Write a Java program to create a class called "CustomerOrder" with attributes for order ID, customer, and order date. Create a subclass "OnlineOrder" that adds attributes for delivery address and tracking number. Implement methods to calculate delivery time based on the address and update the tracking status.

Sample Solution:

Java Code:

CustomerOrder.java

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

// Define the CustomerOrder class
public class CustomerOrder {
    // Attributes for order ID, customer, and order date
    private String orderId;
    private String customer;
    private LocalDate orderDate;

    // Constructor to initialize the CustomerOrder object
    public CustomerOrder(String orderId, String customer, LocalDate orderDate) {
        this.orderId = orderId;
        this.customer = customer;
        this.orderDate = orderDate;
    }

    // Getter for order ID
    public String getOrderId() {
        return orderId;
```

```
    }

    // Getter for customer
    public String getCustomer() {
        return customer;
    }

    // Getter for order date
    public LocalDate getOrderDate() {
        return orderDate;
    }

    // Method to display order details
    public void displayOrderDetails() {
        System.out.println("Order ID: " + orderId);
        System.out.println("Customer: " + customer);
        System.out.println("Order Date: " + orderDate);
    }
}
```

Explanat

Custome

- Attril
- Cons
- Meth
  -
  -

OnlineOrder.java

```
// Import necessary classes for date handling
import java.time.LocalDate;
import java.time.temporal.ChronoUnit;
// Define the OnlineOrder subclass that extends CustomerOrder
class OnlineOrder extends CustomerOrder {
    // Additional attributes for delivery address and tracking number
    private String deliveryAddress;
    private String trackingNumber;

    // Constructor to initialize the OnlineOrder object
    public OnlineOrder(String orderId, String customer, LocalDate orderDate, String deliveryAddress, S
        // Call the superclass constructor to initialize common attributes
        super(orderId, customer, orderDate);
        this.deliveryAddress = deliveryAddress;
        this.trackingNumber = trackingNumber;
    }

    // Getter for delivery address
    public String getDeliveryAddress() {
        return deliveryAddress;
    }

    // Getter for tracking number
    public String getTrackingNumber() {
        return trackingNumber;
    }

    // Method to calculate delivery time based on the address (dummy logic for demonstration)
    public int calculateDeliveryTime() {
```

```
        // Dummy logic: Assuming delivery time is based on the length of the address string
        return deliveryAddress.length() % 10 + 1; // Just a placeholder logic
    }

    // Method to update the tracking status (dummy logic for demonstration)
    public void updateTrackingStatus(String newStatus) {
        // Dummy logic: Print the updated tracking status
        System.out.println("Tracking Number: " + trackingNumber + " - Status: " + newStatus);
    }

    // Override the displayOrderDetails method to include additional details
    @Override
    public void displayOrderDetails() {
        // Call the superclass method to display common details
        super.displayOrderDetails();
        // Display additional details for online order
        System.out.println("Delivery Address: " + deliveryAddress);
        System.out.println("Tracking Number: " + trackingNumber);
    }
}
```

Explanation:

OnlineOrder Subclass:

- Extends CustomerOrder.
- Additional Attributes: deliveryAddress and trackingNumber.
- Constructor: Initializes the attributes, calling the superclass constructor for the common attributes.
- Methods:
    - calculateDeliveryTime(): Dummy logic to calculate delivery time based on the address.

- updateTrackingStatus(String newStatus): Dummy logic to update the tracking status.
- Overrides displayOrderDetails() to include additional details for online orders.
- Getters: getDeliveryAddress(), getTrackingNumber().

Main.java

```
// Import necessary classes for date handling
import java.time.LocalDate;
import java.time.temporal.ChronoUnit;
// Main class to test the CustomerOrder and OnlineOrder classes
public class Main {
    public static void main(String[] args) {
        // Create a CustomerOrder object
        CustomerOrder order = new CustomerOrder("ORD023", "Asih Wanjiku", LocalDate.now());
        // Display the order details
        order.displayOrderDetails();
        System.out.println();

        // Create an OnlineOrder object
        OnlineOrder onlineOrder = new OnlineOrder("ORD034", "Kai Biserka", LocalDate.now(), "123 ABC S
        // Display the online order details
        onlineOrder.displayOrderDetails();
        // Calculate and display the delivery time
        int deliveryTime = onlineOrder.calculateDeliveryTime();
        System.out.println("Estimated Delivery Time: " + deliveryTime + " days");
        // Update and display the tracking status
        onlineOrder.updateTrackingStatus("In Transit");
    }
}
```

Explanation:

- Main Method:
  - public static void main(String[] args): The entry point of the program.
- Creating a CustomerOrder Object:
  - CustomerOrder order = new CustomerOrder("ORD023", "Asih Wanjiku", LocalDate.now());: Creates a CustomerOrder object with order ID "ORD023", customer name "Asih Wanjiku", and the current date.
- Displaying CustomerOrder Details:
  - order.displayOrderDetails();: Calls the method to display details of the CustomerOrder object.
  - System.out.println();: Prints an empty line for separation.
- Creating an OnlineOrder Object:
  - OnlineOrder onlineOrder = new OnlineOrder("ORD034", "Kai Biserka", LocalDate.now(), "123 ABC Street, Springfield", "STR455");: Creates an OnlineOrder object with order ID "ORD034", customer name "Kai Biserka", current date, delivery address "123 ABC Street, Springfield", and tracking number "STR455".
- Displaying OnlineOrder Details:
  - onlineOrder.displayOrderDetails();: Calls the method to display details of the OnlineOrder object.
- Calculating and Displaying Delivery Time:
  - int deliveryTime = onlineOrder.calculateDeliveryTime();: Calls the method to calculate the delivery time based on the delivery address.
  - System.out.println("Estimated Delivery Time: " + deliveryTime + " days");: Prints the estimated delivery time.
- Updating and Displaying Tracking Status:
  - onlineOrder.updateTrackingStatus("In Transit");: Calls the method to update the tracking status to "In Transit".

Output:

```
Order ID: ORD023
Customer: Asih Wanjiku
Order Date: 2024-06-10

Order ID: ORD034
Customer: Kai Biserka
Order Date: 2024-06-10
Delivery Address: 123 ABC Street, Springfield
Tracking Number: STR455
Estimated Delivery Time: 8 days
Tracking Number: STR455 - Status: In Transit
```

Java Code Editor: