VIVA QUESTIONS

HTML

1. What is HTML, and what does it stand for?
   - HTML stands for HyperText Markup Language. It is used to structure content on the web.

2. What is the basic structure of an HTML document?
   - The basic structure includes <!DOCTYPE html>, <html>, <head>, and <body> tags.

3. What is the purpose of the <head> section in an HTML document?
   - It contains meta-information, links to stylesheets, scripts, and the title of the document.

4. What is the purpose of the <title> tag?
   - The <title> tag sets the title of the webpage, which appears in the browser tab.

5. What is an HTML element?
   - An HTML element is a component of an HTML document, consisting of an opening tag, content, and a closing tag.

6. What is an attribute in HTML?
   - An attribute provides additional information about an HTML element and is included in the opening tag. It usually consists of a name and a value (e.g., src="image.jpg").

7. What is the difference between block-level and inline elements?
   - Block-level elements (e.g., <div>, <h1>) take up the full width available and start on a new line. Inline elements (e.g., <span>, <a>) only take up as much width as necessary and do not start on a new line.

8. How do you create a hyperlink in HTML?
   - Use the <a> tag with the href attribute to specify the URL. Example: <a href="https://example.com">Visit Example</a>.

9. What does the target="_blank" attribute do in a hyperlink?
   - It opens the linked document in a new tab or window.

10. How do you include an image in an HTML document?
    - Use the <img> tag with the src attribute to specify the image source and the alt attribute for alternative text. Example: <img src="image.jpg" alt="Description">.

11. What is the purpose of the alt attribute in the <img> tag?

- The alt attribute provides alternative text for the image if it cannot be displayed. It also improves accessibility.

12. What is an unordered list, and how is it created?

- An unordered list is a list where items are not numbered. It is created using the <ul> tag, with each item marked by <li> tags. Example:

```
<ul>
  <li>Item 1</li>
  <li>Item 2</li>
</ul>
```

13. How do you create an ordered list in HTML?

- Use the <ol> tag to create an ordered list, with each item marked by <li> tags. Example:

```
<ol>
  <li>First item</li>
  <li>Second item</li>
</ol>
```

14. What is the purpose of the <form> tag in HTML?

- The <form> tag is used to create a form for collecting user input.

15. How do you create a text input field in a form?

- Use the <input> tag with the type="text" attribute. Example: <input type="text" name="username">.

16. What is the <textarea> element used for?

- The <textarea> element is used to create a multi-line text input field.

17. How do you create a dropdown menu in HTML?

- Use the <select> tag with <option> tags for each dropdown item. Example:

```
<select name="options">
  <option value="1">Option 1</option>
  <option value="2">Option 2</option>
</select>
```

18. What is the <button> element used for?

- The <button> element is used to create a clickable button. It can be used within forms or as standalone buttons.

19. How do you create a table in HTML?

- Use the <table> tag with <tr> for rows, <th> for header cells, and <td> for data cells. Example:

```
<table>
  <tr>
    <th>Header 1</th>
    <th>Header 2</th>
  </tr>
  <tr>
    <td>Data 1</td>
    <td>Data 2</td>
  </tr>
</table>
```

20. What is the <caption> tag used for in a table?

- The <caption> tag provides a title or description for the table.

21. What is the purpose of the <meta> tag?

- The <meta> tag provides metadata about the HTML document, such as character encoding and viewport settings.

22. How do you specify the character encoding of an HTML document?

- Use the <meta charset="UTF-8"> tag in the <head> section.

23. What is the <link> tag used for?

- The <link> tag is used to link external resources, such as stylesheets, to the HTML document.

24. How do you include an external CSS file in an HTML document?

- Use the <link> tag with rel="stylesheet" and href attributes. Example: <link rel="stylesheet" href="styles.css">.

25. What is the <script> tag used for in HTML?

- The <script> tag is used to include JavaScript code or link to external JavaScript files.

26. How do you add comments in HTML?

- Use the <!-- comment --> syntax to add comments.

27. What is the <div> element used for?

- The <div> element is used as a container to group and style sections of a webpage.

28. What is the <span> element used for?

- The <span> element is used to group and style inline portions of text or other elements.

29. What are semantic HTML elements?

- Semantic HTML elements clearly describe their content and purpose, such as <header>, <footer>, <article>, and <section>.

30. How do you create a line break in HTML?

- Use the <br> tag to insert a line break.

31. What is the difference between the <strong> and <b> tags?

- The <strong> tag indicates that text is of strong importance, while <b> simply applies bold styling without implying importance.

32. What does the class attribute do in HTML?

- The class attribute assigns one or more class names to an element, which can be used for styling with CSS or targeting with JavaScript.

33. What does the id attribute do in HTML?

The id attribute assigns a unique identifier to an element, which can be used for styling, JavaScript targeting, or linking.

34. How do you create a checkbox in HTML?

- Use the <input> tag with the type="checkbox" attribute. Example: <input type="checkbox" name="accept" value="yes">.

35. How do you create a radio button in HTML?

- Use the <input> tag with the type="radio" attribute. Example: <input type="radio" name="gender" value="male"> Male.

36. What is the <iframe> tag used for?

- The <iframe> tag is used to embed another HTML document within the current document.

37. How do you make a form submit to a specific URL?

- Use the action attribute of the <form> tag to specify the URL where the form data will be sent. Example: <form action="submit.php">.

38. What does the method attribute do in a form?

- The method attribute specifies how the form data should be sent to the server. Common values are GET and POST.

39. How do you make text italic in HTML?

- Use the <i> or <em> tags to make text italic. <em> is semantically preferred as it indicates emphasis.

40. What is the data-* attribute used for in HTML?

- The data-* attribute is used to store custom data that can be accessed using JavaScript. It allows embedding of custom information into HTML elements.

Stylesheets – CSS

1. What is CSS, and what does it stand for?
   - CSS stands for Cascading Style Sheets. It is used to control the presentation, layout, and design of HTML elements on a webpage.

2. How can CSS be included in an HTML document?
   - CSS can be included in three ways:
     - **Inline CSS**: Using the style attribute within HTML elements (e.g., <p style="color: red;">).
     - **Internal CSS**: Using a <style> tag within the <head> section of the HTML document.
     - **External CSS**: Linking to a separate CSS file using a <link> tag in the <head> section.

3. What is the syntax for a CSS rule?
   - A CSS rule consists of a selector and a declaration block. Example:

     ```
     selector {
       property: value;
     }
     ```

     For example:

     ```
     p {
       color: blue;
     }
     ```

4. What is the purpose of the class selector in CSS?
   - The class selector is used to apply styles to elements with a specific class attribute. It is defined with a dot before the class name (e.g., .my-class).

5. How does the id selector differ from the class selector in CSS?
   - The id selector targets a single unique element with a specific id attribute, while the class selector can target multiple elements with the same class. The id selector is defined with a hash symbol (e.g., #my-id), and the class selector is defined with a dot (e.g., .my-class).

6. What is specificity in CSS?
   - Specificity determines which CSS rule is applied when multiple rules match the same element. It is calculated based on the types of selectors used (e.g., inline styles, IDs, classes, and elements).

7. What is the !important declaration in CSS?
   - The !important declaration is used to give a CSS rule higher priority and override other rules. It should be used sparingly. Example: color: red !important;.

8. What are pseudo-classes in CSS?
   - Pseudo-classes are keywords added to selectors that specify a special state of the selected elements. Examples include :hover, :focus, and :nth-child.

9. What are pseudo-elements in CSS?

- Pseudo-elements are used to style specific parts of an element. Examples include ::before, ::after, and ::first-line.

10. What is the box model in CSS?

- The box model describes the rectangular boxes generated for elements in the document tree. It includes content, padding, border, and margin areas.

11. How do you center a block-level element horizontally using CSS?

- Set the margin property to auto and specify a width for the element. Example:

```
.centered {
  width: 50%;
  margin: 0 auto;
}
```

12. what is the difference between padding and margin in CSS?

- Padding is the space between the content of an element and its border. Margin is the space outside the border, separating the element from other elements.

13. How do you apply a style to multiple elements using CSS?

- List the elements separated by commas in the selector. Example:

```
h1, h2, h3 {
  color: green;
}
```

14. What is the display property in CSS, and what are its common values?

- The display property specifies how an element is displayed on the page. Common values include block, inline, inline-block, none, and flex.

15. How do you create a responsive design using CSS?

- Use media queries to apply different styles for different screen sizes and devices. Example:

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

16. What is Flexbox in CSS?

- Flexbox is a layout model that allows you to design complex layouts with flexible and responsive arrangements. It uses properties like display: flex, justify-content, align-items, and flex-direction.

17. What is CSS Grid Layout?

- CSS Grid Layout is a two-dimensional layout system for the web that allows you to create complex grid-based designs with rows and columns. It uses properties like display: grid, grid-template-rows, and grid-template-columns.

18. What does the float property do in CSS?

- The float property is used to position an element to the left or right within its containing block, allowing other content to wrap around it.

19. What is the position property in CSS, and what are its values?

- The position property specifies how an element is positioned in the document. Common values include static, relative, absolute, fixed, and sticky.

20. How do you use the z-index property in CSS?

- The z-index property specifies the stack order of elements. Elements with a higher z-index value are displayed in front of elements with a lower value. It only works on positioned elements (position set to relative, absolute, fixed, or sticky).

21. What is the opacity property in CSS?

- The opacity property sets the transparency level of an element, with values ranging from 0 (completely transparent) to 1 (completely opaque).

22. How do you change the font size of an element using CSS?

- Use the font-size property. Example: font-size: 16px;.

23. What is the background shorthand property in CSS?

- The background shorthand property allows you to set multiple background properties in one declaration, such as background-color, background-image, background-repeat, and background-position.

24. What is the purpose of the border-radius property in CSS?

- The border-radius property is used to create rounded corners on elements.

25. How do you apply a gradient background using CSS?

- Use the background-image property with a linear-gradient or radial-gradient function. Example:
  background-image: linear-gradient(to right, red, yellow);

26. What are media queries in CSS?

- Media queries are used to apply different styles based on the characteristics of the device, such as screen width, height, or orientation.

27. How do you set the width of an element to 100% of its parent container?

- Use width: 100%; in the CSS rule for the element.

28. What is the text-align property used for?

- The text-align property is used to set the horizontal alignment of text within an element. Common values include left, right, center, and justify.

29. What is the line-height property in CSS?

- The line-height property specifies the amount of space between lines of text.

30. How do you remove the default underline from links using CSS?

- Use the text-decoration property with the value none. Example: a { text-decoration: none; }.

31. What is the box-shadow property used for?

- The box-shadow property adds shadow effects around an element's frame. Example:

  ```
  box-shadow: 2px 2px 5px grey;
  ```

32. How do you change the font family of an element using CSS?

- Use the font-family property. Example: font-family: Arial, sans-serif;.

33. What is the text-transform property in CSS?

- The text-transform property is used to control the capitalization of text. Values include uppercase, lowercase, and capitalize.

34. How do you use the :hover pseudo-class in CSS?

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

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

35. What is the cursor property in CSS?

- The cursor property specifies the type of cursor to be displayed when hovering over an element. Common values include pointer, default, and wait.

36. How do you apply different styles to even and odd table rows?

- Use the :nth-child pseudo-class with the even or odd keyword. Example:

  ```
  tr:nth-child(even) {
    background-color: #f2f2f2;
  }
  ```

37. What is the visibility property in CSS, and how does it differ from display?

- The visibility property controls whether an element is visible or not, with values visible and hidden. Unlike display: none, visibility: hidden keeps the element in the layout but hides it

PHP

1. What is PHP?
    - PHP (Hypertext Preprocessor) is a widely-used, open-source server-side scripting language designed primarily for web development. It can be embedded into HTML and interacts with databases to generate dynamic content.

2. How does PHP handle form data?
    - PHP handles form data through the $_GET and $_POST superglobals. $_GET is used for retrieving data sent via HTTP GET method, while $_POST is used for data sent via HTTP POST method.

3. What are the different types of variables in PHP?
    - PHP supports several types of variables including:
        - **Integers**: Whole numbers.
        - **Floats**: Decimal numbers.
        - **Strings**: Textual data.
        - **Booleans**: true or false.
        - **Arrays**: Collections of values.
        - **Objects**: Instances of classes.
        - **NULL**: Represents a variable with no value.

4. What is the difference between == and === in PHP?
    - == is the equality operator that checks if two values are equal after type juggling (type conversion). === is the identity operator that checks if two values are equal and of the same type.

5. How do you define a function in PHP?
    - A function in PHP is defined using the function keyword.

```
function myFunction($param) {
   // function body
}
```

6. What is the purpose of the include and require statements in PHP?
    - Both include and require are used to include the content of one PHP file into another. The main difference is that require will produce a fatal error and stop script execution if the file is not found, whereas include will only produce a warning and continue execution.

7. What are PHP superglobals?
    - Superglobals are built-in global arrays in PHP that are always accessible, regardless of scope. Examples include:
        - $_GET, $_POST, $_REQUEST, $_SESSION, $_COOKIE, $_FILES, $_SERVER, $_ENV, $_GLOBALS.

8. How do you connect to a MySQL database in PHP?

- You can use the mysqli or PDO extension to connect to a MySQL database. Example using mysqli:

  ```
  $conn = new mysqli("hostname", "username", "password", "database");
  if ($conn->connect_error) {
     die("Connection failed: " . $conn->connect_error);
  }
  ```

9. How do you execute a SQL query in PHP?

- To execute a SQL query, you use the query method of the mysqli object or exec method of the PDO object.

  ```
  $sql = "SELECT * FROM table";
  $result = $conn->query($sql);

  if ($result->num_rows > 0) {
     while($row = $result->fetch_assoc()) {
        // Process each row
     }
  }
  ```

10. How do you handle errors in PHP?

- PHP provides various methods for error handling, including:
  - **Error reporting functions**: error_reporting(), set_error_handler().
  - **Exception handling**: Using try, catch, and finally blocks to handle exceptions.

  ```
  try {
     // Code that may throw an exception
  } catch (Exception $e) {
     echo 'Caught exception: ',  $e->getMessage(), "\n
  ```

11. What is SQL injection and how can you prevent it?

- SQL injection is a security vulnerability where an attacker can manipulate SQL queries by injecting malicious input. To prevent it, use prepared statements and parameterized queries:

  ```
  $stmt = $conn->prepare("SELECT * FROM users WHERE username = ?");
  $stmt->bind_param("s", $username);
  $stmt->execute();
  ```

12. How do you manage sessions in PHP?

- Sessions are managed using the $_SESSION superglobal. Start a session with session_start() and use $_SESSION to store session data.

  ```
  session_start();
  $_SESSION['username'] = 'JohnDoe';
  ```

13. How do you upload a file in PHP?

- File uploads are handled through the $_FILES superglobal. You use the move_uploaded_file() function to move the uploaded file from the temporary directory to a desired location.

```
if (isset($_FILES['file'])) {
    $file = $_FILES['file'];
    $destination = 'uploads/' . $file['name'];
    move_uploaded_file($file['tmp_name'], $destination);
}
```

14. How do you read and write files in PHP?

- Use functions like fopen(), fread(), fwrite(), and fclose() to handle file operations.

```
$file = fopen("example.txt", "r");
$content = fread($file, filesize("example.txt"));
fclose($file);

$file = fopen("example.txt", "w");
fwrite($file, "New content");
fclose($file);
```

JavaScript

1. What is JavaScript and what is it used for?
   - JavaScript is a high-level, interpreted programming language primarily used to create dynamic and interactive content on websites. It allows for client-side scripting to enhance user experiences.

2. **Explain the difference between `var`, `let`, and `const`.**
   - var is function-scoped and can be redeclared. let is block-scoped and cannot be redeclared within the same block. const is also block-scoped but is used for variables that should not be reassigned after initialization.

3. What are data types in JavaScript?
   - JavaScript has primitive data types like number, string, boolean, undefined, null, symbol, and bigint. There are also non-primitive types like object and array.

4. What is the difference between == and ===?
   - == (loose equality) compares values after performing type coercion if necessary, while === (strict equality) compares both value and type without type coercion.

5. How does prototypal inheritance work in JavaScript?
   - JavaScript uses prototypes for inheritance. Each object has a prototype object from which it can inherit properties and methods. This is achieved through the prototype chain.

6. What is the this keyword in JavaScript?
   - this refers to the context in which a function is executed. It can vary depending on how the function is called: it refers to the global object in non-strict mode, the object that is calling the function in method calls, or it can be explicitly bound using call, apply, or bind.

7. What are JavaScript prototypes and how do you use them?
   - Prototypes are a mechanism by which JavaScript objects inherit features from one another. You can set the prototype of an object using Object.create() or by setting the prototype property of a constructor function.

8. What are callbacks in JavaScript?
   - Callbacks are functions passed as arguments to other functions and are executed after the completion of the other function. They are used to handle asynchronous operations.

9. Explain promises and how they are used.
   - Promises represent the eventual completion (or failure) of an asynchronous operation and its resulting value. They provide a cleaner way to handle asynchronous code compared to callbacks, using methods like then(), catch(), and finally().

10. What is async/await and how does it improve asynchronous code?

- async and await are syntactic sugar built on top of promises, making asynchronous code look and behave more like synchronous code. async functions return a promise, and await pauses the function execution until the promise is resolved.

11. What are JavaScript modules and why are they used?

- JavaScript modules are a way to organize code by exporting and importing functions, objects, or primitives from one file to another. This improves code maintainability and encapsulation. Modules are typically managed using the import and export keywords.

12. Explain event delegation in JavaScript.

- Event delegation is a technique where a single event listener is added to a parent element to manage events for its child elements. This leverages event bubbling and can be more efficient than adding multiple listeners to individual elements.

13. What are the different ways to handle errors in JavaScript?

- Errors can be handled using try, catch, finally, and throw. try contains the code that might throw an error, catch handles the error, and finally executes code regardless of the result. throw is used to manually trigger an error.

AJAX

1. What is AJAX and what are its primary components?

- AJAX stands for Asynchronous JavaScript and XML. It allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. Its primary components include JavaScript, the XMLHttpRequest object, and often XML or JSON for data interchange.

2. How does AJAX improve user experience on web pages?

- AJAX improves user experience by enabling web pages to update content dynamically without needing a full page reload. This results in a smoother, faster, and more responsive user experience.

3. Can you explain the process of making an AJAX request?

- Making an AJAX request typically involves creating an instance of the XMLHttpRequest object, setting up the request (specifying the HTTP method, URL, and whether the request should be asynchronous), sending the request, and handling the server's response in a callback function.

4. What are the key methods and properties of the XMLHttpRequest object?

- Key methods include `open()`, `send()`, `setRequestHeader()`, and `abort()`. Key properties include `readyState`, `status`, and `responseText` or `responseXML`.

5. What is the purpose of the `readyState` property in XMLHttpRequest?
   - The `readyState` property indicates the current state of the XMLHttpRequest object. It can have values from 0 (uninitialized) to 4 (complete), which correspond to different stages of the request lifecycle.

6. How can you handle errors in an AJAX request?
   - Errors can be handled by checking the `status` property of the XMLHttpRequest object in the `onreadystatechange` callback function. You can also use the `onerror` event handler to catch network-level errors.

7. What is JSON, and how is it used with AJAX?
   - JSON (JavaScript Object Notation) is a lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate. In AJAX, JSON is often used as the format for sending and receiving data between the client and server due to its simplicity and ease of use with JavaScript.

8. Explain the difference between synchronous and asynchronous requests in AJAX.
   - In synchronous requests, the browser waits for the server response before continuing with the execution of the script, which can lead to a frozen user interface. In asynchronous requests, the browser can continue to process other tasks while waiting for the server response, providing a more fluid user experience.

9. What are some common uses of AJAX in web development?
   - Common uses include form submission without reloading the page, loading new content dynamically (e.g., infinite scroll), updating specific parts of a page (e.g., live search suggestions), and interacting with APIs to fetch or send data.

10. How does AJAX differ from traditional full-page form submission?
    - Traditional form submission involves sending the entire form data to the server and refreshing the page with the server's response. AJAX allows for sending and receiving data asynchronously without reloading the page, which enables partial updates and a more seamless user experience.

JQUERY

1. What is jQuery, and why is it used in web development?
   - jQuery is a fast, lightweight, and feature-rich JavaScript library designed to simplify HTML document traversal, event handling, animation, and Ajax interactions. It is used to streamline and simplify tasks that would otherwise require more verbose JavaScript code.

2. How do you include jQuery in a web page?
   - jQuery can be included in a web page by linking to a jQuery CDN (Content

Delivery Network) or by downloading and hosting the jQuery file locally. The typical inclusion in HTML is done with a `<script>` tag, either referencing the jQuery library from a CDN or from a local file:

```
<script      src="https://code.jquery.com/jquery-3.6.0.min.js">
</script>
```

or

```
<script src="path/to/your/local/jquery.min.js"></script>
```

3. What is the purpose of the `$` function in jQuery?
   - The `$` function is a shorthand for the `jQuery` function. It is used to select elements, create new jQuery objects, and call jQuery methods.

     For example, `$('#myElement')` selects an element with the ID `myElement`.

4. How do you handle events in jQuery?
   - Events in jQuery are handled using the `.on()` method or its shorthand `.click()`, `.hover()`, etc.

     For example:

     ```
     $('#myButton').on('click', function() {
       alert('Button clicked!');
     });
     ```

     or

     ```
     $('#myButton').click(function() {
       alert('Button clicked!');
     });
     ```

5. **What is the difference between `.hide()` and `.fadeOut()` in jQuery?**
   - `.hide()` immediately hides the selected elements by setting their `display` property to `none`. `.fadeOut()` gradually hides the elements by animating the opacity from visible to hidden, creating a fading effect.

6. How can you make an AJAX request using jQuery?
   - jQuery provides several methods for making AJAX requests, such as `.ajax()`, `.get()`, and `.post()`.

     For example:

     ```
     $.ajax({
       url: 'example.com/data',
       method: 'GET',
       success: function(response) {
         console.log(response);
       },
       error: function(error) {
         console.log('Error:', error);
       }
     });
     ```

7. What is chaining in jQuery, and how does it work?
   - Chaining in jQuery allows you to perform multiple operations on the same jQuery object in a single line of code. It works by returning the jQuery object itself from a method, enabling subsequent methods to be called. For example:

     ```
     $('#myElement').css('color',
     'red').slideUp(2000).slideDown(2000);
     ```

8. How do you traverse the DOM with jQuery?
   - jQuery provides various methods for traversing the DOM, such as `.parent()`,

`.children()`, `.find()`, `.siblings()`, and `.prev()`.

For example:

```
$('#myElement').parent(); // Selects the parent element
$('#myElement').find('.child'); // Selects child elements with
class 'child'
```

9. What is the purpose of the .data() method in jQuery?

- The `.data()` method is used to store and retrieve data associated with DOM elements. It provides a way to attach arbitrary data to elements that can be accessed later.

  For example:

  ```
  $('#myElement').data('key', 'value'); // Store data
  var value = $('#myElement').data('key'); // Retrieve data
  ```

10. How do you handle form submission with jQuery?

- You can handle form submission using the `.submit()` event handler. You can prevent the default form submission behavior and perform actions like AJAX requests:

  ```
  $('#myForm').on('submit', function(event) {
    event.preventDefault(); // Prevent default form submission
    // Perform AJAX request or other actions
  });
  ```