**Task1: Library Management System Application**

Requirements:

1. **Create Classes:**
   * **Base Class:**
     + Properties:
       - (string): The title of the book.
       - (string): The author of the book.
       - (string): The International Standard Book Number.
       - (bool): The availability status of the book (checked out or not).
     + Methods:
       - : Change the status to checked out.
       - : Change the status to available.
   * **Derived Class:** (inherits from )
     + Properties:
       - (double): The size of the file.
       - (string): The format of the book (e.g., PDF, EPUB).
     + Methods:
       - : Execute the download process.
   * **Another Derived Class:** (inherits from )
     + Properties:
       - (int): The duration of the audiobook (in seconds).
       - (string): The name of the narrator.
     + Methods:
       - : Start playing the audiobook.
2. **Implement Encapsulation:**
   * Ensure that the core properties and derived properties (like ) are private or protected, and provide public methods to access them.
3. **Test the System:**
   * Create instances of , , and .
   * Test the different methods to ensure that inheritance and encapsulation are working correctly.

**Part 2**

Requirements:

1. Create Enums:
   * BookStatus: To represent the status of a book.
     + Available
     + Checked out
     + Reservesd
   * BookType: To represent different types of books.
     + (paperback)
     + (electronic book)
     + (audio book)
2. Update the Book Class:
   * Add a property to differentiate between the types of books.
3. Modify the Derived Classes:
   * Use the enumeration when creating instances of the derived classes (EBook and AudioBook).
4. Implement Encapsulation:
   * Ensure that properties are properly encapsulated and provide methods to manipulate book states (borrow, return, reserve).
5. Test the System:
   * Create instances of , , and .
   * Test the different methods to ensure that the implementation of inheritance, encapsulation, and enums work correctly.

Criteria:

* Use object-oriented programming principles correctly.
* Ensure effective use of encapsulation and enums.
* Tests should be comprehensive for all classes and methods.

**Notes:**

Students can also think about adding additional features, such as introducing new classes (like to manage a collection of books) or implementing a user interface as part of the enhancements.