Experiment – 11

Aim –Decorator design pattern (Structural pattern)

Concept -**Decorator** is a structural design pattern that lets you attach new behaviors to objects by placing these objects inside special wrapper objects that contain the behaviors.

cake.java

```java
public interface cake {
    void makeCake();
}
```

RedVelvet.java

```java
public class RedVelvet implements cake{
    @Override
    public void makeCake() {
        System.out.println("RedVelvet Cake 🍥🍥");
    }
}
```

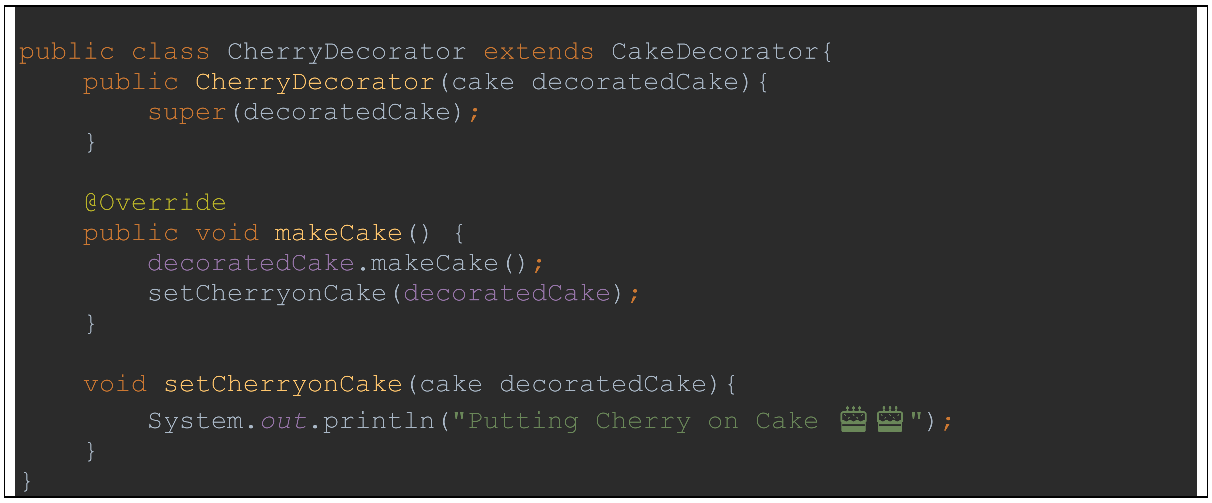
BlackForest.java

```java
public class BlackForest implements cake{
    @Override
    public void makeCake() {
        System.out.println("BlackForest Cake 🍰🍰🍰    ");
    }
}
```

CakeDecorator.java

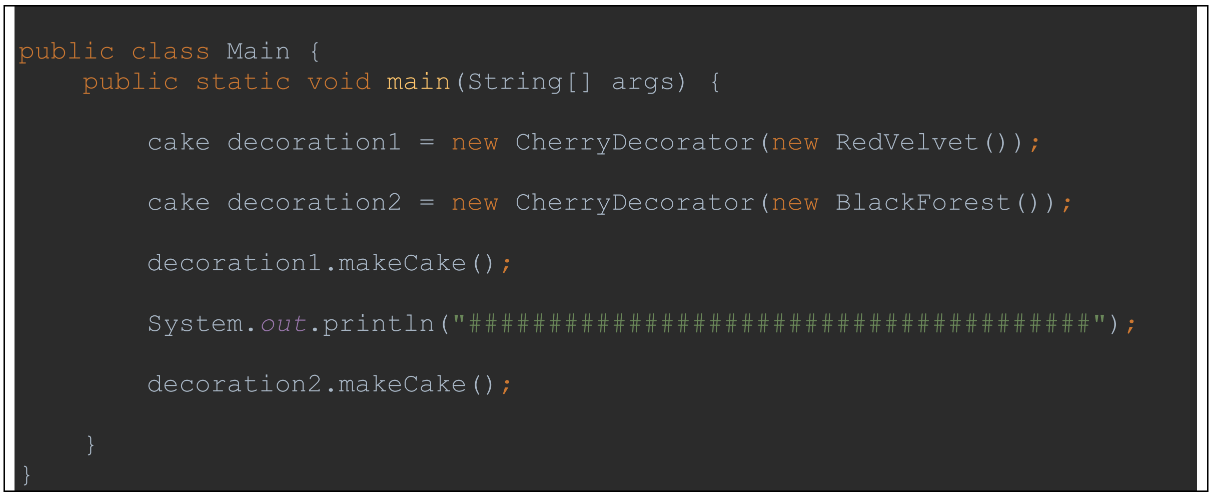
```java
public class CakeDecorator implements cake {
    cake decoratedCake;

    public CakeDecorator(cake decoratedCake){
        this.decoratedCake = decoratedCake;
    }

    @Override
    public void makeCake() {
        decoratedCake.makeCake();
    }
}
```

CherryDecorator.java



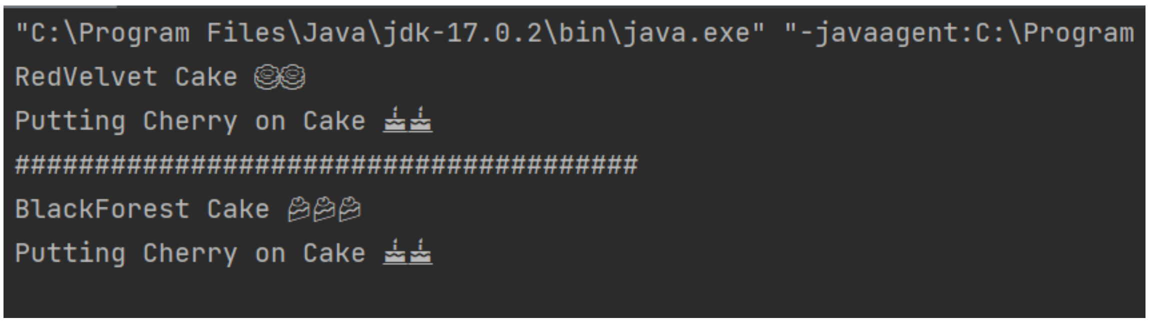
```java
public class CherryDecorator extends CakeDecorator{
    public CherryDecorator(cake decoratedCake){
        super(decoratedCake);
    }

    @Override
    public void makeCake() {
        decoratedCake.makeCake();
        setCherryonCake(decoratedCake);
    }

    void setCherryonCake(cake decoratedCake){
        System.out.println("Putting Cherry on Cake 🎂🎂");
    }
}
```

Main.java



```java
public class Main {
    public static void main(String[] args) {

        cake decoration1 = new CherryDecorator(new RedVelvet());

        cake decoration2 = new CherryDecorator(new BlackForest());

        decoration1.makeCake();

        System.out.println("#######################################");

        decoration2.makeCake();

    }
}
```

OUTPUT :



```
"C:\Program Files\Java\jdk-17.0.2\bin\java.exe" "-javaagent:C:\Program
RedVelvet Cake 🍩🍩
Putting Cherry on Cake 🎂🎂
#######################################
BlackForest Cake 🍰🍰🍰
Putting Cherry on Cake 🎂🎂
```