4. For simulating alarms, use a simple data structure (e.g., a List or Map) to store alarms, and group them based on the interface name and alarm type

```
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

class Alarm {
    private final String interfaceName;
    private final String alarmType;
    private final String description;
    private final long timestamp;

    public Alarm(String interfaceName, String alarmType, String description) {
        this.interfaceName = interfaceName;
        this.alarmType = alarmType;
        this.description = description;
        this.timestamp = System.currentTimeMillis();
    }

    public String getInterfaceName() {
        return interfaceName;
    }

    public String getAlarmType() {
        return alarmType;
    }

    public String getDescription() {
        return description;
    }

    public long getTimestamp() {
        return timestamp;
    }

    @Override
    public String toString() {
        return "Alarm{" +
                "interfaceName='" + interfaceName + '\'' +
                ", alarmType='" + alarmType + '\'' +
                ", description='" + description + '\'' +
                ", timestamp=" + timestamp +
                '}';
    }
}

class AlarmManager {
    private final Map<String, List<Alarm>> alarmMap = new HashMap<>();

    // Add an alarm to the map, grouping by interfaceName and alarmType
    public void addAlarm(String interfaceName, String alarmType, String description) {
        String key = interfaceName + ":" + alarmType;
        alarmMap.computeIfAbsent(key, k -> new ArrayList<>()).add(new Alarm(interfaceName,
alarmType, description));
```

```
        System.out.println("Alarm added for " + interfaceName + " - " + alarmType);
    }

    // Retrieve all alarms for a specific interface and alarm type
    public List<Alarm> getAlarmsByGroup(String interfaceName, String alarmType) {
        String key = interfaceName + ":" + alarmType;
        return alarmMap.getOrDefault(key, new ArrayList<>());
    }

    // Display all stored alarms grouped by interface and alarm type
    public void displayAllAlarms() {
        for (Map.Entry<String, List<Alarm>> entry : alarmMap.entrySet()) {
            System.out.println("Alarms for group: " + entry.getKey());
            for (Alarm alarm : entry.getValue()) {
                System.out.println(alarm);
            }
        }
    }
}

public class AlarmSimulation {
    public static void main(String[] args) {
        AlarmManager manager = new AlarmManager();

        // Adding alarms for different interfaces and types
        manager.addAlarm("eth0", "LINK_DOWN", "Link went down unexpectedly");
        manager.addAlarm("eth0", "HIGH_TEMP", "Temperature exceeds threshold");
        manager.addAlarm("eth1", "LINK_DOWN", "Link went down during maintenance");
        manager.addAlarm("eth1", "HIGH_CPU", "CPU usage is above 90%");

        // Retrieve alarms for specific group
        System.out.println("\nAlarms for eth0 - LINK_DOWN:");
        List<Alarm> eth0LinkDownAlarms = manager.getAlarmsByGroup("eth0", "LINK_DOWN");
        eth0LinkDownAlarms.forEach(System.out::println);

        // Display all alarms grouped by interface and alarm type
        System.out.println("\nAll grouped alarms:");
        manager.displayAllAlarms();
    }
}
```

Output:

```
Alarm added for eth0 - LINK_DOWN
Alarm added for eth0 - HIGH_TEMP
Alarm added for eth1 - LINK_DOWN
Alarm added for eth1 - HIGH_CPU

Alarms for eth0 - LINK_DOWN:
Alarm{interfaceName='eth0', alarmType='LINK_DOWN', description='Link went down unexpectedly',
timestamp=...}

All grouped alarms:
Alarms for group: eth0:LINK_DOWN
Alarm{interfaceName='eth0', alarmType='LINK_DOWN', description='Link went down unexpectedly',
```

```
timestamp=...}
Alarms for group: eth0:HIGH_TEMP
Alarm{interfaceName='eth0', alarmType='HIGH_TEMP', description='Temperature exceeds threshold',
timestamp=...}
Alarms for group: eth1:LINK_DOWN
Alarm{interfaceName='eth1', alarmType='LINK_DOWN', description='Link went down during maintenance',
timestamp=...}
Alarms for group: eth1:HIGH_CPU
Alarm{interfaceName='eth1', alarmType='HIGH_CPU', description='CPU usage is above 90%',
timestamp=...}
```