Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И**

**РАДИОЭЛЕКТРОНИКИ (ТУСУР)**

Кафедра компьютерных систем в управлении и проектировании (КСУП)

Discounts

Проектная документация

Студент гр. 586-2

\_\_\_\_\_\_\_\_ Исанов А.А.

«\_\_» \_\_\_\_\_\_\_\_\_ 2018 г

Руководитель:

\_\_\_\_\_\_\_\_ А.А. Калентьев

«\_\_» \_\_\_\_\_\_\_\_\_ 2018 г

Томск 2018

Оглавление

[Техническое задание 3](#_Toc516693561)

[Проблема: 3](#_Toc516693562)

[Цель: 3](#_Toc516693563)

[Задачи: 3](#_Toc516693564)

[Контекст использования и конечный пользователь: 3](#_Toc516693565)

[Критерии качества: 3](#_Toc516693566)

[Функциональные возможности: 3](#_Toc516693567)

1. [Введение 4](#_Toc516693568)

2. [Описание программной системы 5](#_Toc516693569)

2.1. [UML диаграмма вариантов использования: 5](#_Toc516693570)

2.2. [UML диаграмма классов бизнес логики: 6](#_Toc516693571)

2.3. [UML диаграмма классов Приложения: 7](#_Toc516693572)

2.4. Описание классов скидок………………………………………...……… ……………..8

2.5. [Дерево ветвления git, полученное по окончанию работы над проектом: 11](#_Toc516693573)

3. [Тестирование 15](#_Toc516693574)

4. [Заключение 18](#_Toc516693575)

[Список использованной литературы: 19](#_Toc516693576)

# Техническое задание[1]

Проблема:

У кассиров в различных сферах (быстрое питание, продажа различных товаров, салоны красоты и так далее) отсутствуют программные средства, позволяющие оперативно вести учёт позиций в чеке для клиента, при этом автоматически высчитывая скидку.

Цель:

Разработать программное обеспечения для расчёта различных типов скидок к продаваемым услуге/товару, для дальнейшего занесения их в чек с возможностью поиска, удаления, редактирования, а также сохранения и открытия ранее созданных списков

Задачи:

-Работа с позициями в чеке в количестве не меньше тысячи

-Программа должна сохранять файлы в формате Json

-В программе должны быть предусмотрены следующие типы скидок: по купону, по процентам

-Архитектура должна позволять добавлять новые виды скидок

-Программа должна поставляться в установщике

Контекст использования и конечный пользователь:

Возрастная группа: средняя и старшая

Профессия: менеджер по продажам, кассир

Условия пользования и контекст: в любой организации, занимающейся продажами товаров и услуг, за рабочим компьютером возле кассы во время расчёта стоимости всех позиций в чеке для клиента.

Специальные ограничения: Не предусматриваются

Критерии качества:

-Интуитивно понятный интерфейс

Функциональные возможности:

1. Добавление позиции в чеке в каталог
2. Удаление позиции в чеке из каталога
3. Изменение позиции в чеке, находящейся в базе данных
4. Проверить наличие позиции в чеке в каталоге

# 1 Введение

Данная проектная документация предназначена для ознакомления пользователя с основным функционалом программы, с целью содействия в понимании, как её использовать. Также ответить на возможно возникающие в процессе использования вопросы. В ней описаны основные принципы работы программы, а также внутренние взаимосвязи сущностей и объектов, что упрощает изучение её работы, что в будущем позволяет меньше времени тратить при подключении к работе новых сотрудников, при развитии проекта другим специалистом, исправлении ошибок и поддержке данного приложения. Эффективная эксплуатация программного обеспечения достигается путём понимания всех аспектов его работы как разработчиком, так и пользователем, а также позволяет утвердить, что полученный результат соответствует ожиданиям заказчика, описанными в разделе «Техническое задание».

# 2 Описание программной системы

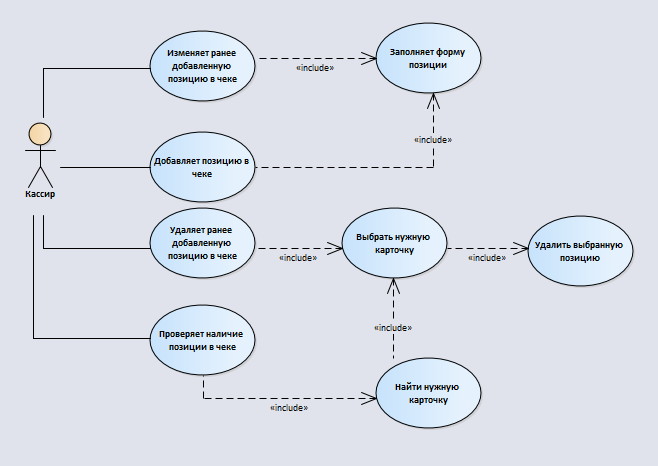
2.1. UML диаграмма вариантов использования[1]: 

Рисунок 2.1 – «Use case» model

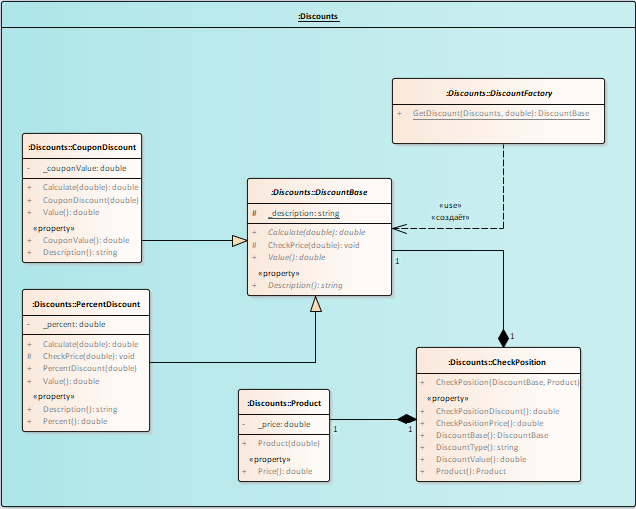
2.2. UML диаграмма классов бизнес логики[1]: 

Рисунок 2.2 – Library class model

2.3. UML диаграмма классов приложения[1]:

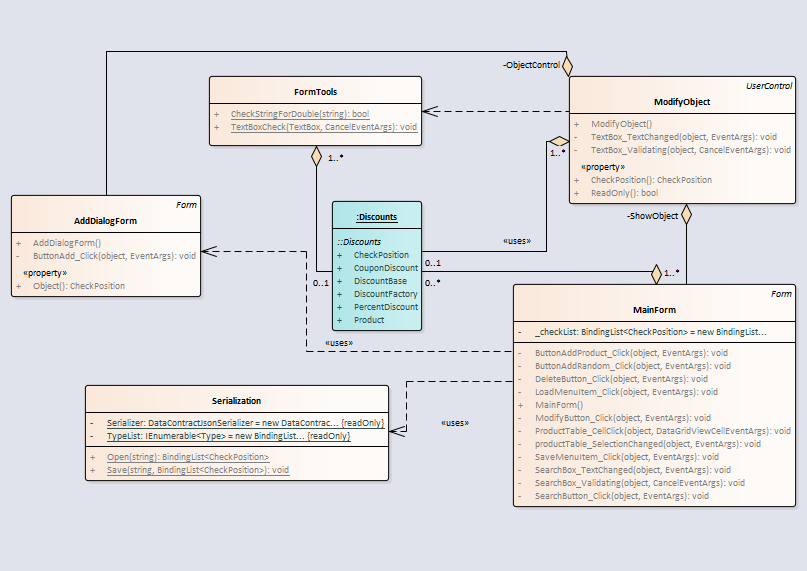


Рисунок 2.3 – Full architecture model

**2.4** Описание классов скидок:

Таблица 2.1 – Описание базового класса DiscountBase

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Базовый класс DiscountBase – Базовая скидка | | |
| Свойства | | |
| + Description | string | Формирует строку типа string которая является информацией о типе скидки |
| Методы | | |
| + Calculate(price) | double | Рассчитать итоговую стоимость товара  price – цена товара |
| + Value() | double | Вернуть значение скидки в зависимости от её типа |
| #CheckPrice(price) | void | Проверка корректного значения на логическое ограничение к полю Price  price – цена товара |

Таблица 2.2 – Описание класса CouponDiscount

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс CouponDiscount – сущность скидки по купону | | |
| Свойства | | |
| + CouponValue | double | Вернуть и установить размер скидки по купону |
| Методы | | |
| + Description | string | Метод, возвращающий string-описание скидки |
| +Value() | double | Метод, возвращающий значение скидки в зависимости от её типа: CouponDiscount - CouponValue |
| Calculate(price) | double | Расчёт итоговой стоимости товара со скидкой  price – цена товара |

Таблица 2.3 – Описание класса PercentDiscount

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс PercentDiscount – сущность скидки по процентам | | |
| Свойства | | |
| + Percent | double | Вернуть и установить размер скидки по процентам |
| Методы | | |
| + Description | string | Метод, возвращающий string-описание скидки |
| +Value() | double | Метод, возвращающий значение скидки в зависимости от её типа: PercentDiscount – Percent |
| Calculate(price) | double | Расчёт итоговой стоимости товара со скидкой  price – цена товара |

Таблица 2.4 – Описание класса CheckPosition

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс CheckPosition – сущность для описания позиции в чеке | | |
| Свойства | | |
| + DiscountBase | DiscountBase | Скидка |
| + Product | Product | Товар |
| + CheckPositionPrice | double | Вернуть цену товара в чеке |
| + DiscountType | string | Вернуть через string название типа скидки |
| + CheckPositionDiscount | double | Вернуть итоговую стоимость товара |
| + DiscountValue | double | Вернуть значение скидки в зависимости от типа |

Таблица 2.5 – Описание класса Product

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс Product – сущность для описания продукта | | |
| Свойства | | |
| + Price | double | Вернуть и установить цену товара |

Таблица 2.6 – Описание класса DiscountFactory

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс DiscountFactory – фабрика класса DiscountBase и наследников | | |
| Методы | | |
| + GetDiscount(key, value) | DiscountBase | Вернуть скидку определённого типа  key – тип скидки  value – значение скидки |

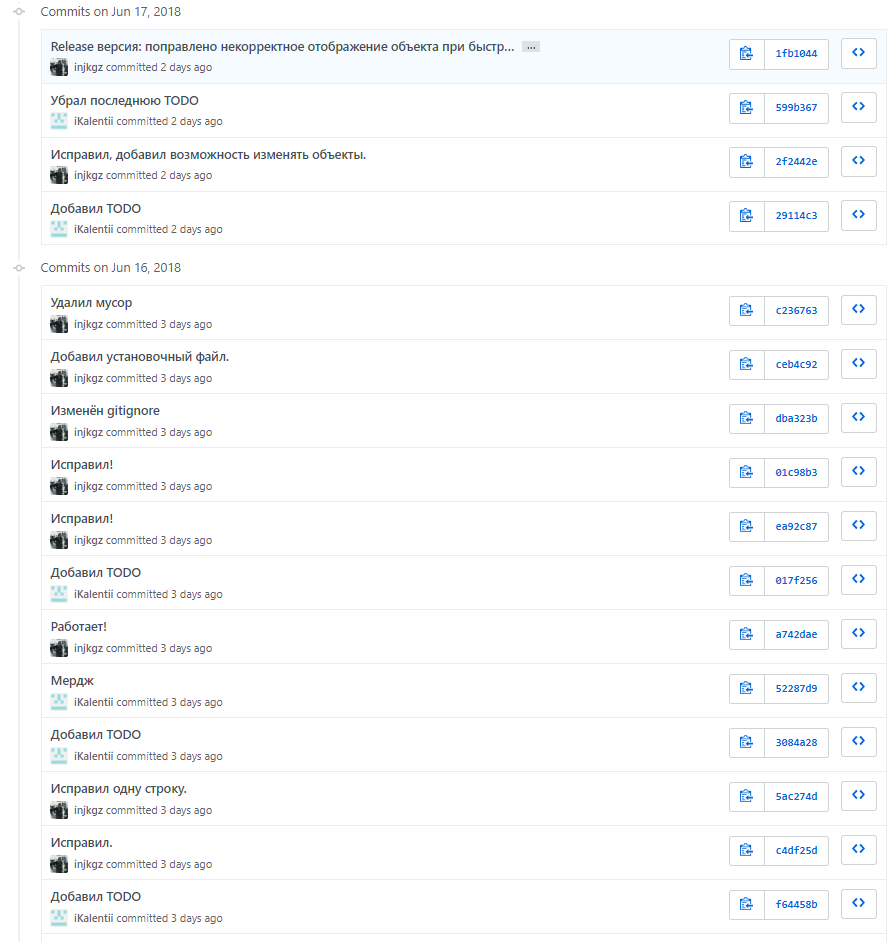
2.5 Дерево ветвления Git, полученное по окончанию работы над проектом[3]: 

Рисунок 2.4 – Дерево коммитов Git[3]

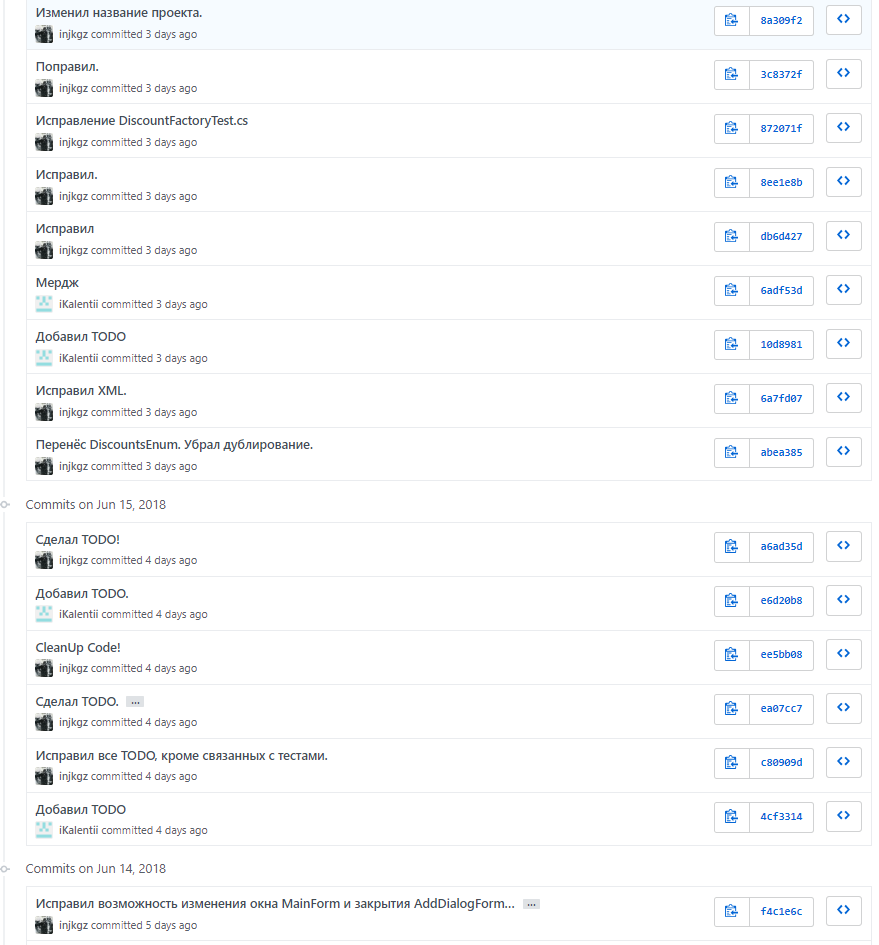


Рисунок 2.5 – Дерево коммитов Git[3]

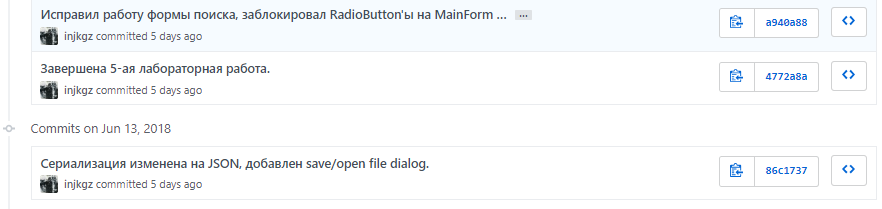
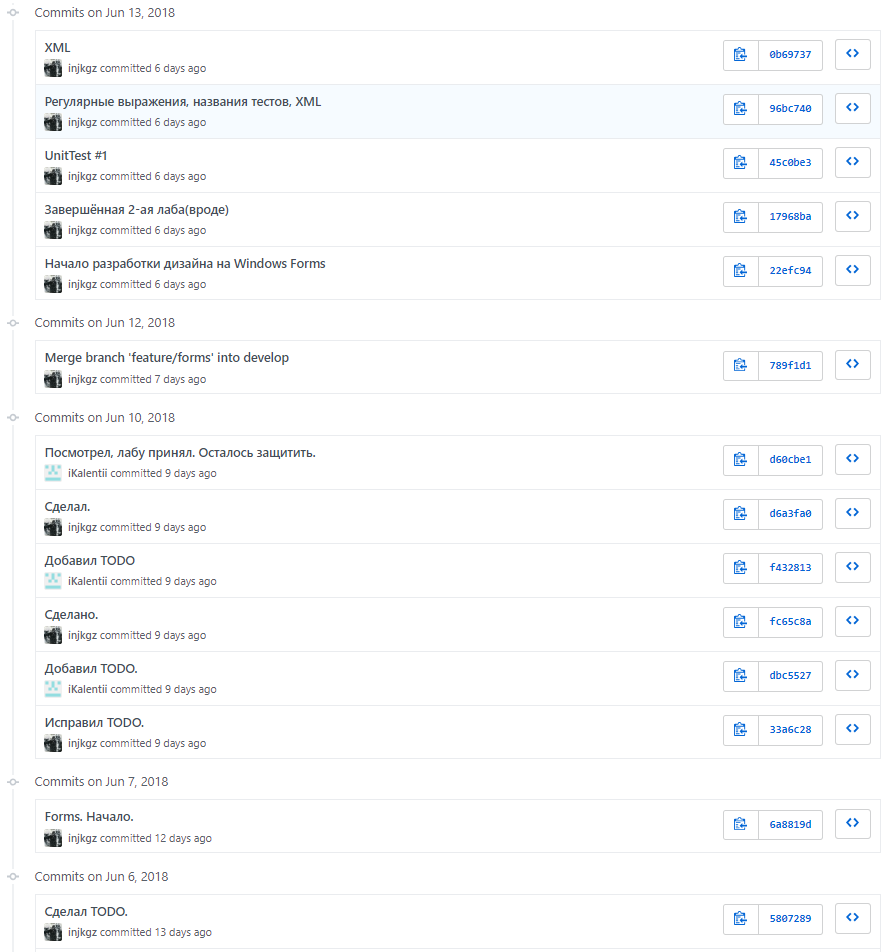
 

Рисунок 2.6 – Дерево коммитов Git[3]

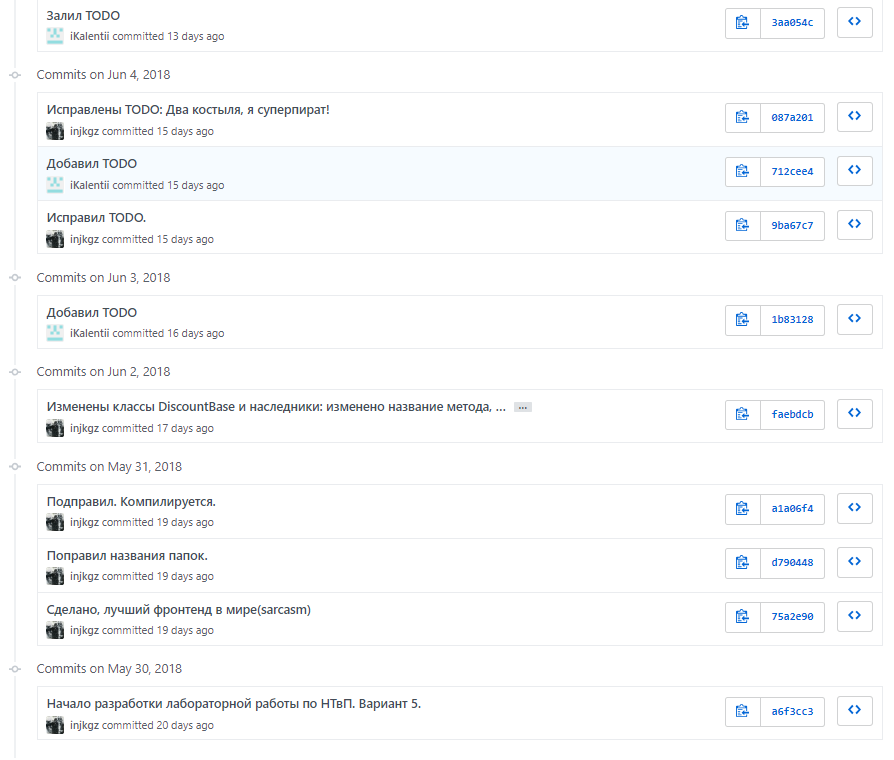


Рисунок 2.7 – Дерево коммитов Git[3]

# 3 Тестирование

На протяжении всего процесса выполнения работы программа тестировалась функционально по методу «white box»[1]. При создании сущности «позиция в чеке», в поля вводились как заведомо некорректные, так и корректные данные, после чего проверялась реакция системы. Процесс тестирования заведомо некорректными данными проходил в следующем порядке: тестирование добавления, удаления, изменения, поиска позиции в чеке, сохранения базы в файл и его открытия.

При тестировании функции создания новых сущностей, совершались попытки внести в поля данные, которые не соответствуют критериям выбранного поля. Программа выполнена таким образом, чтобы пользователь либо не мог ввести некорректные данные, либо высвечивалось окошко с ошибкой, в котором описывалась проблема, после чего все изменения сбрасывались. Данные тесты были успешно пройдены, после чего началось тестирование функции удаления.

Из созданного списка позиций в чеке случайно выбирались объекты, после чего производилось их удаление. Далее происходила проверка корректного удаления, удостоверившись, что удаление произошло, началось тестирование функции изменения.

Выбирался случайный объект из таблицы, далее были попытки не менять поля, оставить их пустыми, изменить на некорректные/корректные данные. Программа успешно отрабатывала, что ошибку при некорректном заполнении полей, что при корректном изменении карточки. Далее проводился тест поиска позиции в чеке по цене товара.

Вводились как существующие, так и не существующие данные. Во всех случаях реакция программы была соответствующая ожиданиям, после чего была протестирована функция удаления найденной карточки. После выбора позиции в чеке из списка найденных, была нажата кнопка удаления, после чего, данные выбранной позиции в чеке были удалены из всех таблиц программы.

Наконец были проверены функции сохранения и открытия файлов базы данных. Программа выдавала ошибку, если сохранённый файл подвергался некорректным изменениям данных извне, и отрабатывала корректно, в случае если файл и данные внутри соответствовали требованиям.

Далее последовал тест по методу «black box»[1]: ей пользовался человек, не знакомый с программой. Поведение программы соответствовало ожиданиям.

В конце было проведено модульное тестирование[1] открытых методов и свойств библиотеки Discount. Результатом стали 53 модульных теста, которые были успешно пройдены. Результаты приведены на рисунках 3.1 и 3.2.

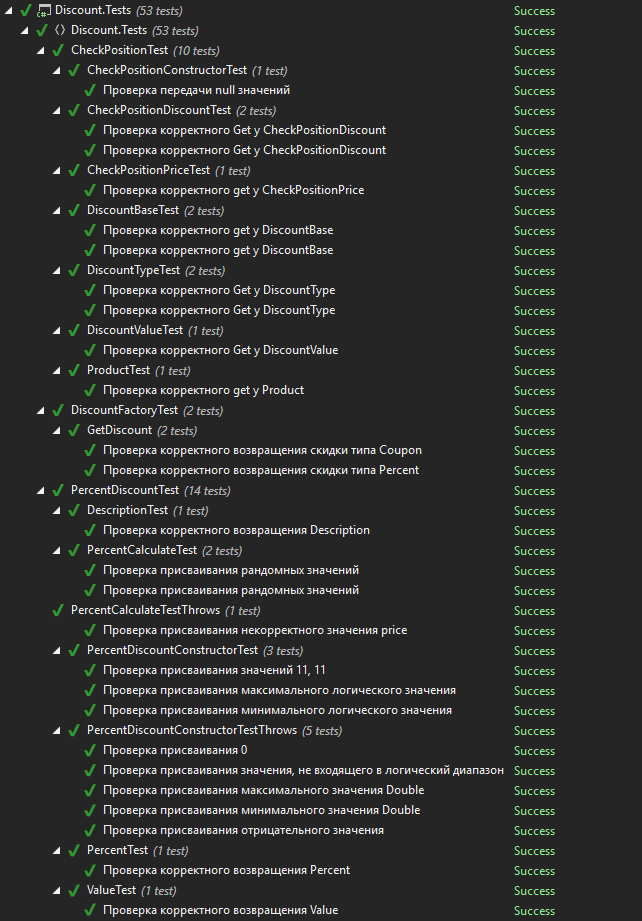
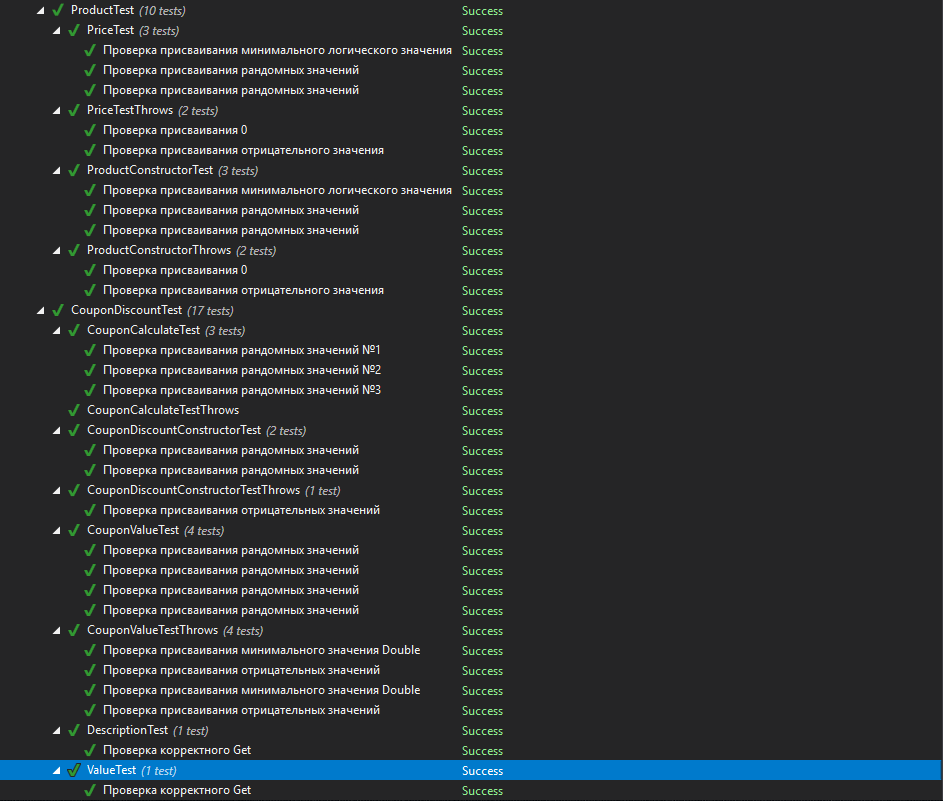
Рисунок 3.1 – Результаты тестов №1

Рисунок 3.2 – Результаты тестов №2

# 4 Заключение

В ходе выполнения работы, была разработана система хранения позиций в чеке: товар и скидка. Была разработана и протестирована .dll библииотека классов «Discount» на языке С#. После этого был разработан пользовательский интерфейс на WinForms[4], который использовал ранее разработанную библиотеку для добавления, удаления, изменения, поиска и хранения позиций в чеке.

В процессе выполнения работы активно использовался сервис GitHub, основанный на системе версионного контроля Git[3].

Для модульного тестирования использовалось расширение NUnit[6] для Visual Studio. Все классы в библиотеке «Discount» были полностью протестированы.

UML диаграмма классов отображает, что архитектура программы написана так, что дальнейшее развитие проекта и поддержка программы не потребуют значительных изменений в её основной архитектуре.

Программа полностью соответствует техническому заданию, готова к эксплуатации, была написана документация, а также скрипт установщика.

# Список использованной литературы:

1. Калентьев А. А. Новые технологии в программировании : учеб.

пособие / А. А. Калентьев, Д. В. Гарайс, А. Е. Горяинов. — Томск : Эль

Контент, 2014.

2. Работы студенческие по направлениям подготовки и специально-

стям технического профиля. Общие требования и правила оформления. ОС

ТУСУР 01-2013 53 с.

3. GitHub [Электронный ресурс]. URL: github.com (Дата обращения 11.06.2018)

4. Windows Forms | Microsoft Docs [Электронный ресурс].

URL: docs.microsoft.com/ru-ru/dotnet/framework/winforms (Дата обращения 12.06.2018)

5. Шпаргалка по git-flow [Электронный ресурс]. URL: danielkummer.github.io/git-flow-cheatsheet/index.ru\_RU.html (Дата обращения 13.06.2018)

6. NUnit.org [Электронный ресурс]. URL: nunit.org (Дата обращения 14.06.2018)