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

- **Block-level elements** always start at new line and takes up the full width available

e.g. <div> , <p>, <pre>, <form> etc.

- **Inline element** does not start on a new line and only takes up as much width as necessary.

e.g. <span>, <button>, <i>, <input>, etc.

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

- **Visibility:hidden** means that the tag with which it is used is not visible, but the space is still allocated for it on the page.

Hello | <span style="visibility:hidden;">I am hidden</span> | Hello

It will result in :

Hello | | Hello

- **Display:none** means that the tag with which it is used will not be visible and unlike visibility:hidden, the space for it won't be allocated on the page.

Hello | <span style="display:none;">I am hidden</span> | Hello

It will result in :

Hello | | Hello

3. Explain clear and float properties.

- **The float property** is used for positioning and formatting content

e.g. Letting an image floating to the left or right of a table

img {

float : right ;
}

<p> <img src="tonyStark.jpg"> Iron Man </p>

Float can have the following values:

1. Left
2. Right
3. None
4. Inherit

- **The clear property** specifies what elements can float beside the cleared element and on which side. It basically clears elements to the direction specified.

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

- **Static :** This is the default value. All elements are in order as they appear in the document.

- **Relative :** the element is positioned relative to it's normal position

When we set position relative to an element, without adding any other positioning attributes(top, bottom, right, left), nothing will happen. When we add additional position, such as, left:20px, the element will move 20px to the right from its normal position. So, clearly it is moving relative to itself.

#some_element {

Position : relative ;

Left : 20px;

}

- **Absolute :** The element is positioned absolutely to it's first positioned parent

It allows us to place our element precisely where we want it.
If, there is no positioned parent element, it will be positioned related directly to the HTML element (The page itself)

```
#parent_element {

Position : relative;
Width : 600px;
Height : 600px;
}

#child_element {
Position : absolute;
Right : 40px;

}
```

- **Fixed :** The element is positioned related to the browser

5. . 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.

6. Why do we use meta tags?

The **<meta>** tag provides metadata about the HTML document. Metadata will not be displayed on the page, but will be machine parsable.

<meta> tag is typically used to specify page description, keywords, author of the document, last modified and other meta data

The metadata can be used by the browsers, like how to display content or reload page.

E.g.

```
<head>
        <meta charset="UTF-8">
        <meta name="keywords" content="HTML, CSS, XML, JavaScript">
</head>
```

7. Explain box-model.

- All HTML elements can be considered as boxes. In CSS, this term is used when talking about design and layout.

- It is essentially a box that wraps around every HTML element.

- It consists of : **margins, borders, padding and the content.**

8. What are the different types of CSS Selectors?

- **CSS selectors are used to select the content you want to style**. They are part of CSS rule set.

- There are several types of selectors in CSS :

8.1 CSS Element Selector:

```
<head>
<style>
        p{
                color : red;
          }
</syle>
</head>
<body>
        <p>Hello</p>
</body>
```

8.2 CSS id Selector

```
<head>
<style>
        #key {
                color : red;
          }
</syle>
</head>
<body>
        <p id="key">Hello</p>
</body>
```

8.3 CSS class selector

```
<head>
<style>
        .key {
                color : red;
          }
</syle>
</head>
<body>
        <p class=“key”>Hello</p>
</body>
```

8.4 CSS Universal selector

```
<head>
<style>
        * {
                color : red;
          }
</syle>
</head>
<body>
        <h1>Big Hello</h1>
        <p>Hello</p>
</body>
```

8.5 CSS group selector

```
<head>
<style>
        h1, p {
                color : red;
          }
</syle>
</head>
<body>
        <h1>Big Hello</h1>
        <p>Hello</p>
        <h2>Bye</h2>
</body>
```

9. Define Doctype

- The <!DOCTYPE> declaration must be the very first thing in your HTML document, before the <html> tag.

- The <!DOCTYPE> declaration is not an HTML tag; it is an instruction to the web browser about what version of HTML the page is written in.

- In HTML 4.01, the <!DOCTYPE> declaration refers to a DTD, because HTML 4.01 was based on SGML. The DTD specifies the rules for the markup language, so that the browsers render the content correctly.

- HTML5 is not based on SGML, and therefore does not require a reference to a DTD.

10. Explain 5 HTML5 semantic tags

10.1 <section> (Section)

A section is thematic grouping of content, typically with a heading.

10.2 <article> (Article)

This 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

10.3 <header> (Article)

It should be used as a container for introductory content at the top generally.

10.4 <footer> (Footer)

It should contain information about it’s containing element. A footer typically contains the author of the document, copyright information, links to terms of use, etc.

10.5 <nav> (Navigation)

This tag defines a set of navigation links.