2.2 practise:

1. Update the JavaBank.java application to implement a new light blue company color that is to be used across all of the graphical user interfaces in the Java application. The values to be used are Red: 173, Green 216 and Blue 230.

A.
```
package javabank;


import java.awt.Color;


public class JavaBank {


    // Define the company color as a constant
    public static final Color COMPANY_COLOR = new Color(173,
216, 230);


    public static void main(String[] args) {
        // Your main application logic here
        // Example: creating a new JFrame and applying the
company color
    }
}
```

2. Create the following interface in the bike project that sets the name of the bike company as an unchangeable value. It also defines the methods that must be implemented by any class that uses the interface .

A.
```
package bikeproject;


public interface BikeCompany {
    // Unchangeable value (constant)
    String COMPANY_NAME = "Oracle Bikes";


    // Methods to be implemented by any class that uses this
interface
    String getHandleBars();
    void setHandleBars(String newValue);


    String getTyres();
    void setTyres(String newValue);


    String getSeatType();
    void setSeatType(String newValue);
}
```

3. Create an interface named MountainParts that has a constant named TERRAIN that will store the String value “off_road”. The interface will define two methods that accept a String argument name newValue and two that will return the current value of an instance field. The methods are to be named: getSuspension, setSuspension, getType , setType.

A.
```
package bikeproject;


public interface MountainParts {
    // Constant declaration
    String TERRAIN = "off_road";


    // Method signatures
    String getSuspension();
    void setSuspension(String newValue);


    String getType();
    void setType(String newValue);
}
```

4. Create a RoadParts interface that has a constant named terrain that will store the String value “track_racing”. The interface will define two methods that accept a String argument name newValue and two that will return the current value of an instance field. The methods are to be named: getTyreWidth, setTyreWidth, getPostHeight, setPostHeight.

A.

```
package bikeproject;

public interface RoadParts {
    // Constant declaration
    String terrain = "track_racing";

    // Method signatures
    String getTyreWidth();
    void setTyreWidth(String newValue);

    String getPostHeight();
    void setPostHeight(String newValue);
}
```

5. Use the MountainParts interface with the MountainBike class adding any unimplemented methods required. Add the required internal code for each of the added methods.

A.
```
package bikeproject;

public interface MountainParts {
    // Constant declaration
    String TERRAIN = "off_road";
    String getSuspension();
    void setSuspension(String newValue);

    String getType();
    void setType(String newValue);
}
```

6. Use the RoadParts interface with the RoadBike class adding any unimplemented methods required. Add the required internal code for each of the added methods.

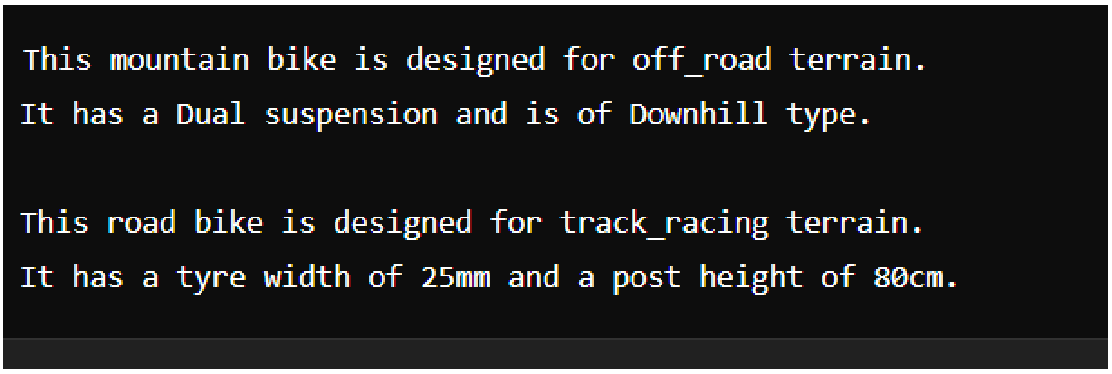
A.

```
package bikeproject;

public class RoadBike implements RoadParts {

    private String tyreWidth;
    private String postHeight;

    // Implementing methods from RoadParts
    @Override
    public String getTyreWidth() {
        return tyreWidth;
    }

    @Override
    public void setTyreWidth(String newValue) {
        this.tyreWidth = newValue;
    }

    @Override
    public String getPostHeight() {
        return postHeight;
    }

    @Override
    public void setPostHeight(String newValue) {
        this.postHeight = newValue;
    }

    // Example method to demonstrate usage
    public void printDescription() {
        System.out.println("This road bike is designed for " +
terrain + " terrain.");
```

```
        System.out.println("It has a tyre width of " + tyreWidth
+ " and a post height of " + postHeight + ".");
    }
}
```

7. Run and test your program, it should do exactly as it did before.

A.



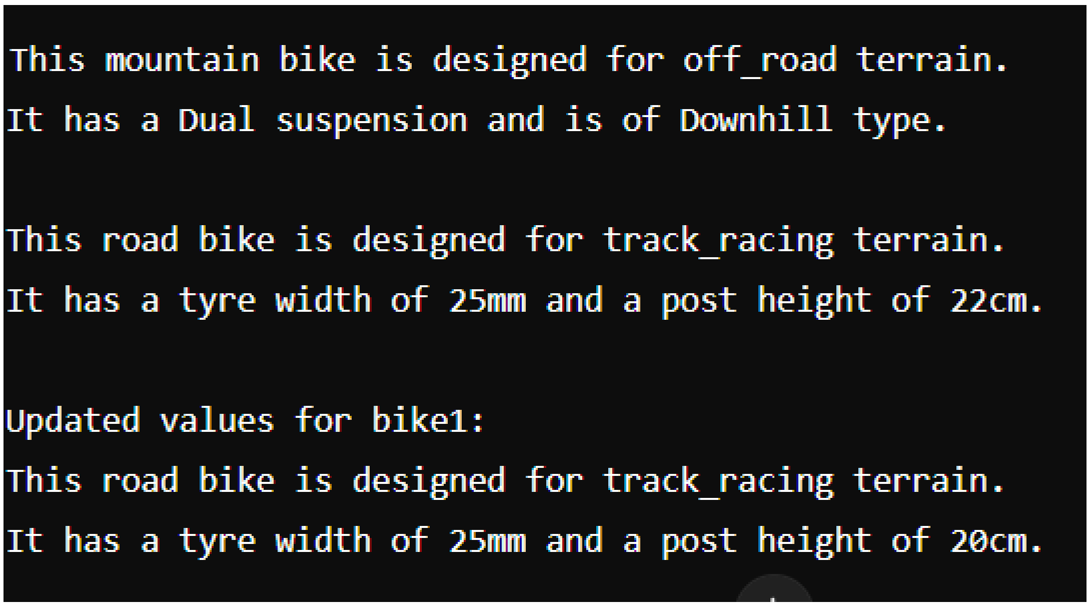
```
This mountain bike is designed for off_road terrain.
It has a Dual suspension and is of Downhill type.

This road bike is designed for track_racing terrain.
It has a tyre width of 25mm and a post height of 80cm.
```

8. At the bottom of the driver class update the height of the post for bike1 to 20 instead of 22.

A.
```
package bikeproject;


public class BikeApp {
    public static void main(String[] args) {
        // Test MountainBike
        MountainBike myMountainBike = new MountainBike();
        myMountainBike.setSuspension("Dual");
        myMountainBike.setType("Downhill");
        myMountainBike.printDescription();


        // Test RoadBike
        RoadBike bike1 = new RoadBike();
        bike1.setTyreWidth("25mm");
        bike1.setPostHeight("22cm"); // Initially set to 22cm
        bike1.printDescription();


        // Update the post height of bike1 to 20cm
        bike1.setPostHeight("20cm");
        System.out.println("Updated post height for bike1:");
        bike1.printDescription();
    }
}
```

9. Display the values of bike1 to screen to confirm the change.

A.



```
This mountain bike is designed for off_road terrain.
It has a Dual suspension and is of Downhill type.

This road bike is designed for track_racing terrain.
It has a tyre width of 25mm and a post height of 22cm.

Updated values for bike1:
This road bike is designed for track_racing terrain.
It has a tyre width of 25mm and a post height of 20cm.
```

10. Run and test your program.