Library Management System Case Study:

Classes:

1. Book:

- Attributes: ISBN, title, author, price, availability, genre, etc.
- Methods: getters and setters, toString() for displaying book details.

2. Library:

- Attributes: List of books, library name, address, etc.
- Methods:
- addBook(Book book): Adds a book to the library inventory.
- removeBook(String ISBN)`: Removes a book from the library inventory based on ISBN.
- searchByAuthor(String author): Returns a list of books by a specific author.
- searchByGenre(String genre): Returns a list of books in a particular genre.
- displayAvailableBooks(): Displays a list of available books.

3. Member:

- Attributes: Member ID, name, borrowedBooks (a list of books a member has borrowed).
- Methods:
- borrowBook(String ISBN): Allows a member to borrow a book.
- returnBook(String ISBN): Allows a member to return a book.
- displayBorrowedBooks(): Displays the list of books a member has borrowed.

4. SerializationHandler:

- Methods:
- serializeLibrary(Library library, String fileName): Serializes the library object to a file.
- deserializeLibrary(String fileName): Deserializes the library object from a file.

Exception Handling:

1. BookNotFoundException:

- Custom exception class for handling cases where a book is not found in the library.

2. NotEnoughBooksException:

- Custom exception class for handling cases where there are not enough copies of a book available for borrowing.

Implementation:

- Use encapsulation to hide the internal details of classes and provide access through getter and setter methods.

- Utilize inheritance to create a base class for common attributes/methods shared among different classes (e.g., Book as a base class).

- Demonstrate polymorphism by allowing different methods to accept objects of the base class

but work with objects of derived classes.

- Implement abstraction by creating abstract methods or classes that represent common behavior without providing a complete implementation.

- Handle exceptions appropriately, for instance, throw `BookNotFoundException` when trying to

perform an operation on a non-existing book or `NotEnoughBooksException` when there are not

enough copies of a book for borrowing.

Serialization:

- Implement serialization by using the Serializable interface for classes that need to be serialized (e.g., Library).

- Use ObjectOutputStream and ObjectInputStream to write and read objects to/from a file.

Solution:

https://github.com/TharunPatel20/UST-techAcademy/tree/08b6492af473bec8f410895e03d7f9285de62240/USTJavaCourse/LibraryManagementSystemCaseStudy/src