**INP Practical Problem Statements**

1. Design a Registration Form for Covid vaccine using HTML5 and inline/external/internal CSS.

| <!DOCTYPE html>  <html>  <head>  <title>COVID-19 Vaccine Registration</title>    <style>  /\* Google Font \*/  @import url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600&display=swap');  body {  font-family: 'Poppins', sans-serif;  background-color: #f2f2f2;  }    .container {  max-width: 600px;  margin: 2rem auto;  padding: 2rem;  background-color: #fff;  border-radius: 10px;  box-shadow: 0 10px 20px rgba(0,0,0,0.1);  }    h1 {  font-size: 1.5rem;  color: #1b468c;  text-align: center;  }  label {  font-size: 1.1rem;  font-weight: 500;  }  input, select {  width: 100%;  padding: 0.8rem;  border-radius: 5px;  border: 1px solid #ccc;  outline: none;  }  input[type="submit"] {  display: block;  width: 100%;  padding: 1rem;  margin-top: 2rem;  background-color: #1b468c;  color: #fff;  border: none;  border-radius: 5px;  font-weight: 600;  cursor: pointer;  }  input[type="submit"]:hover {  background-color: #143366;  }  </style>    </head>  <body>  <div class="container">  <h1>COVID-19 Vaccine Registration</h1>  <form>    <label for="fullname">Full Name:</label>  <input type="text" id="fullname" name="fullname" required>  <label for="email">Email:</label>  <input type="email" id="email" name="email" required>  <label for="dob">Date of Birth:</label>  <input type="date" id="dob" name="dob" required>    <label for="address">Address:</label>  <input type="text" id="address" name="address" required>  <label for="pincode">Pincode:</label>  <input type="text" id="pincode" name="pincode" required>  <label for="city">City:</label>  <input type="text" id="city" name="city" required>  <label for="state">State:</label>  <input type="text" id="state" name="state" required>    <label>Gender:</label>  <div>  <input type="radio" id="male" name="gender" value="male">  <label for="male">Male</label>  <input type="radio" id="female" name="gender" value="female">  <label for="female">Female</label>  </div>  <label for="mobile">Mobile Number:</label>  <input type="tel" id="mobile" name="mobile" required>  <label for="vaccine">Vaccine:</label>  <select id="vaccine" name="vaccine">  <option value="pfizer">Pfizer</option>  <option value="moderna">Moderna</option>  <option value="jj">Johnson & Johnson</option>  <option value="astrazeneca">AstraZeneca</option>  </select>  <input type="submit" value="Register">    </form>  </div>    </body>  </html> |
| --- |

1. Design a Course Registration Form using HTML5 and inline/external/internal CSS.

| <!DOCTYPE html>  <html>  <head>  <title>Course Registration</title>  <style>  /\* Google font \*/  @import url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600&display=swap');  body {  font-family: 'Poppins', sans-serif;  background: #f1f1f1;  }    .container {  max-width: 600px;  margin: 2rem auto;  padding: 2rem;  background: #fff;  border-radius: 10px;  box-shadow: 0 10px 30px rgba(0,0,0,0.1);  }  h1 {  font-size: 1.5rem;  text-align: center;  margin-bottom: 1rem;  }  label {  font-size: 1.1rem;  margin-bottom: 0.5rem;  display: block;  }  input, select {  width: 100%;  padding: 0.8rem;  border-radius: 5px;  border: 1px solid #ccc;  }  input[type="radio"] {  margin-right: 10px;  }  input[type="submit"] {  display: block;  width: 100%;  margin-top: 1rem;  padding: 1rem;  background: #1b468c;  color: #fff;  border: none;  border-radius: 5px;  font-weight: 600;  cursor: pointer;  }  input[type="submit"]:hover {  background: #0e385f;  }  </style>    </head>  <body>  <div class="container">  <h1>Course Registration</h1>  <form>    <label for="firstName">First Name:</label>  <input type="text" id="firstName" name="firstName">  <label for="lastName">Last Name:</label>  <input type="text" id="lastName" name="lastName">  <label for="age">Age:</label>  <input type="number" id="age" name="age">  <label>Gender:</label>  <input type="radio" id="male" name="gender" value="male">  <label for="male">Male</label>  <input type="radio" id="female" name="gender" value="female">  <label for="female">Female</label>  <label for="mobile">Mobile Number:</label>  <input type="tel" id="mobile" name="mobile">  <label for="address">Address:</label>  <input type="text" id="address" name="address">  <label for="city">City:</label>  <input type="text" id="city" name="city">    <label for="state">State:</label>  <input type="text" id="state" name="state">  <label for="country">Country:</label>  <input type="text" id="country" name="country">  <label for="email">Email:</label>  <input type="email" id="email" name="email">  <label for="course">Course:</label>  <select id="course" name="course">  <option value="math">Mathematics</option>  <option value="history">History</option>  <option value="cs">Computer Science</option>  <option value="bio">Biology</option>  </select>  <label for="regDate">Registration Date:</label>  <input type="date" id="regDate" name="regDate">  <input type="submit" value="Register">  </form>  </div>  </body>  </html> |
| --- |

1. Design a personal profile using HTML5 and internal/external/inline CSS.

| <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">  <title>My Resume</title>  </head>    <body style="background-color: beige;">  <div>  <h1 style="text-align: center;">EXP 1</h1>  </div>    <div style="margin: 20px;">  <div style="display: flex; align-items: center; margin-bottom: 20px;">  <img src="PROFILE.png" alt="Profile picture" style="height: 200px;">  <div>  <h2 style="margin: 0;">Sherwin Samuel Rienzo</h2>  <p style="margin-top: 0; font-style: italic;">"If you want to lift yourself up, lift someone else." - Booker T. Washington</p>  </div>  </div>    <div style="margin-bottom: 20px;">  <h3>Address:</h3>  <address>  <b>103, Shradhha Tower, Shanti Park, Mira Road (E), Thane-401107</b><br>  Phone: +91 7977029766<br>  Email: <a href="mailto:sherwin.rienzo@student.sfit.ac.in">sherwin.rienzo@student.sfit.ac.in</a>  </address>  </div>    <div style="margin-bottom: 20px;">  <h3>Academic Qualifications:</h3>  <table style="width: 100%; border-collapse: collapse;">  <thead>  <tr>  <th style="padding: 10px; text-align: center; border: 1px solid black; background-color: #dae063;">Serial No.</th>  <th style="padding: 10px; text-align: center; border: 1px solid black; background-color: rgb(98, 203, 112);">Exams</th>  <th style="padding: 10px; text-align: center; border: 1px solid black; background-color: #109dcc;">Percentage/Pointer</th>  </tr>  </thead>  <tbody>  <tr>  <td style="padding: 10px; text-align: center; border: 1px solid black;">1</td>  <td style="padding: 10px; text-align: center; border: 1px solid black;">SSC</td>  <td style="padding: 10px; text-align: center; border: 1px solid black;">60%</td>  </tr>  <tr>  <td style="padding: 10px; text-align: center; border: 1px solid black;">2</td>  <td style="padding: 10px; text-align: center; border: 1px solid black;">Diploma (EXTC)</td>  <td style="padding: 10px; text-align: center; border: 1px solid black;">87%</td>  </tr>  <tr>  <td style="padding: 10px; text-align: center; border: 1px solid black;">3</td>  <td style="padding: 10px; text-align: center; border: 1px solid black;">BE (3rd SEM)</td>  <td style="padding: 10px; text-align: center; border: 1px solid black;">7</td>  </tr>  </tbody>  </table>  </div>    <div style="margin-bottom: 20px;">  <h3>Skills:</h3>  <ul>  <li>Technical Support Freelancing</li>  <li>Freelancing</li>  <li>CCTV and Computer Repairs</li>  <li>Python Programming</li>  </ul>  </div>    <div style="margin-bottom: 20px;">  <h3>Hobbies:</h3>  <ol>  <li>Reading</li>  <li>Cycling</li>  <li>Cooking</li>  <li>Weightlifting</li>  </ol>  </div>    <div style="margin-bottom: 20px;">  <h3>Project:</h3>  <iframe width="400" height="300" src="https://www.youtube.com/embed/fKJVxItLiUw" title="Complete MERN Stack Developer Roadmap For Beginners (2023)" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share" allowfullscreen></iframe>  </div>    <div style="margin-bottom: 20px;">  <h3>Certifications:</h3>  <ul>  <li><img src="C:\Users\Student\Desktop\SHERWIN\CF1.jpg" alt="Certification 1" height="200"width="200"></li>  <li><img src="C:\Users\Student\Desktop\SHERWIN\CF2.jpg" alt="Certification 2"height="200"width="200"></li>  </ul>  </div>    <div style="margin-bottom: 20px;">  <h3>Achievements:</h3>  <ol>  <li>First place in an inter-college Hackathon competition</li>  <li>Published research paper in college</li>  </ol>  </div>  </div>  <h2>Reference</h2>  <p>MuscleWiki LLC. "Muscle Wiki." <em>MuscleWiki</em>, <a href="https://musclewiki.com/">https://musclewiki.com/</a>.</p>      <div style="background-color: #ddd; padding: 20px; text-align: center;">  <p>&copy; 2023 Sherwin Samuel Rienzo. All rights reserved.</p>  </div>  </body>  </html> |
| --- |

1. Design a homepage of National Park website using HTML5 and CSS. Show logo of National park, Quote related to travelling, History of SGNP and Gallery of images. Design at least one additional page to show the information about spots to be visited. Make use of anchor tag to show the image click event

| <html>  <head>  <style *type*="text/css">  *.head*{  display:flex;  justify-content: space-around;  }  *.Name*{  color:red;  background-color: lightgrey;  width: 900px;  border: 15px solid #5B9BD5;  padding: 5px;  margin: 10px;  height: 50px;  */\* margin-top: 10%; \*/*  text-align:center;  background-position: center;  }  *.About*{  background-color: #C5E0B4;  text-align: center;  padding: 10px;  }  *.one*, *.two*, *.three*, *.four*,*.five*{  border: 2px solid #F8CBAD;  background-color: #F8CBAD;  margin: 20px;  padding: 08px;  height: 400px;  width: 100px;  text-align: center;  }  *.image-section* {  display: flex;  }  *.image-container* {  display: flex;  flex-direction: column;  gap: 20px;  max-width: 900px;  height: 700px;  background-color: #B4C7E7;  justify-content: center;  align-items: center;  }  *.section2* {  display: inline;  width: 1120px;  height: 700px;  background-color: rgb(6, 85, 55);  }  *.exp*{  background-color: #ffff;  height: 40px;  width: 600px;  font-size: 25px;  text-align: center;  margin-left: 250px;  }  *.section* img {  height: 250px;  }  *.head* img{  padding: 20px;  height: 100px;  }  *.slogan*{  background-color: #F8CBAD;  text-align: left;  padding: 10px;  }  *.section2* *.image*{  width: 100px;  height: 50px;  }  *.imgs*{  display: grid;  grid-template-columns: auto auto auto;  align-items: center;  justify-content: center;  gap: 40px;  margin-top: 60px;  }  *#im:hover*{  border: 2px solid #F8CBAD;  height: 180px;  width: 310px;  }  *.visit*{  margin-left: 150px;  margin-top: 50px;  background-color: #E2F0D9;  height: 40px;  width: 600px;  }  *.links*{  margin-top: 50px;  margin-left: 150px;  background-color: #E2F0D9;  height: 40px;  width: 600px;  }  </style>  </head>  <body *style*="background-color:#B4C7E7">  <div *class*="head" >  <div *class*="img">  <img *src*="parklogo.jfif">  </div>  <div *class*="Name">  <h1 *class*="heading">SGNP - Sanjay Gandhi National Park</h1>  </div></div>  <div *class*="Slogan">  <h2>"Travel - Its a thing that you buy and it makes you  richer"</h2>  </div>  <div *class*="About">  <h2>About Us:</h2>  <p>Sanjay Gandhi National Park, also known as SGNP, is an 87 km2  (34 sq mi) protected area in Mumbai, Maharashtra. It was established in 1969  with its headquarters situated at Borivali.</p>  </div>  <div *class*="image-section">  <div *class*="image-container">  <div *class*="one" *border*="2px solid blue">  <a *href*="https://sgnp.maharashtra.gov.in/1215/History-of-  SGNP">History</a>  </div>  <br><br>  <div *class*="two">  <a  *href*="https://sgnp.mahaonline.gov.in/agreement/agreement.aspx?serviceid=2139?M  enuID=1125">Online Booking</a>  </div>  <br><br>  <div *class*="three">  <a  *href*="https://sgnp.maharashtra.gov.in/1115/Photo">Gallery</a>  </div>  <br> <br>  <div *class*="four">  <a *href*="https://sgnp.maharashtra.gov.in/1222/Dos-and-  Donts">Do's Don't</a>  </div>  <br> <br>  <div *class*="five">  <a *href*="https://sgnp.maharashtra.gov.in/1114/Contact-  Us">Contact us</a>  </div>  </div>  <div *class*="Section2" *style*="background-color:#B4C7E7">  <div *class*="exp">Come and Experience</div>  <div *class*="imgs">  <img *id*="im" *src*="lion.jfif" *width*="300px" *height*="170px">  <img *id*="im"*src*="train.jfif" *width*="300px">  <img *id*="im" *src*="boating.jfif" *width*="300px"  *height*="170px">  <img *id*="im" *src*="caves.jpg" *width*="300px" *height*="170px">  <img *id*="im" *src*="butter.jfif" *width*="300px" *height*="170px">  <img *id*="im" *src*="gandhi.jfif" *width*="300px" *height*="170px">  </div>  <div *class*="visit">Feedback from Visitors: </div>  <div *class*="links">Important Link for visitors: <a  *href*="https://en.wikipedia.org/wiki/Sanjay\_Gandhi\_National\_Park#:~:text=This%2  0park%20is%20home%20to,and%2040%20species%20of%20mammals.">For more info click  here</a> </div>  </div>  </div>  </div>  </div>  </body> |
| --- |

.

1. Design a homepage of National Park website using HTML5 and CSS. Show logo of National park, History of SGNP, Do’s and Don’ts and Feedback from visitors. Design at least one additional page to show the information about spots to be visited. Make use of anchor tag to show the image click event.
2. Write a Menu driven program in JavaScript to carry out the following in order to demonstrate the use of control structures -

To find greatest among three numbers.

To display even numbers less than 20.

To calculate area of a triangle.

To find if the entered year is a leap year or not

To display a table of number n.

To find the entered number is a perfect number or not.

| <!DOCTYPE html>  <html>  <head>  <title>Number Operations</title>  </head>  <body>  <script>  // Taking user input  const choice = parseInt(prompt(  "Enter your choice:\n" +  "1. Find the greatest among three numbers\n" +  "2. Display even numbers less than 20\n" +  "3. Calculate the area of a triangle\n" +  "4. Check if it's a leap year\n" +  "5. Display the table of a number\n" +  "6. Check if it's a perfect number"  ));  switch (choice) {  case 1:  // Find the greatest among three numbers  const num1 = parseFloat(prompt("Enter the first number:"));  const num2 = parseFloat(prompt("Enter the second number:"));  const num3 = parseFloat(prompt("Enter the third number:"));  let max = num1;  if (num2 > max) {  max = num2;  }  if (num3 > max) {  max = num3;  }  console.log(`The greatest number is: ${max}`);  break;  case 2:  // Display even numbers less than 20  console.log("Even numbers less than 20:");  for (let i = 2; i < 20; i += 2) {  console.log(i);  }  break;  case 3:  // Calculate the area of a triangle  const base = parseFloat(prompt("Enter the base of the triangle:"));  const height = parseFloat(prompt("Enter the height of the triangle:"));  const area = 0.5 \* base \* height;  console.log(`The area of the triangle is: ${area}`);  break;  case 4:  // Check if it's a leap year  const year = parseInt(prompt("Enter a year:"));  if ((year % 4 === 0 && year % 100 !== 0) || year % 400 === 0) {  console.log(`${year} is a leap year.`);  } else {  console.log(`${year} is not a leap year.`);  }  break;  case 5:  // Display the table of a number  const n = parseInt(prompt("Enter a number:"));  console.log(`Table of ${n}:`);  for (let i = 1; i <= 10; i++) {  console.log(`${n} x ${i} = ${n \* i}`);  }  break;  case 6:  // Check if it's a perfect number  const num = parseInt(prompt("Enter a number:"));  let sum = 0;  for (let i = 1; i <= num / 2; i++) {  if (num % i === 0) {  sum += i;  }  }  if (sum === num) {  console.log(`${num} is a perfect number.`);  } else {  console.log(`${num} is not a perfect number.`);  }  break;  default:  console.log("Invalid choice.");  }  </script>  </body>  </html> |
| --- |

1. Write a Menu driven program in JavaScript to carry out the following in order to demonstrate the use of arrow functions/anonymous function-
   1. to accept an array of subject marks of a student and calculate its average.
   2. to display Fibonacci series.
   3. to check whether the entered string is palindrome or not.

| <!DOCTYPE html>  <html>  <head>  <title>Arrow Function Examples</title>  </head>  <body>  <script>  // Arrow function to calculate average of an array of numbers  const calculateAverage = marks => {  const sum = marks.reduce((total, mark) => total + mark, 0);  return sum / marks.length;  };  // Arrow function to generate Fibonacci series  const generateFibonacciSeries = count => {  const series = [0, 1];  for (let i = 2; i < count; i++) {  series.push(series[i - 1] + series[i - 2]);  }  return series;  };  // Arrow function to check if a string is palindrome  const isPalindrome = str => {  const cleanStr = str.toLowerCase().replace(/[^a-zA-Z0-9]/g, '');  const reversedStr = cleanStr.split('').reverse().join('');  return cleanStr === reversedStr;  };  // Main menu function  const mainMenu = () => {  const choice = parseInt(prompt(`Select an option:  1. Calculate average of subject marks  2. Generate Fibonacci series  3. Check for palindrome  4. Exit`));  switch (choice) {  case 1:  const marks = prompt("Enter subject marks separated by commas:").split(',').map(Number);  const average = calculateAverage(marks);  alert(`Average: ${average}`);  break;  case 2:  const count = parseInt(prompt("Enter the count of Fibonacci numbers to generate:"));  const fibonacciSeries = generateFibonacciSeries(count);  alert(`Fibonacci Series: ${fibonacciSeries.join(', ')}`);  break;  case 3:  const inputString = prompt("Enter a string:");  const palindromeResult = isPalindrome(inputString) ? "is" : "is not";  alert(`The entered string ${palindromeResult} a palindrome.`);  break;  case 4:  alert("Exiting the program.");  break;  default:  alert("Invalid choice.");  break;  }  if (choice !== 4) {  mainMenu(); // Show the menu again  }  };  // Start the program  mainMenu();  </script>  </body>  </html> |
| --- |

1. Write a Menu driven program in JavaScript to carry out the following in order to demonstrate the use of classes and inheritance -
2. (a) Create a class ‘accholder’ with following attribute as ‘accnum’, ‘name’, ‘age’, and ‘balance’.

(b) Print details of two employees using display function.

1. (a) Create a parent class “Calculator” with attributes length, width.

(b) Create one derived class named “rectArea” from “calculator” with attribute radius to calculate and display area of a rectangle. Use super keyword to call parent’s constructor.

(c) Create derived class “circleArea” from “rectArea” to calculate and display area of a circle.

1. (a) Create a class “CheckNum” with attributes num1 and num2.

(b) Create a derived class “CheckNeonSpy” from “CheckNum”. Define methods CheckNeon() and CheckSpy().

| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>Document</title>  </head>  <body>  <script>  class Accholder {  constructor(accnum, name, age, balance) {  this.accnum = accnum;  this.name = name;  this.age = age;  this.balance = balance;  }    display() {  console.log("Account Number:", this.accnum);  console.log("Name:", this.name);  console.log("Age:", this.age);  console.log("Balance:", this.balance);  console.log();  }  }    class Calculator {  constructor(length, width) {  this.length = length;  this.width = width;  }  }    class RectArea extends Calculator {  constructor(length, width, radius) {  super(length, width);  this.radius = radius;  }    calculateRectangleArea() {  return this.length \* this.width;  }  }    class CircleArea extends RectArea {  constructor(length, width, radius) {  super(length, width, radius);  }    calculateCircleArea() {  return Math.PI \* Math.pow(this.radius, 2);  }  }    class CheckNum {  constructor(num1, num2) {  this.num1 = num1;  this.num2 = num2;  }  }    class CheckNeonSpy extends CheckNum {  constructor(num1, num2) {  super(num1, num2);  }    sumOfDigits(number) {  return number.toString().split('').reduce((sum, digit) => sum + parseInt(digit), 0);  }    productOfDigits(number) {  return number.toString().split('').reduce((product, digit) => product \* parseInt(digit), 1);  }    checkNeon() {  return this.sumOfDigits(this.num1 \* this.num1) === this.num1;  }    checkSpy() {  return this.sumOfDigits(this.num2) === this.productOfDigits(this.num2);  }  }  // Part (a) - Input for employee details  const accnum1 = parseInt(prompt("Enter Account Number for emp1:"));  const name1 = prompt("Enter Name for emp1:");  const age1 = parseInt(prompt("Enter Age for emp1:"));  const balance1 = parseFloat(prompt("Enter Balance for emp1:"));  const accnum2 = parseInt(prompt("Enter Account Number for emp2:"));  const name2 = prompt("Enter Name for emp2:");  const age2 = parseInt(prompt("Enter Age for emp2:"));  const balance2 = parseFloat(prompt("Enter Balance for emp2:"));  const emp1 = new Accholder(accnum1, name1, age1, balance1);  const emp2 = new Accholder(accnum2, name2, age2, balance2);  emp1.display();  emp2.display();  // Part (b) - Calculate Rectangle and Circle Area  const length = parseFloat(prompt("Enter Length:"));  const width = parseFloat(prompt("Enter Width:"));  const radius = parseFloat(prompt("Enter Radius for circle:"));  const rect = new RectArea(length, width, radius);  console.log("Rectangle Area:", rect.calculateRectangleArea());  const circle = new CircleArea(length, width, radius);  console.log("Circle Area:", circle.calculateCircleArea());  // Part (c) - Check Neon and Spy Numbers  const num1 = parseInt(prompt("Enter a number to check Neon:"));  const num2 = parseInt(prompt("Enter a number to check Spy:"));  const checkNeonSpy = new CheckNeonSpy(num1, num2);  console.log("Is", num1, "a neon number?", checkNeonSpy.checkNeon());  console.log("Is", num2, "a spy number?", checkNeonSpy.checkSpy());  </script>  </body>  </html> |
| --- |

Write a Menu driven program in JavaScript to carry out the following in order to demonstrate the use of classes and inheritance –

1. Create a class ‘student’ with following attribute as ‘Student\_name’, ‘roll\_no’, ‘age’, and ‘branch’. Print details of two students using display function.

| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>Student Details</title>  <style>  body {  font-family: Arial, sans-serif;  text-align: center;  background-color: #f2f2f2;  }  .container {  background-color: #ffffff;  padding: 20px;  border-radius: 10px;  width: 400px;  margin: 0 auto;  margin-top: 20px;  }  </style>  </head>  <body>  <h1>Student Details</h1>  <div class="container">  <label for="name">Student Name:</label>  <input type="text" id="name" placeholder="Enter Student Name">  <label for="roll">Roll No:</label>  <input type="text" id="roll" placeholder="Enter Roll No">  <label for="age">Age:</label>  <input type="number" id="age" placeholder="Enter Age">  <label for="branch">Branch:</label>  <input type="text" id="branch" placeholder="Enter Branch">  <button onclick="displayStudentDetails()">Display Student Details</button>  <div id="output"></div>  </div>  <script>  class Student {  constructor(Student\_name, roll\_no, age, branch) {  this.Student\_name = Student\_name;  this.roll\_no = roll\_no;  this.age = age;  this.branch = branch;  }  display() {  return `  <p><strong>Student Name:</strong> ${this.Student\_name}</p>  <p><strong>Roll No:</strong> ${this.roll\_no}</p>  <p><strong>Age:</strong> ${this.age}</p>  <p><strong>Branch:</strong> ${this.branch}</p>  <hr>  `;  }  }  function displayStudentDetails() {  const name = document.getElementById("name").value;  const roll = document.getElementById("roll").value;  const age = parseInt(document.getElementById("age").value);  const branch = document.getElementById("branch").value;  const student = new Student(name, roll, age, branch);  const output = document.getElementById("output");  output.innerHTML = student.display();  }  </script>  </body>  </html> |
| --- |

Create a react app “INP\_batch2”.

I) Create a function component. Show the Use HTML5 tags ( paragraph, heading, table, image etc.).

II) Create a class component. Show the Use HTML5 tags ( paragraph, heading, table, image etc.).

| Function component & class component  Create react app using npx create-react-app <app name>  Then in **app.js** paste follo code  import logo from "./logo.svg";  import "./App.css";  *function* App() {  return (  <div>  <h1>This is a Heading</h1>  <p>This is a paragraph of text in a React component.</p>  <h2>Another Heading</h2>  <p>More text in another paragraph.</p>  <table>  <thead>  <tr>  <th>Header 1</th>  <th>Header 2</th>  </tr>  </thead>  <tbody>  <tr>  <td>Data 1</td>  <td>Data 2</td>  </tr>  <tr>  <td>Data 3</td>  <td>Data 4</td>  </tr>  </tbody>  </table>  <img *src*="https://example.com/image.jpg" *alt*="An example image" />  </div>  );  }  export default App;  Then above code is for function component  If asked for class component dont use the above code create new file called appc.js  And paste this code  import React, { Component } from "react";  *class* appc *extends* *Component* {  render() {  return (  <div>  <h1>This is a Heading</h1>  <p>This is a paragraph of text in a React class component.</p>  <h2>Another Heading</h2>  <p>More text in another paragraph.</p>  <table>  <thead>  <tr>  <th>Header 1</th>  <th>Header 2</th>  </tr>  </thead>  <tbody>  <tr>  <td>Data 1</td>  <td>Data 2</td>  </tr>  <tr>  <td>Data 3</td>  <td>Data 4</td>  </tr>  </tbody>  </table>  <img *src*="https://example.com/image.jpg" *alt*="An example image" />  </div>  );  }  }  export default appc;  In index.js create paste following code  import React from "react";  import ReactDOM from "react-dom/client";  import "./index.css";  import App from "./App";  import reportWebVitals from "./reportWebVitals";  import MyComponent from "./appc";  *const* root = ReactDOM.createRoot(document.getElementById("root"));  *//uncomment for class component root.render(<MyComponent />);*  *//uncomment for function component root.render(<App />);*  *// If you want to start measuring performance in your app, pass a function*  *// to log results (for example: reportWebVitals(console.log))*  *// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals*  reportWebVitals(); |
| --- |

Design a Single Page Website in React using React Router. Make us of heading, paragraph, bgcolor, color, font, table, image tags to display contents on the webpage.

| 1. Install node using setup 2. Install react : **npm install -g create-react-app** in cmd 3. **npm i react-router-dom** 4. Create folder: then write command: **npx create-react-app react\_spa** 5. After that delete all files from public and src folder 6. Then create new file in public folder by name of index.html   index.html  <!DOCTYPE *html*>  <html *lang*="en">  <head>  <meta *charset*="utf-8" />  <meta  *name*="viewport"  *content*="width=device-width, initial-scale=1, shrink-to-fit=no"  />  <title>React Router Example</title>  </head>  <body>  <div *id*="root"></div>  </body>  </html>  Then create Contact.js, Home.js, index.js, Main.js,Stuff.js  Contact.js  import React, { Component } from "react";  *class* Contact *extends* *Component* {  render() {  return (  <div>  <h2>GOT QUESTIONS?</h2>  <p>  The easiest thing to do is post on our{" "}  <a *href*="http://forum.kirupa.com">forums</a>.  </p>  </div>  );  }  }  export default Contact;  Home.js  import React, { Component } from "react";  *class* Home *extends* *Component* {  render() {  return (  <div>  <h2>HELLO</h2>  <p>  Cras facilisis urna ornare ex volutpat, et convallis erat elementum.  Ut aliquam, ipsum vitae gravida suscipit, metus dui bibendum est, eget  rhoncus nibh metus nec massa. Maecenas hendrerit laoreet augue nec  molestie. Cum sociis natoque penatibus et magnis dis parturient  montes, nascetur ridiculus mus.  </p>  <p>Duis a turpis sed lacus dapibus elementum sed eu lectus.</p>  </div>  );  }  }  export default Home;  Stuff.js  import React, { Component } from "react";  *class* Stuff *extends* *Component* {  render() {  return (  <div>  <h2>STUFF</h2>  <p>  Mauris sem velit, vehicula eget sodales vitae, rhoncus eget sapien:  </p>  <ol>  <li>Nulla pulvinar diam</li>  <li>Facilisis bibendum</li>  <li>Vestibulum vulputate</li>  <li>Eget erat</li>  <li>Id porttitor</li>  </ol>  </div>  );  }  }  export default Stuff;  Index.js  import React from "react";  import ReactDOM from "react-dom";  import Main from "./Main";  ReactDOM.render(<Main />, document.getElementById("root"));  Main.js  import React, { Component } from "react";  import { Route, NavLink, HashRouter } from "react-router-dom";  import Home from "./Home";  import Stuff from "./Stuff";  import Contact from "./Contact";  *class* Main *extends* *Component* {  render() {  return (  <HashRouter>  <div>  <h1>Simple SPA</h1>  <ul *className*="header">  <li>  <NavLink *to*="/">Home</NavLink>  </li>  <li>  <NavLink *to*="/stuff">Stuff</NavLink>  </li>  <li>  <NavLink *to*="/contact">Contact</NavLink>  </li>  </ul>  <div *className*="content">  <Route *exact* *path*="/" *component*={Home} />  <Route *path*="/stuff" *component*={Stuff} />  <Route *path*="/contact" *component*={Contact} />  </div>  </div>  </HashRouter>  );  }  }  export default Main;  After this open terminal  Cd <your path>  npm start |
| --- |

WAP to perform following file operations using Node.js.

a)Access b) open c) Append d) Read e)delete f)rename g)copy

| const fs = require('fs');  const prompt = require('prompt-sync')({ sigint: true });  function asynchronousRead() {  fs.readFile('file.txt', function (error, data) {  if (error) {  return console.error(error);  }  console.log("Asynchronous read: " + data.toString());  });  }  function synchronousRead() {  try {  var data = fs.readFileSync('file.txt');  console.log("Synchronous read: " + data.toString());  } catch (error) {  console.error(error);  }  }  function openFile() {  console.log("Opening the file");  fs.open('file.txt', 'r+', function (err, fd) {  if (err) {  return console.error(err);  }  console.log("File Opened Successfully");  });  }  function writeFile() {  fs.writeFile('file.txt', 'Hello World!', function (err) {  if (err) {  console.log(err);  } else {  console.log('Write operation complete.');  }  });  }  function appendToFile() {  var data = "\nLearn Node.js with the help of a well-built Node.js Tutorial.";  fs.appendFile('file.txt', data, 'utf8', function (err) {  if (err) throw err;  console.log("Data is appended to the file successfully.");  });  }  function renameFile() {  const oldFileName = prompt("Enter the name of the old file: ");  const newFileName = prompt("Enter the name of the new file: ");  fs.rename(oldFileName, newFileName, (err) => {  if (err) {  console.error("Error renaming the file:", err);  } else {  console.log('File Renamed!');  }  });  }  function copyFile() {  const sourceFile = prompt("Enter the name of the source file: ");  const destinationFile = prompt("Enter the name of the destination file: ");  fs.copyFile(sourceFile, destinationFile, (err) => {  if (err) {  console.error("Error copying the file:", err);  } else {  console.log('File is copied!');  }  });  }  function deleteFile() {  const fileName = prompt("Enter the name of the file to delete: ");  fs.unlink(fileName, (err) => {  if (err) {  console.error("Error deleting the file:", err);  } else {  console.log('File deleted!');  }  });  }  console.log("Choose a file operation:");  console.log("1. Asynchronous Read from 'text.txt'");  console.log("2. Synchronous Read from 'text.txt'");  console.log("3. Open 'text.txt'");  console.log("4. Write to 'file.txt'");  console.log("5. Append to 'file.txt'");  console.log("6. Rename a file");  console.log("7. Copy a file");  console.log("8. Delete a file");  var choice = parseInt(prompt("Enter your choice: "));  switch (choice) {  case 1:  asynchronousRead();  break;  case 2:  synchronousRead();  break;  case 3:  openFile();  break;  case 4:  writeFile();  break;  case 5:  appendToFile();  break;  case 6:  renameFile();  break;  case 7:  copyFile();  break;  case 8:  deleteFile();  break;  default:  console.log("Invalid choice. Please select a valid option.");  }  console.log("Program Ended"); |
| --- |

Set, get and delete the cookie with express

| | // Import the Express framework  var express = require('express');  // Create an Express application  var app = express();  // Define a route for handling GET requests at the root path '/'  app.get('/',  function(req, res){  // Set a cookie named 'name' with the value 'express'  res.cookie('name', 'express').send('Cookie set'); // Sets name = express  });  // Start the Express application and listen on port 3000  app.listen(3000); | | --- |        | // Import the Express framework  var express = require('express');  // Create an Express application  var app = express();  // Define a route for handling GET requests at the path '/clear\_cookie\_foo'  app.get('/clear\_cookie\_foo',  function(req, res){  // Clear the 'foo' cookie  res.clearCookie('foo');  res.send('Cookie "foo" cleared');  });  // Start the Express application and listen on port 3000  app.listen(3000); | | --- | |
| --- | --- | --- |

**Note: Any combination of problem statements will be given.**