1. What is CSS?
👉 CSS (Cascading Style Sheets) is used to style and format HTML documents (colors, layouts, fonts). It separates design from structure.

2. How do you include CSS in an HTML document?

- **Inline**: <p style="color:red">
- **Internal**: <style>p { color:red; }</style>
- **External**: <link rel="stylesheet" href="style.css">

3. What are the different types of CSS?

- Inline CSS
- Internal CSS
- External CSS

4. What is the box model in CSS?
Every element is a **box** made of:

- Content → Padding → Border → Margin

```
div {

  padding: 10px;

  border: 2px solid black;

  margin: 20px;

}
```

5. How do you center a block element horizontally?

```
div {

  width: 200px;

  margin: 0 auto;

}
```

6. What is the difference between padding and margin?

- **Padding**: Space *inside* the element (between content & border).
- **Margin**: Space *outside* the element (between border & neighbors).

7. What is a CSS selector?
👉 A pattern used to target elements (e.g., p { color:blue; }).

8. What are the different types of selectors in CSS?

- Universal: * {}
- Type: p {}
- Class: .class {}
- ID: #id {}
- Grouping: h1, h2 {}
- Attribute: input[type="text"] {}
- Pseudo-classes/elements

9. What is the difference between class and ID selectors?

- **Class (.class)**: Can be used on multiple elements.
- **ID (#id)**: Must be unique per page.

10. How do you apply multiple classes to an element in HTML?

```
<p class="text bold">Hello</p>
```

11. What is specificity in CSS?
👉 Determines which style is applied when multiple rules conflict.
Order: Inline > ID > Class/Attribute/Pseudo-class > Type

12. What is the !important declaration in CSS?
👉 Overrides all other rules regardless of specificity.

```
p { color: red !important; }
```

13. What are pseudo-classes in CSS?
👉 Styles applied based on **state** of element.

```
a:hover { color: red; }
```

14. What are pseudo-elements in CSS?
👉 Style **parts of elements**.

```
p::first-letter { font-size: 24px; }
```

15. How do you create a CSS grid layout?

```
.container {

  display: grid;

  grid-template-columns: 1fr 1fr 1fr;

  gap: 10px;

}
```

16. What is the flexbox layout in CSS?
👉 A layout model for aligning & distributing space.

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

17. How do you make a responsive design using CSS?
👉 Use **media queries**.

```
@media (max-width: 600px) {
  body { background: lightblue; }
}
```

18. What is a CSS preprocessor?
👉 A tool like **Sass/LESS** that extends CSS with variables, nesting, and functions.

19. What is a CSS framework?
👉 Pre-written CSS libraries (e.g., **Bootstrap, Tailwind, Bulma**) for faster design.

20. What is the difference between em and rem units in CSS?

- **em**: Relative to the parent's font size.
- **rem**: Relative to the root (html) font size.

21. What are the different position values in CSS?

- static (default)
- relative
- absolute
- fixed
- sticky

22. What is the z-index property in CSS?
👉 Controls **stacking order** of elements. Higher = in front.

23. How do you create a fixed header in CSS?

```
header {
  position: fixed;
  top: 0;
  width: 100%;
}
```

24. What is the float property used for?
👉 Used to move elements **left/right**, allowing text to wrap.

25. How do you clear floated elements in CSS?

```
.clearfix::after {
  content: "";
  display: block;
  clear: both;
}
```

26. What is the display property and its possible values?
👉 Defines how an element is displayed.

- block, inline, inline-block, flex, grid, none

27. How do you create a two-column layout using CSS?

```
.container {
  display: grid;
  grid-template-columns: 1fr 1fr;
}
```

28. What is the overflow property in CSS?
👉 Defines how content is handled when it overflows its container.

- visible (default), hidden, scroll, auto

29. How do you make an element invisible but still take up space in the layout?
👉 Use visibility: hidden; (not display: none;).

```
div {
  visibility: hidden;
}
```

30. What is the clip-path property in CSS?
👉 Used to define a **clipping region** that hides parts of an element.

```
img {
  clip-path: circle(50% at center);
}
```

31. How do you create a simple CSS transition?

```
div {
  transition: background 0.5s;
}
div:hover {
  background: red;
}
```

32. What is the transform property in CSS?
👉 Applies 2D/3D transformations like rotate, scale, translate.

```
div { transform: rotate(45deg) scale(1.2); }
```

33. How do you create a CSS animation?

```
@keyframes slide {
  from { transform: translateX(0); }
  to { transform: translateX(100px); }
}
div {
  animation: slide 2s infinite;
}
```

34. What is the difference between transition and animation in CSS?

- **Transition**: Happens only when triggered (hover, click).
- **Animation**: Runs automatically with @keyframes.

35. What is the animation-delay property?
👉 Delays start of animation.

```
div { animation: fade 2s ease 1s; }
```

36. What are CSS keyframes?
👉 Define steps/states of an animation.

```
@keyframes fade {
  0% { opacity: 0; }
  100% { opacity: 1; }
}
```

37. How do you create a bounce effect using CSS?

```
@keyframes bounce {
  0%, 100% { transform: translateY(0); }
  50% { transform: translateY(-50px); }
}
div { animation: bounce 1s infinite; }
```

38. What is the @keyframes rule?
👉 Defines an animation's intermediate steps.

39. What is the transition-timing-function property?
👉 Defines speed curve of a transition.
Values: ease, linear, ease-in, ease-out, cubic-bezier().

40. How do you create a fade-in effect using CSS?

```
.fade {
  opacity: 0;
  animation: fadeIn 2s forwards;
}
@keyframes fadeIn {
  to { opacity: 1; }
}
```

41. What is responsive web design?
👉 Design that adapts to screen size (desktop, tablet, mobile).

42. What is mobile-first design in CSS?
👉 Start designing for **small screens** first, then expand with media queries.

43. What are CSS variables?
👉 Custom properties for reusability.

```
:root { --main-color: blue; }

div { color: var(--main-color); }
```

44. How do you use media queries in CSS?

```
@media (max-width: 768px) {

  body { font-size: 14px; }

}
```

45. What is the calc() function in CSS?
👉 Perform calculations directly in CSS.

```
div { width: calc(100% - 50px); }
```

46. What are vendor prefixes in CSS?
👉 Browser-specific properties for compatibility.

```
div { -webkit-transform: rotate(45deg); }
```

47. What is BEM in CSS?
👉 Naming convention: **Block__Element--Modifier**
Example:

```
.card__title--highlight { color: red; }
```

48. What are CSS frameworks and name a few popular ones?
👉 Prebuilt CSS libraries for UI.
Examples: Bootstrap, Tailwind CSS, Bulma, Foundation.

49. What is the difference between inline, inline-block, and block elements?

- **inline** → No line break, only width of content.
- **block** → Full width, starts on new line.
- **inline-block** → Behaves inline but can set width/height.

50. What is the box-sizing property in CSS?
👉 Defines how width/height are calculated.

- content-box (default) → only content.
- border-box → includes padding + border.

51. How can you optimize CSS for better performance?

- Minify CSS
- Combine files
- Remove unused CSS
- Use shorthand properties
- Load critical CSS first

52. What are CSS sprites?

👉 A single image containing multiple icons → reduce HTTP requests.

53. How do you minify CSS?

👉 Remove spaces/comments using tools like cssnano, uglifycss.

54. What is critical CSS?

👉 The CSS needed for above-the-fold content (loaded first).

55. How do you use font-face in CSS?

```
@font-face {

  font-family: "MyFont";

  src: url("myfont.woff2") format("woff2");

}

body { font-family: "MyFont", sans-serif; }
```

56. What is the will-change property in CSS?

👉 Tells browser which properties will change → better performance.

```
div { will-change: transform; }
```

57. How can you defer loading of non-critical CSS?

- Load critical CSS inline.
- Use media="print" or rel="preload".

58. What is the content-visibility property?

👉 Skips rendering of off-screen elements until needed.

```
div { content-visibility: auto; }
```

59. How do you improve rendering performance with CSS?

- Use GPU-friendly transforms (translate/opacity).
- Avoid large repaints.
- Use will-change.
- Optimize selectors.

60. What are CSS modules?

👉 A system where CSS is scoped locally to components (used in React).

61. What is CSS Grid?

👉 A 2D layout system for rows & columns.

```
.container {
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  gap: 10px;
}
```

62. How do you create a grid container in CSS?

👉 Use display: grid;

```
.container { display: grid; }
```

63. What is the grid-template-areas property?

👉 Defines named grid areas for layout.

```
.container {
  display: grid;
  grid-template-areas:
    "header header"
    "sidebar main"
    "footer footer";
}
```

64. How do you use grid-template-columns and grid-template-rows?

👉 Define number/size of rows & columns.

```
.container {
  grid-template-columns: 200px 1fr;
  grid-template-rows: 100px auto 50px;
}
```

65. What is the flex property in CSS?
👉 Shorthand for flex-grow, flex-shrink, flex-basis.

```
.item { flex: 1 1 100px; }
```

66. How do you create a flex container in CSS?

```
.container { display: flex; }
```

67. What is the flex-direction property?
👉 Defines main axis direction.
Values: row | row-reverse | column | column-reverse.

68. What is the justify-content property?
👉 Aligns items along **main axis**.
Values: flex-start, flex-end, center, space-between, space-around.

69. What is the align-items property?
👉 Aligns items along **cross axis**.
Values: flex-start, flex-end, center, stretch.

70. What is the align-self property?
👉 Overrides align-items for a single flex item.

71. What is the calc() function in CSS?
👉 Perform calculations.

```
div { width: calc(100% - 50px); }
```

72. What is the var() function in CSS?
👉 Access CSS variables.

```
:root { --main: red; }

p { color: var(--main); }
```

73. What is the clamp() function in CSS?
👉 Sets value within a min, preferred, max range.

```
p { font-size: clamp(14px, 2vw, 20px); }
```

74. What is the difference between px, em, rem, %, and vh/vw units?

- **px** → fixed pixels.
- **em** → relative to parent’s font-size.
- **rem** → relative to root (html) font-size.
- **%** → relative to parent size.
- **vh/vw** → relative to viewport height/width.

75. How do you use nth-child selector in CSS?

👉 Selects elements by order.

```
li:nth-child(2) { color: red; }

li:nth-child(odd) { background: lightgray; }
```

76. What is the nth-of-type selector?

👉 Selects by element type & order.

```
p:nth-of-type(2) { color: blue; }
```

77. What is the not() selector in CSS?

👉 Excludes elements.

```
p:not(.highlight) { color: gray; }
```

78. How do you use attr() in CSS?

👉 Inserts attribute values in content.

```
a::after { content: " (" attr(href) ")"; }
```

79. What are viewport units in CSS?

- **vw** → 1% of viewport width.
- **vh** → 1% of viewport height.
- **vmin** → smaller of vw/vh.
- **vmax** → larger of vw/vh.

80. How do you use custom properties (CSS variables) in CSS?

```
:root { --padding: 10px; }

div { padding: var(--padding); }
```

81. What are shorthand properties in CSS?

👉 Properties that combine multiple styles into one line.
Examples: margin, padding, border, font.

82. How do you use the margin shorthand property?

```
div { margin: 10px 20px 30px 40px; }

/* top right bottom left */
```

83. How do you use the padding shorthand property?

```
div { padding: 10px 20px; }

/* top-bottom | left-right */
```

84. How do you use the border shorthand property?

```
div { border: 2px solid black; }
```

85. How do you use the background shorthand property?

```
div {

  background: url("img.jpg") no-repeat center/cover;

}
```

86. How do you use the font shorthand property?

```
p { font: italic bold 16px/1.5 Arial, sans-serif; }
```

87. How do you use the flex shorthand property?

```
.item { flex: 1 0 auto; }

/* grow shrink basis */
```

88. How do you use the list-style shorthand property?

```
ul { list-style: square inside url("icon.png"); }
```

89. How do you use the transition shorthand property?

```
div { transition: all 0.5s ease-in-out; }
```

90. How do you use the animation shorthand property?

```
div { animation: bounce 2s ease-in-out 1s infinite; }
```

91. How do you debug CSS issues?

- Use **browser DevTools** (inspect element).
- Check **specificity & overrides**.
- Test **in multiple browsers**.

92. What are CSS resets?
👉 Normalize/remove browser default styles for consistency.
Example: * { margin:0; padding:0; box-sizing:border-box; }

93. How do you use the outline property for debugging?
👉 Helps visualize element boundaries.

```
* { outline: 1px solid red; }
```

94. What is the difference between outline and border?

- **Outline**: Drawn outside the element's border edge, does not affect layout, can overlap other elements.
- **Border**: Drawn around the element's box, affects layout and element dimensions.

95. How do you use the :hover pseudo-class for testing?

```
button:hover {
  background-color: lightblue;
  cursor: pointer;
}
```

- The :hover pseudo-class applies styles when the user hovers over the element with the mouse.

96. What are CSS custom properties (variables) and how do you debug them?

- **CSS Variables**: Defined using -- and accessed with var().

```
:root {
  --main-color: #3498db;
}
p {
  color: var(--main-color);
}
```

- **Debugging**: Use browser dev tools to inspect computed styles, check variable values, or override them temporarily in the inspector.

97. How do you test for responsive design?

- Resize the browser window.
- Use developer tools to toggle device emulation (Chrome DevTools → Toggle Device Toolbar).
- Check layout on multiple devices (desktop, tablet, mobile).
- Use media queries in CSS to ensure adaptability.

98. What are CSS grid inspector tools?

- Browser dev tools (Chrome, Firefox, Edge) provide a **Grid Inspector** to visualize grid lines, areas, and gaps, making it easier to debug grid layouts.

99. How do you handle browser-specific CSS issues?

- Use **vendor prefixes** (e.g., -webkit-, -moz-, -ms-).
- Test across browsers.
- Use feature queries (@supports).
- Avoid unsupported properties or provide fallback styles.

100. How do you validate your CSS?

- Use online validators like W3C CSS Validator.
- Check for syntax errors, unsupported properties, and browser compatibility warnings.

101. Write CSS rules to set the font family to Arial, color to dark blue, and font size to 16px for all paragraphs.

```
p {
  font-family: Arial, sans-serif;
  color: darkblue;
  font-size: 16px;
}
```

102. Create a class named .highlight to change the background color to yellow and font weight to bold, and apply it to some text.

```
.highlight {
  background-color: yellow;
  font-weight: bold;
}
```

```
<p>This is <span class="highlight">highlighted text</span> in a paragraph.</p>
```