EXPERIMENT 3.3

AIM: Person Class Hierarchy with Student and Teacher Subclasses

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Class Hierarchy</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      line-height: 1.6;
      padding: 20px;
      background-color: #f4f4f4;
      color: #333;
    }
    .container {
      max-width: 800px;
      margin: auto;
      background: #fff;
      padding: 20px;
      border-radius: 8px;
      box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    }
    h1, h2 {
      color: #34495e;
      border-bottom: 2px solid #34495e;
      padding-bottom: 5px;
    }
    pre {
      background: #ecf0f1;
      padding: 15px;
      border-radius: 5px;
      overflow-x: auto;
    }
    .person, .student, .teacher {
      margin-bottom: 20px;
      padding: 15px;
      border-left: 5px solid;
      background-color: #f9f9f9;
      border-radius: 5px;
    }
    .person { border-color: #2ecc71; }
    .student { border-color: #3498db; }
    .teacher { border-color: #e74c3c; }
    .output-box {
```

```
            margin-top: 20px;
        }
        .output-box p {
            margin: 5px 0;
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>Person Class Hierarchy</h1>
        <p>This page demonstrates a simple class hierarchy in JavaScript with a base
**Person** class and two subclasses, **Student** and **Teacher**.</p>

        <h2>Class Definitions</h2>
        <pre id="class-definitions"></pre>

        <h2>Output</h2>
        <div id="output" class="output-box"></div>
    </div>

    <script>
        // --- JavaScript Code for Class Hierarchy ---

        // 1. Parent Class: Person
        class Person {
            constructor(name, age) {
                this.name = name;
                this.age = age;
            }

            introduce() {
                return `My name is ${this.name} and I am ${this.age} years old.`;
            }
        }

        // 2. Subclass: Student (inherits from Person)
        class Student extends Person {
            constructor(name, age, major) {
                // Call the parent class constructor using super()
                super(name, age);
                this.major = major;
            }

            // Override the introduce method
            introduce() {
                return super.introduce() + ` I am a student studying ${this.major}.`;
            }

            study() {
                return `${this.name} is studying ${this.major}.`;
```

```
        }
    }

    // 3. Subclass: Teacher (inherits from Person)
    class Teacher extends Person {
        constructor(name, age, subject) {
            super(name, age);
            this.subject = subject;
        }

        // Override the introduce method
        introduce() {
            return super.introduce() + ` I am a teacher who teaches ${this.subject}.`;
        }

        teach() {
            return `${this.name} is teaching ${this.subject}.`;
        }
    }

    // --- Creating Instances and Displaying Output ---

    const outputDiv = document.getElementById('output');
    const classDefinitionsDiv = document.getElementById('class-definitions');

    // Display class definitions as text
    classDefinitionsDiv.textContent = `
class Person {
    constructor(name, age) {
        this.name = name;
        this.age = age;
    }
    introduce() { ... }
}

class Student extends Person {
    constructor(name, age, major) {
        super(name, age);
        this.major = major;
    }
    introduce() { ... }
    study() { ... }
}

class Teacher extends Person {
    constructor(name, age, subject) {
        super(name, age);
        this.subject = subject;
    }
    introduce() { ... }
```

```
    teach() { ... }
}
        `;

        // Create instances
        const person = new Person('Alice', 30);
        const student = new Student('Bob', 22, 'Computer Science');
        const teacher = new Teacher('Carol', 45, 'Physics');

        // Function to create and append an output block
        function createOutputBlock(instance, className) {
            const div = document.createElement('div');
            div.className = className.toLowerCase();
            div.innerHTML = `
                <h4>${className} Instance</h4>
                <p><strong>Name:</strong> ${instance.name}</p>
                <p><strong>Age:</strong> ${instance.age}</p>
                <p><strong>Method introduce():</strong> ${instance.introduce()}</p>
            `;
            // Add specific properties and methods for subclasses
            if (className === 'Student') {
                div.innerHTML += `<p><strong>Major:</strong> ${instance.major}</p>`;
                div.innerHTML += `<p><strong>Method study():</strong>
${instance.study()}</p>`;
            }
            if (className === 'Teacher') {
                div.innerHTML += `<p><strong>Subject:</strong> ${instance.subject}</p>`;
                div.innerHTML += `<p><strong>Method teach():</strong>
${instance.teach()}</p>`;
            }
            outputDiv.appendChild(div);
        }

        // Display the output
        createOutputBlock(person, 'Person');
        createOutputBlock(student, 'Student');
        createOutputBlock(teacher, 'Teacher');

    </script>
</body>
</html>
```

Person Class Hierarchy

This page demonstrates a simple class hierarchy in JavaScript with a base **Person** class and two subclasses, **Student** and **Teacher**.

Class Definitions

```
class Person {
    constructor(name, age) {
        this.name = name;
        this.age = age;
    }
    introduce() { ... }
}

class Student extends Person {
    constructor(name, age, major) {
        super(name, age);
        this.major = major;
    }
    introduce() { ... }
    study() { ... }
}

class Teacher extends Person {
    constructor(name, age, subject) {
```

Output

Person Instance

Name: Alice

Age: 30

Method introduce(): My name is Alice and I am 30 years old.

Student Instance

Name: Bob

Age: 22

Method introduce(): My name is Bob and I am 22 years old. I am a student studying Computer Science.

Major: Computer Science

Method study(): Bob is studying Computer Science.

Teacher Instance

Name: Carol