Experiment 1.2

Student Name: Jatin Singh **UID: 22BCS10887**
Branch: CSE **Section: 631**
Semester: 6th **DOP: 31/01/25**
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:

```
// Video.java
class Video {
    private String videoId;
    private String title;
    private String genre;
    private boolean isAvailable;

    public Video(String videoId, String title, String genre)
        { this.videoId = videoId;
        this.title = title;
        this.genre = genre;
        this.isAvailable = true;
    }
    public String getVideoId() { return videoId; }
    public String getTitle() { return title; }
    public String getGenre() { return genre; }
    public boolean isAvailable() { return isAvailable; }
    public void rentVideo() {
        isAvailable = false;
    }
    public void returnVideo()
        { isAvailable = true;
    }
    @Override
    public String toString() {
        return String.format("ID: %s, Title: %s, Genre: %s, Available: %s",
            videoId, title, genre, isAvailable ? "Yes" : "No");
    }
}
// Customer.java
import java.util.ArrayList;
import java.util.List;
class Customer {
    private String customerId;
    private String name;
    private List<Video> rentedVideos;
    public Customer(String customerId, String name)
        { this.customerId = customerId;
        this.name = name;
        this.rentedVideos = new ArrayList<>();
    }
    public String getCustomerId() { return customerId; }
    public String getName() { return name; }
    public List<Video> getRentedVideos() { return rentedVideos; }
    public void rentVideo(Video video) {
        if (video.isAvailable())
            { rentedVideos.add(video);
            video.rentVideo();
        }
    }
    public void returnVideo(Video video)
        { if (rentedVideos.remove(video)) {
            video.returnVideo();
        }
    }
```

```
    @Override
    public String toString() {
        return String.format("Customer ID: %s, Name: %s, Videos Rented: %d",
            customerId, name, rentedVideos.size());
    }
}
// Inventory.java
class Inventory {
    private List<Video> videos;

    public Inventory() {
        this.videos = new ArrayList<>();
    }
    public void addVideo(Video video)
        { videos.add(video);
    }
    public void removeVideo(Video video)
        { videos.remove(video);
    }
    public List<Video> getAllVideos()
        { return new ArrayList<>(videos);
    }
    public List<Video> getAvailableVideos()
        { return videos.stream()
            .filter(Video::isAvailable)
            .collect(Collectors.toList());
    }
    public Video findVideo(String videoId)
        { return videos.stream()
            .filter(v -> v.getVideoId().equals(videoId))
            .findFirst()
            .orElse(null);
    }
    public void displayInventory()
        { System.out.println("\nCurrent Inventory:");
        videos.forEach(System.out::println);
    }
}
// RentalSystem.java
import java.util.ArrayList;
import java.util.List;
import java.util.stream.Collectors;

public class RentalSystem
    { private Inventory inventory;
    private List<Customer> customers;

    public RentalSystem()
        { this.inventory = new Inventory();
        this.customers = new ArrayList<>();
    }
    public void addCustomer(Customer customer)
        { customers.add(customer);
    }
    public Customer findCustomer(String customerId)
        { return customers.stream()
            .filter(c -> c.getCustomerId().equals(customerId))
```

```
            .findFirst()
            .orElse(null);
    }
    public void rentVideo(String customerId, String videoId)
        { Customer customer = findCustomer(customerId);
        Video video = inventory.findVideo(videoId);

        if (customer == null)
            { System.out.println("Customer not found!");
            return;
        }
        if (video == null)
            { System.out.println("Video not found!");
            return;
        }
        if (!video.isAvailable())
            { System.out.println("Video is not available!");
            return;
        }
        customer.rentVideo(video);
        System.out.printf("Video '%s' rented to customer '%s'\n",
            video.getTitle(), customer.getName());
    }
    public void returnVideo(String customerId, String videoId)
        { Customer customer = findCustomer(customerId);
        Video video = inventory.findVideo(videoId);
        if (customer == null || video == null) {
            System.out.println("Customer or video not found!");
            return;
        }
        customer.returnVideo(video);
        System.out.printf("Video '%s' returned by customer '%s'\n",
            video.getTitle(), customer.getName());
    }
    public void displayAvailableVideos()
        { System.out.println("\nAvailable Videos:");
        inventory.getAvailableVideos().forEach(System.out::println);
    }
    public void displayCustomerInfo(String customerId)
        { Customer customer = findCustomer(customerId);
        if (customer != null)
            { System.out.println("\nCustomer Information:");
            System.out.println(customer);
            System.out.println("Rented Videos:");
            customer.getRentedVideos().forEach(System.out::println);
        }
    }
    public static void main(String[] args) {
        RentalSystem rentalSystem = new RentalSystem();

        // Add videos to inventory
        System.out.println("### Initializing the Video Rental System ###");
        Video v1 = new Video("V001", "3 Idiots", "Comedy-Drama");
        Video v2 = new Video("V002", "Dilwale Dulhania Le Jayenge", "Romance");
        Video v3 = new Video("V003", "Sholay", "Action-Adventure");
        Video v4 = new Video("V004", "PK", "Comedy-Drama");
        Video v5 = new Video("V005", "Lagaan", "Sports-Drama");
        rentalSystem.inventory.addVideo(v1);
```

```
        rentalSystem.inventory.addVideo(v2);
        rentalSystem.inventory.addVideo(v3);
        rentalSystem.inventory.addVideo(v4);
        rentalSystem.inventory.addVideo(v5);
        // Add customers
        Customer c1 = new Customer("C001", "Rahul Sharma");
        Customer c2 = new Customer("C002", "Priya Patel");
        rentalSystem.addCustomer(c1);
        rentalSystem.addCustomer(c2);
        System.out.println("\n### Initial Inventory ###");
        rentalSystem.displayAvailableVideos();
        System.out.println("\n### Renting Videos ###");
        // Rahul rents 3 Idiots
        rentalSystem.rentVideo("C001", "V001");
        // Priya rents DDLJ
        rentalSystem.rentVideo("C002", "V002");
        // Try to rent already rented video
        System.out.println("\n### Attempting to rent an unavailable video ###");
        rentalSystem.rentVideo("C002", "V001");
        System.out.println("\n### Checking Customer Information ###");
        rentalSystem.displayCustomerInfo("C001");
        rentalSystem.displayCustomerInfo("C002");
        System.out.println("\n### Current Available Videos ###");
        rentalSystem.displayAvailableVideos();
        System.out.println("\n### Returning Videos ###");
        // Rahul returns 3 Idiots
        rentalSystem.returnVideo("C001", "V001");
        System.out.println("\n### Final Inventory Status ###");
        rentalSystem.displayAvailableVideos();
    }
}
```

Output:

```
### Initializing the Video Rental System ###

### Initial Inventory ###
Available Videos:
ID: V001, Title: 3 Idiots, Genre: Comedy-Drama, Available: Yes
ID: V002, Title: Dilwale Dulhania Le Jayenge, Genre: Romance, Available: Yes
ID: V003, Title: Sholay, Genre: Action-Adventure, Available: Yes
ID: V004, Title: PK, Genre: Comedy-Drama, Available: Yes
ID: V005, Title: Lagaan, Genre: Sports-Drama, Available: Yes

### Renting Videos ###
Video '3 Idiots' rented to customer 'Rahul Sharma'
Video 'Dilwale Dulhania Le Jayenge' rented to customer 'Priya Patel'

### Attempting to rent an unavailable video ###
Video is not available!

### Checking Customer Information ###
Customer Information:
Customer ID: C001, Name: Rahul Sharma, Videos Rented: 1
Rented Videos:
ID: V001, Title: 3 Idiots, Genre: Comedy-Drama, Available: No
```

```
Customer Information:
Customer ID: C002, Name: Priya Patel, Videos Rented: 1
Rented Videos:
ID: V002, Title: Dilwale Dulhania Le Jayenge, Genre: Romance, Available: No

### Current Available Videos ###
Available Videos:
ID: V003, Title: Sholay, Genre: Action-Adventure, Available: Yes
ID: V004, Title: PK, Genre: Comedy-Drama, Available: Yes
ID: V005, Title: Lagaan, Genre: Sports-Drama, Available: Yes

### Returning Videos ###
Video '3 Idiots' returned by customer 'Rahul Sharma'

### Final Inventory Status ###
Available Videos:
ID: V001, Title: 3 Idiots, Genre: Comedy-Drama, Available: Yes
ID: V003, Title: Sholay, Genre: Action-Adventure, Available: Yes
ID: V004, Title: PK, Genre: Comedy-Drama, Available: Yes
ID: V005, Title: Lagaan, Genre: Sports-Drama, Available: Yes
```

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.