RAJIV GANDHI INSTITUTE OF PETROLEUM TECHNOLOGY

B.Tech. 2nd Year

MID-SEM EXAMINATION, 2023-24

INSTRUCTOR: Dr. GARGI SRIVASTAVA

**SUBJECT**: Web Technology **COURSE NUMBER**: CS222

**DATE & TIME**: 16.03.2024 01:00 P.M. – 03:00 P.M. **FULL MARKS**: 20

**INSTRUCTIONS**:

i. There are 16 questions in this question paper in 3 pages.

ii. You have to attempt all the questions.

iii. All questions carry equal marks.

iv. Attempt the question in serial order (if possible).

v. Make suitable assumptions wherever required and mention them in your answer.

**Q1**. Define Servers. What are the key functions and responsibilities of a server?

**Ans:** A server is a computer or software system that provides services or resources to other computers, known as clients, over a network.

**Key functions and responsibilities of a server:**

- Servers host and deliver web content to users.
- Servers store web content, including HTML documents, images, stylesheets, scripts, and multimedia files.
- Web content is organized in a file structure on servers, and each file is identified by a unique address, usually a URL (Uniform Resource Locator).
- Servers respond to requests from clients, typically web browsers, by providing the requested resources.
- For dynamic websites, servers execute server-side scripts or applications to generate content based on user requests.

**Q2.** What are the key roles and functions of web browsers?

- Web browsers interpret and render HTML (Hypertext Markup Language), CSS (Cascading Style Sheets), and JavaScript to display web pages as intended by web developers.
- Browsers provide a user-friendly interface, allowing users to interact with and navigate through web content.
- Browsers initiate HTTP (Hypertext Transfer Protocol) requests to servers to retrieve web resources such as HTML files, images, stylesheets, and scripts.
- Browsers implement security features like phishing protection, pop-up blockers, and warning messages for potentially harmful websites.
- Browsers maintain a history of visited websites, allowing users to revisit pages and track their online activity.
- Browsers support extensions and add-ons that enhance functionality and allow users to customize their browsing experience with features like ad blockers, password managers, and developer tools.

---

Q3. Write an HTML code to create a button and input box. Both of them should be disabled.

```
<button disabled>Button</button>
<input disabled>
```

Q4. Write the code for the following expected browser view:

| Header | |
|---|---|
| Main Content | Sidebar |
| Footer | |

Use a grid display. The background for Header and Footer should be black, the background for Sidebar should be gray.

```
<head>
```

```
  <style>
    header, footer{
      background-color: black;
      color: white;
    }
    aside{
      background-color: gray;
    }
    .container{
      display: grid;
      grid-template-columns: 2fr 1fr;
    }
  </style>
</head>
<body>
  <header>
    Header
  </header>
  <div class="container">
    <main>
      Main Content
    </main>
    <aside>
      Sidebar
    </aside>
  </div>
  <footer>
    Footer
  </footer>
</body>
```

**Q5.** Create two elements paragraph and span belonging to the same class and initialized by appropriate text content. Use the method 'getElementsByClassName' to display the text content of the first element in the log.

```
<p class="myClass">Paragraph 1</p>
<span class="myClass">Span 2</span>
<script>
let elementsByClass = document.getElementsByClassName("myClass");
// Access the first element:
console.log(elementsByClass[0].textContent); // Output: "Paragraph 1"
</script>
```

**Q6**. Write a Javascript code to press any key and see its name in the console.

```
<body>
 <h1>Press any key to see its name in the console!</h1>
<script>
document.addEventListener("keydown", function(event) {
 console.log("Key pressed:", event.key);
});
</script>
</body>
```

---

**Q7.** Explain the publish and subscribe design pattern.

- Publisher: This is an entity that produces messages or events. It does not need to know who is interested in these messages. When an event occurs, the publisher sends it to the message broker.
- Message Broker: This is a component that receives messages from publishers and routes them to the appropriate subscribers based on predefined criteria, such as message type or topic.

- Subscriber: This is an entity that expresses interest in receiving messages of a particular type or from a specific publisher. When a subscriber subscribes to a topic, it receives all messages published to that topic.

**Q8.** What is JSON? Explain the working of the primary methods of JSON objects with examples.

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.

The primary methods of the JSON object are stringify and parse.

The stringify method takes an object as a parameter and returns a JSON string.

JSON.stringify([1,2,3.1415, 4])

Type: string

"[1, 2, 3.1415, 4]"

JSON.parse(JSON.stringify([1,2,3.1415, 4]))

Type: object

[1,2,3.1415, 4]

---

**Q9**. What are query string parameters? Give an example. Write a small python code to extract the hostname and pathname of any input URL.

The query string is used to send data from the browser to the server when an HTTP GET request is made. As the name suggests it is a special string that is appended to the end of the URL.

https://example.com/api/getdata?id=1234&date=now&apikey=1234567

```
function query_string_parser(qs){
const url=new URL(qs)
console.log(url.hostname)
console.log(url.pathname)
}
query_string_parser('https://example.com/api/getdata?id=1234&date=now&apikey=1234567')
```

**Q10.** Write the output of this code:

```
function Person(name, age) {
 this.name = name;
 this.age = age;
}
Person.prototype.sayHello = function() {
 console.log("Hello, my name is " + this.name);
};
let person1 = new Person("Alice", 30);
let person2 = new Person("Bob", 25);
person1.sayHello();
person2.sayHello();
```

Modify the above code to add the following:

- An **addFriend function** which allows each instance to add friends to an array. Note: The list of friends should be separate for each instance.

```
// Constructor function
function Person(name, age) {
 // Instance members
 this.name = name;
 this.age = age;
 this.friends = []; // An instance member that will be unique to each instance
}
// Prototype member - shared among all instances
Person.prototype.addFriend = function(friendName) {
 this.friends.push(friendName);
};
```

- A variable **location** specific only to the instance **person1**.

```
// Instance member - unique to each instance
```

```
person1.location = "New York"
```

What changes would you make if you want both the instances to have a common friend list?

```
// Prototype member - shared among all instances
Person.prototype.friends = [];
```

---

**Q11.** Design the following page using React.

---

**Priyanka Chopra**

---

```
const user={
name: 'Priyanka Chopra',
imageUrl:
'https://upload.wikimedia.org/wikipedia/commons/thumb/6/6c/Priyanka-chopra-gesf-2018
-7565.jpg/640px-Priyanka-chopragesf-2018-7565.jpg',
imageSize:90,
};
export default function Profile0{
return(
<>
```

```
<h1>{user.name}</h1>
<img
className="avatar"
src={user.imageUrl}
alt={'Photo of '+ user.name}
style={{
width:user.imageSize,
height:user.imageSize
}}
/>
</>
)
}
```

```
.avatar {
 border-radius: 50%;
}
```

Q12. A web page contains two buttons: one plays a movie and another uploads an image. Pass the movie name (for the first button) and the content of the alert box (for both the buttons) as props for event handlers. The page should be completely designed in React.

```
function Button({ onClick, children }) {
 return (
 <button onClick={onClick}>
 {children}
 </button>
 );
}
function PlayButton({ movieName }) {
 function handlePlayClick() {
 alert(`Playing ${movieName}!`);
 }
```

```
function UploadButton() {
 return (
 <Button onClick={() =>
alert('Uploading!')}>
 Upload Image
 </Button>
 );
}
export default function Toolbar() {
 return (
 <div>
```

```
 return (
 <Button onClick={handlePlayClick}>
 Play "{movieName}"
 </Button>
 );
}
```

```
 <PlayButton movieName="Kiki's
Delivery Service" />
 <UploadButton />
 </div>
 );
}
```

---

**Q13.** Implement a simple server using **Node and Express** which simply responds with “Hello World” on the homepage.

```
const express = require(‘express’);
const app = express();
app.get(‘/’, (req, res)=>{
res.send(‘Hello World’);
});
app.listen(3000, ()=>console.log(‘Listening on port 3000...’));
```

**Q14.** Implement a simple server using **only Node** which simply responds with “Hello World” on the homepage.

```
const http = require('node:http');
const hostname = '127.0.0.1';
const port = 3000;
const server = http.createServer((req, res) => {
 res.statusCode = 200;
 res.setHeader('Content-Type', 'text/plain');
 res.send('Hello World\n');
});
server.listen(port, hostname, () => {
 console.log(`Server running at http://${hostname}:${port}/`);
```

```
});
```

---

**Q15.** Make a webpage that uses a table to create two columns. In the left column should be a bulleted list of 5 countries. In the right column should be a numbered list of capitals of those countries.

```
<table>
  <tr>
    <td>
      <ul>
        <li>United States</li>
        <li>France</li>
        <li>Japan</li>
        <li>India</li>
        <li>Australia</li>
      </ul>
    </td>
    <td>
      <ol>
        <li>Washington D.C.</li>
        <li>Paris</li>
        <li>Tokyo</li>
        <li>New Delhi</li>
        <li>Canberra</li>
      </ol>
    </td>
  </tr>
</table>
```

**Q16.** Using the nav element, make a navbar for at the top of a sample webpage. Add at least three links to the navbar using an unordered list that is displayed as inline rather than

block. Make the navbar have a light blue background color, but change the background of the title to a darker blue when the mouse hovers over it.

```
<head>
<style>
nav {
	background-color: lightblue;
	}
nav ul li {
	display: inline;
	 }
nav ul li a {
	text-decoration: none;
}
nav ul li a:hover {
	background-color: darkblue;
   }
</style>
</head>
<nav>
   <ul>
	<li><a href="#">Home</a></li>
	<li><a href="#">About</a></li>
	<li><a href="#">Contact</a></li>
   </ul>
</nav>
```

---