**Assignment 3: Object-Oriented JavaScript**

**Task: Create a simple library system using object-oriented principles in JavaScript. Include classes for books, patrons, and transactions.**

Below is a simple implementation of a library system using object-oriented principles in JavaScript. The system includes classes for books, patrons, and transactions.

| class Book {  constructor(title, author, ISBN) {  this.title = title;  this.author = author;  this.ISBN = ISBN;  this.isAvailable = true;  }  displayInfo() {  console.log(`Title: ${this.title}`);  console.log(`Author: ${this.author}`);  console.log(`ISBN: ${this.ISBN}`);  console.log(`Available: ${this.isAvailable ? 'Yes' : 'No'}`);  console.log('----------------------');  }  }  class Patron {  constructor(name, libraryCardNumber) {  this.name = name;  this.libraryCardNumber = libraryCardNumber;  }  displayInfo() {  console.log(`Name: ${this.name}`);  console.log(`Library Card Number: ${this.libraryCardNumber}`);  console.log('----------------------');  }  }  class Transaction {  constructor(book, patron, dueDate) {  this.book = book;  this.patron = patron;  this.dueDate = dueDate;  this.isReturned = false;  }  displayInfo() {  console.log('Transaction Details:');  console.log('----------------------');  this.book.displayInfo();  this.patron.displayInfo();  console.log(`Due Date: ${this.dueDate}`);  console.log(`Returned: ${this.isReturned ? 'Yes' : 'No'}`);  console.log('----------------------');  }  returnBook() {  this.isReturned = true;  this.book.isAvailable = true;  console.log(`Book "${this.book.title}" has been returned by ${this.patron.name}.`);  }  }  // Example usage  const book1 = new Book('The Hobbit', 'J.R.R. Tolkien', '9780547928227');  const book2 = new Book('To Kill a Mockingbird', 'Harper Lee', '0061120081');  const patron1 = new Patron('John Doe', 'A12345');  const patron2 = new Patron('Jane Smith', 'B67890');  const transaction1 = new Transaction(book1, patron1, '2024-02-01');  const transaction2 = new Transaction(book2, patron2, '2024-03-01');  // Display book and patron information  book1.displayInfo();  book2.displayInfo();  patron1.displayInfo();  patron2.displayInfo();  // Display transaction information  transaction1.displayInfo();  transaction2.displayInfo();  // Simulate returning a book  transaction1.returnBook();  transaction1.displayInfo(); // Display updated transaction information |
| --- |

In this implementation, there are three classes: Book, Patron, and Transaction. Each class has a constructor to initialize its properties and methods for displaying information. The Transaction class also has a returnBook method to simulate returning a book.

The example usage section demonstrates creating instances of books, patrons, and transactions, displaying their information, and simulating the return of a book. Feel free to modify and extend the code based on the specific requirements of your library system.