1.) โจทย์: ระบบพนักงาน (Employee System)

สถานการณ์: บริษัทแห่งหนึ่งต้องการระบบจัดการพนักงาน โดยพนักงานทุกคนมีชื่อและเงินเดือนเหมือนกัน แต่การทำงานของแต่ละตำแหน่งไม่เหมือนกัน

คำสั่ง: จงเขียนโปรแกรมภาษา Java โดยมีข้อกำหนดดังนี้:

1. **สร้างคลาสแม่ Employee:**
    - มีตัวแปร private String name และ private double salary
    - มี **Constructor** รับค่า name และ salary
    - มีเมธอด work() ที่พิมพ์ว่า "[ชื่อ] is doing general work."
    - มีเมธอด getSalary() คืนค่าเงินเดือน
2. **สร้างคลาสลูก Programmer (สืบทอดจาก Employee):**
    - มี **Constructor** ที่รับ name และ salary แล้วส่งต่อให้คลาสแม่ (ใช้ super)
    - **Override** เมธอด work() ให้พิมพ์ว่า "[ชื่อ] is coding Java application."
3. **สร้างคลาสลูก Manager (สืบทอดจาก Employee):**
    - มี **Constructor** ที่รับ name และ salary แล้วส่งต่อให้คลาสแม่ (ใช้ super)
    - **Override** เมธอด work() ให้พิมพ์ว่า "[ชื่อ] is managing the team."
4. **สร้างคลาส MainClass:**
    - สร้าง **Array ของ Employee** ขนาด 3 ช่อง (Employee[] staff = new Employee[3];)
    - ช่องที่ 0: ใส่ Employee ธรรมดา (ชื่อ "Somchai", เงินเดือน 15000)
    - ช่องที่ 1: ใส่ Programmer (ชื่อ "Somsri", เงินเดือน 30000)
    - ช่องที่ 2: ใส่ Manager (ชื่อ "Somboon", เงินเดือน 50000)
    - ใช้ **Loop** (for หรือ for-each) วนลูปเพื่อเรียกเมธอด work() ของทุกคนออกมา

ตัวอย่างผลการรัน
Somchai is doing general work. Somsri is coding Java application. Somboon is managing the team.

2.) โจทย์: Smart Device System

สถานการณ์: บ้านอัจฉริยะต้องการระบบควบคุมอุปกรณ์ต่างๆ ซึ่งอุปกรณ์ทุกตัวต้องมีชื่อและสั่งงานได้ แต่อุปกรณ์บางตัวเป็นแบบไร้สายที่ต้องชาร์จแบตเตอรี่

คำสั่ง:

1. **สร้าง Abstract Class Device:**
    - มีตัวแปร private String name
    - มี Constructor รับค่า name
    - มีเมธอด getName()
    - มี **Abstract Method** void turnOn() (สั่งเปิดเครื่อง)
2. **สร้าง Interface Rechargeable:**
    - มีเมธอด void chargeBattery() (สั่งชาร์จไฟ)
3. **สร้างคลาส RobotVacuum (หุ่นยนต์ดูดฝุ่น):**
    - สืบทอดจาก Device และใช้ Rechargeable (เพราะต้องชาร์จ)
    - turnOn() -> พิมพ์ "[ชื่อ] is cleaning the floor."
    - chargeBattery() -> พิมพ์ "[ชื่อ] is docking to charge station."
4. **สร้างคลาส SmartLight (หลอดไฟอัจฉริยะ):**
    - สืบทอดจาก Device อย่างเดียว (เพราะต่อไฟบ้าน ไม่ต้องชาร์จ)
    - turnOn() -> พิมพ์ "[ชื่อ] is lighting up the room."
5. **MainClass:**
    - สร้างอุปกรณ์ทั้ง 2 ตัว
    - สั่ง turnOn()
    - เช็คว่าตัวไหนชาร์จได้ (instanceof Rechargeable) ให้สั่งชาร์จด้วย

ตัวอย่างผลการรัน
--- Starting Devices --- Robo-Clean is cleaning the floor. Robo-Clean is docking to charge station. Living Room Light is lighting up the room.

โจทย์ข้อที่ 1: Abstract Class

ธีม: ระบบค่าจอดรถ (Parking System) **คอนเซปต์:** ยานพาหนะแต่ละชนิดมีอัตราค่าจอดรถไม่เท่ากัน

สิ่งที่คุณต้องทำ:

1. **สร้าง Abstract Class ชื่อ Vehicle**
    - **ตัวแปร:** ประกาศตัวแปร private String licensePlate (ทะเบียนรถ)
    - **Constructor:** รับค่า licensePlate เข้ามาเพื่อกำหนดค่าให้ตัวแปร
    - **Getter:** สร้างเมธอด getLicensePlate()
    - **Abstract Method:** สร้างเมธอดชื่อ calculateParkingFee(int hours) ที่รับจำนวนชั่วโมง (int) และคืนค่าเป็นค่าจอดรถ (double) แต่ยังไม่ต้องเขียนไส้ใน
2. **สร้างคลาส Car (รถยนต์) สืบทอดจาก Vehicle**
    - **Constructor:** รับค่า licensePlate แล้วส่งต่อให้ Constructor ของแม่ (super)
    - **Override Method:** เขียนทับเมธอด calculateParkingFee
        - *สูตรคำนวณ:* ชั่วโมงละ **50 บาท** (คืนค่า hours * 50)
3. **สร้างคลาส Motorcycle (มอเตอร์ไซค์) สืบทอดจาก Vehicle**
    - **Constructor:** รับค่า licensePlate แล้วส่งต่อให้ Constructor ของแม่ (super)
    - **Override Method:** เขียนทับเมธอด calculateParkingFee
        - *สูตรคำนวณ:* ชั่วโมงละ **20 บาท** (คืนค่า hours * 20)
4. **สร้างคลาส MainClass**
    - สร้างตัวแปร Vehicle ชื่อ v1 เก็บ object ของ Car (ทะเบียน "A-123")
    - สร้างตัวแปร Vehicle ชื่อ v2 เก็บ object ของ Motorcycle (ทะเบียน "B-999")
    - สั่งให้ v1 และ v2 คำนวณค่าจอดรถสำหรับ **3 ชั่วโมง** และแสดงผลออกทางหน้าจอ

ตัวอย่างการรันโปรแกรม
Vehicle: A-123 fee for 3 hours: 150.0 Vehicle: B-999 fee for 3 hours: 60.0
Vehicle: 9999-BKK type Car Parking Fee for 24 hours: 1200.0 Baht

โจทย์ข้อที่ 2: Interface

ธีม: ระบบการเชื่อมต่ออุปกรณ์ (Connection System) **คอนเซปต์:** อุปกรณ์ต่างๆ เชื่อมต่อได้เหมือนกัน แต่มีวิธีการทำงานต่างกัน

สิ่งที่คุณต้องทำ:

1. **สร้าง Interface ชื่อ Connectable**
    - ประกาศเมธอดชื่อ connect() (ไม่มีการรับค่า, คืนค่าเป็น void)
    - ประกาศเมธอดชื่อ disconnect() (ไม่มีการรับค่า, คืนค่าเป็น void)
2. **สร้างคลาส BluetoothSpeaker (ลำโพงบลูทูธ)**
    - **ตัวแปร:** private String deviceName
    - **Constructor:** รับค่าชื่ออุปกรณ์
    - **Implement:** สืบทอด Interface Connectable
    - **Override connect():** พิมพ์ว่า "[ชื่อ] paired via Bluetooth."
    - **Override disconnect():** พิมพ์ว่า "[ชื่อ] disconnected."
3. **สร้างคลาส WiredKeyboard (คีย์บอร์ดมีสาย)**
    - **ตัวแปร:** private String brand
    - **Constructor:** รับค่าชื่อยี่ห้อ
    - **Implement:** สืบทอด Interface Connectable
    - **Override connect():** พิมพ์ว่า "[ยี่ห้อ] plugged into USB port."
    - **Override disconnect():** พิมพ์ว่า "[ยี่ห้อ] unplugged."
4. **สร้างคลาส MainClass**
    - สร้าง Array ของ Connectable ขนาด 2 ช่อง
    - ช่องที่ 0: ใส่ BluetoothSpeaker (ชื่อ "JBL Go")
    - ช่องที่ 1: ใส่ WiredKeyboard (ยี่ห้อ "Logitech")
    - **วนลูป (Loop):** ดึงอุปกรณ์ออกมาทีละตัวแล้วสั่ง connect() และตามด้วย disconnect()
    - *(Challenge เพิ่มเติม)*: ในลูป ลองใช้ if (obj instanceof BluetoothSpeaker) เพื่อเช็คดูว่า ถ้าเป็นลำโพง ให้พิมพ์ข้อความพิเศษเพิ่มว่า "Play sound!"

ตัวอย่างรันโปรแกรม
[JBL PartyBox] paired via Bluetooth. [JBL PartyBox] disconnected.
[Logitech K120] plugged into USB port. [Logitech K120] unplugged.
--- Device 1 --- [Marshall] paired via Bluetooth. [Marshall] disconnected. --- Device 2 --- [Razer] plugged into USB port. [Razer] unplugged.