DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Experiment 1.2

Student Name: Deep Baliyan **UID: 22BCS13734**
Branch: CSE **Section: 631-A**
Semester: 6th **DOP: 15/01/2025**
Subject: Java **Subject Code: 22CSH-359**

Aim: Design and implement a simple inventory control system for a small video rental store

Objective: To design and implement a user-friendly inventory control system for a small video rental store, enabling efficient management of video inventory, including functionalities for adding, renting, and returning videos.

Algorithm:

- **Define Classes:**
- **Video**: To represent each video, with attributes such as video ID, title, genre, and availability status.
- **Inventory**: To manage the list of videos, including adding and removing videos from the inventory.
- **Customer**: To represent customers, with attributes such as customer ID, name, and rented videos.
- **RentalSystem**: To control the process of renting and returning videos.
- **Video Class:**
- Define the video with attributes such as `videoID`, `title`, `genre`, and `isAvailable`.
  Define methods to mark the video as rented and returned.
- **Inventory Class:** Maintain a list of videos (`ArrayList<Video>`).
- Implement methods to add new videos, display available videos, and check if a video is available.
- **Customer Class:**
- Define a list to store rented videos.
- Implement methods to rent a video (if available) and return it.
- **RentalSystem Class:**

- Handle the main functionality: list available videos, allow customers to rent and return videos, and display the inventory status.

Code:

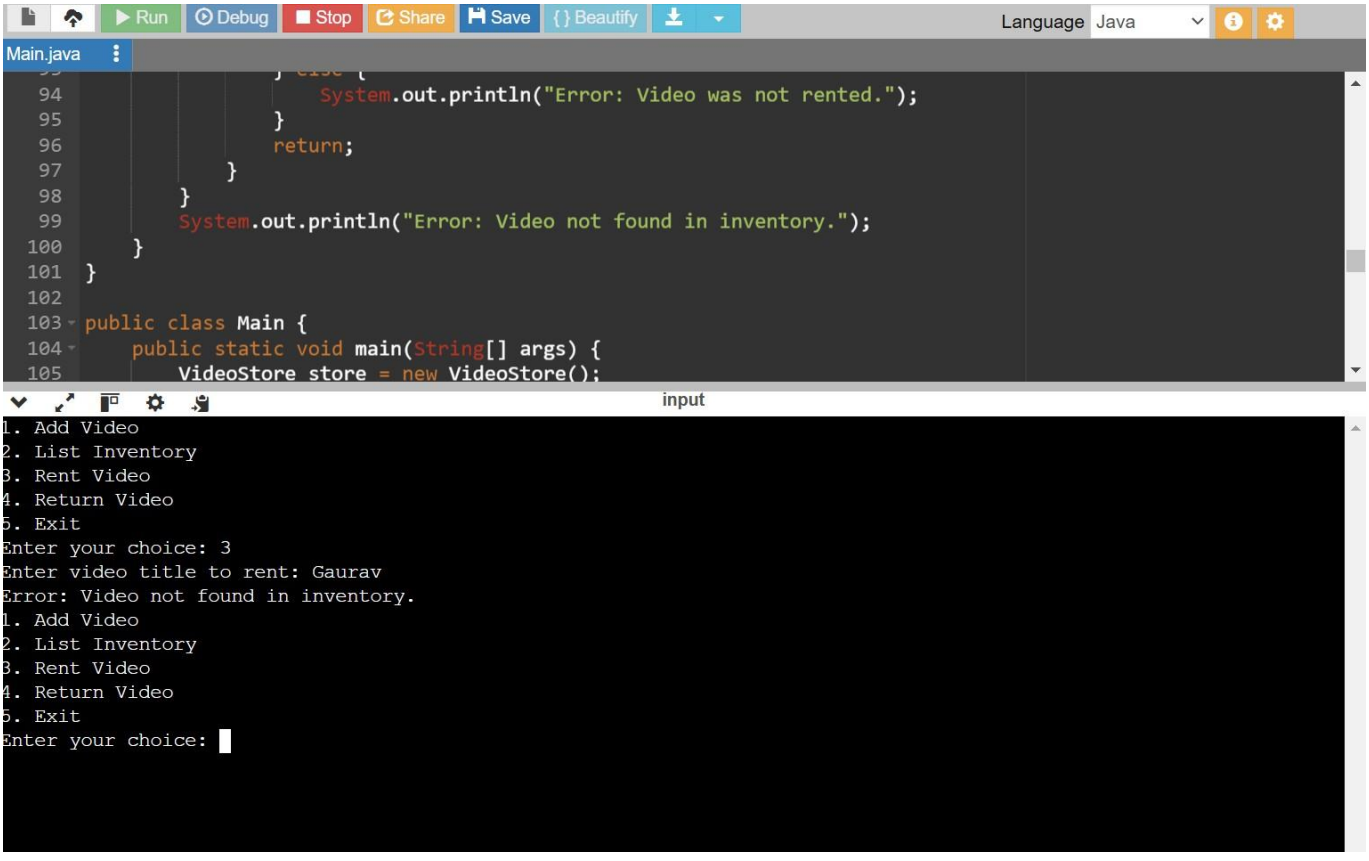
```
import java.util.ArrayList; import
java.util.Scanner;

// Class representing a Video class
Video {  private
   String title;
   private boolean isAvailable;

   public Video(String title) {
      this.title     =          title;
      this.isAvailable = true;
   }

   public String getTitle() { return title;
   }

   public boolean isAvailable() {  return
      isAvailable;
   }

   public void rent() {  if
      (isAvailable) {
      isAvailable = false;
      } else {
System.out.println("Error: Video is already rented out.");


   }

}

   public void returnVideo() { if
         (!isAvailable)    {
      isAvailable = true;
      } else {
         System.out.println("Error: Video was not rented."); }
   }
```

```
    @Override  public String toString() { return "Title: " + title + " |
    Available: " + (isAvailable ? "Yes" : "No"); }
}

// Class representing the Video Store class
VideoStore {  private
    ArrayList<Video> inventory;

    public VideoStore() {  inventory =
        new ArrayList<>();
    }

    // Add a new video to the inventory public void
    addVideo(String title) {
        for  (Video  video  :  inventory)  {  if
            (video.getTitle().equalsIgnoreCase(title)) {
                System.out.println("Error: Video already exists in the inventory."); return;
            }
        }  inventory.add(new
        Video(title));
        System.out.println("Video added successfully: " + title); }

    // List all videos in the inventory
    public void listInventory() { if
    (inventory.isEmpty()) {
System.out.println("No videos in inventory.");  }
        else {
            System.out.println("Inventory:");      for
            (int i = 0; i < inventory.size(); i++) {
                System.out.println((i + 1) + ". " + inventory.get(i)); }
        }
    }


  // Rent a video  public void rentVideo(String title)
  { for (Video video
        :              inventory)              {              if
        (video.getTitle().equalsIgnoreCase(title)) {
 if (video.isAvailable()) {  video.rent();
                System.out.println("You rented: " + title);
            } else {
                System.out.println("Video is currently unavailable.");
            } return;
        }
        }
        System.out.println("Error: Video not found in inventory."); }

    // Return a video
```

```
    public void returnVideo(String title) {
        for (Video video : inventory) { if
            (video.getTitle().equalsIgnoreCase(title)) {
        if (!video.isAvailable()) {
video.returnVideo();
                System.out.println("You returned: " + title);
            } else {
                System.out.println("Error: Video was not rented.");
            } return;
        }
    }
    System.out.println("Error: Video not found in inventory."); }
}

// Main class to run the Video Rental System public
class VideoRentalSystem {
  public static void main(String[] args) {
    VideoStore store = new VideoStore(); Scanner
    scanner = new Scanner(System.in);  while
    (true) {
      System.out.println("\n--- Video Rental Store ---");
          System.out.println("1. Add Video");
          System.out.println("2. List Inventory");
          System.out.println("3. Rent Video");
          System.out.println("4. Return Video");
          System.out.println("5. Exit");


System.out.print("Enter your choice: ");

// Handle invalid input for menu choices


          int choice = -1;
if (scanner.hasNextInt()) {


        choice = scanner.nextInt();
        } else {
          System.out.println("Invalid choice. Please enter a
        number."); scanner.next(); // Consume invalid input
        continue;  } scanner.nextLine();

        switch (choice) {
          case 1:
            System.out.print("Enter video title to add: ");
            String titleToAdd =
```

```
                    scanner.nextLine().trim();
                    store.addVideo(titleToAdd); break;
                case 2: store.listInventory();
                    break;
                case 3:
                    System.out.print("Enter video title to rent: ");
                    String titleToRent =
                    scanner.nextLine().trim();
                    store.rentVideo(titleToRent); break;
                case 4:
                    System.out.print("Enter video title to return: ");
                    String titleToReturn =
                    scanner.nextLine().trim();
                    store.returnVideo(titleToReturn); break;
                case 5:  System.out.println("Exiting the
                    system.
                    Goodbye!"); scanner.close(); return;  default:
                    System.out.println("Invalid choice. Please try again.");
            }
        }
    }
}
```

Output:



Learning Outcomes:

- **Object-Oriented Design:** Learn to create and use classes for real-world entities. **Core Programming Skills:** Practice loops, conditionals, and methods for inventory operations.
- **Data Structure Usage:** Use `arraylist` to manage dynamic data effectively.
- **User-Friendly Systems:** Design intuitive interfaces and handle errors smoothly.