CS680 HomeWork – 19
Vinit R. Tambekar

Show and explain how your code looks like if the Decorator design pattern is not used.

Let's consider example of different kinds of cars
we can create class Car to define the getPrice method and then we can have a Sedan car, we can extend it to Sports car and Luxury Car.

Our psuedo code will look like :

```
public class Car{
	private int price = 5000;
	public int getPrice()
	{
		return price;
	}

}

public class SedanCar extends Car
{
	private Car car;
	 public SedanCar(Car c)
	{
		this.car = c;
	}
	public int getPrice()
	{
		return this.car.getPrice() * 2;
	}
}

public class SportsCar extends Car
{
	private boolean isSedan = false;

	public SportsCar(boolean isSedan)
	{
		this.isSedan = isSedan;
	}

	public int getPrice()
	{
```

```
            if(isSedan)
            {
                return super.getPrice() * 10;
            }

            else
            {
                return super.getPrice() * 5;
            }
        }
}

public class LuxuryCar extends Car
{
        private boolean isSedan = false;

        public LuxuryCar(boolean isSedan)
        {
            this.isSedan = isSedan;
        }

        public int getPrice()
        {
            if(isSedan)
            {
                return super.getPrice() * 20;
            }

            else
            {
                return super.getPrice() * 10;
            }
        }
}
```

Now if we want to add new class ElectricSedanSportsCar or ElectricSedanLuxuryCar it will create more complications with more condittional statements.
Having Decorator pattern is much better than having single class of every possible combination

With Decorator pattern

```
Car car = new SedanCar(new LuxuryCar(new Car));
```

Explain why Java API designers decided to use Decorator in java.io.

Similar behaviour is found in java.io
InputStream is an abstract class. Implementations of this class are like FileInputStream, BufferedInputStream, GizpInputStream,ObjectInputStream etc.

It is much beneficial to have decorator pattern to expand with any combination as per requirement instread of maining single class ObjectGzipBufferedFileInputStream, ObjectBufferedFileInputStream etc.

With Decorator Pattern
BufferedInputStream bis = new BufferedInputStream(new FileInputStream(new File(“abc.txt”)));