Lab 03

CODE:

transport/Vehicle.java

```
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);


}
```

//Bicycle.java

```
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.");
}


}
```

// transport/EBike.java

```
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);
}


}
```

// Payable.java

```
public interface Payable {

double cost(double distanceKm);

}

// SecurityRules.java

public final class SecurityRules {

private SecurityRules() {

}

public static boolean canFly(String place)

{
    return !place.equals("ExamCell");
}


}
```

// Drone.java

```
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;
}


}
```

// Main.java

```
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 called

Bicycle() constructor called

EBike() constructor called

Checking battery: 50%

Delivering Sandwich to Library by Bicycle.

Vehicle() constructor called

Drone() constructor called

Delivery to ExamCell is blocked by security.

Delivering USB to CSE Block by Drone.

Drone delivery cost: Rs.100.0
```