control the appearance of text?

- **font-size:** Sets the size of the font.
- **font-family:** Specifies the font(s) to be used for text content.
- **font-weight:** Defines the thickness or boldness of the font.
- **font-style:** Specifies the style of the font (normal, italic, or oblique).
- **font-variant:** Controls the use of small-caps characters in the font.
- **line-height:** Sets the space between lines of text.
- **letter-spacing**: Adjusts the space between characters.
- **word-spacing:** Controls the spacing between words.
- **text-align**: Determines the horizontal alignment of text within its container.

● **text-decoration:** Adds visual styling (underline, overline, line-through) to text.

● **text-transform**: Modifies the capitalization of text (uppercase, lowercase, capitalize).

● **text-shadow**: Applies a shadow effect to the text.

● **color:** Sets the color of the text.

100. What is the @font-face rule in CSS? How do you use it to include custom fonts in your web pages?

a. The **@font-face rule** in CSS is used to define custom fonts that can be used on a web page. It allows you to include external font files and use them in your styles.

This is particularly useful when you want to use non-standard fonts that may not be available on all users' devices.

The @font-face rule consists of several descriptors:

● **font-family**: Specifies the name you'll use to refer to the font in your styles.

● **src**: Defines the source of the font files. This includes URLs to font files like WOFF, WOFF2, TTF, or EOT formats.

● **font-weight**: Specifies the font weight (e.g., normal, bold).

● **font-style:** Defines the font style (e.g., normal, italic).

● **font-display**: Specifies how the font should be displayed while it's loading.

● **unicode-range**: Allows you to specify a range of characters that should be included in the font.

AlgoBoost

Example usage of @font-face:

```
@font-face {
  font-family: 'CustomFont';
  src: url('customfont.woff2') format('woff2'),
       url('customfont.woff') format('woff');
  font-weight: normal;
  font-style: normal;
}
```

With @font-face, you can use web fonts that aren't installed on users' systems, ensuring consistent typography across different devices and platforms.

AlgoBoost