```
package transport;
public abstract class Vehicle {
protected String id;
public Vehicle(String id) {
        this.id = id;
        System.out.println("Vehicle() constructor called");
}
public abstract void deliver(String item, String place);
}
package transport;
public class Bicycle extends Vehicle {
        public Bicycle(String id) {
        super(id);
        System.out.println("Bicycle() constructor called");
}

@Override
public void deliver(String item, String place) {
System.out.println("Delivering " + item + " to " + place + " by Bicycle.");
}
}
package transport;
public class EBike extends Bicycle {
private int battery;
public EBike(String id, int battery)
{
        super(id); this.battery = battery;
        System.out.println("EBike() constructor called");
}
@Override
Public void deliver(String item, String place) {
System.out.println("Checking battery: " + battery + "%"); super.deliver(item, place);
```

```java
}
}
package transport;
public class Drone extends Vehicle implements Payable {
public Drone(String id) {
        super(id);
        System.out.println("Drone() constructor called");
}
@Override
public void deliver(String item, String place)
{
        if (!SecurityRules.canFly(place)) {
        System.out.println("Delivery to " + place + " is blocked by security.");
        return;
}
System.out.println("Delivering " + item + " to " + place + " by Drone.");
}
@Override
public double cost(double distanceKm)
{
        return 20 * distanceKm;
}
}
package transport;
public final class SecurityRules{
private SecurityRules() {
}
public static boolean canFly(String place
 {
        return !place.equals("ExamCell");
}
}
```

```
package transport;
public interface Payable
{
        double cost(double distanceKm);
}
import transport.*;
public class Main
{
public static void main(String[] args)
{
        EBike e = new EBike("EB-101", 50);
        e.deliver("Sandwich", "Library");

        Drone d = new Drone("DR-1");
        d.deliver("Notes", "ExamCell");
        d.deliver("USB", "CSE Block");
        double bill = d.cost(5);
        System.out.println("Drone delivery cost: Rs." + bill);
}
}
```

OUTPUT:

```
Vehicle() constructor is called
Bicycle() constructor is called
EBike() constructor is called
checking battery50%
DeliveringSandwitchtoLibraryby Bicycle
Vehicle() constructor is called
Drone() constructor is called
Delivery to examcell is blocked by security
Delivering USB toCSE block by Drone
```

Drone delivery cost Rs100.0