Implement the below Questions and upload it to a github repository. Then share the github repository link after you are done.

Question 1:

You were assigned to Team in Bank where you have to make a software for Bank Accounting system. You task is to design a `BankAccount` class in Python with the following specifications:

1. **Attributes:**
    - `account_number` (string): A unique identifier for the account.
    - `account_holder` (string): The name of the account holder.
    - `balance` (float): The current balance of the account.
2. **Methods:**
    - `__init__(self, account_number, account_holder, initial_balance=0.0)`: Initializes the account with the given account number, account holder, and an optional initial balance.
    - `deposit(self, amount)`: Adds the specified amount to the account balance. Raises a `ValueError` if the amount is non-positive.
    - `withdraw(self, amount)`: Deducts the specified amount from the account balance. Raises a `ValueError` if the amount is non-positive or if there are insufficient funds.
    - `get_balance(self)`: Returns the current balance of the account.

- `__str__(self)`: Returns a string representation of the account with account number, account holder, and current balance.

Instructions:

- Write the implementation of the `BankAccount` class.
- Include proper error handling and ensure the class methods have appropriate validation checks.
- Demonstrate the usage of the class by creating an instance and performing a series of deposits and withdrawals.

Question 2:

Let's build a Library Management System. You task is to design a set of classes to represent a simple library system with the following specifications:

1. **Base Class:**
    - `Item`: Represents a generic item in the library.
        - **Attributes:**
            - `title` (string): The title of the item.
            - `author` (string): The author or creator of the item.
            - `year` (int): The year the item was published.
        - **Methods:**
            - `__init__(self, title, author, year)`: Initializes the item with the given title, author, and year.
            - `get_info(self)`: Returns a string with the title, author, and year of the item.
            - `__str__(self)`: Returns a string representation of the item.
2. **Derived Classes:**
    - Book: Inherits from `Item`.
        - **Attributes:**
            - `title`, `author`, `year` (inherited from `Item`).
            - `publisher` (string): The publisher of the book.
            - `isbn` (string): The ISBN number of the book.
        - **Methods:**

        - `__init__(self, title, author, year, publisher, isbn)`: Initializes the book with the given title, author, year, publisher, and ISBN.
        - `get_info(self)`: Returns a string with all the details of the book, including publisher and ISBN.
- DVD: Inherits from `Item`.
    - **Attributes:**
        - `title`, `author`, `year` (inherited from `Item`).
        - `duration` (int): The duration of the DVD in minutes.
        - `region_code` (int): The region code of the DVD.
    - **Methods:**
        - `__init__(self, title, author, year, duration, region_code)`: Initializes the DVD with the given title, author, year, duration, and region code.
        - `get_info(self)`: Returns a string with all the details of the DVD, including duration and region code.

Instructions:

- Write the implementation of the `Item`, Book, and DVD classes.
- Ensure that the `get_info` method in each derived class overrides the base class method and includes additional details specific to the derived class.
- Demonstrate the usage of these classes by creating instances of Book and DVD and calling their `get_info` methods.