WEB COMPUTING PRACTICAL EXAM SOLUTIONS

# Q1: Asynchronous File Reading Node.js Program

const fs = require('fs');  
  
fs.readFile('example.txt', 'utf8', (err, data) => {  
 if (err) {  
 console.error('Error reading file:', err);  
 return;  
 }  
 console.log('File content:', data);  
});

# Q2: React JSX with Greeting Message Example

import React from 'react';  
import ReactDOM from 'react-dom';  
  
const element = <h1>Hello! Welcome to React</h1>;  
ReactDOM.render(element, document.getElementById('root'));

# Q3: JavaScript to Check if a Number is Even

let number = prompt("Enter a number:");  
if (number % 2 === 0) {  
 alert("The number is even.");  
} else {  
 alert("The number is odd.");  
}

# Q4: Creating a Simple Text File in Node.js

const fs = require('fs');  
const content = "This is the content of the file.";  
  
fs.writeFile('output.txt', content, err => {  
 if (err) {  
 console.error('Error writing file', err);  
 } else {  
 console.log('File written successfully');  
 }  
});

# Q5: React useState Hook to Count Button Clicks

import React, { useState } from 'react';  
  
function ClickCounter() {  
 const [count, setCount] = useState(0);  
  
 return (  
 <div>  
 <p>You clicked {count} times</p>  
 <button onClick={() => setCount(count + 1)}>  
 CLICK  
 </button>  
 </div>  
 );  
}  
  
export default ClickCounter;

# Q6: React Class Component with Constructor to Update State

import React, { Component } from 'react';  
  
class Car extends Component {  
 constructor(props) {  
 super(props);  
 this.state = { brand: "Ford" };  
 }  
  
 render() {  
 return <h1>This car is a {this.state.brand}</h1>;  
 }  
}  
  
export default Car;

# Q7: React Class Component Without Constructor to Update State

import React, { Component } from 'react';  
  
class Car extends Component {  
 state = { brand: "Tesla" };  
  
 render() {  
 return <h1>This car is a {this.state.brand}</h1>;  
 }  
}  
  
export default Car;

# Q8: React Functional Component to Update State with Props

import React, { useState } from 'react';  
  
function Car(props) {  
 const [brand, setBrand] = useState(props.brand);  
  
 return <h1>This car is a {brand}</h1>;  
}  
  
export default Car;

# Q9: React Functional Component to Update State with Props

import React, { useState } from 'react';  
  
function Bike(props) {  
 const [model, setModel] = useState(props.model);  
  
 return <h1>This bike is a {model}</h1>;  
}  
  
export default Bike;

# Q10: React JSX Example Rendering "Hello! Welcome to React"

import React from 'react';  
import ReactDOM from 'react-dom';  
  
const element = <h1>Hello! Welcome to React</h1>;  
ReactDOM.render(element, document.getElementById('root'));

# Q11: React JSX Example Rendering "Hello! Welcome to React"

import React from 'react';  
import ReactDOM from 'react-dom';  
  
const element = <h1>Hello! Welcome to React</h1>;  
ReactDOM.render(element, document.getElementById('root'));

# Q12: Create a React Class Component "Car" and Invoke it in index.js

import React, { Component } from 'react';  
  
class Car extends Component {  
 render() {  
 return <h2>I am a Car!</h2>;  
 }  
}  
  
export default Car;  
  
// index.js  
import React from 'react';  
import ReactDOM from 'react-dom';  
import Car from './Car';  
  
ReactDOM.render(<Car />, document.getElementById('root'));

# Q13: React Hooks Example with Color Buttons

import React, { useState } from 'react';  
  
function ColorSelector() {  
 const [color, setColor] = useState('');  
  
 return (  
 <div>  
 <button onClick={() => setColor('Red')}>Red</button>  
 <button onClick={() => setColor('Blue')}>Blue</button>  
 <button onClick={() => setColor('Green')}>Green</button>  
 <button onClick={() => setColor('Yellow')}>Yellow</button>  
 <p>You selected {color}</p>  
 </div>  
 );  
}  
  
export default ColorSelector;

# Q14: JavaScript Digital Clock on Web Page

function showTime() {  
 const date = new Date();  
 const time = date.toLocaleTimeString();  
 document.getElementById('clock').innerText = time;  
}  
setInterval(showTime, 1000);

# Q15: JavaScript Program with Radio Buttons Changing Background Color

function changeBackground(color) {  
 document.body.style.backgroundColor = color;  
}  
  
document.querySelectorAll('input[name="color"]').forEach(radio => {  
 radio.addEventListener('change', function() {  
 changeBackground(this.value);  
 });  
});

# Q16: Web Page with Moving Image (Left to Right)

let img = document.getElementById("image");  
let position = 0;  
let interval = setInterval(moveImage, 10);  
  
function moveImage() {  
 if (position >= window.innerWidth) {  
 position = 0;  
 } else {  
 position += 1;  
 img.style.left = position + "px";  
 }  
}

# Q17: JavaScript Code to Change Background Color Based on Button Click

function changeColor(color) {  
 document.body.style.backgroundColor = color;  
}  
  
document.getElementById('redButton').addEventListener('click', () => changeColor('red'));  
document.getElementById('greenButton').addEventListener('click', () => changeColor('green'));

# Q18: JavaScript Code to Set a Cookie

function setCookie(name, value, days) {  
 let expires = "";  
 if (days) {  
 const date = new Date();  
 date.setTime(date.getTime() + (days \* 24 \* 60 \* 60 \* 1000));  
 expires = "; expires=" + date.toUTCString();  
 }  
 document.cookie = name + "=" + (value || "") + expires + "; path=/";  
}

# Q19: JavaScript Code to Accept a Number and Check if it is Even

let number = prompt("Enter a number:");  
if (number % 2 === 0) {  
 alert("The number is even.");  
} else {  
 alert("The number is odd.");  
}

# Q20: JavaScript Code to Accept Two Numbers and Display Their Sum in Popup

let num1 = parseInt(prompt("Enter the first number:"));  
let num2 = parseInt(prompt("Enter the second number:"));  
alert("The sum is: " + (num1 + num2));

# Q21: JavaScript Program to Change Background Color by Refreshing Every 2 Seconds

let colors = ["red", "blue", "green", "yellow"];  
let i = 0;  
  
function changeColor() {  
 document.body.style.backgroundColor = colors[i];  
 i = (i + 1) % colors.length;  
}  
  
setInterval(changeColor, 2000);

# Q22: JavaScript Code to Process Online Alumni Information and Form Validation

// Function to validate form  
function validateForm() {  
 const name = document.forms["alumniForm"]["name"].value;  
 const dob = document.forms["alumniForm"]["dob"].value;  
 const email = document.forms["alumniForm"]["email"].value;  
 const password = document.forms["alumniForm"]["password"].value;  
  
 // Check if all fields are filled  
 if (!name || !dob || !email || !password) {  
 alert("All fields must be filled out");  
 return false;  
 }  
  
 // Validate email format  
 const emailPattern = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;  
 if (!emailPattern.test(email)) {  
 alert("Invalid email format");  
 return false;  
 }  
  
 // Validate age (>= 22 years)  
 const currentYear = new Date().getFullYear();  
 const birthYear = new Date(dob).getFullYear();  
 const age = currentYear - birthYear;  
 if (age < 22) {  
 alert("Age must be 22 years or older");  
 return false;  
 }  
  
 // Validate password length  
 if (password.length < 6) {  
 alert("Password must be at least 6 characters long");  
 return false;  
 }  
  
 return true;  
}