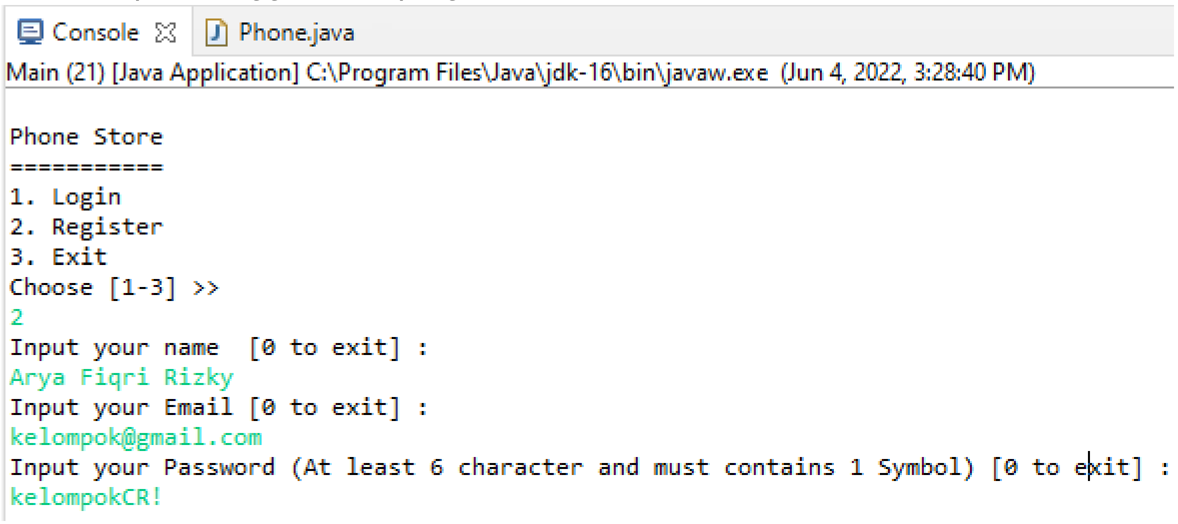
Code Reengineering Document Project

Dosen : Kornelius Irfandhi S.Kom., M.TI.
Kelas : LF01
Anggota kelompok :
● Arya Putra Kartiwa - 2440040343
● Muhammad Fiqri Febriansyah - 2440084995
● Rizky Hertama - 2440062483

A. Introduction

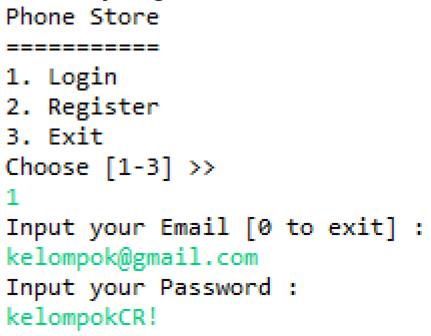
Judul Project : Phone Store App
Phone store app merupakan sebuah program yang dapat memiliki berbagai fitur seperti :

1. Register : ketika menggunakan program tersebut kita harus register untuk dapat menggunakan program.



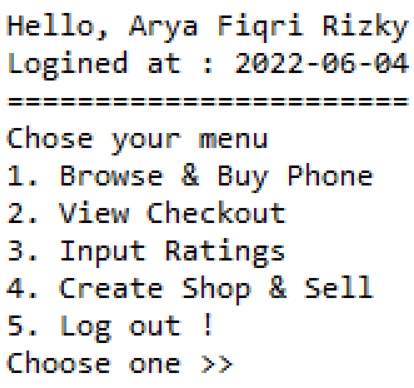
```
Main (21) [Java Application] C:\Program Files\Java\jdk-16\bin\javaw.exe (Jun 4, 2022, 3:28:40 PM)

Phone Store
===========
1. Login
2. Register
3. Exit
Choose [1-3] >>
2
Input your name  [0 to exit] :
Arya Fiqri Rizky
Input your Email [0 to exit] :
kelompok@gmail.com
Input your Password (At least 6 character and must contains 1 Symbol) [0 to exit] :
kelompokCR!
```

2. Login : setelah register maka dapat login menggunakan email dan password yang telah didaftarkan



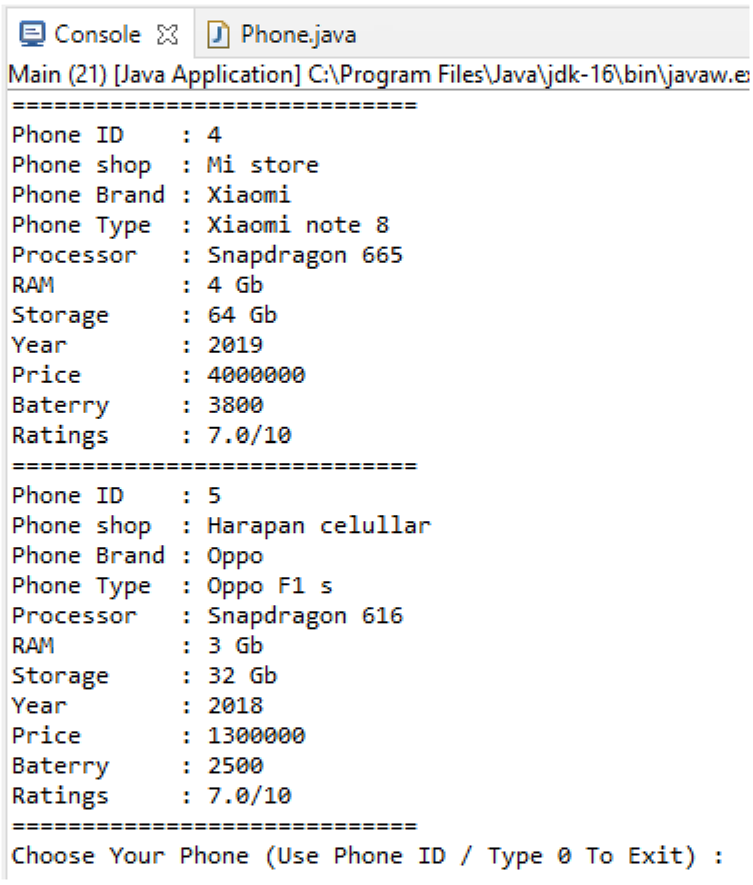
```
Phone Store
===========
1. Login
2. Register
3. Exit
Choose [1-3] >>
1
Input your Email [0 to exit] :
kelompok@gmail.com
Input your Password :
kelompokCR!
```

Setelah login akan muncul menu untuk melakukan



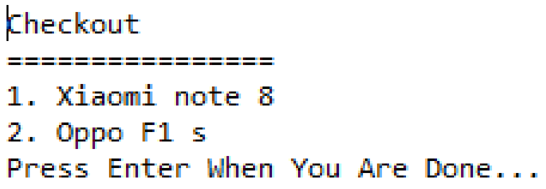
```
Hello, Arya Fiqri Rizky
Logined at : 2022-06-04
=======================
Chose your menu
1. Browse & Buy Phone
2. View Checkout
3. Input Ratings
4. Create Shop & Sell
5. Log out !
Choose one >>
```

3. Browse & buy phone : pengguna dapat membeli phone yang tersedia pada list penjualan.



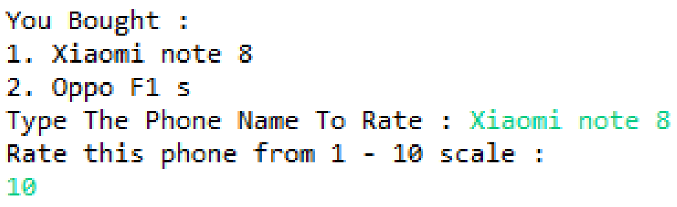
```
Main (21) [Java Application] C:\Program Files\Java\jdk-16\bin\javaw.e
=============================
Phone ID     : 4
Phone shop   : Mi store
Phone Brand  : Xiaomi
Phone Type   : Xiaomi note 8
Processor    : Snapdragon 665
RAM          : 4 Gb
Storage      : 64 Gb
Year         : 2019
Price        : 4000000
Baterry      : 3800
Ratings      : 7.0/10
=============================
Phone ID     : 5
Phone shop   : Harapan celullar
Phone Brand  : Oppo
Phone Type   : Oppo F1 s
Processor    : Snapdragon 616
RAM          : 3 Gb
Storage      : 32 Gb
Year         : 2018
Price        : 1300000
Baterry      : 2500
Ratings      : 7.0/10
=============================
Choose Your Phone (Use Phone ID / Type 0 To Exit) :
```

4. View Checkout : setelah melakukan pembelian pengguna dapat melihat barang yang sudah dibeli pada menu checkout



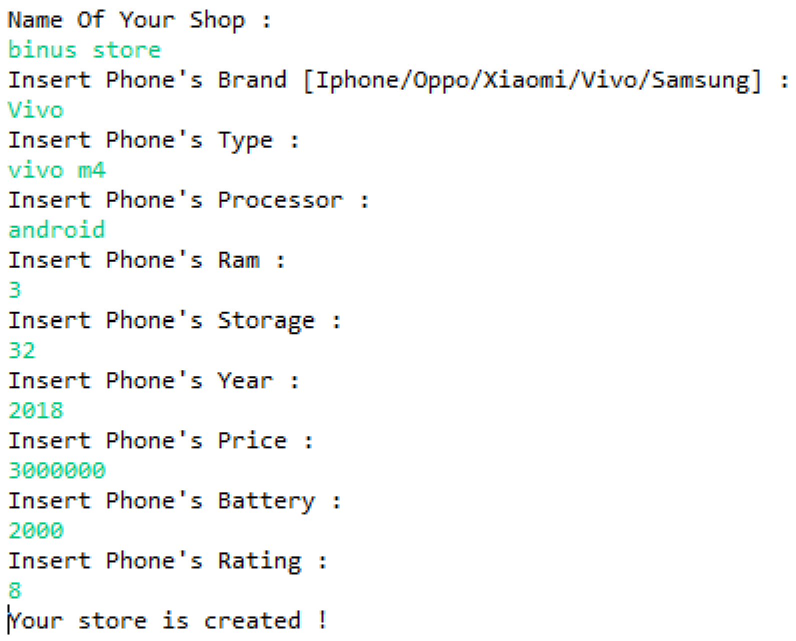
```
Checkout
================
1. Xiaomi note 8
2. Oppo F1 s
Press Enter When You Are Done...
```

5. Input ratings : pengguna dapat memasukkan input 1-10 setelah membeli produk.



```
You Bought :
1. Xiaomi note 8
2. Oppo F1 s
Type The Phone Name To Rate : Xiaomi note 8
Rate this phone from 1 - 10 scale :
10
```

6. Create shop & sell phone : pengguna dapat membuka toko dan menjual handphone.

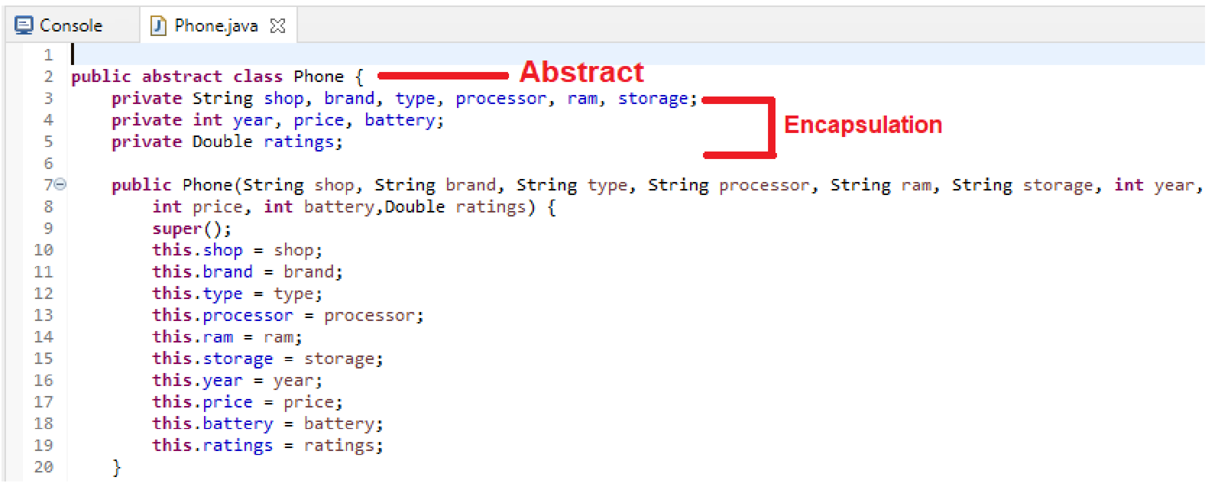


```
Name Of Your Shop :
binus store
Insert Phone's Brand [Iphone/Oppo/Xiaomi/Vivo/Samsung] :
Vivo
Insert Phone's Type :
vivo m4
Insert Phone's Processor :
android
Insert Phone's Ram :
3
Insert Phone's Storage :
32
Insert Phone's Year :
2018
Insert Phone's Price :
3000000
Insert Phone's Battery :
2000
Insert Phone's Rating :
8
Your store is created !
```

7. Logout : pengguna akan kembali ke menu login dan register apabila memilih menu logout.

B. Object-oriented Design Principles

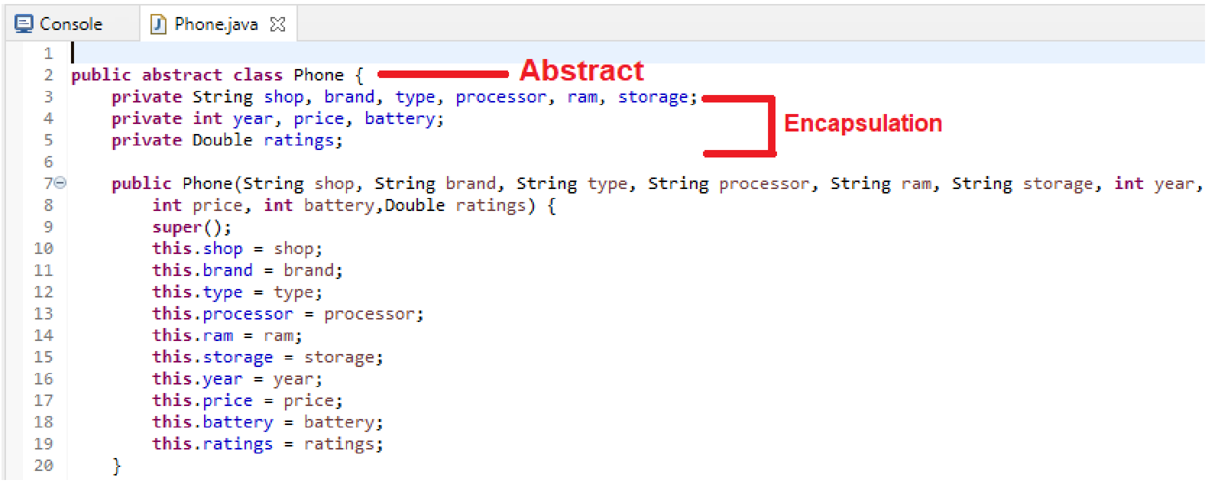
- **Encapsulation implementation**



```java

public abstract class Phone {
    private String shop, brand, type, processor, ram, storage;
    private int year, price, battery;
    private Double ratings;

    public Phone(String shop, String brand, String type, String processor, String ram, String storage, int year,
        int price, int battery,Double ratings) {
        super();
        this.shop = shop;
        this.brand = brand;
        this.type = type;
        this.processor = processor;
        this.ram = ram;
        this.storage = storage;
        this.year = year;
        this.price = price;
        this.battery = battery;
        this.ratings = ratings;
    }
```

- **Abstraction implementation**

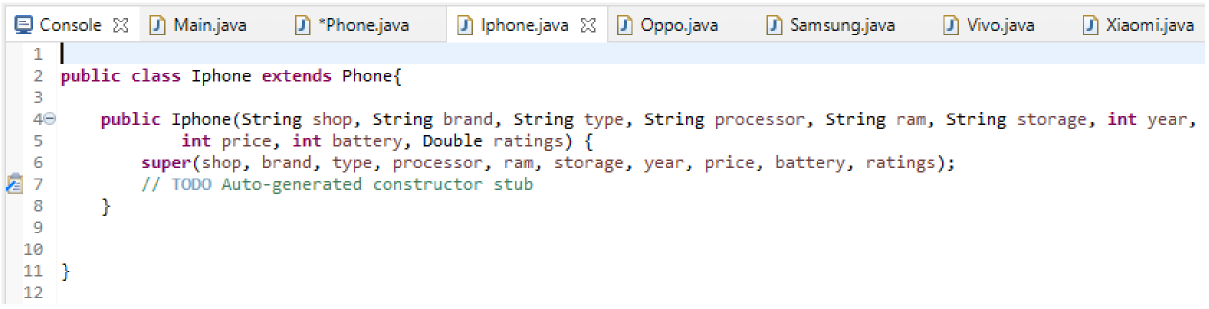
Terdapat class phone pada project kami yang berfungsi sebagai parent class dari berbagai macam merk handphone.



```java

public abstract class Phone {
    private String shop, brand, type, processor, ram, storage;
    private int year, price, battery;
    private Double ratings;

    public Phone(String shop, String brand, String type, String processor, String ram, String storage, int year,
        int price, int battery,Double ratings) {
        super();
        this.shop = shop;
        this.brand = brand;
        this.type = type;
        this.processor = processor;
        this.ram = ram;
        this.storage = storage;
        this.year = year;
        this.price = price;
        this.battery = battery;
        this.ratings = ratings;
    }
```

- **Inheritance implementation**

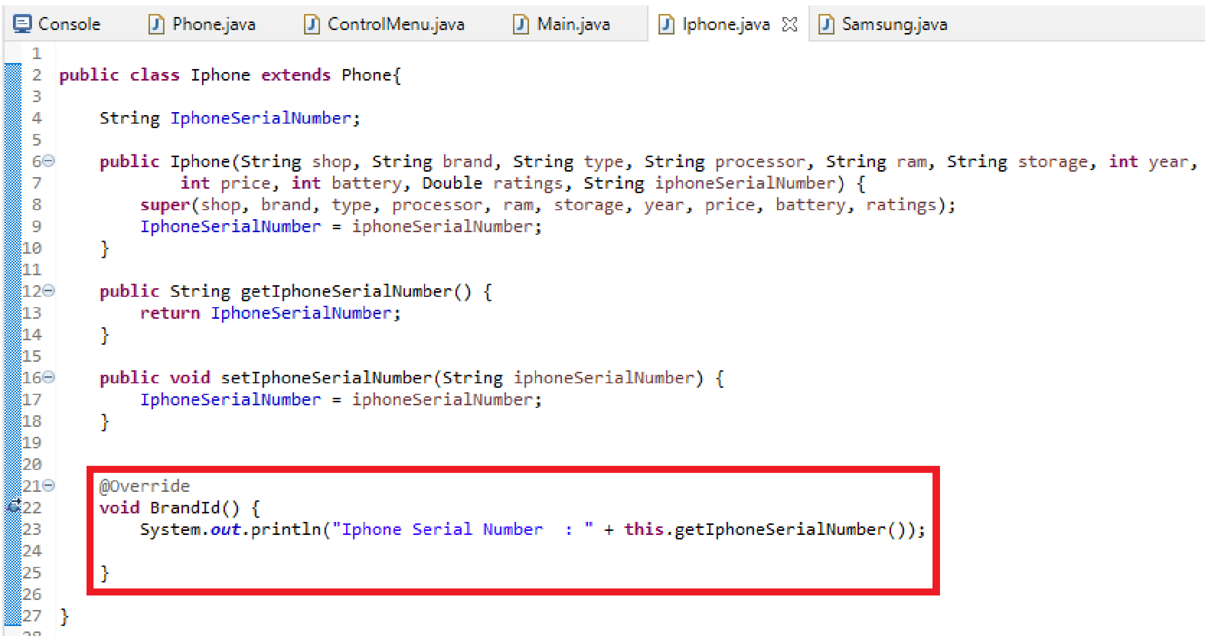
Kemudian, salah satu contohnya class iphone melakukan extends ke class phone.class iphone merupakan child class dari class phone.



```java

public class Iphone extends Phone{

    public Iphone(String shop, String brand, String type, String processor, String ram, String storage, int year,
            int price, int battery, Double ratings) {
        super(shop, brand, type, processor, ram, storage, year, price, battery, ratings);
        // TODO Auto-generated constructor stub
    }


}

```

- **Polymorphism implementation**

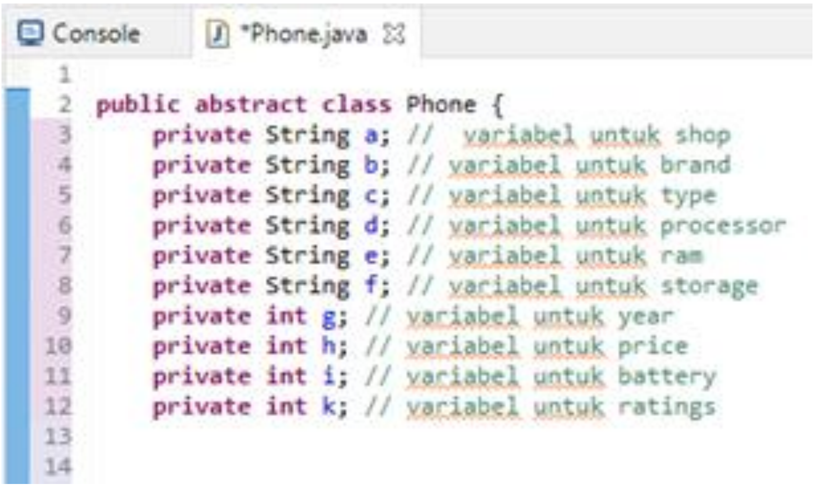
Implementasi polymorphism pada project kami terletak pada setiap class pada merk handphone yang memiliki atribut variabel “serial number”.


Console | Phone.java | ControlMenu.java | Main.java | Iphone.java | Samsung.java

```java

public class Iphone extends Phone{

    String IphoneSerialNumber;

    public Iphone(String shop, String brand, String type, String processor, String ram, String storage, int year,
            int price, int battery, Double ratings, String iphoneSerialNumber) {
        super(shop, brand, type, processor, ram, storage, year, price, battery, ratings);
        IphoneSerialNumber = iphoneSerialNumber;
    }

    public String getIphoneSerialNumber() {
        return IphoneSerialNumber;
    }

    public void setIphoneSerialNumber(String iphoneSerialNumber) {
        IphoneSerialNumber = iphoneSerialNumber;
    }


    @Override
    void BrandId() {
        System.out.println("Iphone Serial Number  : " + this.getIphoneSerialNumber());

    }

}
```

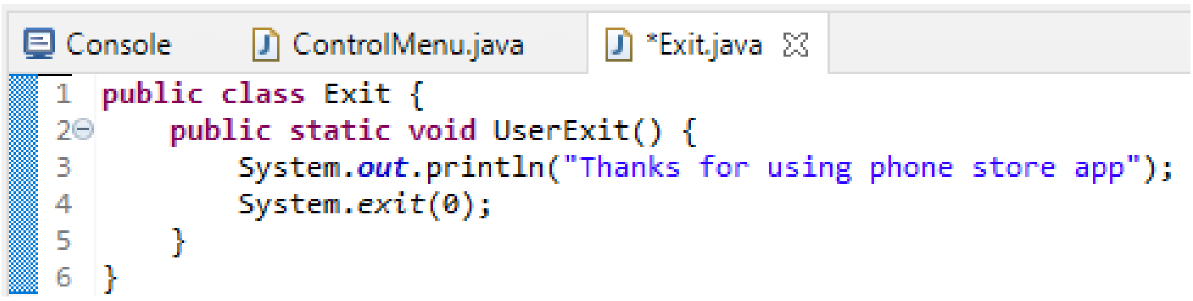
C. Object Smell

1. Dispensable smell : The comments


Console | *Phone.java

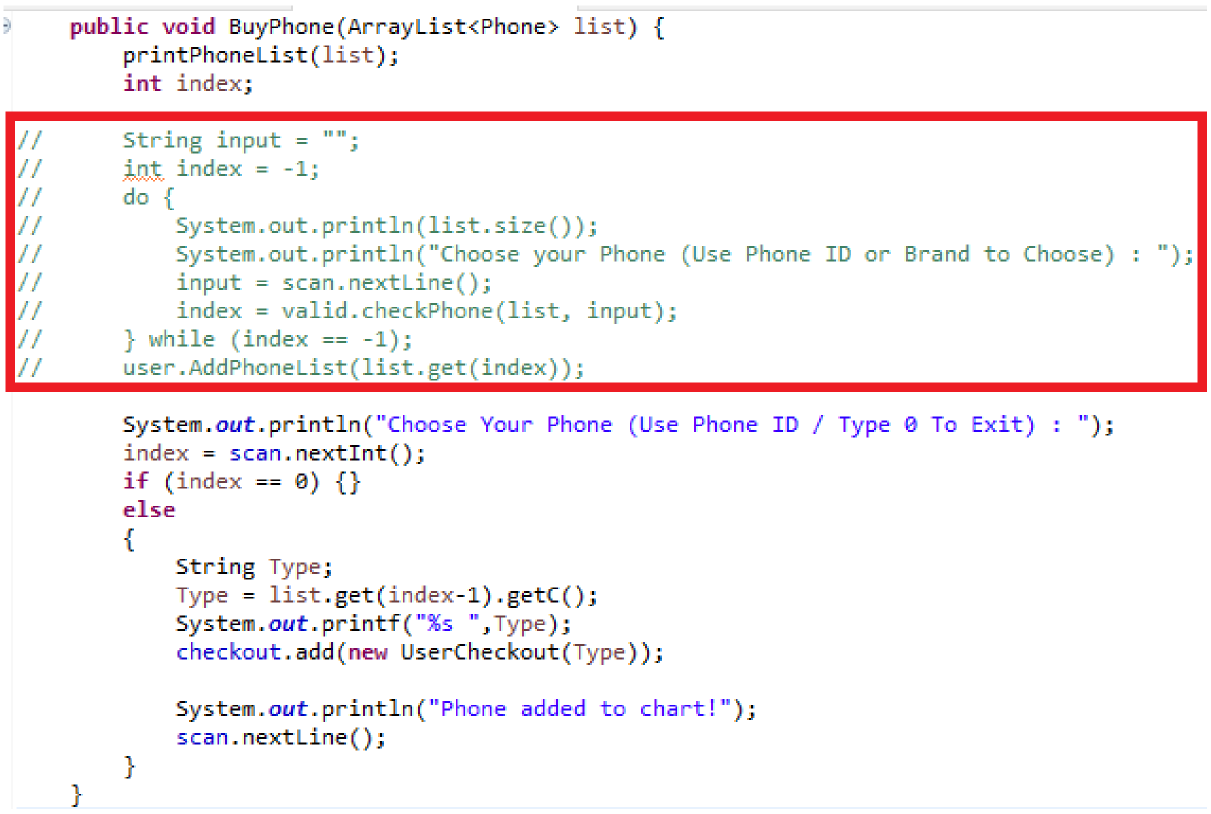
```java

public abstract class Phone {
    private String a; //  variabel untuk shop
    private String b; // variabel untuk brand
    private String c; // variabel untuk type
    private String d; // variabel untuk processor
    private String e; // variabel untuk ram
    private String f; // variabel untuk storage
    private int g; // variabel untuk year
    private int h; // variabel untuk price
    private int i; // variabel untuk battery
    private int k; // variabel untuk ratings


```

2. Dispensable smell : Lazy Class


Console | ControlMenu.java | *Exit.java

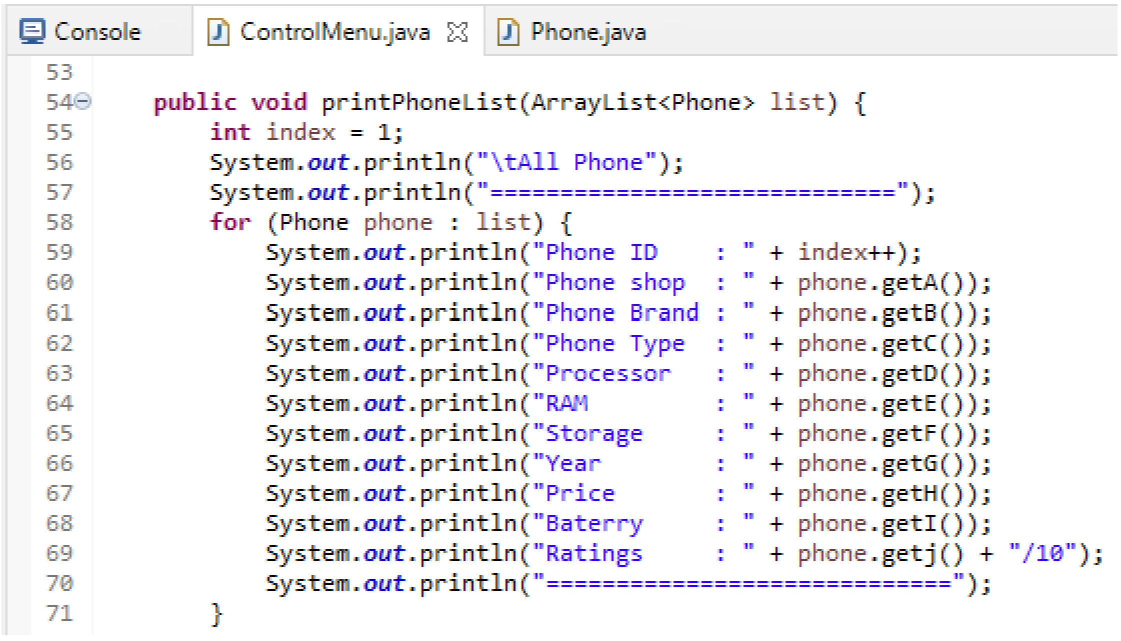
```java
public class Exit {
    public static void UserExit() {
        System.out.println("Thanks for using phone store app");
        System.exit(0);
    }
}
```

3. Dispensable smell : Dead Code



```
    public void BuyPhone(ArrayList<Phone> list) {
        printPhoneList(list);
        int index;
//      String input = "";
//      int index = -1;
//      do {
//          System.out.println(list.size());
//          System.out.println("Choose your Phone (Use Phone ID or Brand to Choose) : ");
//          input = scan.nextLine();
//          index = valid.checkPhone(list, input);
//      } while (index == -1);
//      user.AddPhoneList(list.get(index));

        System.out.println("Choose Your Phone (Use Phone ID / Type 0 To Exit) : ");
        index = scan.nextInt();
        if (index == 0) {}
        else
        {
            String Type;
            Type = list.get(index-1).getC();
            System.out.printf("%s ",Type);
            checkout.add(new UserCheckout(Type));

            System.out.println("Phone added to chart!");
            scan.nextLine();
        }
    }
```

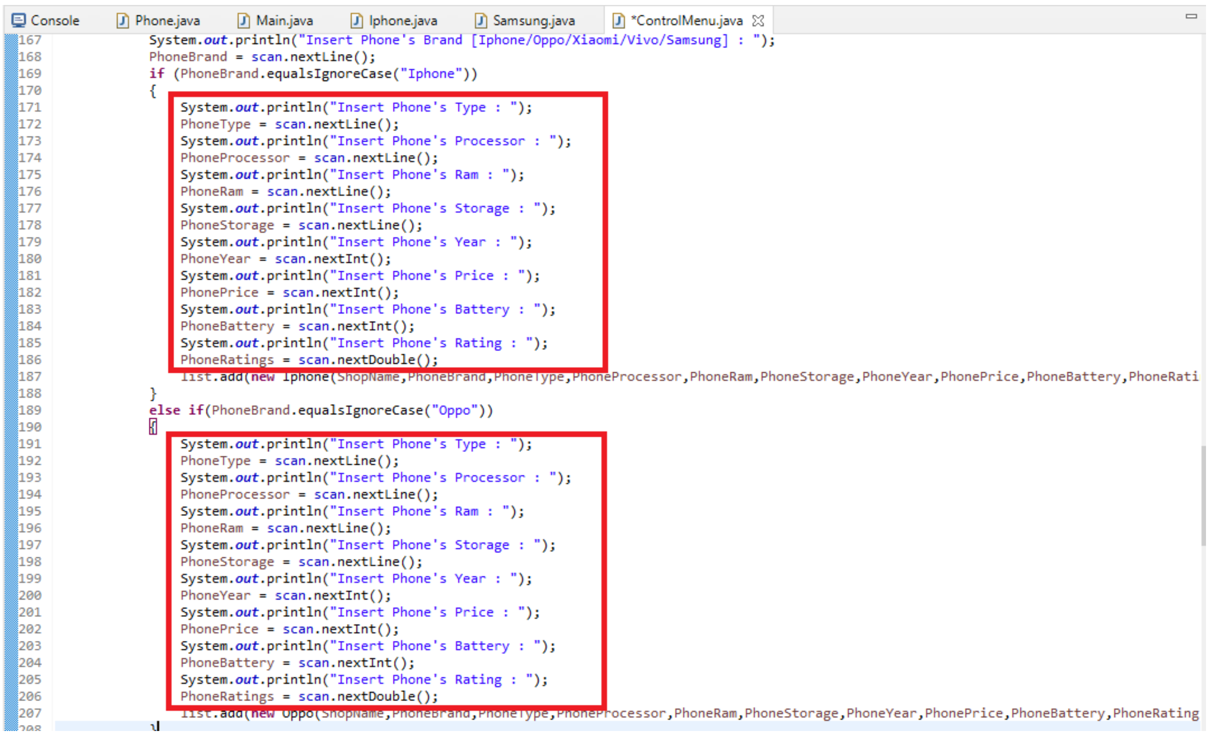
4. **Couplers smell : Inappropriate Intimacy**



Console | ControlMenu.java | Phone.java

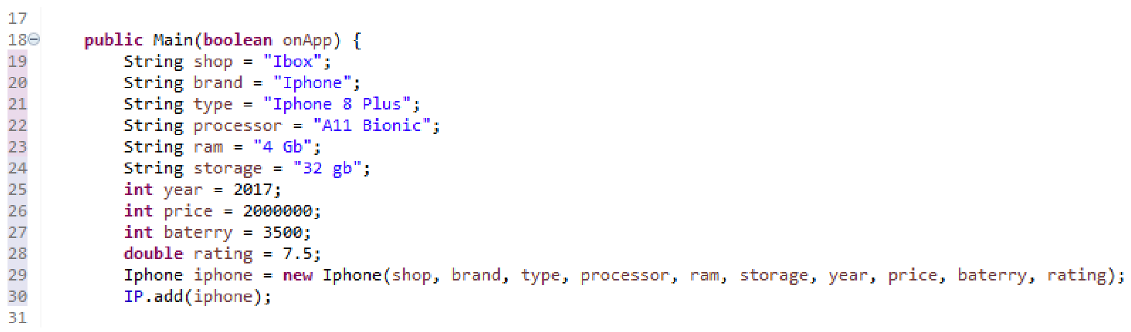
```

    public void printPhoneList(ArrayList<Phone> list) {
        int index = 1;
        System.out.println("\tAll Phone");
        System.out.println("=============================");
        for (Phone phone : list) {
            System.out.println("Phone ID    : " + index++);
            System.out.println("Phone shop  : " + phone.getA());
            System.out.println("Phone Brand : " + phone.getB());
            System.out.println("Phone Type  : " + phone.getC());
            System.out.println("Processor   : " + phone.getD());
            System.out.println("RAM         : " + phone.getE());
            System.out.println("Storage     : " + phone.getF());
            System.out.println("Year        : " + phone.getG());
            System.out.println("Price       : " + phone.getH());
            System.out.println("Baterry     : " + phone.getI());
            System.out.println("Ratings     : " + phone.getj() + "/10");
            System.out.println("=============================");
        }
```

5. **Dispensable smell : Duplicate Code**



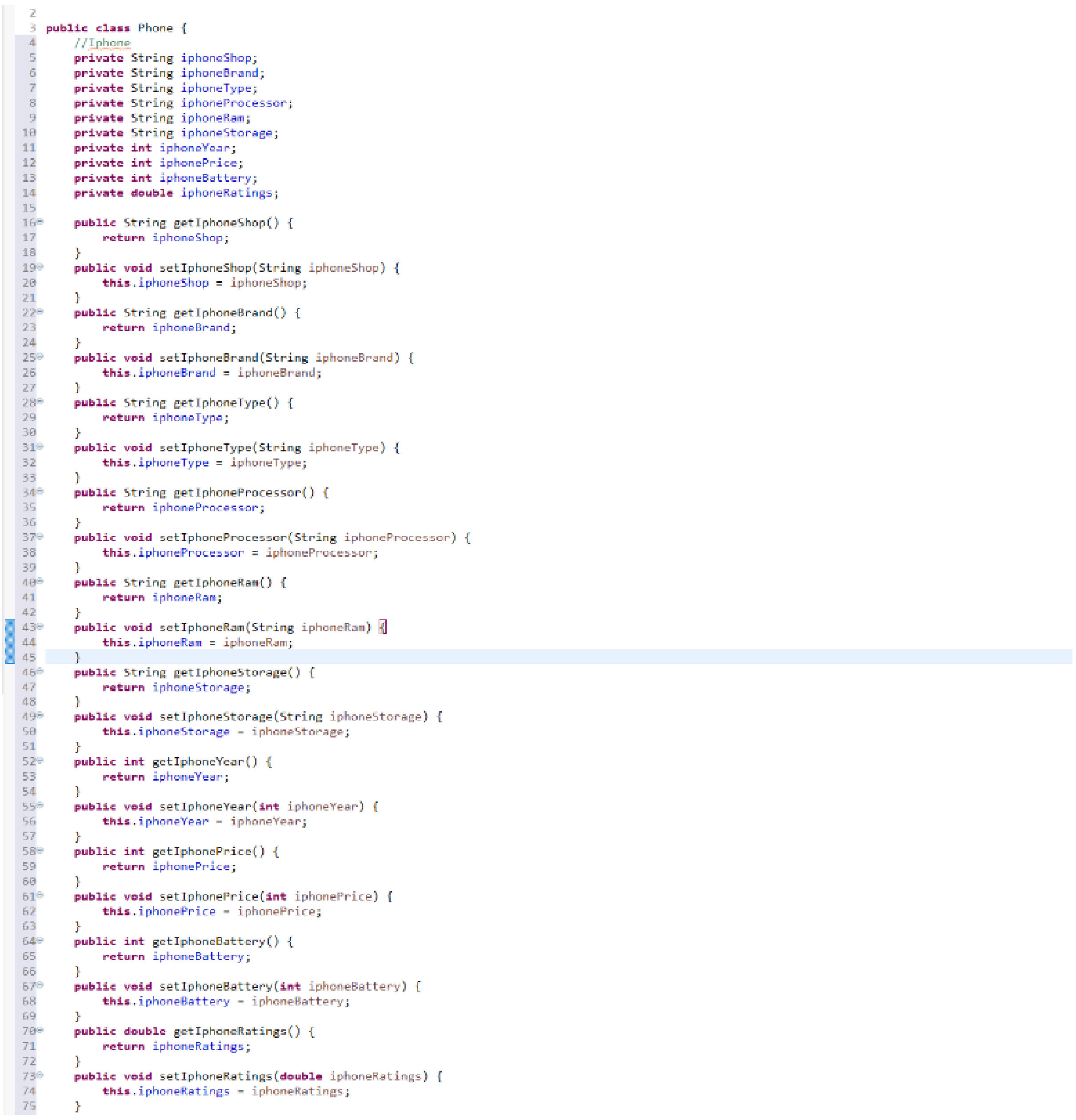
```
System.out.println("Insert Phone's Brand [Iphone/Oppo/Xiaomi/Vivo/Samsung] : ");
PhoneBrand = scan.nextLine();
if (PhoneBrand.equalsIgnoreCase("Iphone"))
{
    System.out.println("Insert Phone's Type : ");
    PhoneType = scan.nextLine();
    System.out.println("Insert Phone's Processor : ");
    PhoneProcessor = scan.nextLine();
    System.out.println("Insert Phone's Ram : ");
    PhoneRam = scan.nextLine();
    System.out.println("Insert Phone's Storage : ");
    PhoneStorage = scan.nextLine();
    System.out.println("Insert Phone's Year : ");
    PhoneYear = scan.nextInt();
    System.out.println("Insert Phone's Price : ");
    PhonePrice = scan.nextInt();
    System.out.println("Insert Phone's Battery : ");
    PhoneBattery = scan.nextInt();
    System.out.println("Insert Phone's Rating : ");
    PhoneRatings = scan.nextDouble();
    list.add(new Iphone(ShopName,PhoneBrand,PhoneType,PhoneProcessor,PhoneRam,PhoneStorage,PhoneYear,PhonePrice,PhoneBattery,PhoneRati
}
else if(PhoneBrand.equalsIgnoreCase("Oppo"))
{
    System.out.println("Insert Phone's Type : ");
    PhoneType = scan.nextLine();
    System.out.println("Insert Phone's Processor : ");
    PhoneProcessor = scan.nextLine();
    System.out.println("Insert Phone's Ram : ");
    PhoneRam = scan.nextLine();
    System.out.println("Insert Phone's Storage : ");
    PhoneStorage = scan.nextLine();
    System.out.println("Insert Phone's Year : ");
    PhoneYear = scan.nextInt();
    System.out.println("Insert Phone's Price : ");
    PhonePrice = scan.nextInt();
    System.out.println("Insert Phone's Battery : ");
    PhoneBattery = scan.nextInt();
    System.out.println("Insert Phone's Rating : ");
    PhoneRatings = scan.nextDouble();
    list.add(new Oppo(ShopName,PhoneBrand,PhoneType,PhoneProcessor,PhoneRam,PhoneStorage,PhoneYear,PhonePrice,PhoneBattery,PhoneRating
}
```

6. Bloaters Smell: Long Method

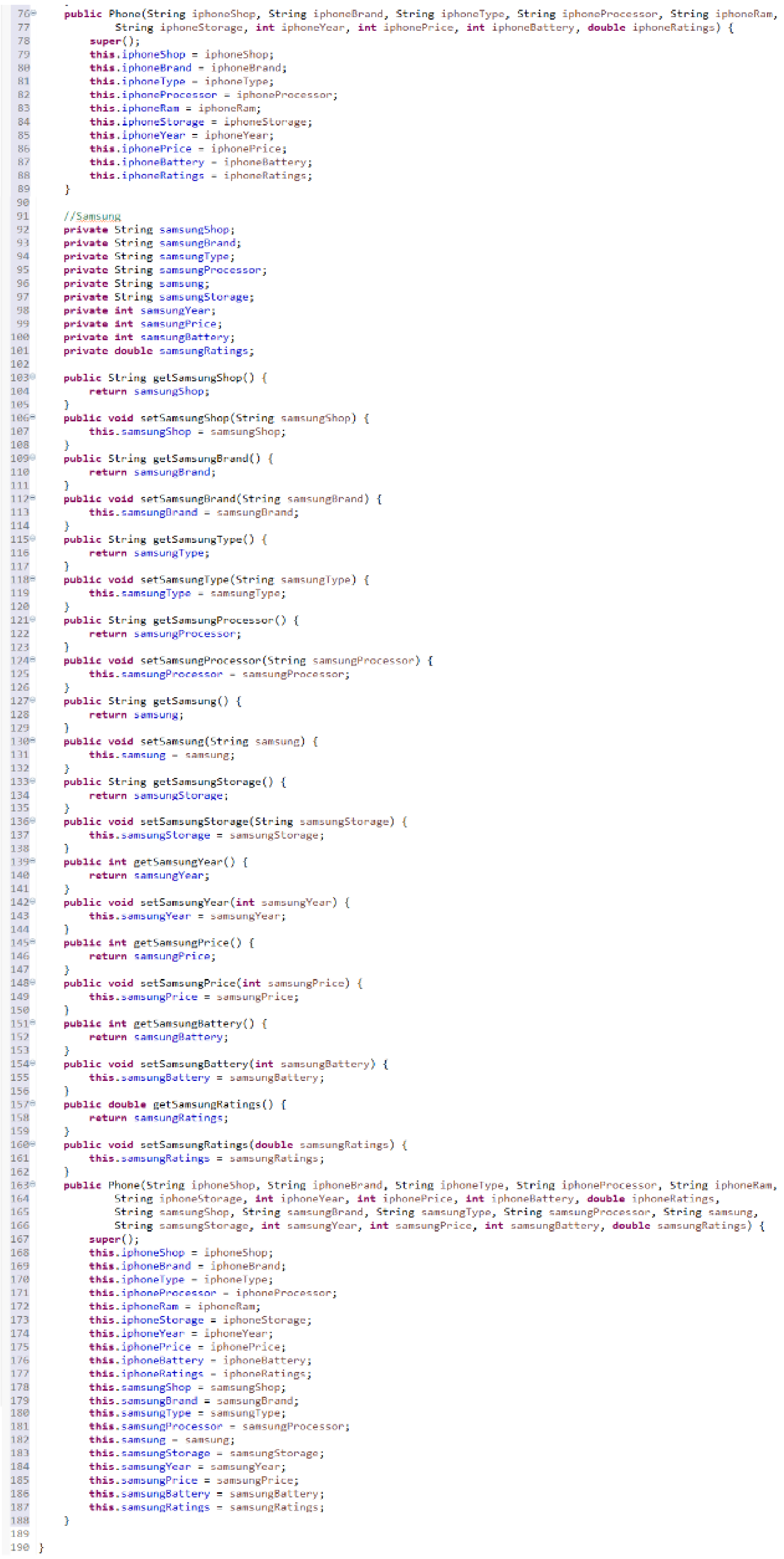


```

public Main(boolean onApp) {
    String shop = "Ibox";
    String brand = "Iphone";
    String type = "Iphone 8 Plus";
    String processor = "A11 Bionic";
    String ram = "4 Gb";
    String storage = "32 gb";
    int year = 2017;
    int price = 2000000;
    int baterry = 3500;
    double rating = 7.5;
    Iphone iphone = new Iphone(shop, brand, type, processor, ram, storage, year, price, baterry, rating);
    IP.add(iphone);

```

7. Bloaters Smell: Large Class

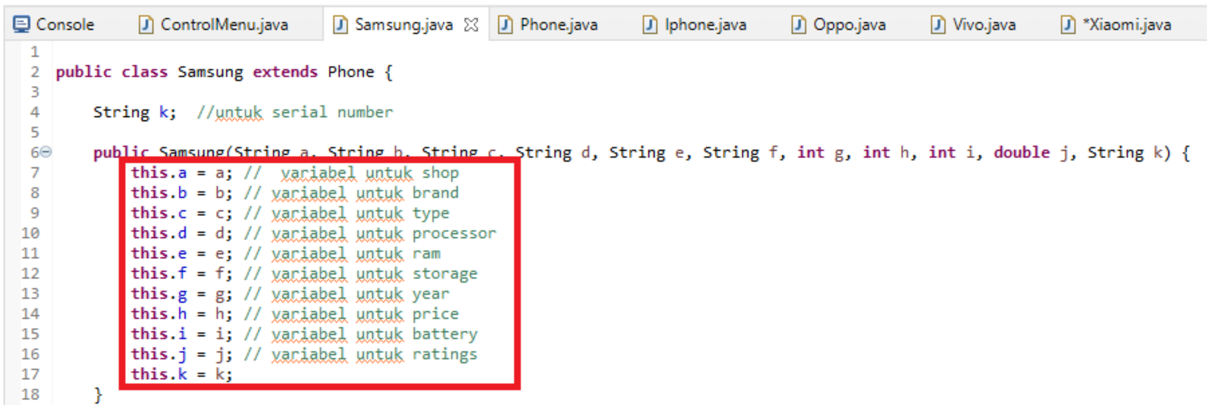


```java

public class Phone {
    //Iphone
    private String iphoneShop;
    private String iphoneBrand;
    private String iphoneType;
    private String iphoneProcessor;
    private String iphoneRam;
    private String iphoneStorage;
    private int iphoneYear;
    private int iphonePrice;
    private int iphoneBattery;
    private double iphoneRatings;

    public String getIphoneShop() {
        return iphoneShop;
    }
    public void setIphoneShop(String iphoneShop) {
        this.iphoneShop = iphoneShop;
    }
    public String getIphoneBrand() {
        return iphoneBrand;
    }
    public void setIphoneBrand(String iphoneBrand) {
        this.iphoneBrand = iphoneBrand;
    }
    public String getIphoneType() {
        return iphoneType;
    }
    public void setIphoneType(String iphoneType) {
        this.iphoneType = iphoneType;
    }
    public String getIphoneProcessor() {
        return iphoneProcessor;
    }
    public void setIphoneProcessor(String iphoneProcessor) {
        this.iphoneProcessor = iphoneProcessor;
    }
    public String getIphoneRam() {
        return iphoneRam;
    }
    public void setIphoneRam(String iphoneRam) {
        this.iphoneRam = iphoneRam;
    }
    public String getIphoneStorage() {
        return iphoneStorage;
    }
    public void setIphoneStorage(String iphoneStorage) {
        this.iphoneStorage = iphoneStorage;
    }
    public int getIphoneYear() {
        return iphoneYear;
    }
    public void setIphoneYear(int iphoneYear) {
        this.iphoneYear = iphoneYear;
    }
    public int getIphonePrice() {
        return iphonePrice;
    }
    public void setIphonePrice(int iphonePrice) {
        this.iphonePrice = iphonePrice;
    }
    public int getIphoneBattery() {
        return iphoneBattery;
    }
    public void setIphoneBattery(int iphoneBattery) {
        this.iphoneBattery = iphoneBattery;
    }
    public double getIphoneRatings() {
        return iphoneRatings;
    }
    public void setIphoneRatings(double iphoneRatings) {
        this.iphoneRatings = iphoneRatings;
    }
```



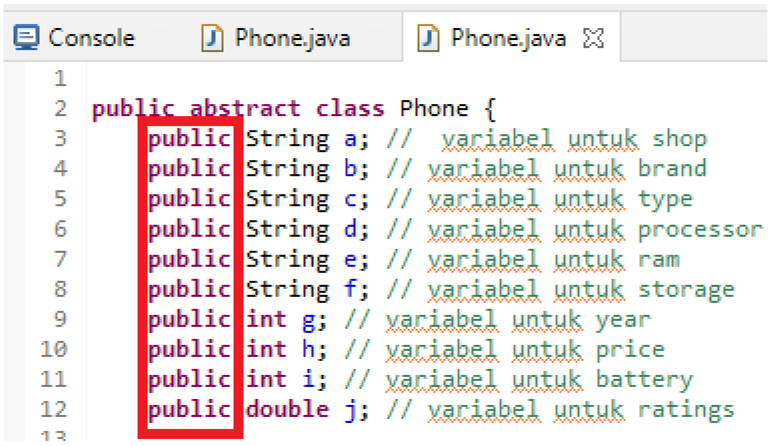
```java
    public Phone(String iphoneShop, String iphoneBrand, String iphoneType, String iphoneProcessor, String iphoneRam,
            String iphoneStorage, int iphoneYear, int iphonePrice, int iphoneBattery, double iphoneRatings) {
        super();
        this.iphoneShop = iphoneShop;
        this.iphoneBrand = iphoneBrand;
        this.iphoneType = iphoneType;
        this.iphoneProcessor = iphoneProcessor;
        this.iphoneRam = iphoneRam;
        this.iphoneStorage = iphoneStorage;
        this.iphoneYear = iphoneYear;
        this.iphonePrice = iphonePrice;
        this.iphoneBattery = iphoneBattery;
        this.iphoneRatings = iphoneRatings;
    }

    //Samsung
    private String samsungShop;
    private String samsungBrand;
    private String samsungType;
    private String samsungProcessor;
    private String samsung;
    private String samsungStorage;
    private int samsungYear;
    private int samsungPrice;
    private int samsungBattery;
    private double samsungRatings;

    public String getSamsungShop() {
        return samsungShop;
    }
    public void setSamsungShop(String samsungShop) {
        this.samsungShop = samsungShop;
    }
    public String getSamsungBrand() {
        return samsungBrand;
    }
    public void setSamsungBrand(String samsungBrand) {
        this.samsungBrand = samsungBrand;
    }
    public String getSamsungType() {
        return samsungType;
    }
    public void setSamsungType(String samsungType) {
        this.samsungType = samsungType;
    }
    public String getSamsungProcessor() {
        return samsungProcessor;
    }
    public void setSamsungProcessor(String samsungProcessor) {
        this.samsungProcessor = samsungProcessor;
    }
    public String getSamsung() {
        return samsung;
    }
    public void setSamsung(String samsung) {
        this.samsung = samsung;
    }
    public String getSamsungStorage() {
        return samsungStorage;
    }
    public void setSamsungStorage(String samsungStorage) {
        this.samsungStorage = samsungStorage;
    }
    public int getSamsungYear() {
        return samsungYear;
    }
    public void setSamsungYear(int samsungYear) {
        this.samsungYear = samsungYear;
    }
    public int getSamsungPrice() {
        return samsungPrice;
    }
    public void setSamsungPrice(int samsungPrice) {
        this.samsungPrice = samsungPrice;
    }
    public int getSamsungBattery() {
        return samsungBattery;
    }
    public void setSamsungBattery(int samsungBattery) {
        this.samsungBattery = samsungBattery;
    }
    public double getSamsungRatings() {
        return samsungRatings;
    }
    public void setSamsungRatings(double samsungRatings) {
        this.samsungRatings = samsungRatings;
    }
    public Phone(String iphoneShop, String iphoneBrand, String iphoneType, String iphoneProcessor, String iphoneRam,
            String iphoneStorage, int iphoneYear, int iphonePrice, int iphoneBattery, double iphoneRatings,
            String samsungShop, String samsungBrand, String samsungType, String samsungProcessor, String samsung,
            String samsungStorage, int samsungYear, int samsungPrice, int samsungBattery, double samsungRatings) {
        super();
        this.iphoneShop = iphoneShop;
        this.iphoneBrand = iphoneBrand;
        this.iphoneType = iphoneType;
        this.iphoneProcessor = iphoneProcessor;
        this.iphoneRam = iphoneRam;
        this.iphoneStorage = iphoneStorage;
        this.iphoneYear = iphoneYear;
        this.iphonePrice = iphonePrice;
        this.iphoneBattery = iphoneBattery;
        this.iphoneRatings = iphoneRatings;
        this.samsungShop = samsungShop;
        this.samsungBrand = samsungBrand;
        this.samsungType = samsungType;
        this.samsungProcessor = samsungProcessor;
        this.samsung = samsung;
        this.samsungStorage = samsungStorage;
        this.samsungYear = samsungYear;
        this.samsungPrice = samsungPrice;
        this.samsungBattery = samsungBattery;
        this.samsungRatings = samsungRatings;
    }

}
```

8. **Object oriented design smell : Pull up constructor body**



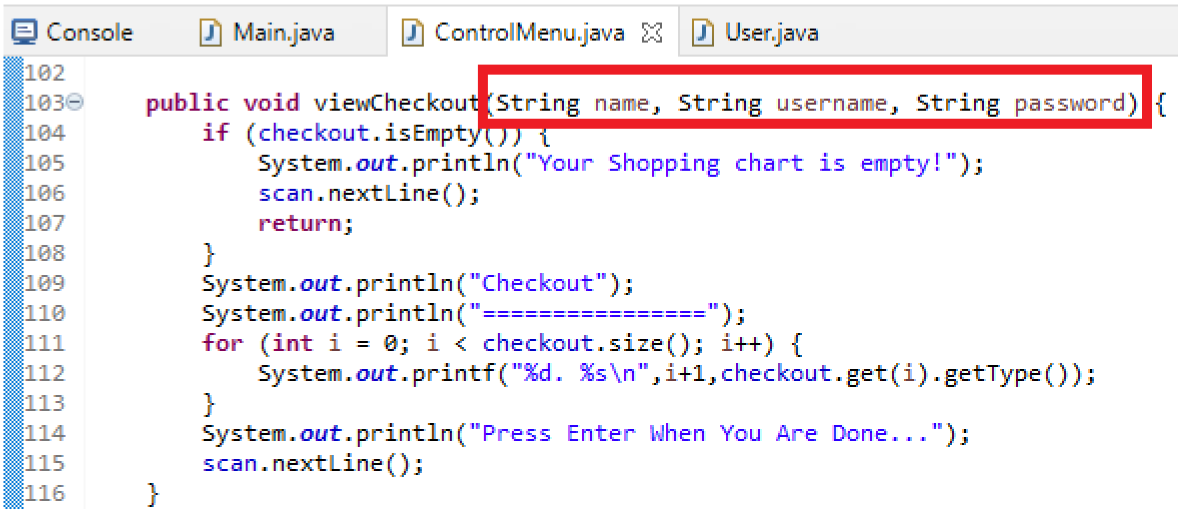
```java

public class Samsung extends Phone {

    String k;  //untuk serial number

    public Samsung(String a, String b, String c, String d, String e, String f, int g, int h, int i, double j, String k) {
        this.a = a; //  variabel untuk shop
        this.b = b; // variabel untuk brand
        this.c = c; // variabel untuk type
        this.d = d; // variabel untuk processor
        this.e = e; // variabel untuk ram
        this.f = f; // variabel untuk storage
        this.g = g; // variabel untuk year
        this.h = h; // variabel untuk price
        this.i = i; // variabel untuk battery
        this.j = j; // variabel untuk ratings
        this.k = k;
    }
```

9. **Encapsulation Smell**



```java

public abstract class Phone {
    public String a; //  variabel untuk shop
    public String b; // variabel untuk brand
    public String c; // variabel untuk type
    public String d; // variabel untuk processor
    public String e; // variabel untuk ram
    public String f; // variabel untuk storage
    public int g; // variabel untuk year
    public int h; // variabel untuk price
    public int i; // variabel untuk battery
    public double j; // variabel untuk ratings
```

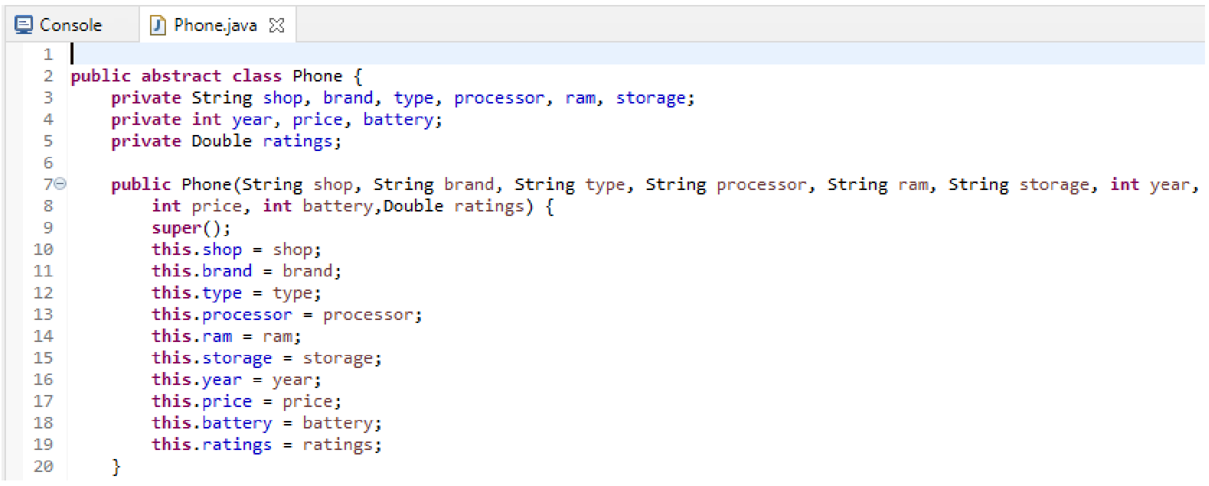
10. **Bloaters Smell : Long Parameter List**



```java

    public void viewCheckout(String name, String username, String password) {
        if (checkout.isEmpty()) {
            System.out.println("Your Shopping chart is empty!");
            scan.nextLine();
            return;
        }
        System.out.println("Checkout");
        System.out.println("================");
        for (int i = 0; i < checkout.size(); i++) {
            System.out.printf("%d. %s\n",i+1,checkout.get(i).getType());
        }
        System.out.println("Press Enter When You Are Done...");
        scan.nextLine();
    }
```

D. Refactoring Process

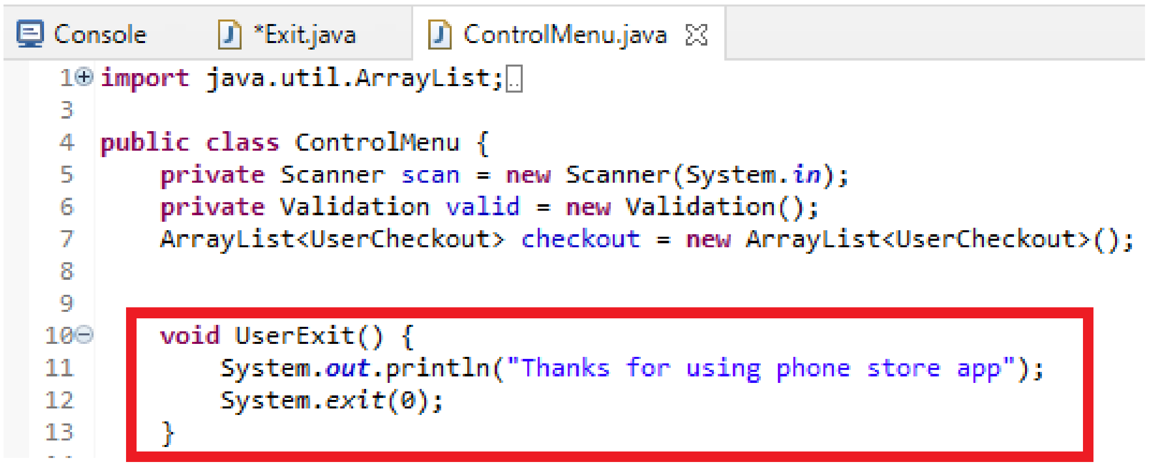
1. **Solusi refactoring Dispensable smell The comments :** dengan cara mengubah nama variabel yang mudah dipahami sehingga tidak membutuhkan comment yang berlebihan.



Console | Phone.java

```java

public abstract class Phone {
    private String shop, brand, type, processor, ram, storage;
    private int year, price, battery;
    private Double ratings;

    public Phone(String shop, String brand, String type, String processor, String ram, String storage, int year,
        int price, int battery,Double ratings) {
        super();
        this.shop = shop;
        this.brand = brand;
        this.type = type;
        this.processor = processor;
        this.ram = ram;
        this.storage = storage;
        this.year = year;
        this.price = price;
        this.battery = battery;
        this.ratings = ratings;
    }
```

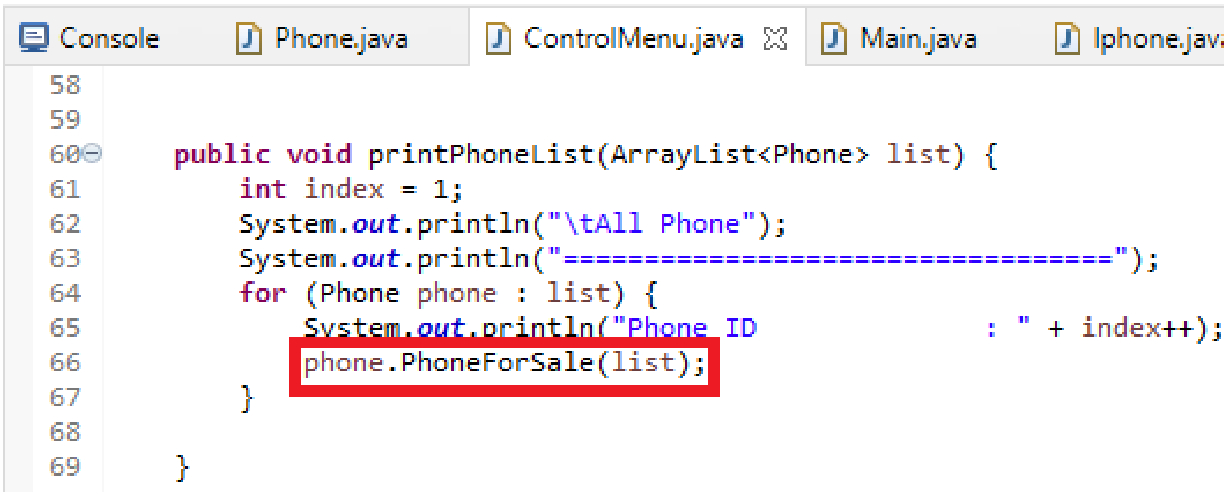
2. **Solusi refactoring Dispensable smell Lazy Class :** dengan cara menggabungkan code tersebut kedalam class yang digunakan menggunakan method baru.



Console | *Exit.java | ControlMenu.java

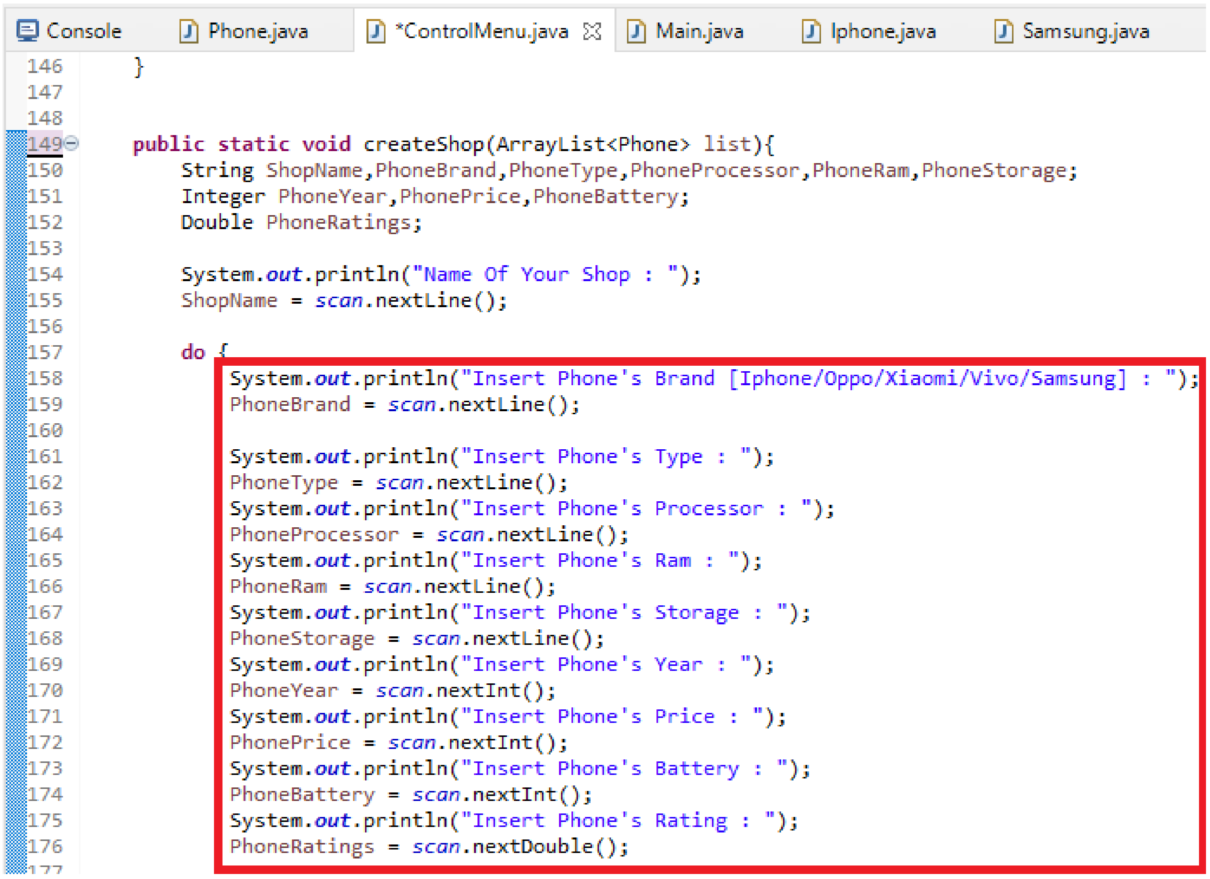
```java
import java.util.ArrayList;

public class ControlMenu {
    private Scanner scan = new Scanner(System.in);
    private Validation valid = new Validation();
    ArrayList<UserCheckout> checkout = new ArrayList<UserCheckout>();


    void UserExit() {
        System.out.println("Thanks for using phone store app");
        System.exit(0);
    }
```

3. **Solusi refactoring Dispensable smell Dead Code :** dengan cara menghapus code yang sudah tidak digunakan.
4. **Solusi refactoring Couplers smell :** Inappropriate Intimacy dengan cara hide data phone pada class “ControlMenu”. Karena class “ControlMenu” telah mengakses internal field dan method dari class “Phone”.



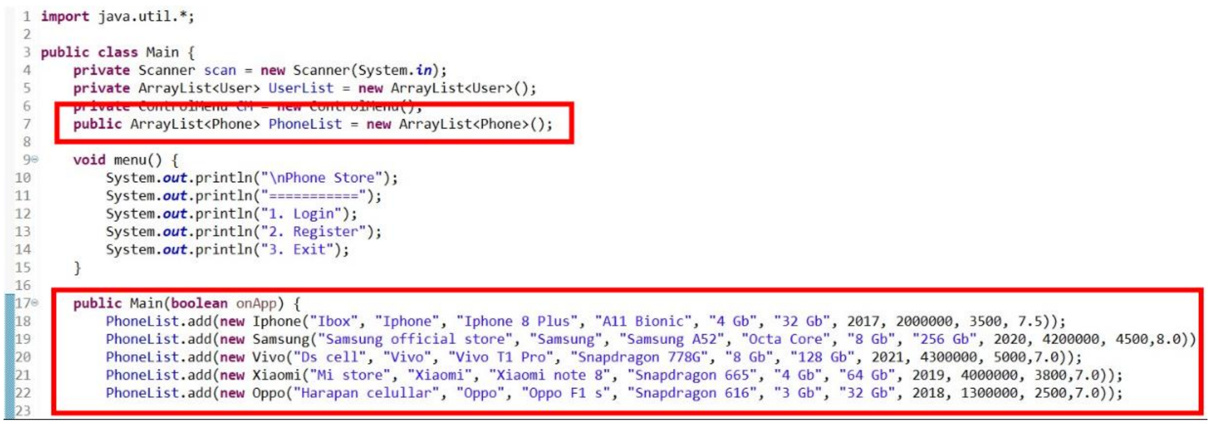
```java
	public void printPhoneList(ArrayList<Phone> list) {
		int index = 1;
		System.out.println("\tAll Phone");
		System.out.println("==================================");
		for (Phone phone : list) {
			System.out.println("Phone ID		: " + index++);
			phone.PhoneForSale(list);
		}

	}
```

5. **Solusi refactoring Dispensable smell Duplicate Code :** dengan cara menggabungkan code yang memiliki fungsional yang sama agar meminimalisir baris code.



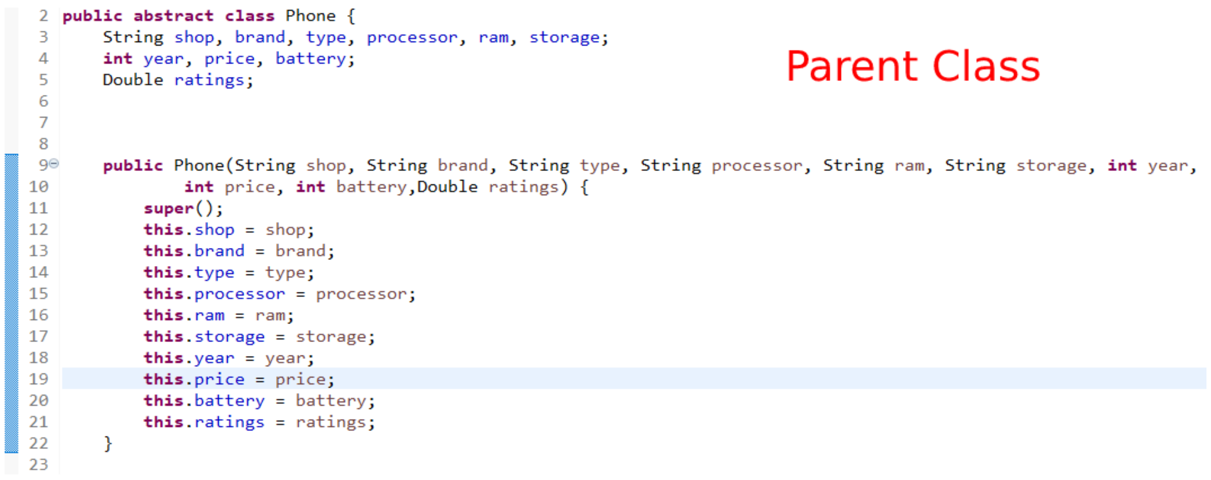
```java
	}


	public static void createShop(ArrayList<Phone> list){
		String ShopName,PhoneBrand,PhoneType,PhoneProcessor,PhoneRam,PhoneStorage;
		Integer PhoneYear,PhonePrice,PhoneBattery;
		Double PhoneRatings;

		System.out.println("Name Of Your Shop : ");
		ShopName = scan.nextLine();

		do {
			System.out.println("Insert Phone's Brand [Iphone/Oppo/Xiaomi/Vivo/Samsung] : ");
			PhoneBrand = scan.nextLine();

			System.out.println("Insert Phone's Type : ");
			PhoneType = scan.nextLine();
			System.out.println("Insert Phone's Processor : ");
			PhoneProcessor = scan.nextLine();
			System.out.println("Insert Phone's Ram : ");
			PhoneRam = scan.nextLine();
			System.out.println("Insert Phone's Storage : ");
			PhoneStorage = scan.nextLine();
			System.out.println("Insert Phone's Year : ");
			PhoneYear = scan.nextInt();
			System.out.println("Insert Phone's Price : ");
			PhonePrice = scan.nextInt();
			System.out.println("Insert Phone's Battery : ");
			PhoneBattery = scan.nextInt();
			System.out.println("Insert Phone's Rating : ");
			PhoneRatings = scan.nextDouble();
```

6. **Solusi refactoring Bloaters smell Long Method:** dengan cara menggunakan ".add new()" pada ArrayList untuk menambahkan data kedalam child class phone sehingga tidak perlu menulis ulang nama variabel

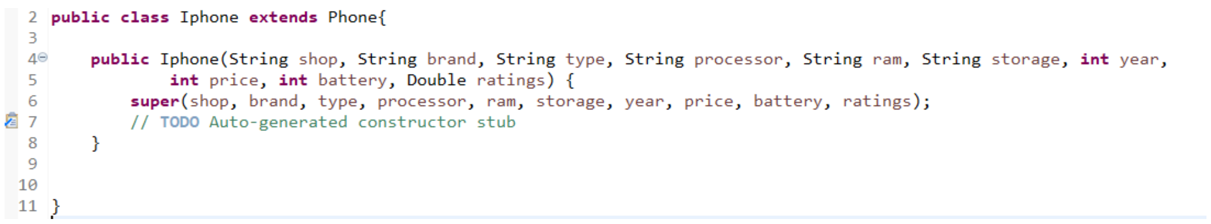


```java
import java.util.*;

public class Main {
    private Scanner scan = new Scanner(System.in);
    private ArrayList<User> UserList = new ArrayList<User>();
    private ControlMenu cm = new ControlMenu();
    public ArrayList<Phone> PhoneList = new ArrayList<Phone>();

    void menu() {
        System.out.println("\nPhone Store");
        System.out.println("===========");
        System.out.println("1. Login");
        System.out.println("2. Register");
        System.out.println("3. Exit");
    }

    public Main(boolean onApp) {
        PhoneList.add(new Iphone("Ibox", "Iphone", "Iphone 8 Plus", "A11 Bionic", "4 Gb", "32 Gb", 2017, 2000000, 3500, 7.5));
        PhoneList.add(new Samsung("Samsung official store", "Samsung", "Samsung A52", "Octa Core", "8 Gb", "256 Gb", 2020, 4200000, 4500,8.0))
        PhoneList.add(new Vivo("Ds cell", "Vivo", "Vivo T1 Pro", "Snapdragon 778G", "8 Gb", "128 Gb", 2021, 4300000, 5000,7.0));
        PhoneList.add(new Xiaomi("Mi store", "Xiaomi", "Xiaomi note 8", "Snapdragon 665", "4 Gb", "64 Gb", 2019, 4000000, 3800,7.0));
        PhoneList.add(new Oppo("Harapan celullar", "Oppo", "Oppo F1 s", "Snapdragon 616", "3 Gb", "32 Gb", 2018, 1300000, 2500,7.0));

```

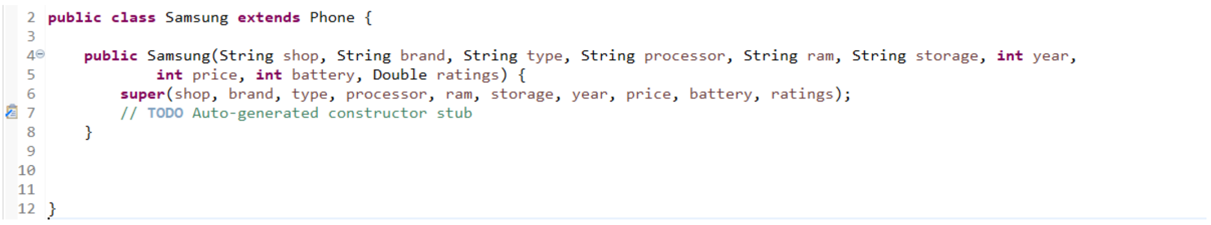
7. **Solusi refactoring Bloaters smell Large Class:** dengan menginmplementasikan Inheritance. Pisahkan brand handphone ke dalam class yang berbeda-beda dengan cara membuat class untuk setiap brand handphone sebagai child class dan buat class baru sebagai parent class.



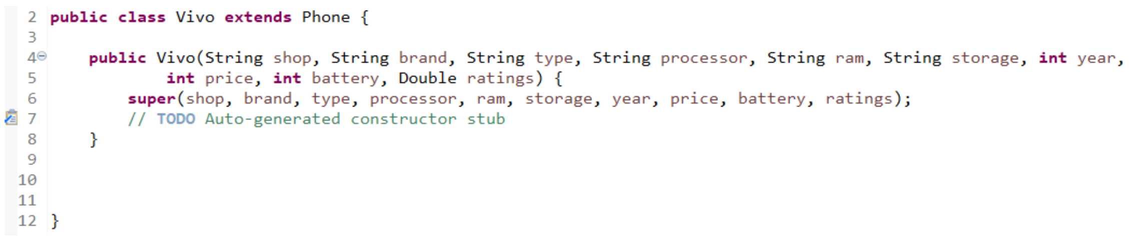
```java
public abstract class Phone {
    String shop, brand, type, processor, ram, storage;
    int year, price, battery;
    Double ratings;



    public Phone(String shop, String brand, String type, String processor, String ram, String storage, int year,
            int price, int battery,Double ratings) {
        super();
        this.shop = shop;
        this.brand = brand;
        this.type = type;
        this.processor = processor;
        this.ram = ram;
        this.storage = storage;
        this.year = year;
        this.price = price;
        this.battery = battery;
        this.ratings = ratings;
    }

```



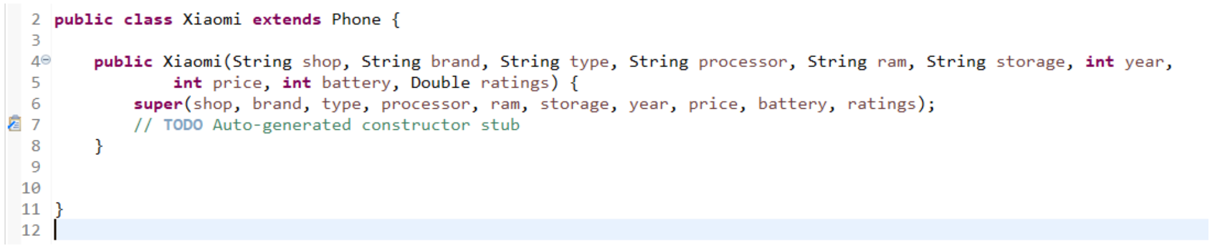
```java
public class Iphone extends Phone{

    public Iphone(String shop, String brand, String type, String processor, String ram, String storage, int year,
            int price, int battery, Double ratings) {
        super(shop, brand, type, processor, ram, storage, year, price, battery, ratings);
        // TODO Auto-generated constructor stub
    }



}
```



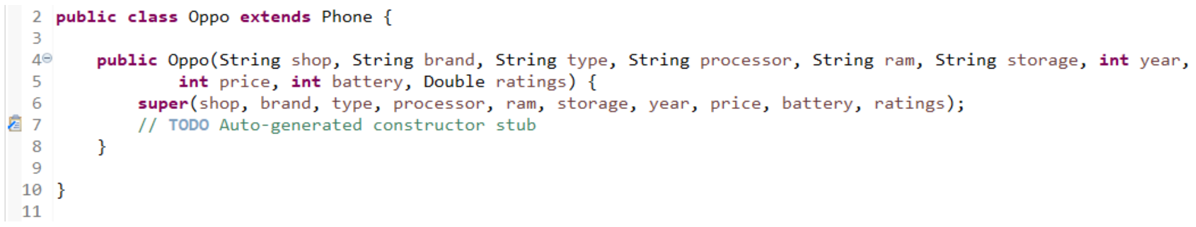
```java
public class Samsung extends Phone {

    public Samsung(String shop, String brand, String type, String processor, String ram, String storage, int year,
            int price, int battery, Double ratings) {
        super(shop, brand, type, processor, ram, storage, year, price, battery, ratings);
        // TODO Auto-generated constructor stub
    }




}
```



```java
public class Vivo extends Phone {

    public Vivo(String shop, String brand, String type, String processor, String ram, String storage, int year,
            int price, int battery, Double ratings) {
        super(shop, brand, type, processor, ram, storage, year, price, battery, ratings);
        // TODO Auto-generated constructor stub
    }



}
```

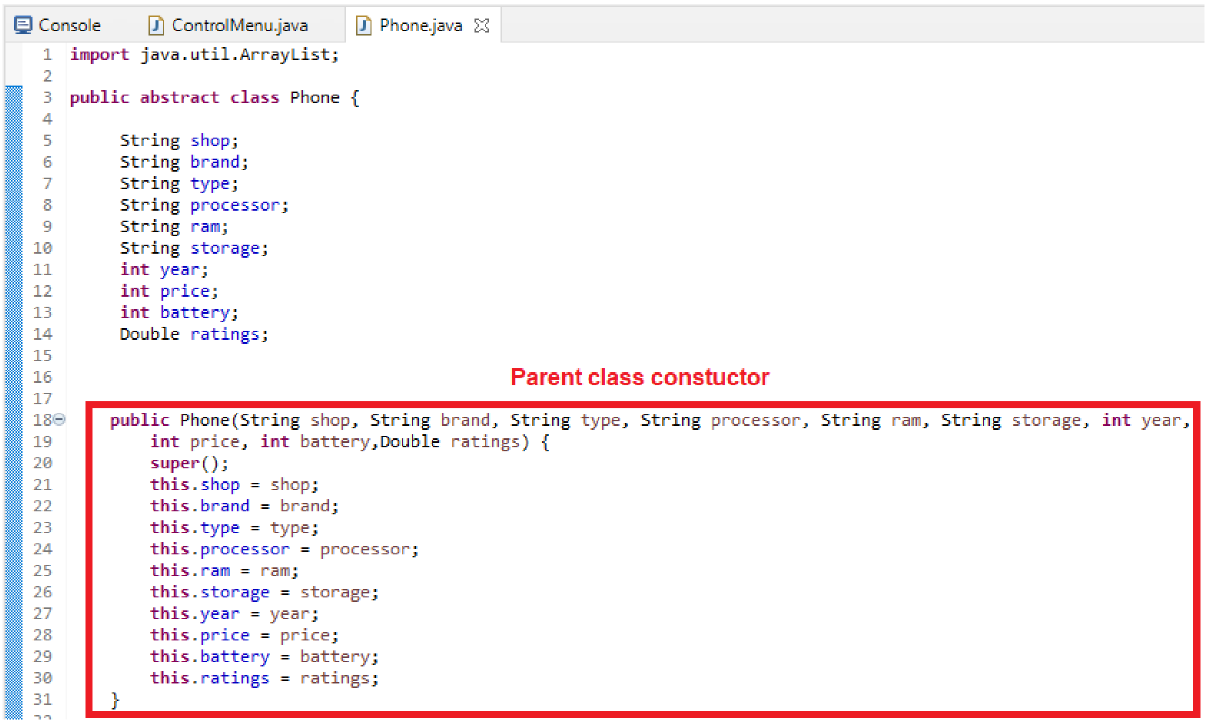


```java
public class Xiaomi extends Phone {

    public Xiaomi(String shop, String brand, String type, String processor, String ram, String storage, int year,
            int price, int battery, Double ratings) {
        super(shop, brand, type, processor, ram, storage, year, price, battery, ratings);
        // TODO Auto-generated constructor stub
    }


}

```

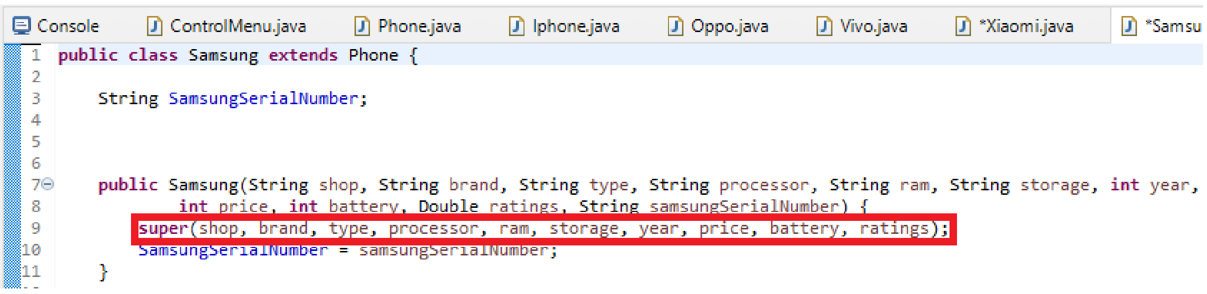


```java
public class Oppo extends Phone {

    public Oppo(String shop, String brand, String type, String processor, String ram, String storage, int year,
            int price, int battery, Double ratings) {
        super(shop, brand, type, processor, ram, storage, year, price, battery, ratings);
        // TODO Auto-generated constructor stub
    }

}

```

8. **Solusi Object oriented design smell : Pull up constructor body** dengan membuat constructor pada parent class lalu menggunakan kata kunci “super” pada child class untuk mengambil constructor dari parent class.

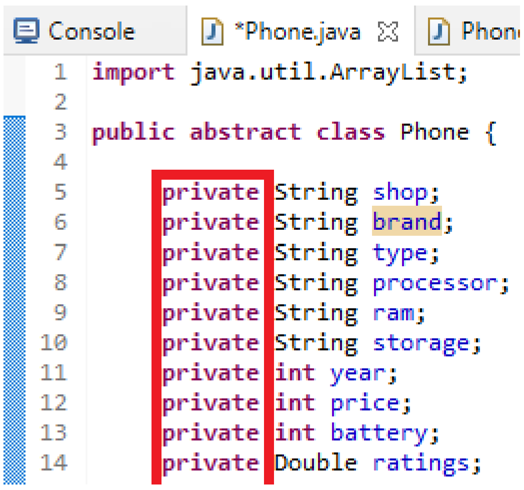


```java
import java.util.ArrayList;

public abstract class Phone {

    String shop;
    String brand;
    String type;
    String processor;
    String ram;
    String storage;
    int year;
    int price;
    int battery;
    Double ratings;



    public Phone(String shop, String brand, String type, String processor, String ram, String storage, int year,
        int price, int battery,Double ratings) {
        super();
        this.shop = shop;
        this.brand = brand;
        this.type = type;
        this.processor = processor;
        this.ram = ram;
        this.storage = storage;
        this.year = year;
        this.price = price;
        this.battery = battery;
        this.ratings = ratings;
    }
```



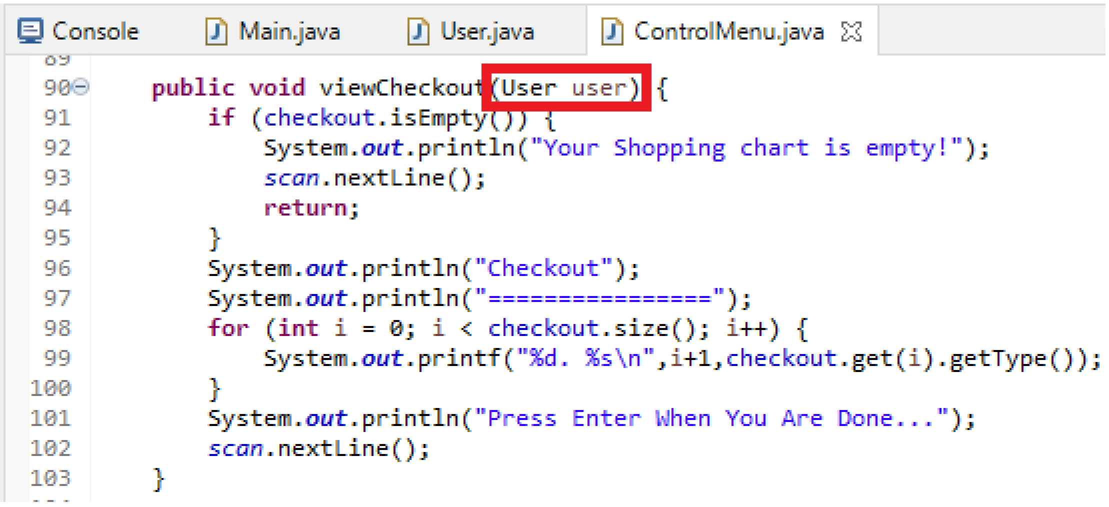
```java
public class Samsung extends Phone {

    String SamsungSerialNumber;



    public Samsung(String shop, String brand, String type, String processor, String ram, String storage, int year,
            int price, int battery, Double ratings, String samsungSerialNumber) {
        super(shop, brand, type, processor, ram, storage, year, price, battery, ratings);
        SamsungSerialNumber = samsungSerialNumber;
    }
```

9. **Solusi Encapsulation smell** dengan cara membuat private pada setiap atribut yang terdapat pada class tersebut



```java
import java.util.ArrayList;

public abstract class Phone {

    private String shop;
    private String brand;
    private String type;
    private String processor;
    private String ram;
    private String storage;
    private int year;
    private int price;
    private int battery;
    private Double ratings;
```

10. **Solusi Bloaters Smell : Long Parameter List** dengan cara melakukan Preserve Whole Object pada method viewCheckout.



```java
    public void viewCheckout(User user) {
        if (checkout.isEmpty()) {
            System.out.println("Your Shopping chart is empty!");
            scan.nextLine();
            return;
        }
        System.out.println("Checkout");
        System.out.println("================");
        for (int i = 0; i < checkout.size(); i++) {
            System.out.printf("%d. %s\n",i+1,checkout.get(i).getType());
        }
        System.out.println("Press Enter When You Are Done...");
        scan.nextLine();
    }
```

E. Implementation

Hasil implementasi kami buat dalam bentuk 2 file kode yang berbeda yaitu file : code smell dan hasil refactoring. Untuk link backup code dapat diakses melalui link berikut int :

https://drive.google.com/drive/folders/1eFhohCF_Pfgk0p4W408maBxa5LfZXfbn?usp=sharing