Refactoring

1.

```
class Library {
    private List<Book> books;

    public Library() {
        books = new ArrayList<>();
    }


    public void calculatePriceForGoldCustomers(Book book, Customer customer) {
       sout('Yo Gold Customer")
        if (book.price > 1000 || customer.age>20 || book.author==customer.author) {
            return price * 0.70;
        } else {
            return price;
        }

    }
    public void calculatePriceForSilverCustomers(Book book, Customer customer) {
       sout("Yo silver candidate")
       If (book.price==500){
            sout('wow')
        }
        if (book.price > 1000 || customer.age>30 || book.author==customer.author) {
            return price * 0.80;
        } else {
            return price;
        }

    }
```

Solution:

Duplication->Extract Method
Long Condition-> Extract method

```
class Library {
    private List<Book> books;

    public Library() {
        books = new ArrayList<>();
    }
```

```
public void calculatePriceForGoldCustomers(Book book, Customer customer) {
   sout('Yo Gold Customer")
   return applyDiscount(Book book, Customer customer, 20 ,  .70)

}
public void calculatePriceForSilverCustomers(Book book, Customer customer) {
   sout("Yo silver candidate")
   If (book.price==500){
        sout('wow')
    }
   return applyDiscount(Book book, Customer customer, 30 ,  .80)
}

public void applyDiscount(Book book, Customer customer,age_limit,charge_rate) {

    if (isDiscountApplcable(Book book, Customer customer,age_limit)) {
        return price * charge_rate;
    } else {
        return price;
    }

}
public void isDiscountApplicable(Book book, Customer customer,age_limit) {

    if (book.price > 1000 || customer.age>age_limit  || book.author==customer.name) {
        return True;
    } else {
        return False;
    }

}
```

2.

```
class Order {
    private Customer customer;
    private double totalPrice;

    public Order(Customer customer, double totalPrice) {
        this.customer = customer;
        this.totalPrice = totalPrice;
    }

    public double calculateDiscountRate() {

        if (customer.glp() > 100) {
            return  0.1;
        }
        return 0;
```

```
    }
}

class Customer {
    private String name;
    private int loyaltyPoints;
    public void PriceRecommendation(double saree_price,double shirt_price,double
panjabi_price,double hat_price){

        //Calculating recommendation rate
        saree_price=2*5*shirt_price;
        panjabi_price=saree_price+hat_price
        Recommendation_rate=panjabi_price*100;

        Sout("Hooray. done. ",Recommendation_rate);

    }

    public Customer(String name, int loyaltyPoints) {
        this.name = name;
        this.loyaltyPoints = loyaltyPoints;
    }

    public int glp() {
        return loyaltyPoints;
    }
}
```

Solution: Feature Envy-> Move field
Inappropriate naming -> Proper naming
Long Parameter-> create class
Comment->Extract method

```
class Order {
    private Customer customer;
    private double totalPrice;

    public Order(Customer customer, double totalPrice) {
        this.customer = customer;
        this.totalPrice = totalPrice;
    }

}
class RecommendedPrice{
    double saree;
    double shirt;
    double panjabi;
    double hat;
```

```
    public RecommendedPrice(double saree,double shirt,double panjabi,double hat){
        this.saree=saree;
        this.shirt=shirt;
        this.panjabi=panjabi;
        this.hat=hat;

    }

}


class Customer {
    private String name;
    private int loyaltyPoints;

    public Customer(String name, int loyaltyPoints) {
        this.name = name;
        this.loyaltyPoints = loyaltyPoints;
    }
    public void PriceRecommendation(RecommendedPrice){

        Recommendaton_rate=calculate_recommendation_rate(RecommendedPrice)

        Sout("Hooray. done. ",Recommendaton_rate);

    }

    public double calculate_recommendation_rate(RecommendedPrice){
        RecommendedPrice.saree=2*5*RecommendedPrice.shirt;
        RecommendedPrice.panjabi=RecommendedPrice.saree+RecommendedPrice.hat
        Recommendation_rate=RecommendedPrice.panjabi*100;

        return  Recommendation_rate;

    }


    public int getLoyaltyPoints() {
        return loyaltyPoints;
    }

    public double calculateDiscountRate(Customer customer, double totalPrice) {

        if (customer.getLoyaltyPoints() > 100) {
            return  0.1;
        }
        return 0;
    }
}
```