**← HTML Questions →**

Answer 1 -> HTML

NO, <!DOCTYPE html> is not a tag of html. It is used to tell the web browsers which version of HTML your document is using and it ensure that your web pages are displayed correctly.

Answer 2 -> HTML

Semantic tags are those tags in html which gives meaningful information to both user as well as the browser about the content.

For example: Header, Footer, Aside, Article etc.

Answer 3 -> HTML

HTML Tags is a keyword surrounded by angle bracket that defines how content is displayed on a webpage .

Ex <p>, <h1>…..<h6>, <img> etc

Answer 4 -> HTML

GitHub-> https://github.com/krishna5867/placement-assignments/tree/main/HTML%20Resume

Live Link-> https://placement-assignments.vercel.app

Answer 5 -> HTML

GitHub->https://github.com/krishna5867/placement-assignments/tree/main/Landing%20page

Answer 6 -> HTML

**Enhanced Forms and Input Types:** HTML5 introduces new form input types, including

date pickers, email fields, number inputs, and more. Additionally, it provides built-in

form validation, reducing the need for JavaScript-based validation and enhancing the user experience.

**Improved Semantics:** HTML5 introduces new semantic elements such as <header> <nav> <article> and <footer> which provide better structure and meaning to web content. This helps search engines and accessibility tools understand the document structure more accurately.

**Media Support:** HTML5 provides native support for audio and video playback without requiring third-party plugins like Flash. This enables developers to easily embed multimedia content and create engaging experiences on websites.

Answer 7 -> HTML

Music Player

GitHub->

Live Link->

Answer 8 -> HTML

<img> tag: The <img> tag is used to insert an image into an HTM document. It is a self-closing tag that required the src attribute, which specifies the URL or path to the image file.

**<figure>** tag: The **<figure>** tag is used to encapsulate self-contained content, typically representing images,

illustrations, diagrams, code snippets, or other media objects. It provides a way to group related content and

associate a caption or description with it using the **<figcaption>** tag

Answer 9 -> HTML

HTML Tags is a keyword surrounded by angle bracket that defines how content is displayed on a webpage .

Ex <p>, <h1>….<h6>, <img> etc.

Where as Attribute tag provides additional information about an HTML element . Always specified in opening tags and can modify an element behavior or appearance.

Ex: href and src in <img> tag.

Answer 10 -> HTML

GitHub-> https://github.com/krishna5867/placement-assignments/tree/main/HTML%20Table

Live Link-> <https://html-table-kappa.vercel.app>

**← CSS Questions →**

Answer 1-> CSS

By default when the browser shows element it automatically shows like a box where every element is surrounded by a invisible box around it . There are 4 different properties of this box are content, padding , margin and border.

Answer 2-> CSS

1. **Element Selector**: Selects elements based on their tag name (e.g., div selects all **<**div**>** elements).
2. **Class Selector**: Selects elements based on their assigned class name (e.g., **.**my-class selects all elements with the class "my-class").
3. **ID Selector**: Selects elements based on their unique ID attribute (e.g., #my-id selects the element with the ID "my-id").
4. **Attribute Selector**: Selects elements based on their attributes and attribute values (e.g., [type="text"] selects elements with type="text**"**).
5. **Pseudo-Class Selector**: Selects elements based on specific states or conditions (e.g., **:**hover selects elements when the user hovers over them).
6. **Pseudo-Element Selector**: Selects specific parts or elements of an element (e.g., ::before inserts content before an element).
7. **Descendant Selector**: Selects elements that are descendants of a specific parent element (e.g., div p selects all **<p>** elements that are descendants of <div> elements).
8. **Child Selector**: Selects elements that are direct children of a specific parent element (e.g., ul > li selects all **<li>** elements that are direct children of <ul> elements).
9. **Adjacent Sibling Selector**: Selects elements that are adjacent siblings of a specific element (e.g., h2 + p selects the <p> element that immediately follows an <h2> element).
10. **General Sibling Selector**: Selects elements that are siblings of a specific element (e.g.,h2 ~ p selects all <p> elements that are siblings of an <h2> element).

Answer 3-> CSS

**VW/VH** are CSS units used to measure view width and view height respectively in percentage form in the responsive design technique.

**PX-** Pixel value provide so precise and fixed measurement that is not dependent on the size of the user’s screen.

Answer 4 -> CSS

Inline, Inline-block and Block are the display property in the CSS. which determine how elements are visually rendered and interact with other elements on a web page.

1. **Inline Elements:**

* Inline elements do not start on a new line; they flow within the text content and occupy only the space necessary for their content.
* Inline elements do not have a specified width or height. Their dimensions are determined by the content they contain.
* Margins and paddings can be applied to inline elements horizontally, but they do not affect the vertical layout or positioning of other elements.
* Examples of inline elements include <span>, <a>, <strong>, and <em>.

1. **Inline-Block Elements:**

* Inline-block elements are similar to inline elements in that they flow within the text content, but they also have block-level properties like width, height, margins, and paddings.
* Inline-block elements start on the same line as text content but can have specified dimensions and vertical alignment.
* They allow for horizontal and vertical spacing, as margins and paddings affect the surrounding elements.
* Examples of inline-block elements include <img>, <input>, and <button>.

1. **Block Elements:**

* Block elements start on a new line and occupy the full width available within their parent container by default.
* Block elements have a specified width, height, and can have margins and paddings that affect both the horizontal and vertical layout of other elements.
* They create a visual separation, typically stacking vertically on top of each other.
* Examples of block elements include <div>, <p>, <h1>, and <ul>.

Answer 5-> CSS

**Border-Box** is used toprovide the inner dimension for the element in the documents by providing padding and border with respect to the length and width of the content.

The "**content box**" box-sizing model is used in CSS by default and is commonly used in scenarios where you want to size elements based on their content and have control over the padding and border widths separately.

Answer 6-> CSS

**z-index** is a CSS property that determines the stacking order of positioned elements on a web page. It controls how elements are layered and displayed in relation to each other along the z-axis, which is a three-dimensional axis that extends from the screen towards the viewer.

**Higher Values = Closer to the Viewer**: The z-index property accepts an integer value. Elements with a higher z-index value are placed closer to the viewer and will be displayed on top of elements with lower z-index values within the same stacking context.

**Negative z-index Values**: Negative z-index values can be used to place an element behind the default stacking order (the background or other elements) within the same stacking context. Elements with negative z-index values are positioned behind elements with positive values or no z-index specified.

Answer 7-> CSS

Grid and Flex are two important display property of the CSS which is used to create a responsive layout of a webpage.

The main difference between them are Grid is a two dimensional layout system along with row and columns where as flex is only one dimensional system either row or column.

Grid is used for creating large sized layout where as flex is used for the component of an application.

Answer 8-> CSS

Periodic table

GitHub->

Answer 9-> CSS

Flex Box layout

GitHub-> https://github.com/krishna5867/placement-assignments/tree/main/Flex-Box%20Layout

Answer 10-> CSS

Responsive layout

GitHub->

Answer 11-> CSS

iNeuron

GitHub->

Live Link->

Answer 12-> CSS

**Pseudo Classes** are the type of Pseudo-elements that don’t exist in a normal document tree.

It allow selecting the regular element under certain condition especially when we try to hover over the links .

Ex: a:hover{

Color: #000;

}

Pseudo Elements provides special effects to some selectors. CSS find it useful in applying styles in HTML markups if additional markups or styles is not feasible for a document the pseudo-elements help by allowing extra markup without interfering with the original document.

Ex: before, after etc.

**← JavaScript Questions →**

Answer 1-> JavaScript

Hoisting refers to the process whereby the interpreter appers to move the declaration of function, variable or classes to the top of their scope prior to the execution of the code. It allows functions to be safely used in code before they are declared.

Answer 2-> JavaScript

HOC are the function which take another function as an argument and returns a new function map, filter, reduce are the example of higher order function in JavaScript.

Map is used to illiterates through every element of the array and do some necessary operations on it.

**Difference between map and forEach:-**

forEach is also same as map but the main difference between them is forEach return undefine and it mutates the original array but map doesn’t mutate and always returns a new output array.

Answer 3-> JavaScript

Call, Apply And Bind help in the function borrowing, we can borrow a function from some object and can use it with data of some other object.

**Difference: -**

**Call** takes two argument first is the reference of the objects which this keyword is pointing and the second is the argument of the function .

**Example:**

const people = {

    name: "Krishna",

}

function sayHello(age){

    return "Hy" + " " + "my name is " + *this*.name + " " + "and my age is" + " " + age;

}

console.log(sayHello.call(people, 22));

*//Output  Hy my name is Krishna and my age is 22*

**Apply** is same as the call the only difference is the argument it takes. It takes in the form of an array where as call take it explicitly.

**Exmple:**

console.log(sayHello.apply(people, [36]));

*//Output  Hy my name is Krishna and my age is 123*

**Bind** is also same as the call instead of just immediately invoking the function it returns the copy of the function which we can invoke later on the code.

**Example:**

const bind = sayHello.bind(people);

console.log(bind(12));

*//Output Hy my name is Krishna and my age is 12*

Answer 4 -> JavaScript

Event Bubbling – JavaScript allows DOM elements to be nested inside each other. In such a case if the handler of the child is clicked the handler of parent will also work as if it were clicked to.

Event Capturing – It is just reverse of the event bubbling the only difference is when handler of parent Is clicked the handler of child will also work as if were clicked to .

Answer 5-> JavaScript

Currying is the process of taking a function with multiple argument and returning it into a sequence of function each with only a single argument.

Example:

function curry(a){

    return function(b){

        return function(c){

            console.log(a+b+c);

        }

    }

}

*// curry(2)(3)(4)*

*// Output 9*

Answer 6-> JavaScript

Output – First

Third

Second

Answer 7-> JavaScript

Promises are the objects in the java script representing the eventual completion or failure of an asynchronous operation.

It has 3 different stages: -

1. Fulfill
2. Rejected
3. Pending

Example:

1. console.log("Start");
2. const promises = new Promise((resolve, reject) => {
3. setTimeout(()=> {
4. const result = true;
5. if(result){
6. resolve("Promise is resolved")
7. }else reject(new Error("Promise Rejected"));
8. }, 2000)
9. });
10. promises
11. .then((res) => {
12. console.log(res);
13. })
14. .catch((err) => {
15. console.log(err);
16. })
17. console.log("Stop");
18. *// Output*
19. start
20. Stop
21. Promise is resolved

Answer 8-> JavaScript

This keyword in JavaScript refers to the object and which object it will refer to it depends on how this is being invoked or called.

In an object method this will refer to object.

Alone this refers to the global object.

In a function this refer to the global object.

In an event this refers to the global object receiving the event.

var obj = {

    name: 'Krishna',

    getName: function(){

        console.log(*this*.name);

    }

}

obj.getName();

*// Krishna*

Answer 9-> JavaScript

Basically, it’s a runtime model present in the JS engine and it continuously keeps on checking the call stack and the call back queue and if it finds any callback function is waiting to be executed either in call stack or micro task queue then event loop will check the callback is either empty or not. If it is empty then it pushes the callback function giving priority to micro task queue first into the call stack queue for its execution.

Answer 10-> JavaScript

Debouncing is a technique used in web development to control the frequency of a particular event, such as input or scroll events, by limiting the execution of a function to occur after a specified delay has passed since the last occurrence of the event.

GitHub -> https://github.com/krishna5867/placement-assignments/tree/main/Debouncing

Answer 11-> JavaScript

Closures is a function bundled together with reference to its lexical environment. Closures are essential concept in programming because they allow us to create function that can remember and access the variable that were present inside the parent scope even after the parent function has been returned.

Answer 12-> JavaScript

Create a Blog web app using JavaScript (10 Marks) - Fetch data from https://jsonplaceholder.typicode.com/posts and show it to ui - User can also add new blog - Add Delete functionality also

**← React Questions →**

Answer 1-> React

React is an open-source front-end JavaScript library that is used for building user interfaces, especially for single-page applications.

The major features of React are:

1. It uses Virtual DOM instead of Real DOM considering that Real DOM manipulations are expensive.

2. Supports server-side rendering.

3. Follows Unidirectional data flow or data binding.

4. Uses reusable/composable UI components to develop the vie

Answer 2-> React

Virtual DOM is a light weight memory representation of the actual DOM. When a component in react updates initially it will get updated in the virtual DOM then React will compare the virtual DOM and the actual DOM and updates only the component that changed on the actual Dom.

By this process react reduces the loading time of the web app and make the application faster.

Answer 3-> React

Life Cycle Method

Component will Mount () -> render when react component loads for the first time.

Component did Mount () -> render state of the component gets updated in the react.

Component will unmount () -> render when state of the component get removed.

Answer 4 -> React

Functional Components are the plain JS functions which return JSX whereas Class Components are JS classes which extends classes.

Answer 5-> React

React hooks are simple JS functions that we can use to isolate the reusable part from a function components .

Different hooks present in react are: -

useState, useEffect, useCallback, useMemo, useRef,useReducer,useSelector, useDispatch, useContext etc

No, we cannot use hooks in the class components can be only used in function component.

Answer 6-> React

Life Cycle Method play very important role in the react application it helps in performing the side effect when state of any component get updated or removed.

Initially before introduction of hooks in react only class-based components were used to perform the life cycle method but after react 16.8 version we can perform all the life cycle method in functional component using use Effect hook only.

Answer 7-> React

**useState**

* Allows you to add and manage state within functional components.
* Simplifies state management by eliminating the need for class components and this.setState().
* Provides a straightforward way to update and access state values.
* Enables you to create multiple state variables in a single component.

Answer 8-> React

**useEffect**

* Enables you to perform side effects in functional components, such as fetching data, subscribing to events, or modifying the DOM.
* Handles the lifecycle of a component by executing code in response to changes or events.
* Offers a clean and concise way to manage asynchronous operations and prevent memory leaks.
* Allows you to specify dependencies, controlling when the effect is triggered.

Answer 9-> React

Context Api is used to create a global state which can be used in the entire application and accessible to any part of the component. It prevents prop-drilling problem in the react.

Answer 10-> React

**use Reducer**

* Provides a way to manage complex state logic by using a reducer function similar to Redux.
* Offers a predictable way to update and handle state changes, especially when dealing with multiple related values.
* Simplifies state updates by providing a dispatch function that triggers actions to update the state.
* Enables you to encapsulate state and related logic in reusable custom hooks.

Answer 11-> React

Todo app using useReducer

GitHub -> https://github.com/krishna5867/placement-assignments/tree/main/todo

Answer 12-> React

React Counter App

GitHub -> https://github.com/krishna5867/placement-assignments/tree/main/React%20Counter%20App%20%2B%20Dark%20Mode

Answer 13-> React

React Calculator

GitHub -> https://github.com/krishna5867/placement-assignments/tree/main/React%20Calculator

Answer 14-> React

Tic Tac Toe Game using Class Based Component

GitHub -> https://github.com/krishna5867/placement-assignments/tree/main/tictactoe

Answer 15-> React

Prop drilling, also known as the "prop chain" refers to the process of passing props through multiple layers of components in a React application, even when intermediate components do not directly use or need those props. It occurs when a prop needs to be accessed by a deeply nested component that is not directly connected to the parent component that possesses the prop.

Answer 16-> React

Task Manager

GitHub ->

**← Express Questions →**

Answer 1-> Express

Server with /post endpoint to send 20 posts.

GitHub ->

Answer 2-> Express

Middleware is a concept commonly used in software development, particularly in web development frameworks. It refers to a piece of code or a function that sits between the incoming request and the application's main logic, allowing for the modification or handling of the request before it reaches the intended endpoint or route

Answer 3-> Express

Blog App

GitHub ->

Answer 4 -> Express

**Authentication** is the process of verifying the identity of a user or entity. It ensures that the user is who they claim to be before granting access to a system or resource. Authentication is typically performed at the beginning of a user session or when accessing protected resources.

**Authorization** on the other hand, is the process of granting or denying access rights and permissions to authenticated users or entities. Once a user's identity has been authenticated, authorization determines what actions or resources that user is allowed to access

Answer 5-> Express

1. **Import/Export Syntax**: CJS uses require () for importing modules and module**.** Exports or exports for exporting values, while EJS uses import and export statements for importing and exporting.
2. **Synchronous vs. Asynchronous**: CJS modules are loaded synchronously, whereas EJS modules are loaded asynchronously, improving performance in certain scenarios.
3. **Browser Support**: CJS modules are primarily used in server-side environments like Node.js and require bundlers or transpilers for browser support, while EJS modules are natively supported in modern browsers.
4. **Static vs. Dynamic**: CJS modules are dynamically loaded at runtime, while EJS modules are statically loaded at parse time, enabling better tooling support and optimizations.

Answer 6-> Express

JWT stands for JSON Web Token. It is an open standard for securely transmitting information between two parties as a JSON object. JWTs are commonly used for authentication and authorization purposes in web applications and APIs.

GitHub ->

Answer 7-> Express

Hash the password using a strong, cryptographically secure hashing algorithm like bcrypt before storing it to the database.

Answer 8-> Express

Basically, it’s a runtime model present in the JS engine and it continuously keeps on checking the call stack and the call back queue and if it finds any callback function is waiting to be executed either in call stack or micro task queue then event loop will check the callback is either empty or not. If it is empty then it pushes the callback function giving priority to micro task queue first into the call stack queue for its execution.

Answer 9-> Express

Full Stack E-commerce App

GitHub -> https://github.com/krishna5867/Full-Stack-E-commerce-App-MERN