DSA PROJECT
(Library Management System)

Code Explanation:

1. Book

Represents a book in the library.

- **Attributes:**
  - **title:** The title of the book.
  - **author:** The author of the book.
  - **isbn:** The ISBN of the book.
  - **status:** The status of the book, which is either "available" or "unavailable". Default is "available".
- **Methods:**
  - No specific methods; the class acts as a data holder for book information.

2. Ebook

Represents an ebook in the library.

- **Attributes:**
  - **title:** The title of the ebook.
  - **author:** The author of the ebook.
  - **format:** The format of the ebook (e.g., PDF, EPUB, MOBI).
- **Methods:**
  - No specific methods; the class acts as a data holder for ebook information.

3. LinkedList

Manages a collection of Book objects using a linked list data structure.

- **Structure:**
  - Contains an inner class Node representing a node in the linked list, with data (Book object) and reference to the next node (next).
  - Has pointers to head (first node) and tail (last node).
- **Methods:**
  - **append(data):** Adds a new Book object to the end of the linked list.
  - **remove(data):** Removes a Book object from the linked list.
  - **search(query):** Searches for books in the linked list based on a query (title, author, or ISBN) and returns a list of matching books.

4. Stack

Manages a collection of Ebook objects using a stack data structure.

- **Structure:**
  - Uses a Python list (self.items) to implement the stack.
- **Methods:**
  - **push(item):** Adds a new Ebook object to the top of the stack.
  - **pop():** Removes and returns the top Ebook object from the stack.
  - **peek():** Returns the top Ebook object without removing it.
  - **is_empty():** Checks if the stack is empty.
  - **search(query):** Searches for ebooks in the stack based on a query (title or author) and returns a list of matching ebooks.

5. Queue

Manages available Book objects using a queue data structure.

- **Structure:**
  - Uses a Python list (self.items) to implement the queue.
- **Methods:**
  - **enqueue(item):** Adds a new Book object to the end of the queue.
  - **dequeue():** Removes and returns the first Book object from the queue.
  - i**s_empty():** Checks if the queue is empty.

6. LibraryManagementSystem

Manages the library's operations, including books, ebooks, and user interactions.

- **Attributes:**
  - **books_linkedlist:** An instance of LinkedList to manage books.
  - **ebooks_stack**: An instance of Stack to manage ebooks.
  - **available_books_queue:** An instance of Queue to manage available books.
  - **available_ebooks:** A list of available ebooks.
- **Methods:**

☐ **Book Management:**

- **receive_book()**: Handles receiving a book from a user, updating the book's status, and saving the transaction.
- **return_book():** Handles returning a book to the library, updating the book's status, and saving the transaction.
- **get_receiver_details()**: Retrieves and displays receiver details.
- **add_book(title, author, isbn):** Adds a new Book object to the linked list and queue, and saves it to file.

- **remove_book(title, author, isbn):** Removes a book from the linked list and queue based on the specified criteria, and updates the file.
- **search_book(query):** Searches for books based on the query and returns matching books.
- **view_available_books():** Displays the list of available books.

☐ **Ebook Management:**

- **add_ebook(title, author, format):** Adds a new Ebook object to the stack and list, and saves it to file.
- **remove_ebook(title, author, format):** Removes an ebook from the stack and list based on the specified criteria, and updates the file.
- **search_ebook(query):** Searches for ebooks based on the query and returns matching ebooks.
- **view_available_ebooks():** Displays the list of available ebooks.

- **Utility Methods:**
  - **load_books_from_file(file_path):** Loads books from a file and appends them to the linked list and queue.
  - **load_ebooks_from_file(file_path):** Loads ebooks from a file and appends them to the stack and array.
  - **update_books_file():** Updates the books file based on the current state of the linked list.
  - **update_ebooks_file():** Updates the ebooks file based on the current state of the ebooks stack and list.
- **Menu Interaction:**
  - **display_menu():** Displays a menu for user interaction and prompts the user to make a selection.
  - **run():** Main function that runs the library management system, handling user input and executing corresponding operations.

Data Structures:

- **Linked List:**

☐ **Purpose:**

1. Manages a collection of Book objects.
2. Operations include appending, removing, and searching for books based on their title, author, and ISBN.

☐ **Structure:**

1. Nodes contain a Book object (data) and a reference to the next node (next).
2. Keeps track of the head (first node) and tail (last node) for efficient appending.

- **Stack:**
  - ☐ **Purpose:**
  1. Manages a collection of Ebook objects.
  2. Operations include pushing, popping, peeking, checking if empty, and searching for ebooks.
  - ☐ **Structure:**
  1. Implemented using a Python list (self.items).
  2. Follows the Last-In-First-Out (LIFO) principle.
- **Queue:**
  - ☐ **Purpose:**
  1. Manages available Book objects.
  2. Operations include enqueuing, dequeuing, and checking if empty.
  - ☐ **Structure:**
  1. Implemented using a Python list (self.items).
  2. Follows the First-In-First-Out (FIFO) principle.
- **Array (Python List):**
  - ☐ **Purpose:**
  1. Manages available Ebook objects in the library.
  2. Provides fast access to available ebooks and supports iteration.
  - ☐ **Structure:**
  1. The list (self.available_ebooks) stores instances of the Ebook class.
  2. It is maintained in parallel with the stack to keep track of available ebooks.
  - ☐ **Operations:**
  1. **Append**: Adds new ebooks to the list when an ebook is added to the stack.
  2. **Remove**: Removes ebooks from the list when an ebook is removed from the stack.
  3. **Search:** Allows iteration and search operations directly on the list.
  4. **Load and Save:** Used for loading and saving ebook data from and to files.

Algorithms:

- **Searching:**
  - ☐ **Linked List Search:**
  1. The search method in LinkedList traverses the linked list from the head node to the tail node.
  2. It compares the query string with the title, author, and ISBN of each book in a case-insensitive manner.
  3. If a match is found, the book is added to the list of results, which is returned at the end.

- ☐ **Stack Search:**

1. The search method in Stack iterates over the stack's items.
2. It compares the query string with the title and author of each ebook in a case-insensitive manner.
3. If a match is found, the ebook is added to the list of results, which is returned at the end.

- **File Loading and Updating:**
  - ☐ **Loading Books from File:**

1. The load_books_from_file method reads a specified file line by line.
2. Each line contains book details (title, author, ISBN, and status), which are used to create Book objects.
3. These book objects are appended to the linked list and, if available, to the queue.

- ☐ **Loading Ebooks from File:**

1. The load_ebooks_from_file method reads a specified file line by line.
2. Each line contains ebook details (title, author, and format), which are used to create Ebook objects.
3. These ebook objects are pushed onto the stack and appended to the available ebook's list.

- ☐ **Updating Books File:**

1. The update_books_file method writes the current state of the linked list to the books file.
2. It traverses the linked list from head to tail and writes each book's details to the file.

- ☐ **Updating Ebooks File:**

1. The update_ebooks_file method writes the current state of the available ebooks list to the ebooks file.
2. It iterates over the list of available ebooks and writes each ebook's details to the file.

Conclusion:

The Library Management System uses various data structures like linked lists, stacks, and queues to manage books and ebooks efficiently. These data structures enable operations such as adding, removing, searching, and viewing available books and ebooks. The system also interacts with files to load and update data as needed.