***CASE STUDY NO. 3***

* **Title**
* A book shop inventory
* **Theory Concepts Of Java/OOP Used in This Case Study**

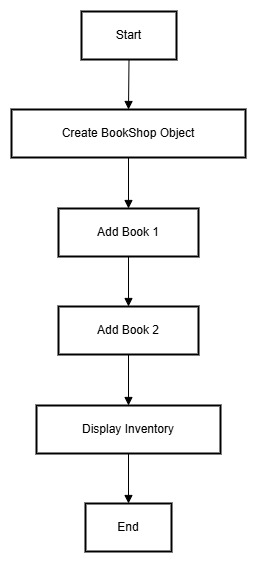
1. **Class**:   
   A class is a template or blueprint for creating objects that encapsulate data and behaviors. It defines the structure and behavior that the objects created from the class will have. In this program, the Book class models individual book details like title, author, and price, while the BookShop class represents the inventory system for managing a collection of books.
2. **Object**:   
   An object is an instance of a class that holds specific data and can perform actions defined by its class methods. For example, Book objects represent individual books with unique details, while a BookShop object manages multiple Book instances, interacting with them to perform operations like adding or displaying books.
3. **Encapsulation**:   
   Encapsulation Encapsulation is a principle of bundling data and methods within a class while restricting direct access to some components to protect the integrity of the data. In the Book and BookShop classes, fields like title and books are declared private, and public methods such as getTitle or addBook provide controlled access to these fields, ensuring data security and consistency.

1. **Array:**   
   An array is a collection of elements of the same type stored in contiguous memory locations, allowing efficient storage and access. In this program, an array of Book objects is used to maintain the inventory of the bookshop. This allows easy iteration and management of books, such as adding or retrieving book details.
2. **Method:**   
   A constructor is a special method in a class that is automatically called when an object is created. It initializes the object’s fields with provided values or defaults. In this program, the Book constructor initializes fields like title, author, and price, while the BookShop constructor sets up the initial state of the bookshop's inventory, such as creating an empty array for storing books.

* **Algorithm:**

1. **Start**
2. **Define the Book class**:
   * Declare private fields: title, author, and price.
   * Define a constructor to initialize title, author, and price.
   * Create a method displayDetails to print the book's details.
3. **Define the BookShop class**:
   * Declare private fields: an array of Book objects (books) and an integer count to track the number of books.
   * Define a constructor to initialize the books array with a given capacity and set count to 0.
   * Implement the addBook method:
     + Check if count is less than the length of the books array:
       - If true, create a new Book object with the provided details and add it to the books array. Increment count.
       - Otherwise, print "Inventory full!".
   * Implement the displayInventory method:
     + Loop through the books array up to count and call the displayDetails method for each book.
4. **In the main method**:
   * Create a BookShop object with a capacity of 5.
   * Call the addBook method to add two books:
     + "The Great Gatsby" by "F. Scott Fitzgerald" priced at $10.99.
     + "1984" by "George Orwell" priced at $8.99.
   * Call the displayInventory method to display the details of all books in the shop.
5. **End**

* **Flowchart:**



* **Program/Code:**
* class Book

{

    private String title;

    private String author;

    private double price;

    public Book(String title, String author, double price)

    {

        this.title = title;

        this.author = author;

        this.price = price;

    }

    public void displayDetails()

    {

        System.out.println("Title: " + title + ", Author: " + author + ", Price: $" + price);

    }

}

class BookShop

{

    private Book[] books;

    private int count;

    public BookShop(int capacity)

    {

        books = new Book[capacity];

        count = 0;

    }

    public void addBook(String title, String author, double price)

    {

        if (count < books.length)

        {

            books[count++] = new Book(title, author, price);

        }

        else

        {

            System.out.println("Inventory full! Cannot add more books.");

        }

    }

    public void displayInventory()

    {

        System.out.println("=== Book Inventory ===");

        for (int i = 0; i < count; i++)

        {

            books[i].displayDetails();

        }

    }

}

public class BookShopInventory

{

    public static void main(String[] args)

    {

        BookShop shop = new BookShop(5);

        shop.addBook("The Great Gatsby", "F. Scott Fitzgerald", 10.99);

        shop.addBook("1984", "George Orwell", 8.99);

        shop.displayInventory();

    }

}

* **Output:**
* === Book Inventory ===

Title: The Great Gatsby, Author: F. Scott Fitzgerald, Price: $10.99

Title: 1984, Author: George Orwell, Price: $8.99

* **Conclusion**
* This case study illustrates how Object-Oriented Programming simplifies managing a bookshop inventory. By using classes to encapsulate data and methods, the program is modular and organized. Arrays efficiently store multiple book objects, and methods enable controlled access and manipulation of inventory data. This design ensures scalability and ease of maintenance.