Write a Java program to create a class called "Reservation" with attributes for reservation ID, customer name, and date. Create subclasses "ResortReservation" and "RailwayReservation" that add specific attributes like room number for hotels and seat number for flights. Implement methods to check reservation status and modify reservation details.

**Sample Solution:**

**Java Code:**

**Reservation.java**

```java
// Import necessary classes for date handling
import java.time.LocalDate;

// Define the Reservation class
public class Reservation {
    // Attributes for reservation ID, customer name, and date
    private String reservationId;
    private String customerName;
    private LocalDate date;

    // Constructor to initialize the Reservation object
    public Reservation(String reservationId, String customerName, LocalDate date) {
        this.reservationId = reservationId;
        this.customerName = customerName;
        this.date = date;
    }

    // Method to check reservation status
    public void checkReservationStatus() {
        System.out.println("Reservation ID: " + reservationId);
        System.out.println("Customer Name: " + customerName);
        System.out.println("Date: " + date);
        System.out.println("Status: Confirmed");
    }

    // Method to modify reservation details
    public void modifyReservation(String newCustomerName, LocalDate newDate) {
        this.customerName = newCustomerName;
        this.date = newDate;
        System.out.println("Reservation modified successfully.");
    }

    // Getters for the attributes
    public String getReservationId() {
        return reservationId;
    }

    public String getCustomerName() {
        return customerName;
    }

    public LocalDate getDate() {
        return date;
    }
}
```

### Explanat

- Defi
- Attril

    - private String reservationId: Stores the reservation ID.
    - private String customerName: Stores the customer's name.
    - private LocalDate date: Stores the reservation date.
- Constructor to initialize the Reservation object: Initializes the reservationId, customerName, and date attributes with the provided values.
- Method to check reservation status:
    - public void checkReservationStatus(): Prints the reservation ID, customer name, date, and a status message ("Confirmed") to the console.
- Method to modify reservation details:
    - public void modifyReservation(String newCustomerName, LocalDate newDate): Updates the customerName and date attributes with the new values provided and prints a confirmation message to the console.
- Getters for the attributes:
    - public String getReservationId(): Returns the reservation ID.
    - public String getCustomerName(): Returns the customer's name.
    - public LocalDate getDate(): Returns the reservation date.

**ResortReservation.java**

```java
// Import necessary classes for date handling
import java.time.LocalDate;
// Define the ResortReservation subclass that extends Reservation
public class ResortReservation extends Reservation {
    // Additional attribute for the room number
    private int roomNumber;

    // Constructor to initialize the ResortReservation object
    public ResortReservation(String reservationId, String customerName, LocalDate date, int roomNumber
        super(reservationId, customerName, date); // Call the superclass constructor
        this.roomNumber = roomNumber;
    }

    // Method to check reservation status including room number
    @Override
    public void checkReservationStatus() {
        super.checkReservationStatus(); // Call the superclass method
        System.out.println("Room Number: " + roomNumber);
    }

    // Method to modify reservation details including room number
    public void modifyReservation(String newCustomerName, LocalDate newDate, int newRoomNumber) {
        super.modifyReservation(newCustomerName, newDate); // Call the superclass method
        this.roomNumber = newRoomNumber;
        System.out.println("Room Number updated successfully.");
    }

    // Getter for the room number
    public int getRoomNumber() {
        return roomNumber;
    }
}
```

**Explanation:**

- Define the ResortReservation subclass that extends Reservation: The ResortReservation class is defined as a subclass of Reservation, inheriting its attributes and methods.
- Additional attribute for the room number:
    - private int roomNumber: Stores the room number specific to resort reservations.

- Constructor to initialize the ResortReservation object:
    - Initializes the reservationId, customerName, and date attributes by calling the superclass (Reservation) constructor.
    - Initializes the roomNumber attribute with the provided value.
- Method to check reservation status including room number:
    - @Override: Indicates that this method overrides the checkReservationStatus method from the Reservation class.
    - public void checkReservationStatus(): Calls the superclass method to print the reservation details and then prints the room number.
- Method to modify reservation details including room number:
    - public void modifyReservation(String newCustomerName, LocalDate newDate, int newRoomNumber): Updates the customerName and date attributes by calling the superclass method and updates the roomNumber attribute with the new value. Prints a confirmation message for the room number update.
- Getter for the room number:
    - public int getRoomNumber(): Returns the room number.

**RailwayReservation.java**

```
// Import necessary classes for date handling
import java.time.LocalDate;
// Define the RailwayReservation subclass that extends Reservation
public class RailwayReservation extends Reservation {
    // Additional attribute for the seat number
    private int seatNumber;

    // Constructor to initialize the RailwayReservation object
    public RailwayReservation(String reservationId, String customerName, LocalDate date, int seatNumbe
        super(reservationId, customerName, date); // Call the superclass constructor
        this.seatNumber = seatNumber;
    }

    // Method to check reservation status including seat number
    @Override
    public void checkReservationStatus() {
        super.checkReservationStatus(); // Call the superclass method
        System.out.println("Seat Number: " + seatNumber);
    }

    // Method to modify reservation details including seat number
    public void modifyReservation(String newCustomerName, LocalDate newDate, int newSeatNumber) {
        super.modifyReservation(newCustomerName, newDate); // Call the superclass method
        this.seatNumber = newSeatNumber;
        System.out.println("Seat Number updated successfully.");
    }

    // Getter for the seat number
    public int getSeatNumber() {
        return seatNumber;
    }
}
```

**Explanation:**

- Define the RailwayReservation subclass that extends Reservation: The RailwayReservation class is defined as a subclass of Reservation, inheriting its attributes and methods.
- Additional attribute for the seat number:
    - private int seatNumber: Defines an attribute to store the seat number specific to railway reservations.
- Constructor to initialize the RailwayReservation object:

    - Calls the superclass (Reservation) constructor to initialize reservationId, customerName, and date.
    - Initializes the seatNumber attribute with the provided value.
- Method to check reservation status including seat number:
    - @Override: Indicates this method overrides the checkReservationStatus method from the Reservation class.
    - Calls the superclass method to print the general reservation details and then prints the seat number.
- Method to modify reservation details including seat number:
    - Updates customerName and date by calling the superclass method.
    - Updates the seatNumber attribute with the new value and prints a confirmation message for the seat number update.
- Getter for the seat number:
    - public int getSeatNumber(): Returns the seat number attribute.

**Main.java**

```
// Import necessary classes for date handling
import java.time.LocalDate;

public class Main {
    public static void main(String[] args) {
        // Create a ResortReservation object
        ResortReservation resortReservation = new ResortReservation("RSV001", "Celestino Aspasia", Loc
        // Display the resort reservation details and status
        resortReservation.checkReservationStatus();
        System.out.println();

        // Modify the resort reservation details
        resortReservation.modifyReservation("Celestino Aspasia", LocalDate.of(2024, 6, 15), 102);
        resortReservation.checkReservationStatus();
```

```
            System.out.println();

            // Create a RailwayReservation object
            RailwayReservation railwayReservation = new RailwayReservation("RSV002", "John Paul Pythios",
            // Display the railway reservation details and status
            railwayReservation.checkReservationStatus();
            System.out.println();

            // Modify the railway reservation details
            railwayReservation.modifyReservation("Bob Smith", LocalDate.of(2024, 6, 20), 23);
            railwayReservation.checkReservationStatus();
        }
    }
```

**Explanation:**

- Define the Main class:
  - Contains the main method to test the functionality of ResortReservation and RailwayReservation.
- Create a ResortReservation object:
  - Initializes a ResortReservation object with reservation ID "RSV001", customer name "Celestino Aspasia", current date, and room number 101.
- Display the resort reservation details and status:
  - Calls checkReservationStatus to display the reservation details including room number.
- Modify the resort reservation details:
  - Calls modifyReservation to update the customer name, date to June 15, 2024, and room number to 102.
- Create a RailwayReservation object:

  - Initializes a RailwayReservation object with reservation ID "RSV002", customer name "John Paul Pythios", current date, and seat number 22.
- Display the railway reservation details and status:
  - Calls checkReservationStatus to display the reservation details including seat number.
- Modify the railway reservation details:
  - Calls modifyReservation to update the customer name to "Bob Smith", date to June 20, 2024, and seat number to 23.
  - Displays the updated reservation details and status by calling checkReservationStatus again.

Output:

```
Reservation ID: RSV001
Customer Name: Celestino Aspasia
Date: 2021-06-10
Status: Confirmed
Room Number: 101

Reservation modified successfully.
Room Number updated successfully.
Reservation ID: RSV001
Customer Name: Celestino Aspasia
Date: 2021-06-15
Status: Confirmed
Room Number: 102

Reservation ID: RSV002
Customer Name: John Paul Pythios
Date: 2021-06-10
Status: Confirmed
Seat Number: 22

Reservation modified successfully.
Seat Number updated successfully.
Reservation ID: RSV002
```

```
Customer Name: Bob Smith
Date: 2021-06-20
Status: Confirmed
Seat Number: 23
```

**Java Code Editor:**