Documentation: Library System Code Overview

This documentation provides a thorough explanation of the library system code, detailing classes, functions, their purpose, and interactions.

Purpose:

The library system code simulates a basic library management system. It includes classes for items (books and videos), transactions, accounts, and the library itself. Users can log in, view borrowable items, borrow and return them, and view their borrowed items.

Classes:

1. `Item` (from `Itemfile.h`):

- **Purpose**: Represents a library item, with common attributes for both books and videos.
- **Attributes**:
  - `theID`: Identifier for the item.
  - `theTitle`: Title of the item.
- **Functions**:
  - `Item()`: Default constructor.
  - `Item(string iid, string ititle)`: Constructor with ID and title.
  - `Item(string iid)`: Constructor with ID.
  - `getID()`, `getTitle()`: Getters for ID and title.
  - `setID(string someid)`, `setTitle(string sometitle)`: Setters for ID and title.
  - `virtual void print()`: Virtual function to print item details (to be overridden in derived classes).
  - `virtual Item* duplicate()`: Virtual function to create a new item object (to be overridden in derived classes).
  - `~Item()`: Destructor.

2. `Video` (Derived from `Item`):

- **Purpose**: Represents a video item.
- **Attributes**:
  - `Genre`: Genre of the video.
  - `director`: Director of the video.
  - `producer`: Producer of the video.
- **Functions**:
  - `Video()`: Default constructor.

    - `Video(string vidId, string Title, string Genre, string director, string producer)`: Constructor with video details.
    - `Video(string theID, string Genre, string director, string producer)`: Constructor with base class ID.
    - `print()`: Overrides `Item`'s print function to print video details.
    - `duplicate()`: Overrides `Item`'s duplicate function to create a new `Video` object.

3. `Book` (Derived from `Item`):

- **Purpose**: Represents a book item.
- **Attributes**:
    - `Author`: Author of the book.
- **Functions**:
    - `Book()`: Default constructor.
    - `Book(string theId, string author)`: Constructor with book ID and author.
    - `Book(string bookId, string Title, string Author)`: Constructor with book details.
    - `print()`: Overrides `Item`'s print function to print book details.
    - `duplicate()`: Overrides `Item`'s duplicate function to create a new `Book` object.

4. `Transaction`:

- **Purpose**: Manages transactions for borrowed items.
- **Attributes**:
    - `itemVector`: Vector to hold borrowed items.
- **Functions**:
    - `Transaction()`: Default constructor.
    - `borrowItems(Item* item)`: Adds an item to the `itemVector`.
    - `getItemVector()`: Returns the `itemVector`.
    - `setItemVector(vector<Item*> iVector)`: Sets the `itemVector`.
    - `findBorrowedItem(Item* item)`: Finds the index of a borrowed item.
    - `~Transaction()`: Destructor.
    - `Transaction(const Transaction& trans)`: Copy constructor.
    - `void operator=(const Transaction& trans)`: Assignment operator.

5. `Account`:

- **Purpose**: Represents a library account.
- **Attributes**:
    - `id`: Account ID.

    - `pass`: Account password.
    - `translist`: Vector to hold transactions.
- **Functions**:
    - `Account()`: Default constructor.
    - `Account(string id, string pass)`: Constructor with ID and password.
    - `getTransaction()`: Returns the vector of all transactions.
    - `addTransaction(const Transaction atransaction)`: Adds a transaction to the account.
    - `setTransaction(vector<Transaction> translist)`: Sets the transaction vector.
    - `getID()`, `getPass()`: Getters for account ID and password.
    - `~Account()`: Destructor.
    - `void operator=(const Account& acc)`: Assignment operator.

6. `Library`:

- **Purpose**: Manages the library system.
- **Attributes**:
    - `libId`: Library ID.
    - `itemList`: Vector to hold library items.
    - `accountList`: Vector to hold library accounts.
- **Functions**:
    - `Library(string Libid)`: Constructor with library ID.
    - `readBookItems()`: Reads book items from file.
    - `readVideoItems()`: Reads video items from file.
    - `readAccount()`: Reads account information from file.
    - `validate(string aid, string apass)`: Validates account ID and password.
    - `getItemList()`, `addItem(Item* item)`: Manages library items.
    - `getAccountList()`: Returns the account list.
    - `removeItem(Item* item)`: Removes an item from the library.
    - `SearchItem(string id)`: Searches for an item in the library.
    - `~Library()`: Destructor.
    - `Library(const Library& lib)`: Copy constructor.
    - `void operator=(const Library& lib)`: Assignment operator.

Usage:

- The `main()` function serves as the entry point and demonstrates the library system's functionalities.
- Users can log in, view items, borrow, return items, and view their borrowed items.

- File I/O is used to read and store account, video, and book information.

Functionality Overview:

1. **Login**: Users enter their ID and password.
2. **Main Menu**:
    - **View Items**: Display all available items.
    - **Borrow Items**: Choose and borrow items.
    - **View Borrowed Items**: Display items currently borrowed.
    - **Return Borrowed Item**: Return an item to the library.
    - **Quit**: Exit the system.
3. **Operations**:
    - Borrowed items are stored in `Transaction` objects.
    - `Library` manages adding, removing, and searching for items.
    - `Account` manages adding and retrieving transactions.

File Usage:

- `account123.txt`: Contains account information (ID and password).
- `vide123.txt`: Contains video items' details.
- `book123.txt`: Contains book items' details.