Q1. How are inline and block elements different from each other?

Ans1. A block-level element always starts on a new line and takes up the full width available i.e stretches out to the left and right as far as it can.
For e.g.- <p>,<table>,<ul>,<ol>,<div>

An inline element does not start on a new line and only takes up as much width as necessary.
For e.g- <span>,<strong>,<br>

Q2. Explain the difference between visibility:hidden and display:none.

Ans2. display:none means that the tag in question will not appear on the page at all (although you can still interact with it through the dom). There will be no space allocated for it between the other tags.
visibility:hidden means that unlike display:none, the tag is not visible, but space is allocated for it on the page. The tag is rendered, it just isn't seen on the page.

Q3. Explain the clear and float properties.

Ans3. The float property is used for positioning and layout on web pages. The float property can have one of the following values:

- left - The element floats to the left of its container
- right- The element floats to the right of its container
- none - The element does not float (will be displayed just where it occurs in the text). This is default
- inherit - The element inherits the float value of its parent

In its simplest use, the float property can be used to wrap text around images.

The clear property specifies what elements can float beside the cleared element and on which side.

The clear property can have one of the following values:

•none - Allows floating elements on both sides. This is default

•left - No floating elements allowed on the left side

•right- No floating elements allowed on the right side

•both - No floating elements allowed on either the left or the right side

- inherit - The element inherits the clear value of its parent

The most common way to use the clear property is after you have used a float property on an element.

Q4. Explain difference between absolute, relative,fixed and static.

Ans 4. HTML elements are positioned static by default.

Static positioned elements are not affected by the top, bottom, left, and right properties.

An element with position: static; is not positioned in any special way; it is always positioned according to the normal flow of the page:

An element with position: relative; is positioned relative to its normal position.

Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.
An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled. The top, right, bottom, and left properties are used to position the element.

A fixed element does not leave a gap in the page where it would normally have been located.

An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed).

However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.

Q5. Write the HTML code to create a table in which there are 4 columns(ID , Employee Name, Designation, Department) and at least 6 rows. Also do some styling to it.

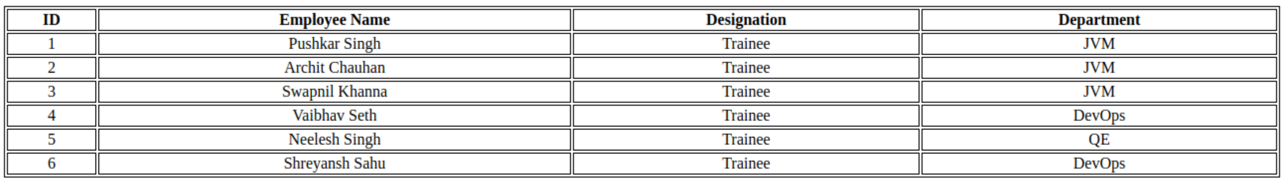
Ans5.

```
<!DOCTYPE html>

<html>

<head>
```

```
<style>

table, th, td {

   border: 1px solid black;

   text-align: center;

}

</style>

<meta charset="utf-8">

<title>Record</title>

</head>

<body>

<table style="width:100%">

        <thead>

                <tr>

                        <th>ID</th>

                        <th>Employee Name</th>

                        <th>Designation</th>

                        <th>Department</th>

                </tr>

        </thead>

        <tbody>

                <tr>

                        <td>1</td>

                        <td>Pushkar Singh</td>

                        <td>Trainee</td>

                        <td>JVM</td>

                </tr>
```

```
<tr>
        <td>2</td>
        <td>Archit Chauhan</td>
        <td>Trainee</td>
        <td>JVM</td>
</tr>
<tr>
        <td>3</td>
        <td>Swapnil Khanna</td>
        <td>Trainee</td>
        <td>JVM</td>
</tr>
<tr>
        <td>4</td>
        <td>Vaibhav Seth</td>
        <td>Trainee</td>
        <td>DevOps</td>
</tr>
<tr>
        <td>5</td>
        <td>Neelesh Singh</td>
        <td>Trainee</td>
        <td>QE</td>
</tr>
<tr>
        <td>6</td>
```

```
            <td>Shreyansh Sahu</td>

            <td>Trainee</td>

            <td>DevOps</td>

        </tr>


    </tbody>


</table>


</body>


</html>
```



ID	Employee Name	Designation	Department
1	Pushkar Singh	Trainee	JVM
2	Archit Chauhan	Trainee	JVM
3	Swapnil Khanna	Trainee	JVM
4	Vaibhav Seth	Trainee	DevOps
5	Neelesh Singh	Trainee	QE
6	Shreyansh Sahu	Trainee	DevOps

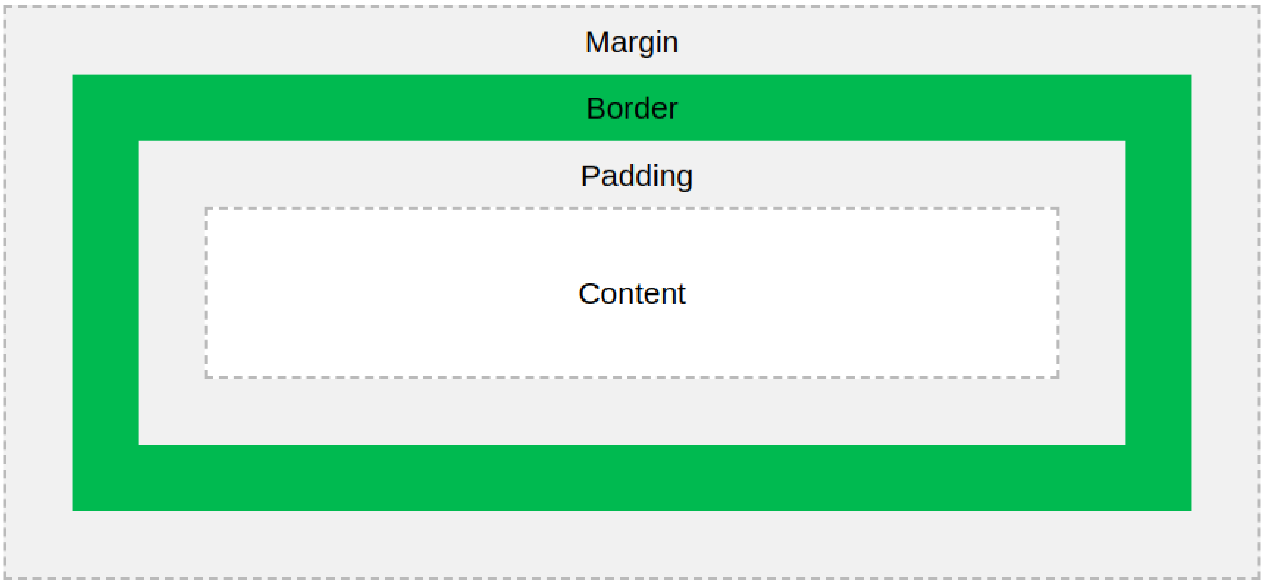
Q6. Why do we use meta tags?

Ans6. The <**meta**> **tag** provides metadata about the HTML document. Metadata will not be displayed on the page, but will be machine parsable. **Meta** elements are typically **used** to specify page description, keywords, author of the document, last modified, and other metadata.

Q7. Explain the Box model.

Ans7. All HTML elements can be considered as boxes. In CSS, the term "box model" is used when talking about design and layout.

The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content.



Explanation of the different parts:

- •**Content**- The content of the box, where text and images appear
- •**Padding**- Clears an area around the content. The padding is transparent
- •**Border**- A border that goes around the padding and content
- •**Margin-** Clears an area outside the border. The margin is transparent

The box model allows us to add a border around elements, and to define space between elements.

Q8. What are the different types of CSS Selectors?

Ans 8. A CSS selector is the part of a CSS rule set that actually selects the content you want to style. Different types of selectors-

Universal Selector - The *universal selector* works like a wild card character, selecting all elements on a page. Every HTML page is built on content placed within HTML tags. Each set of tags represents an element on the page.

Element Type Selector- Also referred to simply as a "type selector," this selector must match one or more HTML elements of the same name. Thus, a selector of nav would match all HTML nav elements, and a selector of <ul> would match all HTML unordered lists, or <ul> elements.

ID Selector- An ID selector is declared using a hash, or pound symbol (#) preceding a string of characters. The string of characters is defined by the developer. This selector matches any HTML element that has an ID attribute with the same value as that of the selector, but minus the hash symbol.

Class Selector- The class selector is the most useful of all CSS selectors. It's declared with a dot preceding a string of one or more characters. Just as is the case with an ID selector, this string of characters is defined by the developer. The class selector also matches all elements on the page that have their class attribute set to the same value as the class, minus the dot.
Descendant Combinator- The descendant selector or, more accurately, the descendant combinator lets you combine two or more selectors so you can be more specific in your selection method.
Child Combinator- A selector that uses the child combinator is similar to a selector that uses a descendant combinator, except it only targets immediate child elements:

Q9. Define Doctype.
Ans9. A document type declaration, or **DOCTYPE**, is an instruction that associates a particular SGML or XML document (for example, a webpage) with a document type **definition** (DTD) (for example, the formal **definition** of a particular version of HTML1.0 - HTML 4.0).

Q10. Explain 5 HTML5 semantic tags.
Ans10. A semantic element clearly describes its meaning to both the browser and the developer.
Examples of non-semantic elements: <div> and <span> - Tells nothing about its content.

Examples of semantic elements: <form>, <table>, and <article> - Clearly defines its content.

HTML5 <section> Element

The <section> element defines a section in a document. According to W3C's HTML5 documentation: "A section is a thematic grouping of content, typically with a heading." A home page could normally be split into sections for introduction, content, and contact information.

HTML5 <article> Element

The <article> element specifies independent, self-contained content. An article should make sense on its own, and it should be possible to read it independently from the rest of the web site.

HTML5 <header> Element

The <header> element specifies a header for a document or section. The <header> element should be used as a container for introductory content. You can have several <header> elements in one document.

HTML5 <footer> Element

The <footer> element specifies a footer for a document or section. A <footer> element should contain information about its containing element. A footer typically contains the author of the document, copyright information, links to terms of use, contact information, etc. You may have several <footer> elements in one document.

HTML5 <nav> Element

The <nav> element defines a set of navigation links.

Q11. Create HTML for web-page.jpg (check resources, highest weightage for answers)

Ans11.

Q12. Create HTML for form.png (check resources, highest weightage for answers)

Ans12.