Министерство науки и высшего образования Российской Федерации

ФГБОУ ВО «Кубанский Государственный Технологический Университет»

Кафедра информационных систем и программирования

Отчет по лабораторной работе №2

По дисциплине «Рефакторинг и работа с унаследованным кодом»

«Базовые техники рефакторинга. Обмен частей между классами»

Выполнил студент

Группы 21-КБ-ПР2

Жулькин Д.П.

Преподаватель:

Степанова Елизавета Владимировна

Краснодар

2024

Цель работы: Изучить базовые техники рефакторинга позволяющие различным образом манипулировать методами между классами программы.

Задание:

1) Преобразовать метод statement класса Bill, и перенести методы получения скидки и начисления бонусов в класс Goods в соответствии с предложенными в пункте 2.2 изменениями, применив описанные в теории методы рефакторинга.

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

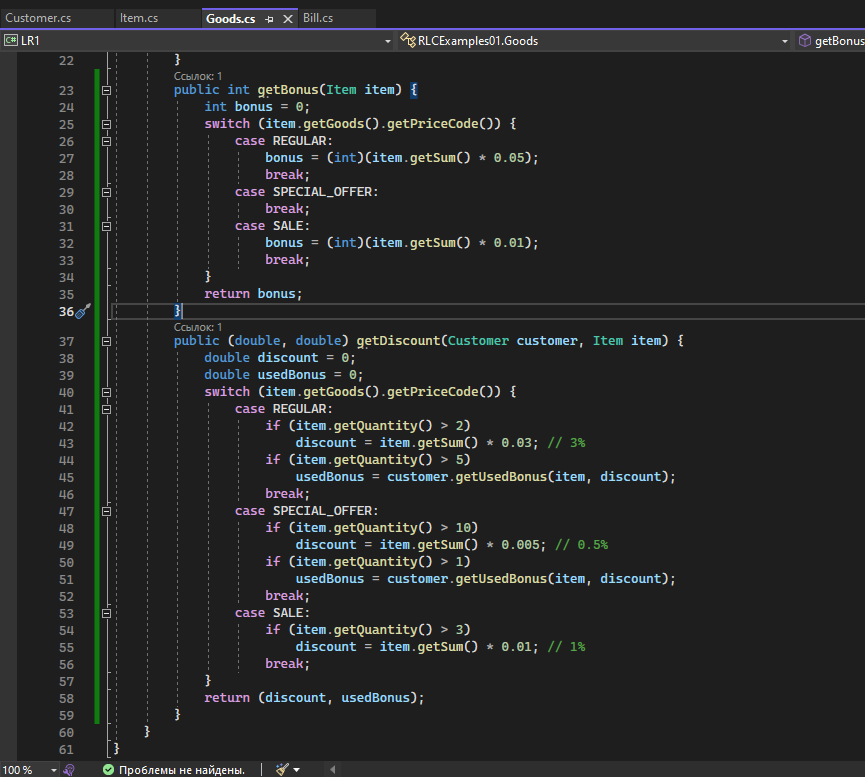


Рисунок 1 – Перенос методов GetBonus и GetDiscount в класс Goods

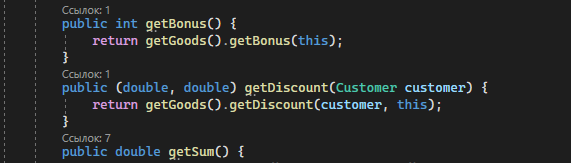


Рисунок 2 – Прокси-методы в классе Item

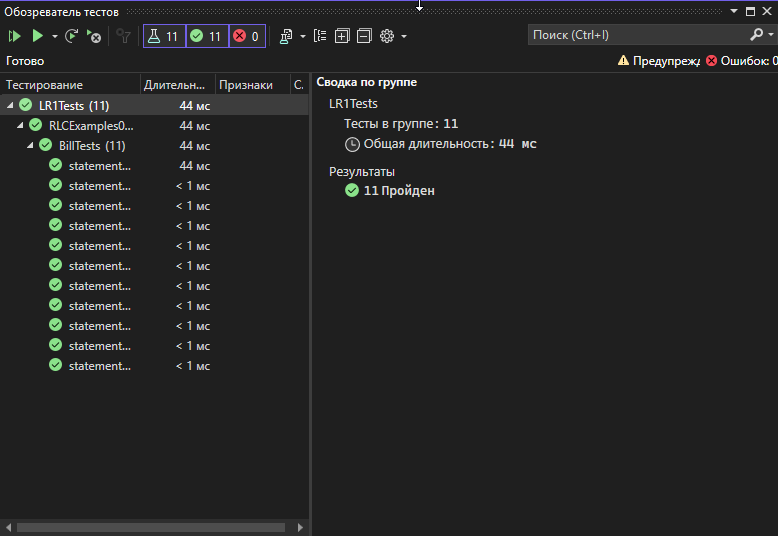


Рисунок 3 – Состояние тестов после преобразования метода statement

Вывод: все задания выполнены успешно, изучены базовые техники рефакторинга позволяющие различным образом манипулировать методами между классами программы.

Листинг:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace RLCExamples01

{

// Класс, представляющий клиента магазина.

public class Customer

{

private int bonus;

private string name;

public Customer(string name, int bonus)

{

this.name = name;

this.bonus = bonus;

}

public string getName()

{

return name;

}

public int getBonus()

{

return bonus;

}

public void receiveBonus(int bonus)

{

this.bonus = bonus;

}

public int useBonus(int needBonus)

{

int bonusTaken;

if (needBonus > bonus)

{

bonusTaken = bonus;

bonus = 0;

}

else

{

bonusTaken = needBonus;

bonus = bonus - needBonus;

}

return bonusTaken;

}

public int getUsedBonus(Item item, double discount) {

return useBonus((int)(item.getSum())) - (int)discount;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace RLCExamples01

{

// Класс, представляющий данные о чеке

public class Item

{

private Goods \_Goods;

private int \_quantity;

private double \_price;

public Item(Goods Goods, int quantity, double price)

{

\_Goods = Goods;

\_quantity = quantity;

\_price = price;

}

public int getQuantity()

{

return \_quantity;

}

public double getPrice()

{

return \_price;

}

public Goods getGoods()

{

return \_Goods;

}

public int getBonus() {

return getGoods().getBonus(this);

}

public (double, double) getDiscount(Customer customer) {

return getGoods().getDiscount(customer, this);

}

public double getSum() {

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

}

}

}

namespace RLCExamples01 {

// Класс, который представляет данные о товаре

public class Goods {

public const int REGULAR = 0;

public const int SALE = 1;

public const int SPECIAL\_OFFER = 2;

private readonly string \_title;

private int \_priceCode;

public Goods(string title, int priceCode) {

\_title = title;

\_priceCode = priceCode;

}

public int getPriceCode() {

return \_priceCode;

}

public void setPriceCode(int arg) {

\_priceCode = arg;

}

public string getTitle() {

return \_title;

}

public int getBonus(Item item) {

int bonus = 0;

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

case REGULAR:

bonus = (int)(item.getSum() \* 0.05);

break;

case SPECIAL\_OFFER:

break;

case SALE:

bonus = (int)(item.getSum() \* 0.01);

break;

}

return bonus;

}

public (double, double) getDiscount(Customer customer, Item item) {

double discount = 0;

double usedBonus = 0;

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

case REGULAR:

if (item.getQuantity() > 2)

discount = item.getSum() \* 0.03; // 3%

if (item.getQuantity() > 5)

usedBonus = customer.getUsedBonus(item, discount);

break;

case SPECIAL\_OFFER:

if (item.getQuantity() > 10)

discount = item.getSum() \* 0.005; // 0.5%

if (item.getQuantity() > 1)

usedBonus = customer.getUsedBonus(item, discount);

break;

case SALE:

if (item.getQuantity() > 3)

discount = item.getSum() \* 0.01; // 1%

break;

}

return (discount, usedBonus);

}

}

}

namespace RLCExamples01

{

public class Bill

{

private List<Item> \_items;

private Customer \_customer;

public double totalAmount = 0;

public int totalBonus = 0;

public Bill(Customer customer)

{

this.\_customer = customer;

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

}

public void addGoods(Item arg)

{

\_items.Add(arg);

}

public string statement()

{

List<Item>.Enumerator items = \_items.GetEnumerator();

string result = getHeader();

while (items.MoveNext())

{

Item each = items.Current;

(double discount, double usedBonus) = each.getDiscount(\_customer);

int bonus = each.getBonus();

double sumWithDiscount = each.getSum() - discount;

double thisAmount = sumWithDiscount - usedBonus;

totalAmount += thisAmount;

totalBonus += bonus;

result += getItemString(each, discount, thisAmount, bonus);

}

\_customer.receiveBonus(totalBonus);

result += getFooter();

return result;

}

string getHeader()

{

return "Счет для " + \_customer.getName() + "\n" +

"\t" + "Название" + "\t" + "Цена" +

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

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

}

string getFooter()

{

return "Сумма счета составляет " + totalAmount.ToString() + "\n" +

"Вы заработали " + totalBonus.ToString() +

" бонусных баллов";

}

string getItemString(Item item, double discount, double thisAmount, int bonus)

{

return "\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";

}

public string oldStatement()

{

List<Item>.Enumerator items = \_items.GetEnumerator();

string result = "Счет для " + \_customer.getName() + "\n";

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

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

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

while (items.MoveNext())

{

double thisAmount = 0;

double discount = 0;

int bonus = 0;

Item each = (Item)items.Current;

//определить сумму для каждой строки

switch (each.getGoods().getPriceCode())

{

case Goods.REGULAR:

if (each.getQuantity() > 2)

discount =

(each.getQuantity() \* each.getPrice()) \* 0.03; // 3%

bonus =

(int)(each.getQuantity() \* each.getPrice() \* 0.05);

break;

case Goods.SPECIAL\_OFFER:

if (each.getQuantity() > 10)

discount =

(each.getQuantity() \* each.getPrice()) \* 0.005; // 0.5%

break;

case Goods.SALE:

if (each.getQuantity() > 3)

discount =

(each.getQuantity() \* each.getPrice()) \* 0.01; // 0.1%

bonus =

(int)(each.getQuantity() \* each.getPrice() \* 0.01);

break;

}

// сумма

thisAmount = each.getQuantity() \* each.getPrice();

// используем бонусы

if ((each.getGoods().getPriceCode() ==

Goods.REGULAR) && each.getQuantity() > 5)

discount +=

\_customer.useBonus((int)(each.getQuantity() \* each.getPrice()));

if ((each.getGoods().getPriceCode() ==

Goods.SPECIAL\_OFFER) && each.getQuantity() > 1)

discount =

\_customer.useBonus((int)(each.getQuantity() \* each.getPrice()));

// учитываем скидку

thisAmount =

each.getQuantity() \* each.getPrice() - discount;

//показать результаты

result += "\t" + each.getGoods().getTitle() + "\t" +

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

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

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

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

totalAmount += thisAmount;

totalBonus += bonus;

}

//добавить нижний колонтитул

result += "Сумма счета составляет " +

totalAmount.ToString() + "\n";

result += "Вы заработали " +

totalBonus.ToString() + " бонусных баллов";

//Запомнить бонус клиента

\_customer.receiveBonus(totalBonus);

return result;

}

}

}