WEB DEVELOPMENT

TEST-2

1) Which HTML attribute is used to define inline styles? **(HTML-Hard)**

A) <style>
/* styles go here */
</style>

B) <inline-style>
/* styles go here */
</inline-style>

C) <css>
/* styles go here */
</css>

D) <style="...">
/* styles go here */
</style>

Correct Answer: D) <style="...">
/* styles go here */
</style>

2) How do you include an external JavaScript file in an HTML document? **(HTML-Hard)**

A) <script type="text/javascript" src="script.js"></script>

B) <javascript src="script.js"></javascript>

C) <javascript src="script.js"></javascript>

D)<script src="script.js"></script>

Correct Answer:A) <script type="text/javascript" src="script.js"></script>

3) .container {

  width: 100%;

  padding-top: 20%;

  margin: 10px;

  border: 1px solid #333;

  box-sizing: border-box;

}

What does this CSS code do? **(CSS-Hard)**

A) Defines a link with a background color that changes on hover.

B) Creates a button with a fixed width and height.

C) Styles a div element with rounded corners.

D) Applies styling to an unordered list.

Correct Answer: A) Defines a link with a background color that changes on hover.

Explanation: The code defines a button class with styling for links. On hover, the background color transitions to create an interactive effect commonly used for buttons.

4) .button {

  display: inline-block;

  padding: 10px 20px;

  margin: 5px;

```
  background-color: #3498db;
  color: #ffffff;
  text-align: center;
  text-decoration: none;
  border-radius: 5px;
  transition: background-color 0.3s ease;
}

.button:hover {
  background-color: #2980b9;
}
```

What does this CSS code do? **(CSS-Hard)**

A) Defines a link with a background color that changes on hover.

B) Creates a button with a fixed width and height.

C) Styles a div element with rounded corners.

D) Applies styling to an unordered list.

Correct Answer: A) Defines a link with a background color that changes on hover.

Explanation: The code defines a button class with styling for links. On hover, the background color transitions to create an interactive effect commonly used for buttons.

5) What is the purpose of the `require` function in Node.js, and how is it used? **(Node-Medium)**

A) The `require` function is used to export modules from a file.

B) The `require` function is used to import external modules or files in Node.js.

C) The `require` function is used to define variables.

D) The `require` function is used to create HTTP servers.

Correct Answer: B) The `require` function is used to import external modules or files in Node.js.

Explanation: The `require` function is fundamental in Node.js for importing external modules or files into your application.

6) const fs = require('fs');

```
const readAndLogFile = (filePath) => {
  fs.readFile(filePath, 'utf8', (err, data) => {
    if (err) {
      console.error('Error reading file:', err);
    } else {
      console.log('File content:', data);
    }
  });
};

readAndLogFile('example.txt');
```

What does this Node.js code do? **(Node-Hard)**

A) Reads the content of a file asynchronously and logs it to the console.

B) Writes content to a file synchronously.

C) Reads the content of a file synchronously and logs it to the console.

D) Creates a new file asynchronously.

Correct Answer: A) Reads the content of a file asynchronously and logs it to the console.

Explanation: The code uses the `fs` (File System) module to read the content of the file 'example.txt' asynchronously and logs it to the console.

7) Explain the difference between process.nextTick() and setImmediate() in Node.js. **(Node-Hard)**

A) Both `process.nextTick()` and `setImmediate()` are used for immediate execution but have different use cases.

B) `process.nextTick()` is used for I/O-related tasks, while `setImmediate()` is used for CPU-intensive tasks.

C) `process.nextTick()` executes before the event loop, and `setImmediate()` executes after the event loop.

D) `setImmediate()` is deprecated, and `process.nextTick()` should be used for immediate execution.

Correct Answer: C) `process.nextTick()` executes before the event loop, and `setImmediate()` executes after the event loop.

Explanation: `process.nextTick()` executes immediately after the current event loop cycle, whereas `setImmediate()` executes in the next iteration of the event loop.

8) What is JSX in React? **(React-Basic)**

A) JavaScript External

B) XML Syntax for JavaScript

C) JavaScript XML

D) JavaScript Extendable

Correct Answer: C) JavaScript XML

Explanation: JSX (JavaScript XML) is a syntax extension for JavaScript recommended by React. It allows you to write HTML elements and components in a syntax similar to XML or HTML within your JavaScript code.

9) In React, what is the purpose of the `setState` method? **(React-Medium)**

A) It is used to define the initial state of a component.

B) It is used to update the state of a component and trigger a re-render.

C) It is used to define the state of child components.

D) It is used to create a static state that doesn't change.

Correct Answer: B) It is used to update the state of a component and trigger a re-render.

Explanation: The `setState` method is used to update the state of a React component, and it triggers a re-render of the component with the updated state.

10) What is the purpose of React Router in a React application? **(React-Medium)**

A) To handle HTTP requests and responses.

B) To manage state and props efficiently.

C) To create navigation and routing in a single-page application.

D) To connect React with a database.

Correct Answer: C) To create navigation and routing in a single-page application.

Explanation:React Router is used for handling navigation and routing in a React application, enabling the creation of single-page applications with different views.

11) ```jsx

```
import React, { Component } from 'react';

class Counter extends Component {
  constructor(props) {
    super(props);
    this.state = {
```

```
      count: 0
    };
  }

  componentDidMount() {
    this.interval = setInterval(() => {
      this.setState((prevState) => ({
        count: prevState.count + 1
      }));
    }, 1000);
  }

  componentWillUnmount() {
    clearInterval(this.interval);
  }

  render() {
    return <div>Count: {this.state.count}</div>;
  }
}

export default Counter;
```

What does this React component do? (React-Hard)

A) It creates a counter that increments every second and displays the count.

B) It creates a static counter that doesn't change.

C) It creates a counter with an interval that increments every 1000 milliseconds.

D) It creates a counter that decrements every second and displays the count.

Correct Answer: A) It creates a counter that increments every second and displays the count.

Explanation: The `componentDidMount` lifecycle method sets up an interval to increment the count every second, and `componentWillUnmount` clears the interval to prevent memory leaks.

12) Explain the concept of React Hooks, and provide an example of one. **(React-Hard)**

A) React Hooks are functions that let you use state and other React features in functional components. Example:

```jsx
import React, { useState } from 'react';

function ExampleComponent() {
  const [count, setCount] = useState(0);

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click me
      </button>
    </div>
  );
}
```

```

B) React Hooks are classes that allow you to manage state in class components.
Example:

```jsx
import React, { Component } from 'react';

class ExampleComponent extends Component {
  constructor(props) {
    super(props);
    this.state = {
      count: 0
    };
  }

  render() {
    return (
      <div>
        <p>You clicked {this.state.count} times</p>
        <button onClick={() => this.setState({ count: this.state.count + 1 })}>
          Click me
        </button>
      </div>
    );
  }
}
```
```

C) React Hooks are used to manipulate the DOM directly. Example:

```jsx
import React, { useEffect } from 'react';

function ExampleComponent() {
  useEffect(() => {
    document.title = 'Updated Title';
  }, []);

  return <div>Example Component</div>;
}
```

D) React Hooks are deprecated and should not be used in modern React applications.

Correct Answer: A) React Hooks are functions that let you use state and other React features in functional components. Example:

Explanation: React Hooks are functions that enable functional components to use state and other React features. In the provided example, the `useState` hook is used to manage state in a functional component.

13) What is the significance of the `ngFor` directive in Angular? **(Angular-Basic)**

A) It is used to handle HTTP requests and responses.

B) It is used to create a loop for iterating over a collection of data in the template.

C) It is used to define the structure of an Angular component.

D) It is used to manage navigation between different components.

Correct Answer: B) It is used to create a loop for iterating over a collection of data in the template.

Explanation: The `ngFor` directive in Angular is employed for iterating over a collection of data, allowing you to dynamically generate HTML elements based on the elements of the collection.

14) What is the purpose of Angular CLI? (Angular- Medium)

A) To manage client-server communication in Angular applications.

B) To provide a command-line interface for building and managing Angular applications.

C) To integrate Angular with server-side languages.

D) To handle authentication and authorization in Angular applications.

Correct Answer: B) To provide a command-line interface for building and managing Angular applications.

Explanation: Angular CLI (Command Line Interface) is a powerful tool that simplifies the process of creating, building, testing, and deploying Angular applications.

15) What is Dependency Injection in Angular and why is it important? **(Angular- Medium)**

A) Dependency Injection is a design pattern that allows components to share data directly.

B) Dependency Injection is a way to manage project dependencies in Angular.

C) Dependency Injection is a process of resolving dependencies and passing them to a dependent object.

D) Dependency Injection is not relevant in Angular.

Correct Answer: C) Dependency Injection is a process of resolving dependencies and passing them to a dependent object.

Explanation:Dependency Injection in Angular is a design pattern where a class receives its dependencies from an external source, rather than creating them itself. It promotes loose coupling and easier testing.

16) import { Component } from '@angular/core';

```
@Component({
  selector: 'app-root',
  template: `
    <div>
      <h1>{{ greeting }}</h1>
      <button (click)="changeGreeting()">Change Greeting</button>
    </div>
  `,
})
export class AppComponent {
  greeting = 'Hello, Angular!';

  changeGreeting() {
    this.greeting = 'Greetings from Angular!';
  }
}
```

What does this Angular component do? **(Angular-Hard)**

A) Displays a button that, when clicked, changes the content of an `<h1>` element.

B) Renders a static greeting message without any interactivity.

C) Fetches data from an API and displays it.

D) Creates a form for user input.

Correct Answer: A) Displays a button that, when clicked, changes the content of an `<h1>` element.

Explanation:The component displays an `<h1>` element with a dynamic greeting. Clicking the "Change Greeting" button triggers the `changeGreeting` method, updating the content dynamically.

17) Explain Angular Change Detection and how it works. (Angular-Hard)

A) Angular Change Detection is a mechanism to detect changes in the DOM and update the model accordingly.

B) Angular Change Detection is a process to detect changes in component state and automatically update the view.

C) Angular Change Detection is not relevant in modern Angular applications.

D) Angular Change Detection is a tool for debugging and profiling Angular applications.

Correct Answer: B) Angular Change Detection is a process to detect changes in component state and automatically update the view.

Explanation: Angular Change Detection is a mechanism that automatically checks for changes in the component's state and updates the view accordingly. It helps to keep the application's user interface in sync with the underlying data.

18) What does the term "REST" stand for in the context of web services? **(Rest_Api-Basic)**

A) Remote Execution of Stateful Transactions

B) Representational State Transfer

C) Real-time Exchange of Server Tasks

D) Responsive Encoding of Server Transactions

Correct Answer: B) Representational State Transfer

Explanation: REST stands for Representational State Transfer. It is an architectural style for designing networked applications, emphasizing a stateless client-server communication model.

19) Explain the difference between PUT and PATCH HTTP methods in a REST API.

(Rest_Api-Medium)

A) PUT is used to update a resource entirely, while PATCH is used to update a resource partially.

B) PUT is used for creating new resources, while PATCH is used for updating existing resources.

C) PUT and PATCH are interchangeable, and there is no significant difference between them.

D) PUT is used for read-only operations, while PATCH is used for write operations.

Correct Answer: A) PUT is used to update a resource entirely, while PATCH is used to update a resource partially.

Explanation: PUT is used to update a resource entirely, replacing the existing resource, while PATCH is used to apply partial modifications to the resource.

20) What is the purpose of Cross-Origin Resource Sharing (CORS) in a RESTful API? **(Rest_Api-Medium)**

A) To allow cookies to be sent across different domains.

B) To restrict access to resources from different origins.

C) To enable or disable caching of resources.

D) To facilitate secure authentication.

Correct Answer: A) To allow cookies to be sent across different domains.

Explanation: CORS enables or restricts web applications running at one origin to request and interact with resources from a different origin. It is crucial for allowing or blocking cross-origin requests, including the exchange of cookies.

21) const express = require('express');

const app = express();

app.get('/api/users', (req, res) => {

  // Code for retrieving a list of users from the database

  // ...

  res.status(200).json(users);

});

app.listen(3000, () => {

  console.log('Server is running on port 3000');

});

```

What does this code do? **(Rest_Api-Hard)**

A) Defines an API endpoint for creating users.

B) Defines an API endpoint for retrieving a list of users.

C) Defines a middleware for authentication.

D) Defines an API endpoint for updating user information.

**Correct Answer: B) Defines an API endpoint for retrieving a list of users.**

**Explanation: The code defines a route for handling GET requests to '/api/users', retrieving a list of users from the database and responding with JSON data**.
```

22) What is the purpose of an ETag in the context of a RESTful API? **(Rest_Api- Hard)**

A) To specify the encoding type of a resource.

B) To provide a unique identifier for a resource that changes whenever the resource is modified.

C) To authenticate requests for secured resources.

D) To define the expiration time of a cached resource.

Correct Answer: B) To provide a unique identifier for a resource that changes whenever the resource is modified.

Explanation: An ETag (Entity Tag) is a unique identifier assigned to a resource. It changes whenever the resource is modified, allowing clients to check if the resource has been updated.

23) What is the purpose of the `express.json()` middleware in an Express application? **(Express-Basic)**

A) To serve static files.

B) To parse incoming JSON requests.

C) To handle session management.

D) To set up route parameters.

Correct Answer: B) To parse incoming JSON requests.

Explanation: The `express.json()` middleware in Express is used to parse incoming JSON requests, making the data available in the `request.body` object.

24) How does middleware work in Express.js, and why is it important? **(Express-Medium)**

A) Middleware is used for handling database operations in Express.

B) Middleware is a mechanism to define routes in Express.

C) Middleware functions are executed in a sequential order, allowing developers to modify request and response objects. It is crucial for adding functionality to the request-response cycle.

D) Middleware is only relevant for error handling in Express.

Correct Answer: C) Middleware functions are executed in a sequential order, allowing developers to modify request and response objects. It is crucial for adding functionality to the request-response cycle.

Explanation: Middleware functions in Express are executed sequentially in the order they are defined. They can modify the request and response objects, adding various functionalities to the request-response cycle.

25) const express = require('express');

```
const app = express();


app.use((req, res, next) => {
  console.log('Middleware 1');
  next();
});


app.use((req, res, next) => {
  console.log('Middleware 2');
  next();
});


app.get('/', (req, res) => {
  res.send('Hello, Express!');
});
```

```
app.listen(3000, () => {
  console.log('Server is running on port 3000');
});
```

What will be logged to the console when a request is made to the root path '/'?
(Express-Hard)

A) "Middleware 1"

B) "Middleware 2"

C) "Middleware 1", "Middleware 2"

D) No log messages will be printed.

Correct Answer: C) "Middleware 1", "Middleware 2"Explanation: The two middleware functions are executed in the order they are defined (`Middleware 1` followed by `Middleware 2`). The `next()` function is called to proceed to the next middleware in the chain.