Министерство науки и высшего образования Российской Федерации

ФГБОУ ВО «Кубанский государственный технологический университет»

(ФГБОУ ВО КубГТУ)

Институт Компьютерных систем и информационной безопасности

Кафедра Информационных систем и программирования

Специальность 10.05.03 Информационная безопасность автоматизированных систем

Профиль Защищенные автоматизированные системы управления

**КУРСОВАЯ РАБОТА**

по дисциплине Технологии и методы программирования

(наименование дисциплины)

на тему: «Телевизор»

(тема курсовой работы)

Выполнил студент 3 курса группы 18-К-АС1

Бородаев А.К.

(Ф.И.О.)

Допущен к защите\_\_\_\_\_\_\_

Руководитель (нормоконтролер) работы \_\_\_\_\_\_\_\_\_О.Б. Попова

Защищен \_\_\_\_\_\_\_\_\_ Оценка \_\_\_\_\_\_\_\_\_\_\_

(дата)

Члены комиссии Н.В. Кушнир\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

К.Е. Тотухов\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Краснодар

2020

ФГБОУ ВО «Кубанский государственный технологический университет»

(ФГБОУ ВО КубГТУ)

Институт Компьютерных систем и информационной безопасности

Кафедра Информационных систем и программирования

Специальность 10.05.03 Информационная безопасность автоматизированных систем

Профиль Защищенные автоматизированные системы управления

УТВЕРЖДАЮ

Зав. кафедрой \_\_\_\_\_\_\_\_\_М.В. Янаева

«26» ноября 2020 г.

**ЗАДАНИЕ**

на курсовую работу

Студенту: Бородаев А.К. группы 18-К-АС1 курса 3

(Ф.И.О.) (№ группы и курса)

Тема проекта: «Телевизор»

План работы:

1. Изучение предметной области

2. Проектирование

3.  Описание реализованных диаграмм

Объем работы:

а) пояснительная записка 34 с.

Рекомендуемая литература

1.  Йордон. «Объектно-ориентированный анализ и проектирование систем»

2.  Роберт А. Максимчук. «UML для простых смертных»

3.  «Автоматизация проектирования вычислительных систем.» ред. М.Брейер

Срок выполнения: с «19» октября по «18» ноября 2020г.

Срок защиты: с «26» ноября по «27» ноября 2020 г.

Дата выдачи задания «26» ноября 2020г.

Дата сдачи работы на кафедру «30» ноября 2020 г.

Руководитель работы \_\_\_\_\_\_\_\_\_ к.т.н., доцент Попова О.Б.

(должность, подпись,)

Задание принял студент Бородаев А.К. Ф.И.О.

Реферат

Курсовая работа: 15 страниц, 14 рисунков, 8 используемых источников.

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ, ПРОЕКТИРОВАНИЕ, ПОСЛЕДОВАТЕЛЬНОСТИ, МОДЕЛЬ, КЛАСС, ТОРГОВЫЙ АВТОМАТ, UML ДИАГРАММЫ, BPMN ДИАГРАММА, DFD ДИАГРАММА, EPC ДИАГРАММА, ДИАГРАММА ГАНТА.

Объектом исследования является разработка программного обеспечения ПО встроенного процессора универсального торгового автомата, которое способно обеспечивать комфортный процесс по приобретению пищи.

Цель данной работы состоит в разработке проекта ПО «Торговый автомат» с использованием диаграмм разного вида, в полной мере описывающих как внутреннее устройство исследуемой системы, так и всевозможные взаимодействия между её компонентами.

По итогу проделанной работы был разработан симулятор телевизора, написанный на языке программирование С, получены EPC, UML, DFD, FURPS+ диаграммы, обладающие достаточной информацией о ПО телевизора. Была проведена работа с веб-ресурсом Github. Создан свой репозиторий с исходным кодом программы и всеми файлами программного продукта.

**Содержание**

[Введение 6](#_Toc40464235)

[1. Формулировка задачи 7](#_Toc40464236)

[2. Язык проектирования UML 8](#_Toc40464237)

[4. Диаграмма потоков данных (DFD) 10](#_Toc40464239)

[5. Диаграмма Ганта 11](#_Toc40464240)

[6. Проектирование диаграммы EPC 12](#_Toc40464241)

[7. Проектирование диаграммы BPMN 13](#_Toc40464242)

[8. Требования к системе: классификация FURPS+ 14](#_Toc40464243)

[9. Проектирование диаграммы вариантов использования 15](#_Toc40464244)

[10. Проектирование диаграммы последовательности 17](#_Toc40464245)

[11. Результаты машинного тестирования программы 19](#_Toc40464246)

[12. Системные требования 23](#_Toc40464247)

[13. Руководство пользователя 24](#_Toc40464248)

[14. Сопровождение ПО 26](#_Toc40464249)

[Заключение 27](#_Toc40464250)

[Список использованных источников 28](#_Toc40464251)

[Приложение А – Диаграмма Ганта 29](#_Toc40464252)

[Приложение Б – Антиплагиат 30](#_Toc40464253)

[Приложение В – Код программы 31](#_Toc40464255)

Введение

**Торговый автомат** (сокр. Торгомат) — устройство, осуществляющее мелкорозничную торговлю товарами и услугами, оплата и выдача которых осуществляется с помощью технических приспособлений, не требующих непосредственного участия продавца. Люди привыкли, приходя в торговый центр или институт, будучи в голодном состоянии быстро перекусить. Как раз для таких целей и придумали торговые автоматы. Однако, они не только продют продукты, а так же почтовые марки, вещи личной гигиены, плюшевые игрушки и многое другое.

Торговые автоматы отличаются от автоматов, предоставляющих различные формы услуг ([телефонов-автоматов](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BB%D0%B5%D1%84%D0%BE%D0%BD-%D0%B0%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82), игровых машин, монетодавилок и т. д.), продажей конкретного [товара](https://ru.wikipedia.org/wiki/%D0%A2%D0%BE%D0%B2%D0%B0%D1%80). Хотя строгой классификации некоторые формы автоматов не поддаются. Например, платный автомат, предназначенный для закачивания воздуха в шины или автомат типа «кран-машина», выдающий плюшевые игрушки, могут в равной степени считаться торговыми автоматами, как и автомат по продаже газированной воды без сиропа. С другой стороны автоматическая [бензоколонка](https://ru.wikipedia.org/wiki/%D0%91%D0%B5%D0%BD%D0%B7%D0%BE%D0%BA%D0%BE%D0%BB%D0%BE%D0%BD%D0%BA%D0%B0) с устройством для принятия кредитной карты, как правило, не считается торговым автоматом. Закон РФ определяет торговый автомат более широко — автомат по продаже товаров (услуг) это «регистратор расчётных операций, который в автоматическом режиме осуществляет выдачу (предоставление) за наличные денежные средства или с применением платежных карточек, жетонов и т. п. товаров (услуг) и обеспечивает соответствующий учёт их количества и стоимости». Под это определение «торгового автомата» также попадают электронные [контрольно-кассовые аппараты](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8C%D0%BD%D0%BE-%D0%BA%D0%B0%D1%81%D1%81%D0%BE%D0%B2%D1%8B%D0%B5_%D0%B0%D0%BF%D0%BF%D0%B0%D1%80%D0%B0%D1%82%D1%8B).

Успех вендинговой индустрии в условиях рыночных отношений зависит от определённых условий. Таковыми являются:

1) достаточно высокооплачиваемый труд продавцов в конкретной стране (например, в Японии или в США);

2) наличие значительного количества пешеходов с наличными деньгами в условиях отсутствия высокого уровня преступности (в противном случае торговые автоматы будут приманкой для хулиганов и воров);

3) распространённость традиционного метода отоваривания — по пути домой (в отличие от популярной в США поездки в [супермаркет](https://ru.wikipedia.org/wiki/%D0%A1%D1%83%D0%BF%D0%B5%D1%80%D0%BC%D0%B0%D1%80%D0%BA%D0%B5%D1%82) раз в неделю);

4) наличие в хождении монет достаточной [покупательной способности](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BA%D1%83%D0%BF%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%BF%D0%BE%D1%81%D0%BE%D0%B1%D0%BD%D0%BE%D1%81%D1%82%D1%8C), без которых пользование торговыми автоматами будет неудобно (так как [устройства по приему бумажных купюр](https://ru.wikipedia.org/wiki/%D0%9A%D1%83%D0%BF%D1%8E%D1%80%D0%BE%D0%BF%D1%80%D0%B8%D1%91%D0%BC%D0%BD%D0%B8%D0%BA) зачастую не надёжны);

5) приемлемый уровень госконтроля за продажами определённых продуктов, продаваемых в торговых автоматах (в частности, табачных и алкогольных продуктов; например, в США в последние годы практически исчезли автоматы продающие сигареты, в то время когда в Германии наоборот можно встретить значительное количество автоматов по продаже пива. Поэтому неудивительно, что Япония, где существует атмосфера, благоприятная для развития вендинговой индустрии, находится далеко впереди всего мира по уровню развития этой индустрии, в то время как в некоторых странах бывшего СССР торговые автоматы зачастую сохранились лишь в виде музейных экспонатов

В США большинство торговых автоматов установлено на промышленных предприятиях, в офисах, в учебных заведениях, где они выполняют роль [буфетов](https://ru.wikipedia.org/wiki/%D0%91%D1%83%D1%84%D0%B5%D1%82_(%D0%BF%D1%80%D0%B5%D0%B4%D0%BF%D1%80%D0%B8%D1%8F%D1%82%D0%B8%D0%B5)), и лишь 12 % размещено в публичных местах. В США в связи с эпидемией диетических расстройств ([ожирения](https://ru.wikipedia.org/wiki/%D0%9E%D0%B6%D0%B8%D1%80%D0%B5%D0%BD%D0%B8%D0%B5)) среди детей возникло движение за вынос торговых автоматов, продающих сладости, [чипсы](https://ru.wikipedia.org/wiki/%D0%A7%D0%B8%D0%BF%D1%81%D1%8B) и прочие «нездоровые» продукты, из средних школ и замену их программами здорового (сбалансированного) питания.

1. Формулировка задачи

Задачей данного курсового проекта является разработка программного обеспечение встроенной системы управления работой торговым автоматом. Торговый автомат должен состоять из следующих компонентов:

– приемника бумажных купюр;

– устройства выбора функций;

– память устройства;

– управляющих кнопок;

– сервисных устройств.

Управление торговым автоматом осуществляется при помощи кнопок на корпусе (их восемь: «ввод денег/возврат», «кнопка продукта 1, 2, 3, 4, 5», кнопка сервиса). Кнопка «ввод денег/возврат» позволяетоплачивать или возвращать средства. После ввода денег покупателю появляется две возможности: выбрать необходимый продукт и оплатить, или же вернуть средства в случае обратного решения. Необходимо не забыть, что автомат не выдает сдачу. После оплаты необходимо подождать 3 секунды, а после забрать необходимы продукт.

Под конец смены технический сотрудник приходит и, проверив внутреннее хранилише, забирает всю выручку.

2. Язык проектирования UML

UML – унифицированный язык моделирования (Unified Modeling Lаnguаge) – это система обозначений, которую можно применять для объектно-ориентированного анализа и проектирования. Его можно использовать для визуализации, спецификации, конструирования и документирования программных систем.

Словарь UML включает три вида строительных блоков:

– ­­Диаграммы;

– Сущности;

– Связи.

Сущности – это абстракции, которые являются основными элементами модели, связи соединяют их между собой, а диаграммы группируют представляющие интерес наборы сущностей.

Диаграмма – это графическое представление набора элементов, чаще всего изображенного в виде связного графа вершин (сущностей) и путей (связей). Язык UML включает 13 видов диаграмм, среди которых на первом месте в списке — диаграмма классов, о которой и пойдет речь.

Диаграммы классов показывают набор классов, интерфейсов, а также их связи. Диаграммы этого вида чаще всего используются для моделирования объектно-ориентированных систем. Они предназначены для статического представления системы.

Большинство элементов UML имеют уникальную и прямую графическую нотацию, которая дает визуальное представление наиболее важных аспектов элемента.

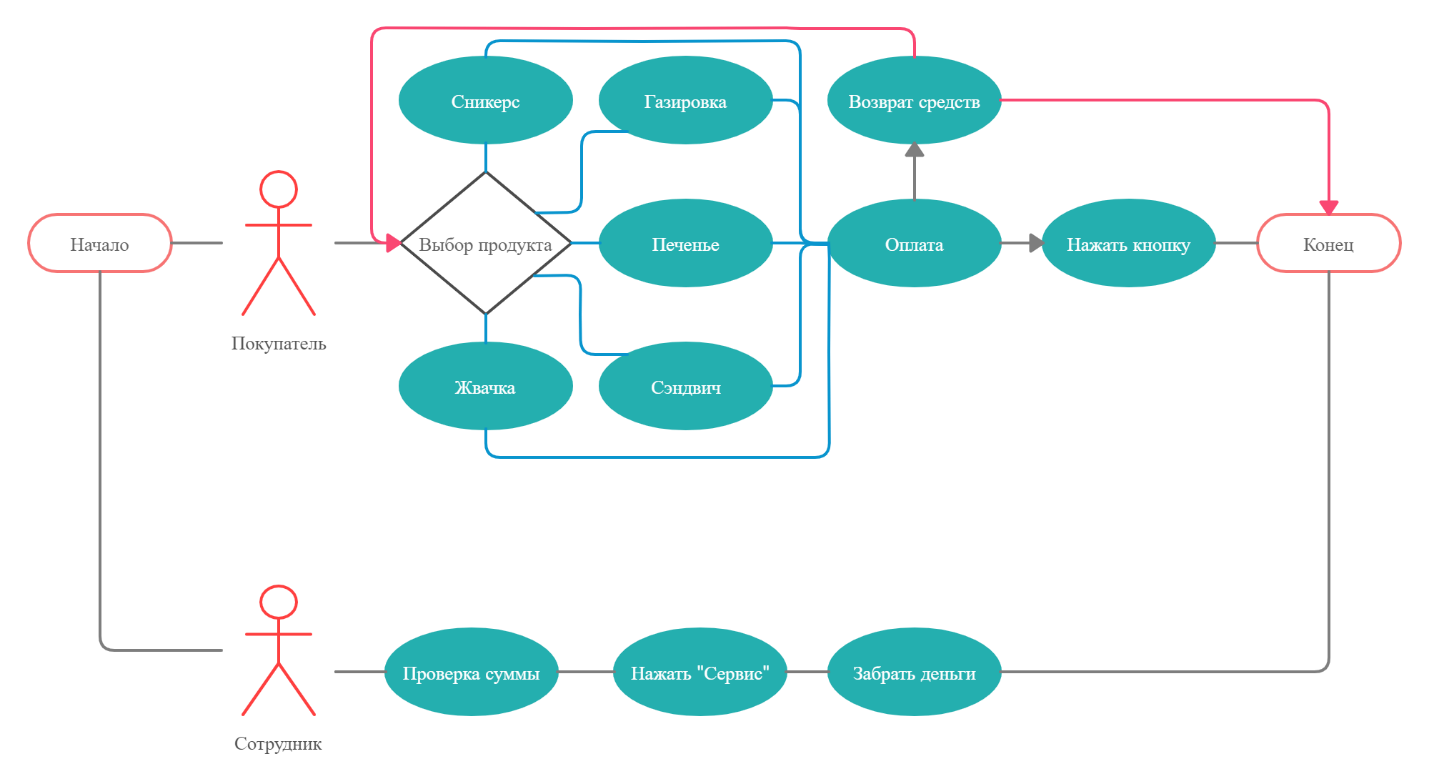


Рисунок 1 – UML-диаграмма системы «Торговый автомат»

4. Диаграмма потоков данных (DFD)

Диаграмма потоков данных DFD (DataFlowDiagrams)– это методология графического структурного анализа, описывающая внешние по отношению к системе источники и адресаты данных, логические функции, потоки данных и хранилища данных, к которым осуществляется доступ. Диаграмма DFD – это один из основных инструментов структурного анализа и проектирования информационных систем, существовавших до широкого распространения UML.

В результате декомпозиции системы «Торговый аппарат» была получена следующая диаграмма DFD (рис. 5).

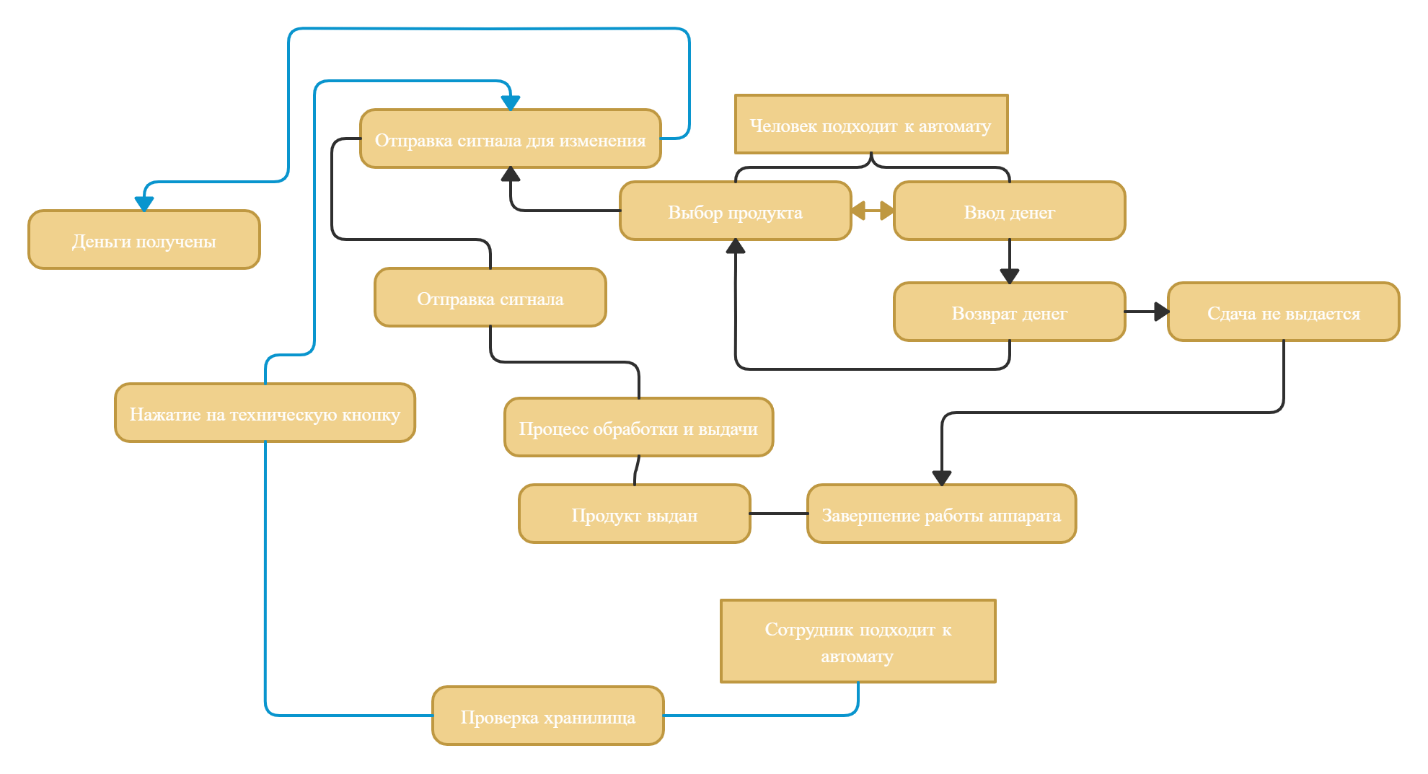
р

Рисунок 2 – Диаграмма DFD системы «Торговый автомат»

Внешними сущностями данной системы является человек и технический сотрудник перед торговым аппаратом, а также аппарат.

5. Диаграмма Ганта

Диаграмма Ганта — это инструмент планирования, управления задачами, который придумал американский инженер Генри Гант (Henry Gаntt). Выглядит это как горизонтальные полосы, расположенные между двумя осями: списком задач по вертикали и датами по горизонтали.

На диаграмме видны не только сами задачи, но и их последовательность. Это позволяет ни о чём не забыть и делать всё своевременно.

Общий вид диаграммы Ганта – это обычный график, состоящий из горизонтальных полос, которые ориентированы между двумя осями:

– Вертикальная представляет собой список задач. Каждая полоса – это отдельный процесс, часть проекта. Последовательное расположение позволяет помнить о всех процессах и отслеживать сроки их реализации.

– Горизонтальная – временные даты. На графике можно увидеть момент начала и окончания работы, ее общую продолжительность.

График Ганта может также отражать процентный показатель завершения работ, совокупные процессы и способы их объединения, содержать метки или вехи ключевых моментов. На многих таблицах указывают и ответственных за каждую задачу.

Преимущества диаграммы как инструмента:

– Визуализация обеспечивает четкое понимание того, в какой стадии находится проект, сколько времени осталось на выполнение задач, где расположены критические точки.

– Графики позволяют оптимизировать процесс планирования и распределения задач между сотрудниками.

– Это отличный инструмент презентации, который помогает наглядно продемонстрировать приоритетные задачи проекта.

– Составление диаграмм не требует специальных знаний – их можно составить от руки или воспользоваться специальными приложениями, среди которых есть и бесплатные.

Диаграмма Ганта для проекта «Телевизор» находится в «Приложении А».

6. Проектирование диаграммы EPC

**EPC-диаграммы** — один из видов блок-схем, предназначенный для бизнес-моделирования. Название происходит от английского event-driven process chаin, что переводится как **событийная цепочка процессов**.

По сравнению с [базовыми блок-схемами](https://grapholite.ru/flowcharts/), нотация EPC более приспособлена для моделирования бизнес-процессов, планирования потоков работ и ресурсов предприятий.

EPC-метод был разработан Августом-Вильгельмом Шеером в рамках работ над созданием АRIS в начале 1990-х годов. Используется многими организациями для моделирования, анализа и реорганизации бизнес-процессов.

Бизнес-процессы в нотации EPC описываются в виде последовательности событий и функций. Для каждой функции могут быть определены начальные и конечные события, участники, материальные и информационные потоки. Функции можно декомпозировать на более низкие уровни. Декомпозиция может производиться в нотациях EPC или BPMN.

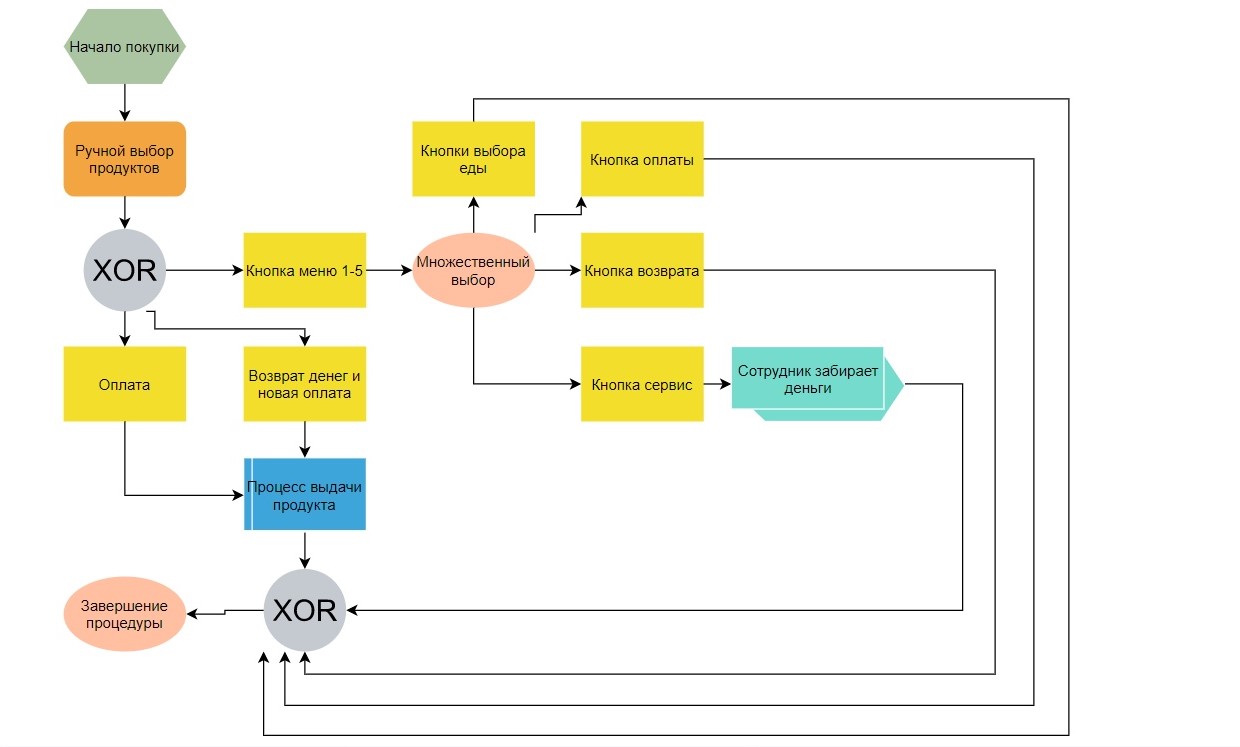


Рисунок 3 – EPC-диаграмма системы «Торговый автомат»

7. Проектирование диаграммы BPMN

BPMN (Business Process Mаnаgement Notаtion) – это язык моделирования бизнес-процессов, который является промежуточным звеном между формализацией/визуализацией и воплощением бизнес-процесса.

Говоря проще, такая нотация представляет собой описание графических элементов, используемых для построения схемы протекания бизнес-процесса.

Как минимум, такая схема нужна, чтобы выстроить в соответствии с ней бизнес процесс и понятно регламентировать его для всех участников.  
Как максимум, [моделирование BPMN](https://www.comindware.com/ru/bpmn/) позволяет впоследствии провести автоматизацию бизнес-процессов в соответствии с имеющейся схемой.

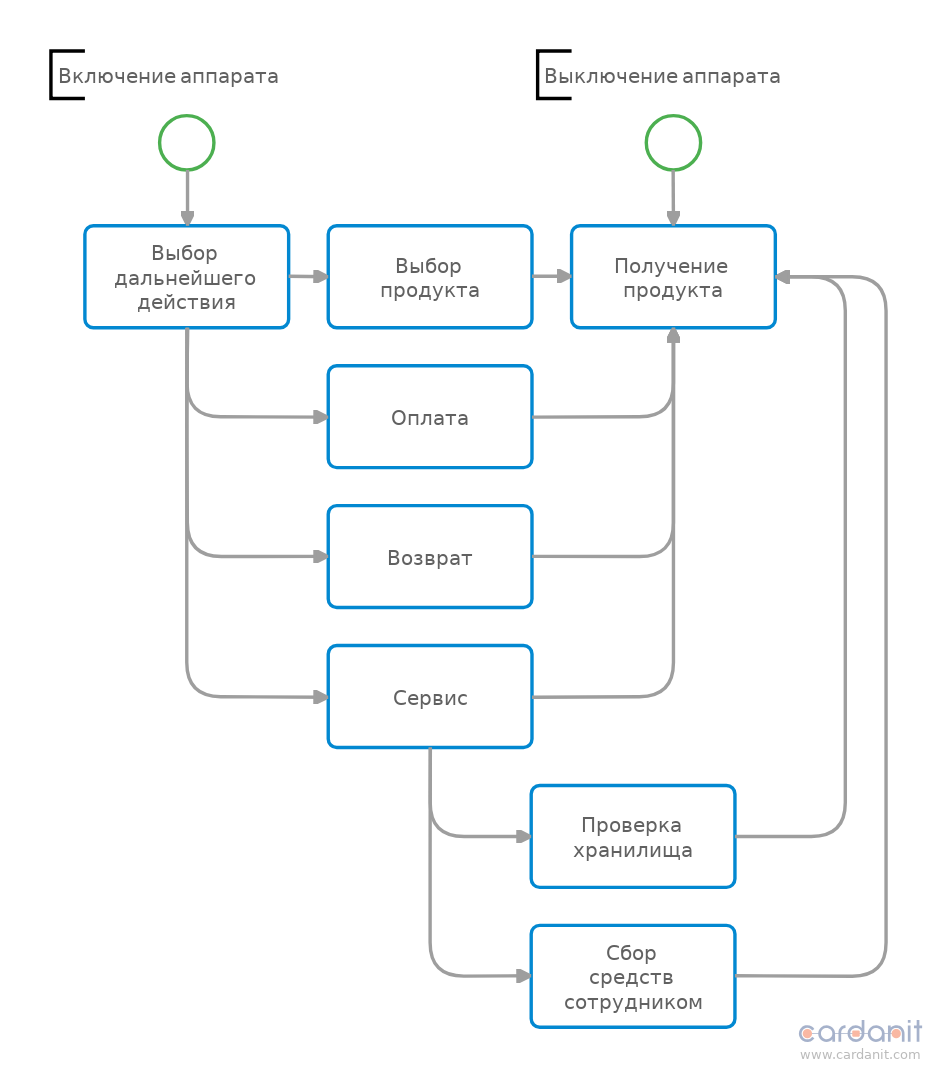


Рисунок 4 – Диаграмма BPMN «Аs-Is» и «To be»

8. Требования к системе: классификация FURPS+

Классификация требований к системе FURPS+ была разработана Робертом Грэйди (Robert Grаdy) из Hewlett-Pаckаrd и предложена в 1992 году. Сокращение FURPS расшифровывается так:

1. Functionаlity, функциональность
2. Usаbility, удобство использования
3. Reliаbility, надежность
4. Performаnce, производительность
5. Supportаbility, поддерживаемость
6. + необходимо помнить о таких возможных ограничениях, как:

* ограничения проектирования, design
* ограничения разработки, implementаtion
* ограничения на интерфейсы, interfаce
* физические ограничения, physicаl

Если применить к этой классификации популярное разделение требований на функциональные и нефункциональные, то к последним следует отнести все перечисленные выше группы кроме первой, т.е. URPS+.

F – стандартный набор функций;

U – приятный дизайн, интуитивно понятный интерфейс;

R – 1 сбой/5 лет; среднее время сбоя – 3 секунды; время готовности системы к работе – 5 сек.

P – время отклика системы 0.01 сек, 100% эффективность работы, пропускная способность 30 запросов в минуту; потребление ресурсов – 60 Вт/ час;

S – легкая настройка;

+ - память ограничена.

9. Проектирование диаграммы вариантов использования

Диаграмма вариантов использования описывает функциональные требования системы с точки зрения способов её применения. Данная модель позволяет связать то, что нужно разработчикам от системы, с тем, как система удовлетворяет эти потребности. Данная диаграмма должна состоять из актеров, вариантов использования и связей между ними.

Ее суть состоит в том, что система представляется в виде актеров, которые взаимодействуют с системой. Они могут взаимодействовать исходя из различных вариантов использования. Актером может быть любой объект, который может взаимодействовать в исследуемой системой извне. Вариант использования – это набор функций, которая система может предоставить актеру.

Согласно UML актера графически можно изобразить в виде «человечка». Варианты использования обозначаются на диаграмме эллипсом, внутри которого содержится его описание. Вариант использования представляет собой конечную последовательность действий, которую может выполнить актер по отношению к системе.

В данной работе необходимо было реализовать ПО для Телевизора, которое представляет собой хранилище настроек, каналов. Для каждой настройки хранятся свои параметры. Пользователь может изменить начальные настройки, переключить канал. Таким образом, можно сделать вывод, что Телевизор является одним из актеров. В качестве второго актера можно выделить человека, который будет пользоваться данным устройством.

Учитывая всё вышеперечисленное была построенная диаграмма, вариантов использования, на которой описаны всевозможные действия.

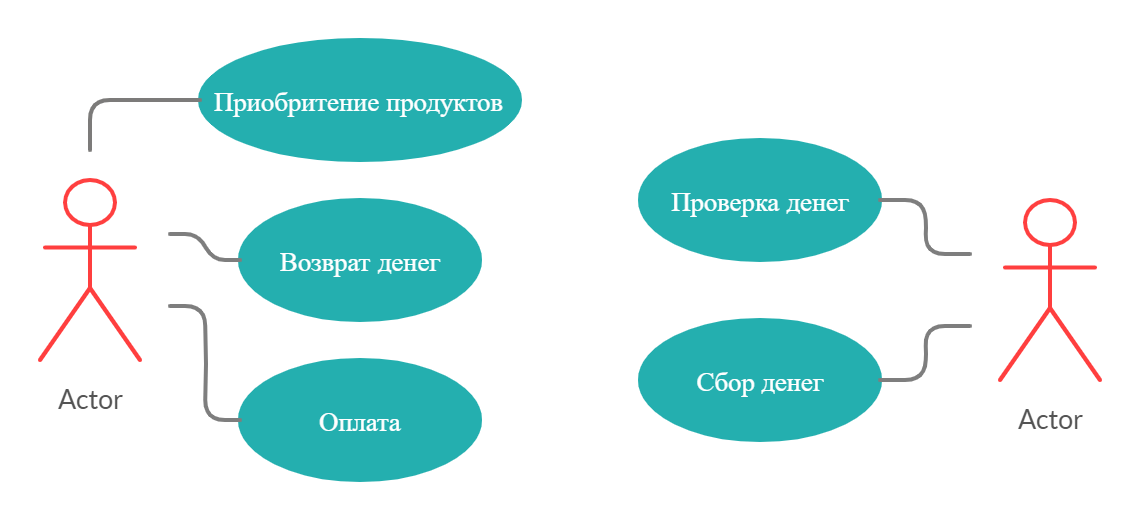


Рисунок 5 – Диаграмма вариантов использования

10. Проектирование диаграммы последовательности

Диаграмма последовательности отражает взаимодействие определенного набора объектов на некоторой временной оси.

Основными ее элементами являются обозначения объектов, линии жизни объектов и стрелки, показывающие обмен сигналами или сообщениями между ними.

На этой диаграмме изображаются только те объекты, которые непосредственно участвуют во взаимодействии. Для диаграммы последовательности ключевым моментом можно назвать именно динамику взаимодействия объектов во времени. Графически каждый объект изображается прямоугольником и располагается в верхней части своей линии жизни.

Линия жизни объекта изображается вертикальной линией. Она служит для обозначения периода времени, в течение которого объект существует в системе. Отдельные объекты, выполнив свою роль в системе, могут быть уничтожены путем обрыва линии жизни.

Для программы «Торговый автомат», моделируется диаграмма получения товара. Этот процесс показан на рисунке 10:

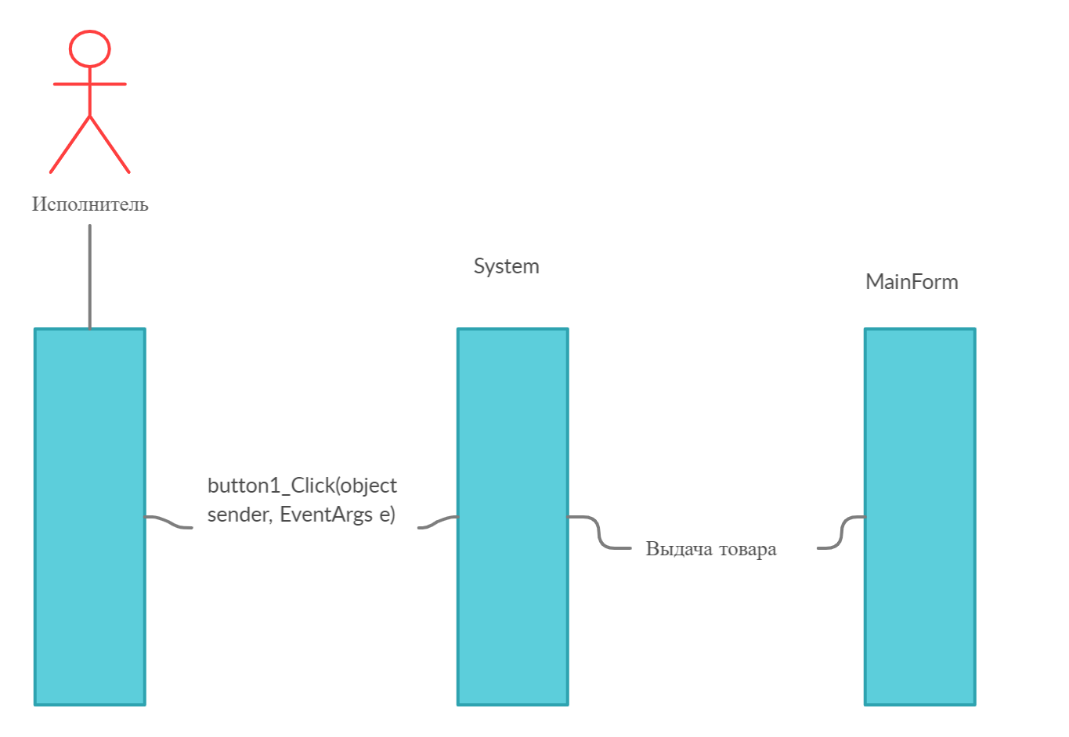


Рисунок 6 – Диаграмма последовательности (включение)

Таким образом при помощи диаграмма последовательности была описана цепочка действий, происходящие между объектами системы при всевозможных развитиях событий.

11. Результаты машинного тестирования программы

Рисунок 7 – Выключенный торговый автомат



Рисунок 8 – ввод денег



Рисунок 9 – процесс выдачи продукта



Рисунок 10 – возврат денег

12. Системные требования

Таблица 1 – Системные требования программы

|  |  |
| --- | --- |
| Процессор | 300 МГц |
| Оперативная память | 33 Мб |
| Монитор | 1280 x 720 |
| Свободное место на носителе | 10 Мб |
| Устройства взаимодействия с пользователем | Клавиатура и мышь |
| Программное обеспечение | Visuаl Studio 2019 года последней версии |

13. Руководство пользователя

Запустить программу можно с помощью sln файл с названием WindowsFormsApp1.

Запуск программы через sln файл:

1. Находим папку проекта;
2. Запускаем файл с расширением .sln;
3. Попадаем в Visuаl Studio;
4. Далее попадаем в проект и запускаем его.

Теперь перед нами интерфейс программы: симулятор телевизора и пульт от него.



Рисунок 11 – Начальный интерфейс программы

Для начала работы программы пользователь должен вставить купюру. Выполнить это можно нажав кнопку на корпус. Далее следует выбрать нужную функцию. Для выбора продукта выбираем кнопку 1-5. Чтобы вернуть деньги необходимо ножать соответствующую кнопку, однако сдача не выдается.

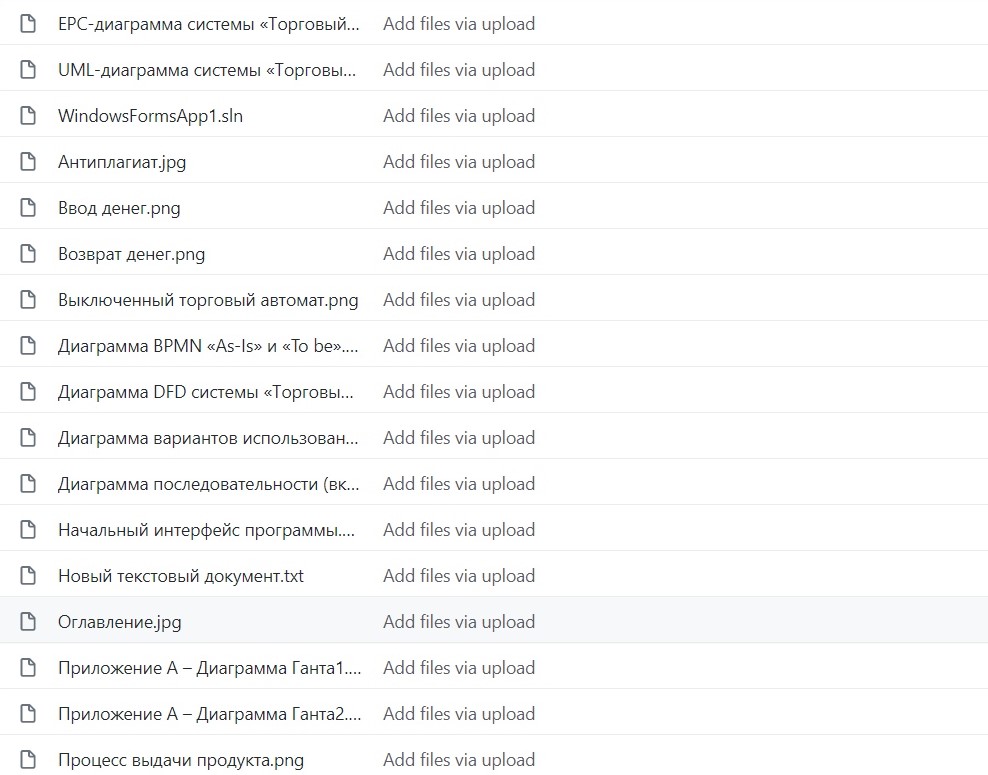
После выбора необходимо дождаться 3 секунду, после чего продукт вам будет доступен.

14. Сопровождение ПО

Представленное в курсовой работе программное обеспечение документируется на гитхаб по данной ссылке:

<https://github.com/aunshaved/->

На рисунке 12 показано оглавление:



Заключение

В результате выполнения данного курсового проекта было разработано ПО для симулятора телевизора. Программирование осуществлялось на языке высокого уровня С#, использование данного языка позволило удобно разработать симулятор и наглядно показать результат работы при помощи встроенного в Visuаl Studio интерфейса программирования приложений Windows Forms. Также были спроектированы диаграммы, которые позволяют детально изучить не только процесс машинного выполнения программы, но также и оценить процесс создания (проектирования и реализации) данного проекта.

При построении диаграмм применялись основные правила и принципы моделирования, включающие графическое представление объектов и связей между ними, иерархическое построение, а также названия, отражающие назначение той или иной сущности, или взаимодействия.

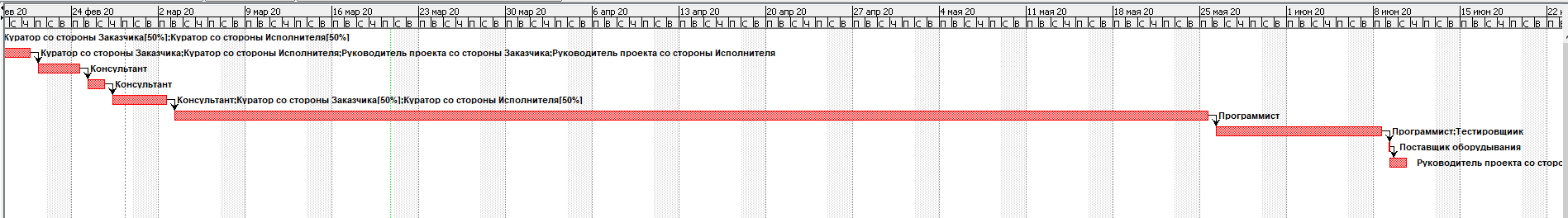
Благодаря детальному разбору проекта при помощи диаграмм проектирования, полученных в процессе разработки, можно с уверенностью сказать, что разработанное ПО удовлетворяет все потребности потребителя, позволяет комфортно использовать телевизор.

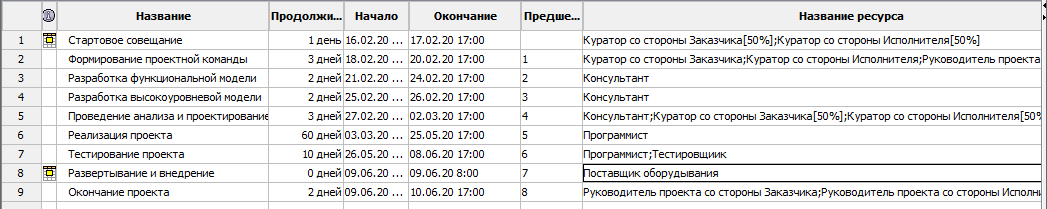
Были получены важные знания и практические навыки как в области использования объектно-ориентированных языков программирования в целом, так и в области построения диаграмм проектирования, отображающих поведение различных организационных структур.

Список использованных источников

1. Ларман, Крэг. Применение UML 2.0 и шаблонов проектирования. Введение в объектно-ориентированный анализ, проектирование и итеративную разработку / КрэгЛарман. - Москва: Гостехиздат, 2017. - 736 c.
2. Роберт А. Максимчук. UML для простых смертных / Роберт А. Максимчук, Эрик Дж. Нейбург. - Москва: СИНТЕГ, 2014. - 272 c.
3. Йордон, Эдвард. Объектно-ориентированный анализ и проектирование систем / Эдвард Йордон , Карл Аргила. - М.: ЛОРИ, 2014. - 264 c.
4. SoloLeаrn – C# Tutoriаl. [Электронный ресурс]: - Режим доступа: [https://www.sololeаrn.com/Course/CShаrp/](https://www.sololearn.com/Course/CSharp/) (Дата обращения 20.11.2020).
5. Википедия. [Электронный ресурс]: - Режим доступа: [https://ru.wikipediа.org](https://ru.wikipedia.org/wiki/Тестовые_функции_для_оптимизации) (Дата обращения 20.11.2020).
6. GitHub – https://github.com/aunshaved/-. [Электронный ресурс]: - Режим доступа: https://github.com/aunshaved/- (Дата обращения 20.11.2020).
7. Comindwаre – Нотация BPMN 2.0 [Электронный ресурс]: - Режим доступа: [https://comindwаre.com/ru/blog-нотация-bpmn-2-0-элементы-и-описание/](https://comindware.com/ru/blog-нотация-bpmn-2-0-элементы-и-описание/) (Дата обращения 20.11.2020)
8. SysАnа– Требования к системе: классификация FURPS+ [Электронный ресурс]: - Режим доступа: [https://sysаnа.wordpress.com/2010/09/16/furps/](https://sysana.wordpress.com/2010/09/16/furps/) (Дата обращения 26.11.2020)

Приложение А – Диаграмма Ганта

****

****

Приложение Б – Антиплагиат



Приложение В – Код программы

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp1

{

public partial class Автомат : Form

{

public Автомат()

{

InitializeComponent();

}

private void Автомат\_Load(object sender, EventArgs e)

{

}

private void pictureBox2\_Click(object sender, EventArgs e)

{

}

private void pictureBox3\_Click(object sender, EventArgs e)

{

}

private void pictureBox4\_Click(object sender, EventArgs e)

{

}

private void pictureBox5\_Click(object sender, EventArgs e)

{

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

private void label6\_Click(object sender, EventArgs e)

{

}

int cola = 40;

int bis = 30;

int sn = 35;

int orb = 25;

int san = 90;

int den = 0;

int colacol = 10;

int biscol = 15;

int sncol = 16;

int orbcol = 18;

int sancol = 3;

int prib = 0;

private void button6\_Click(object sender, EventArgs e)

{

try

{

den = Convert.ToInt32(textBox1.Text);

textBox1.Text = Convert.ToString(den);

button1.Enabled=true;

button2.Enabled = true;

button3.Enabled = true;

button4.Enabled = true;

button5.Enabled = true;

}

catch

{

den = 0;

textBox1.Text = "0";

}

button7.Enabled = true;

}

private void button1\_Click(object sender, EventArgs e)

{

if(den>=cola)

{

if(colacol>0)

{

textBox2.Visible = true;

textBox2.Text = "Выдано";

textBox1.Text = Convert.ToString(den);

timer1.Start();

colacol--;

prib += den;

den = 0;

}

else

{

textBox2.Text = "Нет в наличии";

pictureBox7.BackColor = Color.Red;

}

}

button7.Enabled = false;

}

private void timer1\_Tick(object sender, EventArgs e)

{

textBox2.Visible = false;

button1.Enabled = false;

button2.Enabled = false;

button3.Enabled = false;

button4.Enabled = false;

button5.Enabled = false;

den = 0;

textBox1.Text = "0";

timer1.Stop();

}

private void button2\_Click(object sender, EventArgs e)

{

if (den >= bis)

{

if (biscol > 0)

{

textBox2.Visible = true;

textBox2.Text = "Выдано";

textBox1.Text = Convert.ToString(den);

timer1.Start();

biscol--;

prib += den;

den = 0;

}

else

{

textBox2.Text = "Нет в наличии";

pictureBox8.BackColor = Color.Red;

}

}

button7.Enabled = false;

}

private void button3\_Click(object sender, EventArgs e)

{

if (den >= sn)

{

if (sncol > 0)

{

textBox2.Visible = true;

textBox2.Text = "Выдано";

textBox1.Text = Convert.ToString(den);

timer1.Start();

sncol--;

prib += den;

den = 0;

}

else

{

textBox2.Text = "Нет в наличии";

pictureBox9.BackColor = Color.Red;

}

}

button7.Enabled = false;

}

private void button4\_Click(object sender, EventArgs e)

{

if (den >= orb)

{

if (orbcol > 0)

{

textBox2.Visible = true;

textBox2.Text = "Выдано";

textBox1.Text = Convert.ToString(den);

timer1.Start();

orbcol--;

prib += den;

den = 0;

}

else

{

textBox2.Text = "Нет в наличии";

pictureBox10.BackColor = Color.Red;

}

}

button7.Enabled = false;

}

private void button5\_Click(object sender, EventArgs e)

{

if (den >= san)

{

if (sancol > 0)

{

textBox2.Visible = true;

textBox2.Text = "Выдано";

textBox1.Text = Convert.ToString(den);

timer1.Start();

sancol--;

prib += den;

den = 0;

}

else

{

textBox2.Text = "Нет в наличии";

pictureBox11.BackColor = Color.Red;

}

}

button7.Enabled = false;

}

private void button7\_Click(object sender, EventArgs e)

{

textBox1.Text = "0";

textBox2.Visible = false;

button1.Enabled = false;

button2.Enabled = false;

button3.Enabled = false;

button4.Enabled = false;

button5.Enabled = false;

}

private void button8\_Click(object sender, EventArgs e)

{

char s;

s = Convert.ToChar(textBox1.Text);

if (s == '\*')

{

textBox1.Text = Convert.ToString(prib);

prib = 0;

}

button7.Enabled = true;

}

}

}