```
public class AirlineTest {
   public static void main(String[] args) {
      Flight flight = new Flight(""123""); // Creating a single flight
      Scanner scanner = new Scanner(System.in);
      while (true) {
         System.out.println(""\nMenu:"");
         System.out.println(""1. Reserve a ticket"");
         System.out.println(""2. Cancel a reservation"");
         System.out.println(""3. Check reservation status"");
         System.out.println(""4. Display passengers"");
         System.out.println(""5. Check double-bookings"");
         System.out.println(""6. Exit"");
         System.out.print(""Enter your choice: "");
         int choice = scanner.nextInt();
         scanner.nextLine();
         switch (choice) {
         case 1:
               System.out.print(""Enter passenger name: "");
               String pasName = scanner.nextLine();
               System.out.print(""Enter passenger ID: "");
              ...
               Passenger newPassenger = new Passenger(pasName,pasID,pasNr);
               flight.addPassenger(newPassenger);
               System.out.println(""Ticket reserved for "" + pasName);
               break;
            case 2:
               System.out.print(""Enter passenger name to cancel reservation: "");
               ....
               Passenger passengerD = new Passenger(pasNameToCancel,pasIDToCancel,pasNrToCancel);
               if (flight.removePassenger(passengerD)) {
                  System.out.println(""Reservation canceled for "" + pasNameToCancel);
               } else {
                  System.out.println(""Passenger "" + pasNameToCancel + "" not found."");
               }
               break;
              case 3:
               System.out.print(""Enter passenger name to check reservation: "");
               ....
               Passenger passengerC = new Passenger(pasNameToCheck,pasIDToCheck,pasNrToCheck);
               if (flight.hasPassenger(passengerC))
                  System.out.println(""Reservation found for "" + passengerC);
               else
                  System.out.println(""No reservation found for "" + passengerC);
               break;
            case 4:
               flight.displayPassengers();
               break;
            case 5:
               flight.displayPassengers();
               flight.hasDouble();
              System.out.println(""List of updated passengers:"");
              flight.displayPassengers();
```

```
                break;
            case 6:
                System.out.println(""Exiting..."");
                scanner.close();
                System.exit(0);
            default:
                System.out.println(""Invalid choice. Please choose again."");
        }
      }
   }
}

class Passenger {
   String name;
   String id;
   String contactnr;
   public Passenger(String p_name, String p_id, String p_contact) {
      this.name = p_name;
      this.id = p_id;
      this.contactnr = p_contact;
   }
   public String getName()          {
                        return name;
   }
etc...
  @Override
   public String toString() {
      return name + "" "" + id + "" "" +contactnr;
   }
   @Override
   public boolean equals(Object obj) {
      if (this == obj)
         return true;
      if (obj == null || getClass() != obj.getClass())
         return false;
      Passenger hold = (Passenger) obj;
      if (this.name.equals(hold.name) && this.id.equals(hold.id) && this.contactnr.equals(hold.contactnr))
        return true;
      else
         return false;
   }

class Flight {
   String flightNumber;
   MyLinkedList<Passenger> passengers;
   public Flight(String flightNumber) {
      this.flightNumber = flightNumber;
      passengers = new MyLinkedList<>();
   }
public void addPassenger(Passenger passenger) {
      passengers.append(passenger);
   }
```

```
public boolean removePassenger(Passenger passengerD) {
      if (passengers.delete(passengerD))
          return true;
      else
          return false;
   }
public boolean hasPassenger(Passenger passengerCh) {
      if (passengers.contains(passengerCh))
         return true;
      else
         return false;
   }
public void displayPassengers() {
      System.out.println(""Passengers on Flight "" + flightNumber + "":\n"" + passengers.toString());
   }
public void hasDouble() {
      passengers = passengers.checkDouble();
   }

public MyLinkedList checkDoubles()
{
 MyLinkedList rlist=new MyLinkedList();
 Node ptr = this.head;
 If (head == null)
   return rlist;
 while(ptr!=null)
 {
   boolean pres = false;
   for(Node ptrR=rlist.head; ptrR!=null; ptrR=ptrR.next)
    {
     if(((Comparable)ptr.element).compareTo((Comparable)ptrR.element)==0)
       pres=true;
    }
   if(!pres)
     rlist.append(ptr.element);
   ptr=ptr.next;
 }
 return rlist;
}
```