Experiment-9

Person Class Hierarchy with Student and Teacher Subclasses

Code Implementation

```
// --- Base Class ---
class Person {
  constructor(name, age) {
    this.name = name;
    this.age = age;
  }

  // Method to get basic information
  getInfo() {
    return `${this.name}, Age: ${this.age}`;
  }
}

// --- Subclass 1: Student ---
class Student extends Person {
  constructor(name, age, grade) {
    super(name, age); // Call the parent class constructor
    this.grade = grade;
  }

  // Override the parent method to include the grade
  getInfo() {
    return `${super.getInfo()}, Grade: ${this.grade}`;
  }
}

// --- Subclass 2: Teacher ---
class Teacher extends Person {
  constructor(name, age, subject) {
    super(name, age); // Call the parent class constructor
    this.subject = subject;
  }

  // Override the parent method to include the subject
  getInfo() {
        return `${super.getInfo()}, Subject: ${this.subject}`;
    }
}

// --- Demonstration ---

// 1. Create instances of the subclasses
const student1 = new Student('Alice', 15, '10th');
const teacher1 = new Teacher('Mr. Smith', 45, 'Mathematics');

// 2. Log the created objects and their info to the console
console.log("Student Object:", student1);
console.log("Student Info:", student1.getInfo());

console.log("Teacher Object:", teacher1);
console.log("Teacher Info:", teacher1.getInfo());
```

Output Description

When the code is executed in an HTML file, it will display the information for the created Student and Teacher objects on the webpage. Additionally, it will log the full objects and their information strings to the browser's developer console, demonstrating how the getInfo method is successfully overridden in the subclasses.