Буниф Маждф

18-КБ-ПР2

**Отчет ЭПО Лаб 1**

**4 Вариант Задание**

1) Перед началом модификации кода написать тестовые наборы для проведения регрессионного тестирования метода statement класса Bill, представленного в примере 1.10.

2)Тестовые наборы необходимо составлять, опираясь на структуру метода statement методом белого ящика и учитывая, как количество товара и его категория влияют на сумму бонусов и скидку. (Количество тестов должно быть не менее 7-8 штук).

3)Преобразовать метод statement класса Bill в соответствии с предложенными в пункте 2.10 изменениями, применить описанные в теории методы рефакторинга.

4)Проверить корректность работы нового метода и работоспособность старого метода формирования счета.

**RefBill.cs // Class Bill after Refactoring**

public class RefBill

{

private List<Item> \_items;

private Customer \_customer;

public RefBill(Customer customer)

{

this.\_customer = customer;

this.\_items = new List<Item>();

}

public void addGoods(Item arg)

{

\_items.Add(arg);

}

public string Statement()

{

double totalAmount = 0;

int totalBonus = 0;

String Print = "Счет для " + \_customer.getName() + "\n";

Print += GetHeader();

foreach (Item item in \_items) // Formula in paragraph 2.10-9

{

double itemPrice = Total\_Price(item);

int bonus = GetBonus(item);

double discount = GetDiscount(item);

double PriceWithDiscount = itemPrice - discount;

double usedBonus = UseBonus(item, PriceWithDiscount);

double thisAmount = PriceWithDiscount - usedBonus;

totalAmount = totalAmount + thisAmount;

totalBonus = totalBonus + bonus;

Print += GetItemString(discount, bonus, thisAmount, item); // show the item string

}

Print += GetFooter(totalAmount, totalBonus); //display the footer

\_customer.receiveBonus(totalBonus);

return Print;

}

//The Header

public string GetHeader()

{

return "\t" + "Название" + "\t" + "Цена" +

"\t" + "Кол-во" + "Стоимость" + "\t" + "Скидка" +

"\t" + "Сумма" + "\t" + "Бонус" + "\n";

//string result = "\t" + "Название" + "\t" + "Цена" +

//"\t" + "Кол-во" + "Стоимость" + "\t" + "Скидка" +

//"\t" + "Сумма" + "\t" + "Бонус" + "\n";

//return result;

}

//The Footer

private string GetFooter( double totalAmount, int totalBonus)

{

return "Сумма счета составляет " + totalAmount.ToString() + "\nВы заработали " + totalBonus.ToString() + " бонусных баллов";

}

//String of Items

private string GetItemString(double discount, int bonus, double thisAmount,Item item)

{

return "\t" + item.getGoods().getTitle() + "\t" + // or item

"\t" + item.getPrice() + "\t" + item.getQuantity() +

"\t" + (item.getQuantity() \* item.getPrice()).ToString() +

"\t" + discount.ToString() + "\t" + thisAmount.ToString() +

"\t" + bonus.ToString() + "\n";

/////////// or ////////////

//string result = "\t" + item.getGoods().getTitle() + "\t" +

// "\t" + item.getPrice() + "\t" + item.getQuantity() +

// "\t" + (item.getQuantity() \* item.getPrice()).ToString() +

// "\t" + discount.ToString() + "\t" + thisAmount.ToString() +

// "\t" + bonus.ToString() + "\n";

//return result;

}

//Formula Of Total Cost

private double Total\_Price(Item item)

{

return item.getQuantity() \* item.getPrice();

}

//formula Discount

private double GetDiscount(Item item)

{

double discount = 0.0;

switch (item.getGoods().getPriceCode())

{

case Goods.REGULAR :

if (item.getQuantity() > 2)

return discount=(Total\_Price(item) \* 0.03);

break;

case Goods.SPECIAL\_OFFER:

if (item.getQuantity() > 10)

return discount=(Total\_Price(item) \* 0.005);

break;

case Goods.SALE:

if (item.getQuantity() > 3)

return discount=(Total\_Price(item) \* 0.01);

break;

}

return discount;

}

// For Getting Bonuses

private int GetBonus(Item item)

{

int bonus = 0;

switch (item.getGoods().getPriceCode())

{

case Goods.REGULAR:

bonus =(int)(item.getQuantity() \* item.getPrice() \* 0.05);

break;

case Goods.SALE:

bonus =(int)(item.getQuantity() \* item.getPrice() \* 0.01);

break;

}

return bonus;

}

//using Bonuses

private double UseBonus(Item item, double PriceWithDiscount)

{

double bonus = 0.0;

if ((item.getGoods().getPriceCode() == Goods.REGULAR) && item.getQuantity() > 5)

bonus = \_customer.useBonus((int)PriceWithDiscount);

else if ((item.getGoods().getPriceCode() == Goods.SPECIAL\_OFFER) && item.getQuantity() > 1)

bonus = \_customer.useBonus((int)PriceWithDiscount);

return bonus;

}}}

**TestsClassRefBill.cs**

public class Class1

{

[Test]

public void TestRegularItemWithDiscount()

{

Goods fanta = new Goods("fanta", Goods.REGULAR);

Item i1 = new Item(fanta, 3, 65);

Customer x = new Customer("Sara", 10);

RefBill b1 = new RefBill(x);

b1.addGoods(i1);

string actual = b1.Statement();

string expected = "Счет для Sara\n\tНазвание\tЦена\tКол-воСтоимость\tСкидка\tСумма\tБонус\n\tfanta\t\t65\t3\t195\t5,85\t189,15\t9\nСумма счета составляет 189,15\nВы заработали 9 бонусных баллов";

Assert.AreEqual(expected, actual);

}

[Test]

public void TestRegularItemWithoutDiscount()

{

Goods pepsi = new Goods("pepsi", Goods.REGULAR);

Item i1 = new Item(pepsi, 1, 65);

Customer x = new Customer("Ali", 10);

RefBill b1 = new RefBill(x);

b1.addGoods(i1);

string actual = b1.Statement();

string expected = "Счет для Ali\n\tНазвание\tЦена\tКол-воСтоимость\tСкидка\tСумма\tБонус\n\tpepsi\t\t65\t1\t65\t0\t65\t3\nСумма счета составляет 65\nВы заработали 3 бонусных баллов";

Assert.AreEqual(expected, actual);

}

[Test]

public void TestRegularItemWithoutBonuses()

{

Goods cola = new Goods("Cola", Goods.REGULAR);

Item i1 = new Item(cola, 5, 65);

Customer x = new Customer("Tao", 10);

RefBill b1 = new RefBill(x);

b1.addGoods(i1);

string actual = b1.Statement();

string expected = "Счет для Tao\n\tНазвание\tЦена\tКол-воСтоимость\tСкидка\tСумма\tБонус\n\tCola\t\t65\t5\t325\t9,75\t315,25\t16\nСумма счета составляет 315,25\nВы заработали 16 бонусных баллов";

Assert.AreEqual(expected, actual);

}

[Test]

public void TestRegularItemWithBonuses()

{

Goods cola = new Goods("Cola", Goods.REGULAR);

Item i1 = new Item(cola, 6, 65);

Customer x = new Customer("Aya", 10);

RefBill b1 = new RefBill(x);

b1.addGoods(i1);

string actual = b1.Statement();

string expected = "Счет для Aya\n\tНазвание\tЦена\tКол-воСтоимость\tСкидка\tСумма\tБонус\n\tCola\t\t65\t6\t390\t11,7\t368,3\t19\nСумма счета составляет 368,3\nВы заработали 19 бонусных баллов";

Assert.AreEqual(expected, actual);

}

[Test]

public void TestSpecialItemWithDiscount()

{

Goods Cola = new Goods("Cola", Goods.SPECIAL\_OFFER);

Item i1 = new Item(Cola, 12, 65);

Customer x = new Customer("Ana", 10);

RefBill b1 = new RefBill(x);

b1.addGoods(i1);

string actual = b1.Statement();

string expected = "Счет для Ana\n\tНазвание\tЦена\tКол-воСтоимость\tСкидка\tСумма\tБонус\n\tCola\t\t65\t12\t780\t3,9\t766,1\t0\nСумма счета составляет 766,1\nВы заработали 0 бонусных баллов";

Assert.AreEqual(expected, actual);

}

[Test]

public void TestSpecialItemWithoutDiscount()

{

Goods Mirenda = new Goods("Mirenda", Goods.SPECIAL\_OFFER);

Item i1 = new Item(Mirenda, 10, 65);

Customer x = new Customer("Marwa", 10);

RefBill b1 = new RefBill(x);

b1.addGoods(i1);

string actual = b1.Statement();

string expected = "Счет для Marwa\n\tНазвание\tЦена\tКол-воСтоимость\tСкидка\tСумма\tБонус\n\tMirenda\t\t65\t10\t650\t0\t640\t0\nСумма счета составляет 640\nВы заработали 0 бонусных баллов";

Assert.AreEqual(expected, actual);

}

[Test]

public void TestSaleItemWithoutDiscount()

{

Goods Coffee = new Goods("Coffee", Goods.SALE);

Item i1 = new Item(Coffee, 3, 65);

Customer x = new Customer("Nouha", 10);

RefBill b1 = new RefBill(x);

b1.addGoods(i1);

string actual = b1.Statement();

string expected = "Счет для Nouha\n\tНазвание\tЦена\tКол-воСтоимость\tСкидка\tСумма\tБонус\n\tCoffee\t\t65\t3\t195\t0\t195\t1\nСумма счета составляет 195\nВы заработали 1 бонусных баллов";

Assert.AreEqual(expected, actual);

}

[Test]

public void TestSaleItemWithDiscount()

{

Goods OrangeJus = new Goods("OrangeJus", Goods.SALE);

Item i1 = new Item(OrangeJus, 4, 65);

Customer x = new Customer("David", 10);

RefBill b1 = new RefBill(x);

b1.addGoods(i1);

string actual = b1.Statement();

string expected = "Счет для David\n\tНазвание\tЦена\tКол-воСтоимость\tСкидка\tСумма\tБонус\n\tOrangeJus\t\t65\t4\t260\t2,6\t257,4\t2\nСумма счета составляет 257,4\nВы заработали 2 бонусных баллов";

Assert.AreEqual(expected, actual);

}

}

}

**Результат Работы**

