1. What is the Box Model in CSS?

The CSS box model consists of four parts: content, padding, border, and margin. These properties define the space around an element and how it interacts with other elements.

2. What is the difference between `margin` and `padding`?

- Margin is the space outside an element's border, creating space between elements.

- Padding is the space inside an element's border, between the content and the border.

3. What is the difference between `display: block`, `display: inline`, and `display: inline-block`?

- Block: Takes the full width of its container and starts on a new line (e.g., `<div>`).

- Inline: Takes only as much width as its content and does not start on a new line (e.g., `<span>`).

- Inline-block: Combines features of both; it takes only as much width as its content but allows setting width, height, padding, and margin.

4. What is the purpose of the `z-index` property?

The `z-index` property controls the stacking order of elements. A higher `z-index` value brings an element to the front, while a lower value sends it to the back. It only works on elements with a `position` value other than `static` (e.g., `relative`, `absolute`, `fixed`).

5. What is the difference between `position: absolute` and `position: relative`?

- Relative: Positions the element relative to its normal position in the document flow.

- Absolute: Positions the element relative to its nearest positioned ancestor (or the viewport if no ancestor is positioned).

6. What is the difference between `em` and `rem` units?

- em: Relative to the font size of the parent element.

- rem: Relative to the font size of the root (`<html>`) element.

7. What is the purpose of the `float` property?

The `float` property is used to position an element to the left or right of its container, allowing text or other elements to wrap around it. Common values are `left`, `right`, and `none`.

8. How do you apply styles to multiple elements at once?

Use a comma-separated list of selectors. Example:

h1, h2, h3 { color: blue; }

9. What is the difference between `visibility: hidden` and `display: none`?

- `visibility: hidden`: Hides the element but still occupies space in the layout.

- `display: none`: Completely removes the element from the layout, and no space is reserved for it.

10. What is the `cascade` in CSS?

- The cascade is the process by which CSS determines which styles are applied to an element when multiple rules match. It considers:

- Specificity: More specific selectors override less specific ones.

- Importance: `!important` declarations override normal rules.

13. What is the `grid` layout model, and how is it different from `flexbox`?

- Answer: CSS Grid is a two-dimensional layout model that allows you to create complex layouts with rows and columns. Key differences from Flexbox:

- Flexbox: One-dimensional (either row or column).

- Grid: Two-dimensional (rows and columns simultaneously).

- Example:

.container {

display: grid;

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

gap: 10px;

}

7. How do you make a responsive layout in CSS?

- Answer: Use media queries to apply different styles based on screen size. Example:

```css

@media (max-width: 768px) {

.container {

flex-direction: column;

}

}

3. How do you center a `div` content horizontally and vertically?

- Answer:

.center {

display: flex;

justify-content: center;

align-items: center;

height: 100vh; /\* Full viewport height \*/

}

12. What is the difference between `px`, `%`, `vh`, and `vw` units?

- Answer:

- `px`: Absolute unit (pixels).

- `%`: Relative to the size of the parent element.

- `vh`: Relative to 1% of the viewport height.

- `vw`: Relative to 1% of the viewport width.

10. What is the `flexbox` layout model, and how does it work?

- Answer: Flexbox is a one-dimensional layout model that allows you to distribute space and align items within a container. Key properties include:

- `display: flex`: Enables flexbox for the container.

- `justify-content`: Aligns items along the main axis (e.g., `center`, `space-between`).

- `align-items`: Aligns items along the cross axis (e.g., `center`, `flex-start`).

- `flex-direction`: Defines the direction of the main axis (e.g., `row`, `column`).