**ФГБОУ ВО**

**Национальный исследовательский университет**

**«МЭИ»**

**Лабораторная работа № 7**

по курсу

«Технология программирования»

**МНОЖЕСТВЕННОЕ НАСЛЕДОВАНИЕ**

**Выполнил:**

еще не отчисленный

студент группы А-08-23

Потапов О. В.

Вариант №16

**Москва, 2024**

Задание

1. Построить иерархию классов в соответствии с вариантом задания.
2. Разработать методы и свойства для каждого из определяемых классов.
3. Реализовать программу на C# в соответствии с вариантом задания.
4. Изменить иерархию классов с использованием абстрактного виртуального класса и интерфейсами
5. Реализовать обработку исключительных ситуаций
6. Рассмотреть и описать, какие проблемы возникли при реализации множественного наследования ("алмаз смерти").

Вариант 16

Устройство, принтер, сканер, МФУ

**1. Постановка задачи**

Передо мной стоит задача: создание ромбовидной иерархии классов из устройства, принтера, сканера и МФУ, добавить чистый абстрактный класс во для изменения данной иерархии и создание иерархии интерфейсов.

Разработать несколько методов для понимания структуры работы множественного наследования. Добавить пользовательский интерфейс с использованием Windows Forms.

Пусть наши классы объединяются следующим интерфейсом:

1. Общие состояния:
   * объект включен в сеть/выключен из сети
   * объект разобран/собран
2. Общие методы:
   * включить/выключить устройство из сети
   * разобрать устройство
   * собрать устройство
   * отнести на починку в мастерскую
3. Общие поля:
   * цена устройства
   * производитель устройства

Устройство не будет обладать какими-то уникальными методами или полями, в отличие от принтера и сканера.

Принтер, в свою очередь, будет обладать способностью печатать и вести учет количества бумаги. Сканер будет сканировать. Да, это важно.

МФУ будет лишь объединять функционал этих двух устройств: печатать, вести учет бумаги и сканировать. Добавим ему еще функцию копирования: объединения сканирования и печати.

1. **Разработка приложения**
2. **Разработка структуры приложения**

На Рис. 2.1. изображена иерархия интерфейсов проекта, где видна реализация ромбовидного наследования.

На Рис. 2.2. изображена иерархия классов проекта: диаграмма абстрактного класса Device\_interface, класса «устройств» Device, класса «принтер» Printer, класса «сканер» Scanner, класса «МФУ» MFP, а также используемые делегаты для обработки событий класса.

На Рис. 2.3. изображены вспомогательные классы, такие как обработчик событий, FileReader, класс аргументов для передачи событий.

В Табл. 2.1 описаны вспомогательные функции.

В Табл. 2.2 описан интерфейс абстрактного класса Device\_interface: его методы и их параметры, а также выполняемые ими функции.

В Табл. 2.3 описаны дополнительные методы класса Print и их параметры, а также выполняемые ими функции.

В Табл. 2.4 описаны дополнительные методы класса Scanner и их параметры, а также выполняемые ими функции.

В Табл. 2.5 описаны дополнительные методы класса MFP и их параметры, а также выполняемые ими функции.

В Табл. 2.6 описаны дружественные функции, доступные для всех классов, и их параметры.

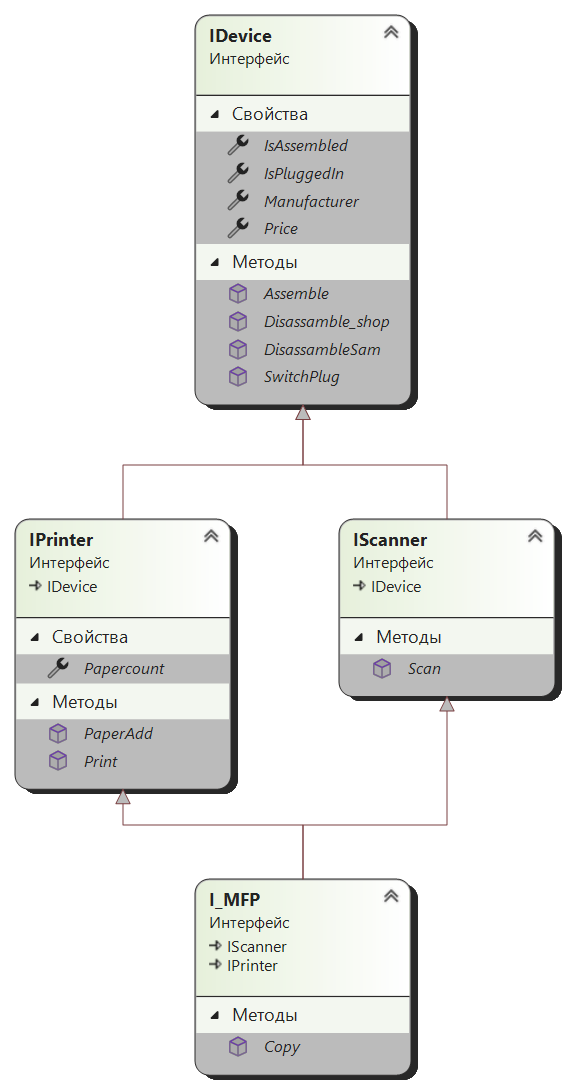


Рис 2.1. Диаграмма интерфейсов

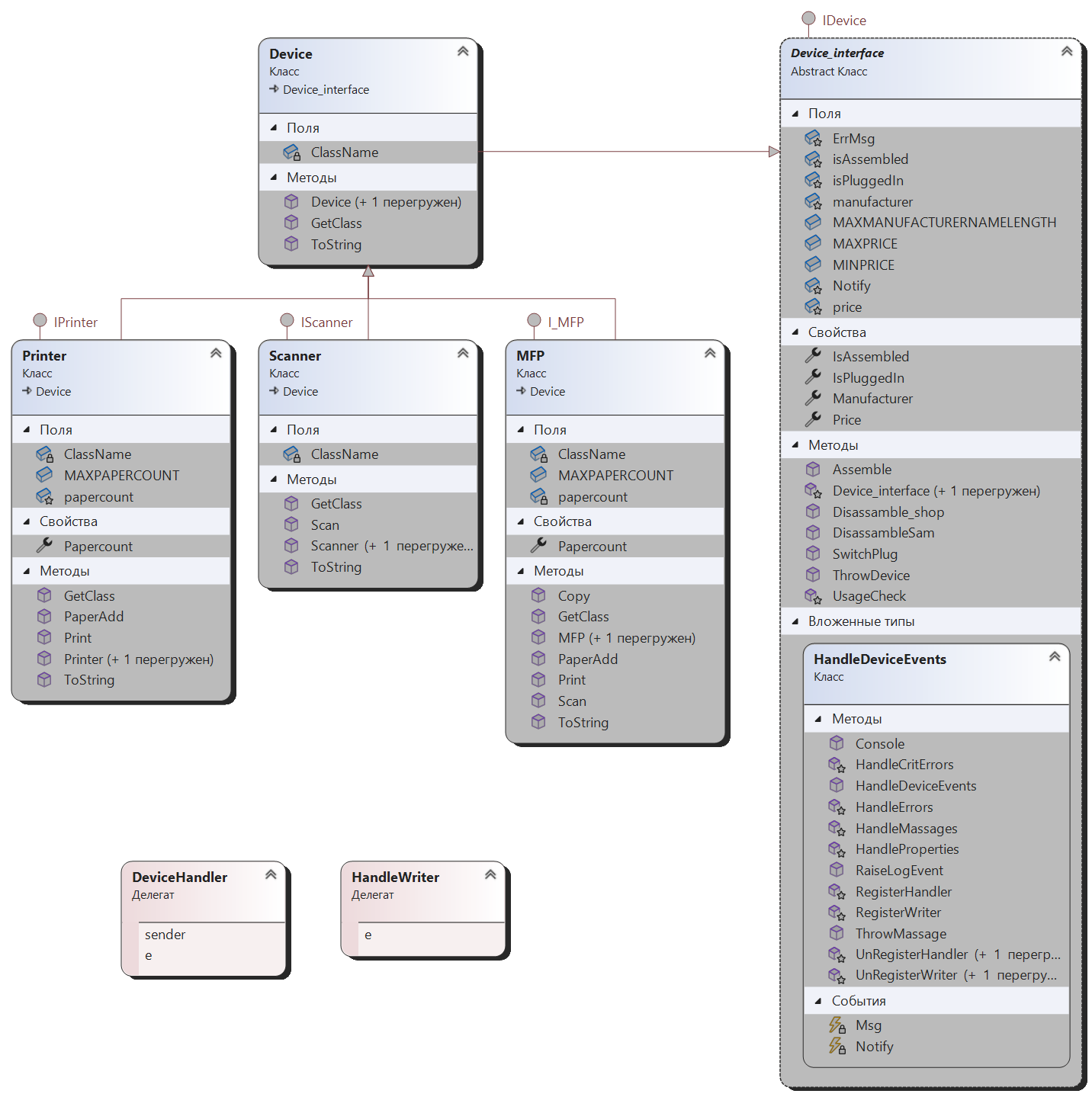


Рис 2.2. Диаграмма классов

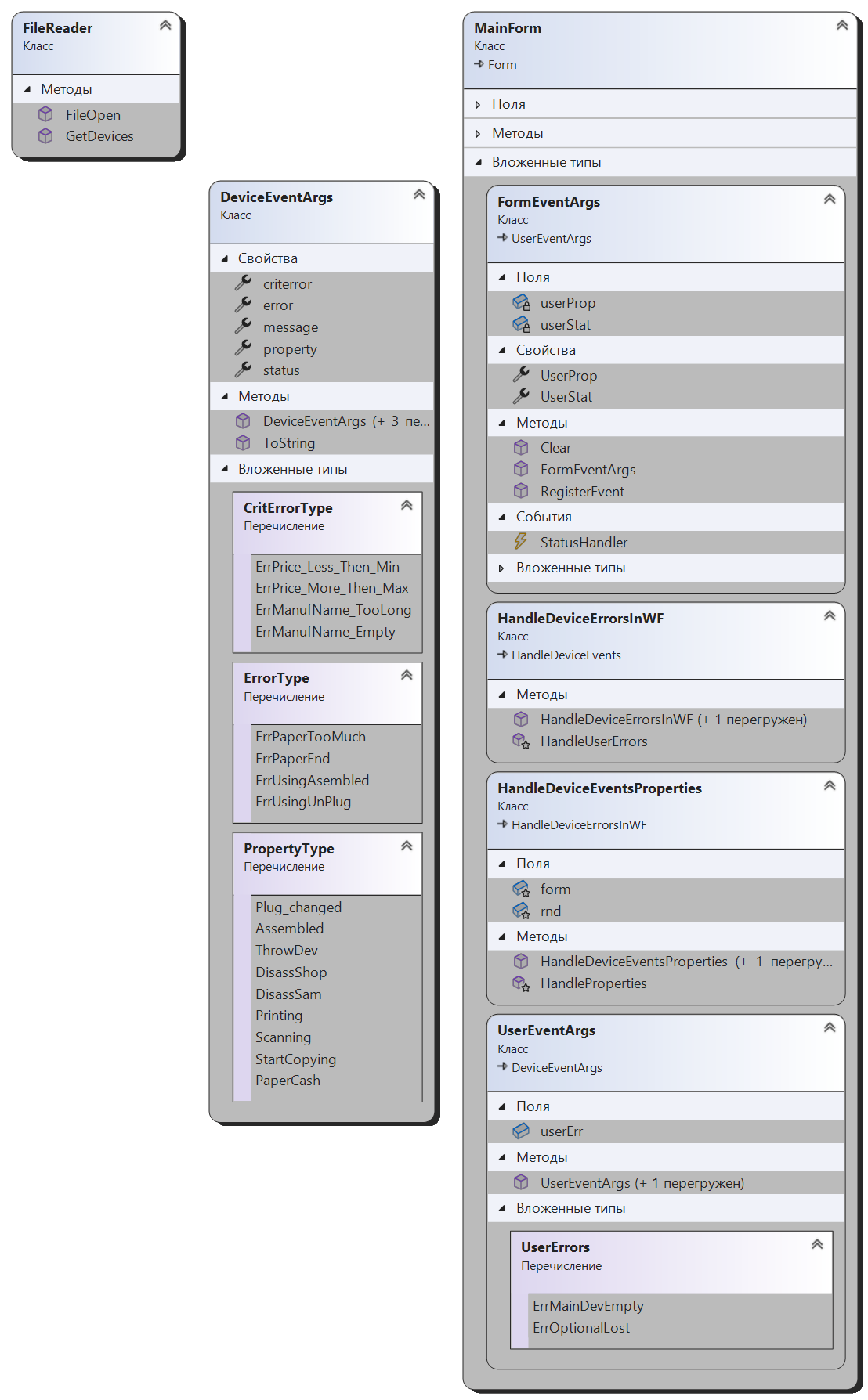


Рис 2.1. Диаграмма вспомогательных классов

Табл. 2.1. интерфейсы

| **Название элемента** | **Назначение, тип элемента** | **Входные данные** | **Выходные данные** |
| --- | --- | --- | --- |
| IDevice | | | |
| Price | int, свойство  цена устройства | - | int |
| Manufacturer | string, свойство  название производителя | - | string |
| IsPluggedIn | bool, свойство  состояние устройства включения в сеть | - | bool |
| IsAssembled | bool, свойство  состояние разобранности устройства | - | bool |
| SwitchPlug | void, функция  изменение сосояния включения в сеть | - | - |
| Assemble | void, функция  разборка усройства | - | - |
| DisassebleSam | void, функция  самостоятельная сборка устройства | - | - |
| Disassemble\_shop | void, функция  отнести устройство в мастерскую | - | - |
| IPrinter: IDevice | | | |
| Papercount | int, свойство  количество бумаги | - | int |
| Print | void, функция  печать | int paperused = 1  количество используемой для печати бумаги | - |
| PaperAdd | void, функция  добивить бумагу | int paper\_add = 1  количество бумаги для добавления | - |
| IScanner: IDevice | | | |
| Scan | void, функция  сканировать | - | - |
| I\_MFP: IPinter, IScanner | | | |
| Copy | void, функция  копировать | int paperused = 1  количество используемой бумаги | - |

Табл. 2.2. методы класса Device\_interface

| **Название элемента** | **Назначение, тип элемента** | **Входные данные** | **Выходные данные** |
| --- | --- | --- | --- |
| Device\_interface | конструктор | int price: цена устройства  string manufacturer: производитль устройства | - |
| Device\_interface | конструктор | int price: цена устройства  string manufacturer: производитель устройства  HandleDeviceEvents notify: обработчик событий объекта устройства  HandleDeviceEvents errMsg = null: обработчик ошибок объекта устройства | - |
| SwitchPlug | void  изменение сосояния включения в сеть | - | - |
| Assemble | void  разборка усройства | - | - |
| DisassebleSam | void  самостоятельная сборка устройства | - | - |
| Disassemble\_shop | void  отнести устройство в мастерскую | - | - |
| ThrowDevice | void  выбросить устройство | - | - |
| UsageCheck | protected void  проверка возможности воспользоваться устройством | - | bool, можно ли пользоваться устройством |

Табл. 2.3. методы класса Device

| **Название элемента** | **Назначение, тип элемента** | **Входные данные** | **Выходные данные** |
| --- | --- | --- | --- |
| Device | конструктор | int price: цена устройства  string manufacturer: производитль устройства | - |
| Device | конструктор | int price: цена устройства  string manufacturer: производитель устройства  HandleDeviceEvents notify: обработчик событий объекта устройства  HandleDeviceEvents errMsg = null: обработчик ошибок объекта устройства | - |
| ToString | override string  название класса | - | string  название класса |
| GetClass | static string  называние класса | - | string  название класса |

Табл. 2.3. методы класса Printer

| **Название элемента** | **Назначение, тип элемента** | **Входные данные** | **Выходные данные** |
| --- | --- | --- | --- |
| Printer | конструктор | int price: цена устройства  string manufacturer: производитль устройства | - |
| Printer | конструктор | int price: цена устройства  string manufacturer: производитель устройства  HandleDeviceEvents notify: обработчик событий объекта устройства  HandleDeviceEvents errMsg = null: обработчик ошибок объекта устройства | - |
| Print | void, функция  печать | int paperused = 1  количество используемой для печати бумаги | - |
| PaperAdd | void, функция  добивить бумагу | int paper\_add = 1  количество бумаги для добавления | - |
| ToString | override string  название класса | - | string  название класса |
| GetClass | static string  называние класса | - | string  название класса |

Табл. 2.3. методы класса Scanner

| **Название элемента** | **Назначение, тип элемента** | **Входные данные** | **Выходные данные** |
| --- | --- | --- | --- |
| Scanner | конструктор | int price: цена устройства  string manufacturer: производитль устройства | - |
| Scanner | конструктор | int price: цена устройства  string manufacturer: производитель устройства  HandleDeviceEvents notify: обработчик событий объекта устройства  HandleDeviceEvents errMsg = null: обработчик ошибок объекта устройства | - |
| Scan | void, функция  сканировать | - | - |
| ToString | override string  название класса | - | string  название класса |
| GetClass | static string  называние класса | - | string  название класса |

Табл. 2.3. методы класса MFP

| **Название элемента** | **Назначение, тип элемента** | **Входные данные** | **Выходные данные** |
| --- | --- | --- | --- |
| MFP | конструктор | int price: цена устройства  string manufacturer: производитль устройства | - |
| MFP | конструктор | int price: цена устройства  string manufacturer: производитель устройства  HandleDeviceEvents notify: обработчик событий объекта устройства  HandleDeviceEvents errMsg = null: обработчик ошибок объекта устройства | - |
| Print | void, функция  печать | int paperused = 1  количество используемой для печати бумаги | - |
| PaperAdd | void, функция  добивить бумагу | int paper\_add = 1  количество бумаги для добавления | - |
| Scan | void, функция  сканировать | - |  |
| Copy | void, функция  копировать | - |  |
| ToString | override string  название класса | - | string  название класса |
| GetClass | static string  называние класса | - | string  название класса |

1. **Разработка схемы алгоритма**

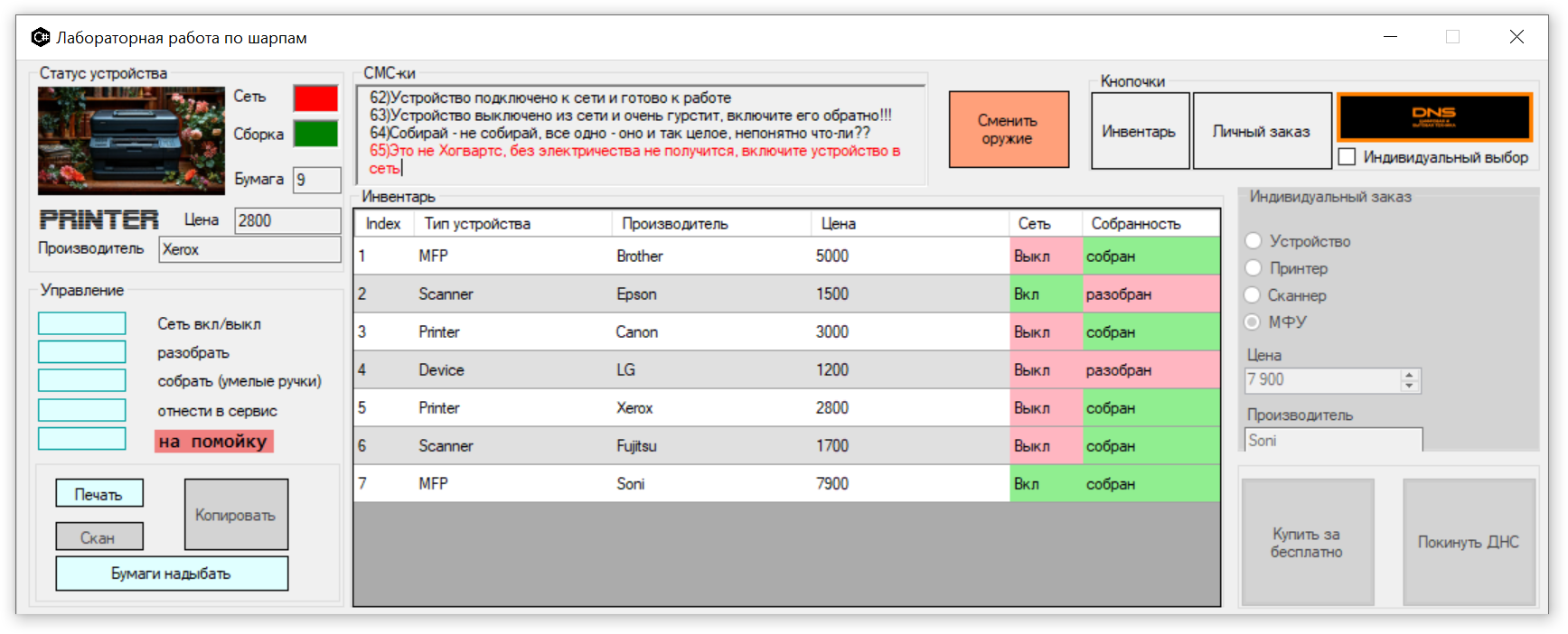
Программа представляет из себя иерархию классов с абстрактным классом в вершине, регулируемая привязанным к классам интерфейсам. Необходима разработка иерархии в соответствии с основными правилами ООП, и привязка выполнения методов к пользовательскому интерфейсу.

Взаимодействие с объектами класса происходит в модели «магазина товаров» с возможностью преобретения и дальнейшего использования:

1. В магазине пользователю предлагается на выбор некоторое количество устройств, из которых он выбирает себе подходящие. Для товаров существуют ограничения: по цене (товаря не могут быть дешевле 1000 рублей и дороже 10000р, а также их наименование не может быть длиннее 100 символов, иначе у покупателя голова закружится. Выйдя из магазина, пользователь может просмотреть свой инвентарь и техническое состояние всех устройств.
2. Из «купленных» товаров пользователь выбирает «текущее устройство», с которым может производить различные манипуляции:
   1. просмотр текущих параметров устройства (например включен он в розетку или нет).
3. Общие для всех устройств действия:
   1. включение/выключение устройства.
   2. разборка устройства.
   3. сборка устройства: здесь вводится элемент случайности: с разной вероятностью пользователь либо может, либо не может собрать устройство самостоятельно.
   4. починка устройства в сервисе.
   5. избавление от девайса: можно удалить устройство своего инвентаря.
4. Другие узкоспециализированные действия, при условии того, что устройство подключено к сети и находится в собранном состоянии:
   1. печать для принтера и МФУ: пользователь должен быть внимателен, потому что при каждой печати в устройстве используется бумага, и если воспользоваться печатью с пустым лотком для бумаги, прибор может сломаться.
   2. сканирование для сканера и МФУ.
   3. сканирование и копирование для МФУ.
5. **Разработка пользовательского интерфейса**

Взаимодействие программы с пользователем происходит в окне программы. Да, мы апнулись настолько, что перестали работать в консоли и узнали о существовании gui.

Общее меню устройства:



Кнопочки:

1. ДНС – вывод данных файла в таблицу для предоставления пользователю выбор для покупки. При включенной галочке индивидуального выбора появляется возможность выбрать файл для загрузки данных.
2. Личный заказ позволяет пользователю сотворить из ничего новое устройство с вымышленным названием и ценой. Просто немыслимо.
3. Покупка осуществляется двойным кликом по таблице или нажатием кнопки Купить за бесплатно.
4. Чтобы выйти в меню инвентаря пользователя требуется нажать кнопку Инвентарь, будучи в магазине можно воспользоватсься кнопкой покинуть ДНС.
5. Слева расположены кнопочки для манипуляций выбраным объектом мучений.
6. **Реализация и тестирование приложения**
7. **Описание разработанной программы**

Программа реализована с использованием языка программирования С#. Код программы содержится в следующих файлах:

* Interfaces.cs, Device\_interface.cs, Sourse.cs – файлы хранения данных класса и его вспомогательных методов и классов.
* MainForm.cs, MainForm\_E.cs, MainForm\_F.cs – файлы для реализации интерфейса класса и обработки событий в пространстве WindowsForms

1. **Тестирование программы**

Табл. 3.1 Функциональные тесты для приложения 1

| **№ теста** | **Элементы формы** | | **Входные данные** | **Ожидаемый результат** | **Смысл теста** |
| --- | --- | --- | --- | --- | --- |
| Пользовательский интерфейс (ввод данных с клавиатуры) | | | | | |
| 1 | Текстовое поле производитель | | ввод пустой строки | сообщение об ошибке с просьбой исправить данные |  |
| ввод слишком длинной строки (больше 15 символов) | сообщение об ошибке |  |
| 2 | Поле цена | | ввод букв | ничего, поле не должно принимать ничего кроме целых чисел |  |
|  |  | | ввод числа меньше 1000 | автоисправление поля на 1000 |  |
|  |  | | ввод числа больше 10000 | автоисправление поля на 10000 |  |
|  | | (Не) Соответствие вводимых данных требуемому типу | | | |
| 3 |  | | 9999 | сообщение об ошибке,  просьба повторить попытку | Неверный тип данных (char) |

|  |  |  |  |
| --- | --- | --- | --- |
| 4 | 9999222222222222222 | сообщение об ошибке,  просьба повторить попытку | Неверный тип данных (int) |
| Работа главного меню | | | |
| 5 | Меню: 1  Внутренние данные: пусто  Входные данные: файл data.txt | 11. Таких мелких ценников [11, 00 Asus] на устройства не бывает. Мы на такое не поведемся. Минимум: 1000  Как отсортирвать устройства: (1) по цене или (2) по производителю, или (3) по типу? 1  --- Доступные устройства ---  1. Device 1100 от известнейшего бренда Asus  2. Device 1200 от известнейшего бренда LG  3. Scanner 1500 от известнейшего бренда Epson | Пользователь попадает в магазин товаров. Происходит сканирование входного файла с отсеиванием неподходящих товаров и выводится список доступных к покупке устройств, отсортированных по желанию пользователя либо по цене, либо по производителю, либо по умолчанию. |
| 6 | Меню: 2  Внутренние данные: устройства  Входные данные: - | --- Мои устройства ---  1. Device Asus за кровные 1100  2. Scanner Epson за кровные 1500 | Вывод инвентаря пользователя |
| 7 | Меню: 3  Внутренние данные: устройства  Входные данные: - | --- Выберите устройство ---  1. Device Asus за кровные 1100  2. Scanner Epson за кровные 1500  3. MFP Panasonic за кровные 4900  Введите номер устройства: 3  Выбор совершен, назад пути нет. | Выбор текущего устройства |
| 8 | Меню: 4  Внутренние данные: устройства, МФУ  Входные данные: - | --- Статус текущего устройства ---  МФУ Status:  Цена: 4900, Производитель: Panasonic  Розетка: Не подключено,  Сборка: Аппарат цел и невредим  Чистой бумаги осталось: 5 | Текущее состояние выбранного устройства |
| 9 | Меню: 6  Внутренние данные: устройства  Входные данные:- | --- Мои устройства ---  Устройство Status:  Цена: 1100, Производитель: Asus  Розетка: Не подключено,  Сборка: Аппарат цел и невредим  ===============================  Сканнер Status:  Цена: 1500, Производитель: Epson  Розетка: Не подключено,  Сборка: Аппарат цел и невредим  ===============================  МФУ Status:  Цена: 4900, Производитель: Panasonic  Розетка: Не подключено,  Сборка: Аппарат цел и невредим  Чистой бумаги осталось: 5  =============================== | Вывод состояния всех устройств в инвентаре |
| 10 | Меню: 5  Внутренние данные: устройства, МФУ  Входные данные | --- Наше текущее устройство ---  МФУ Status:  Цена: 4900, Производитель: Panasonic  Розетка: Не подключено,  Сборка: Аппарат цел и невредим  Чистой бумаги осталось: 5  //////////////////////////////////////////////  // 0 - инструкция //  // 1 - розетка: вкл/выкл //  // 2 - починить (разобрать) //  // 3 - дочинить (собрать) //  // 4 - статус девайса //  // 5 - отнести в ремонт //  // q - избавиться от девайса //  ///////////////////////////////////////////////  // 6 - печатать //  // 7 - сканировать //  // 8 - копировать //  // 9 - назад //  ////////////////////////////////// | Правильность вывода собственного меню для устройства |
| Работа внутреннего меню устройства | | | |
| 11 | Меню: 1  Внутренние данные: МФУ  Входные данные: - | МФУ включен в сеть  БЖ - Ж - Ж  1. (прогревание картриджей)...  2. (просто непонятное жужжание)...  МФУ готов к работе | Проверка правильности работы меню |
| 12 | Меню: 2  Внутренние данные: МФУ  Входные данные: - | МФУ теперь разобрано. Все винтики в коробочку сложили или как обычно?. |
| 13 | Меню: 3  Внутренние данные: МФУ  Входные данные: - | Вот ведь неудача, вы все доломали. Устройство было многофункционально, а вы нет. Теперь все равны. Может в ремонт? |
| 14 | Меню: 4  Внутренние данные: МФУ  Входные данные: - | МФУ Status:  Цена: 4900, Производитель: Panasonic  Розетка: Подключено,  Сборка: Разобран. Собирать когда будем?  Чистой бумаги осталось: 5 |
| 15 | Меню: 5  Внутренние данные: МФУ  Входные данные: - | Все готово! С вас сто мильонов за починку. Мы заменили еще и бумагу вам добавили и материнку поменяли, DOTA3 скачали, и вообще. Да. |
| 16 | Меню: 6  Внутренние данные: МФУ  Входные данные: - | Печать в процессе  ... Печать завершена. Бумаги осталось: 104 |
| 17 | Меню: 7  Внутренние данные: МФУ  Входные данные: - | Сканирование в процессе  ... Сканирование завершено |
| 18 | Меню: 8  Внутренние данные: МФУ  Входные данные: - | Сканирование в процессе  ... Сканирование завершено  Печать в процессе  ... Печать завершена. Бумаги осталось: 103 |
| 19 | Меню: q  Внутренние данные: МФУ  Входные данные: - | Как будем избавляться? Мб на авито его? (1-4)  Ну чтож, на авито кому-нибудь втюхаем за двойную цену |

**Вывод:** самое сложное в написании программ – это оформление отчета и специализации.

Выявленные сложности:

1. Изобретение функционала для разработки приложения
2. Реализация динамического связывания: необходимость выполнения однозначной идентификации экземпляров каждого класса без использования типов.
3. Понимание общей реализации принципов ООП
4. Реализация множественного наследования ромбовидной формы

**Приложение**

Задание 1

Листинг программы main.cpp