МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федерального государственного бюджетного образовательного учреждения высшего образования

«Московский государственный технический университет имени Н.Э.Баумана» (национальный исследовательский университет)

Московский техникум космического приборостроения

СПЕЦИАЛЬНОСТЬ 09.02.03 Программирование в компьютерных системах

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

по учебной практике по профессиональному модулю МДК 03.01

«Технология разработки программного обеспечения»

Группа ТМП-61 Листов 100

|  |  |  |
| --- | --- | --- |
| Студент | (подпись) | В. А. Крапивкин  (Ф.И.О.) |

Москва 2022

Крапивкин Вячеслав ТМП-61

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование | Оценка | Дата | Подпись |
| Программа |  |  |  |
| Листинг |  |  |  |
| Оформление пояснительной  записки |  |  |  |
| ИТОГ |  |  |  |

# СОДЕРЖАНИЕ

1. [Постановка задачи 4](#_TOC_250017)
2. [Логическое проектирование реляционной базы данных 5](#_TOC_250016)
   1. [Построение логической схемы базы данных 5](#_TOC_250015)
   2. [Описание таблиц логической схемы 6](#_TOC_250014)
3. [Структура программы 10](#_TOC_250013)
4. [Отладка программы 13](#_TOC_250012)
5. [Оптимизация программы 15](#_TOC_250011)
6. [Тестирование программы 16](#_TOC_250010)
   1. [Тестирование в нормальных условиях 16](#_TOC_250009)
   2. [Тестирование в экстремальных условиях 24](#_TOC_250008)
   3. [Тестирование в исключительных ситуациях 27](#_TOC_250007)
7. [Модульное тестирование 32](#_TOC_250006)
8. [Руководство пользователя 33](#_TOC_250005)

[Заключение 42](#_TOC_250004)

[Список использованных источников 43](#_TOC_250003)

Приложение А. Листинги программы 44

[А.1 Листинг файла main.cpp 45](#_TOC_250002)

А.2 Листинг файла mainwindow.h 47

А.3 Листинг файла mainwindow.cpp 49

А.4 Листинг файла db\_tur\_ag.h 52

А.5 Листинг файла db\_tur\_ag.cpp 55

А.6 Листинг файла add.h 66

А.7 Листинг файла add.cpp 68

А.8 Листинг файла udal.h 75

А.9 Листинг файла udal.cpp 77

А.10 Листинг файла upd.h 80

А.11 Листинг файла upd.cpp 83

[А.12 Листинг файла Log\_pas.h 90](#_TOC_250001)

[А.13 Листинг файла test.h 91](#_TOC_250000)

Приложение Б. Результаты выполнения программы 92

# Постановка задачи

Разработать информационную систему для интернет-продаж магазинов бытовой техники. Она должна хранить и обрабатывать информацию о магазинах, продаваемых товарах, заказах и доставках.

Необходимо провести анализ предметной области и составить логическую структуру базы данных информационной системы.

Реализовать авторизацию и регистрацию пользователей при начале использования программы, а также проверку логина и пароля посредством регулярных выражений.

Разработанная система должна отвечать следующим требованиям:

1. должна иметь возможность просмотра, добавления, удаления и обновления записей таблиц базы данных;
2. должна иметь возможность сортировки по полям таблиц с помощью выпадающего списка;
3. должна иметь реализацию автоматической загрузки и выгрузки данных из текстового файла;
4. должна иметь реализацию вывода на печать таблиц базы данных;
5. должна иметь форму для авторизации и регистрации.

Для разработанной информационной системы необходимо разработать модульные тесты.

# Логическое проектирование реляционной базы данных

### Построение логической схемы базы данных

На этапе логического проектирования разрабатывается логическая схема, определяются сущности (таблицы), атрибуты (столбцы таблицы), а также указывается вид связей между таблицами. Логическая модель данных является универсальной и никак не связана с конкретной реализацией системы управления баз.

Логическая структура предметной области «Магазины бытовой техники» представлена на рисунке 2.1.



Рисунок 2.1 – Логическая структура базы данных

### Описание таблиц логической схемы

В таблицах 2.1 – 2.5 представлено описание таблиц логической структуры спроектированной информационной системы.

Таблица 2.1 – Описание таблицы «Магазины»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Описание | Тип данного | Размер |
| ID\_magaz | Первичный ключ.  Идентификатор магазина. | Численный | Целый тип |
| Email | Атрибут, содержащий Email магазина. | Строковый | 25 |
| opl\_dost | Атрибут, содержащий информацию об оплате доставки. | Численный | 1 |

Таблица 2.2 – Описание таблицы «Товары»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Описание | Тип данного | Размер |
| ID\_tovar | Первичный ключ.  Идентификатор товара. | Численный | Целый тип |
| nazvanie | Атрибут, содержащий название товара. | Строковый | 20 |
| firma | Атрибут, содержащий фирму товара. | Строковый | 10 |
| model | Атрибут, содержащий модель товара. | Строковый | 25 |
| tx | Атрибут, содержащий технические характеристики товара. | Строковый | 100 |
| cost | Атрибут, содержащий стоимость товара. | Численный | Целый тип |

Продолжение таблицы 2.2

|  |  |  |  |
| --- | --- | --- | --- |
| gar\_srok | Атрибут, содержащий гарантийный срок товара. | Строковый | 11 |

Таблица 2.3 – Описание таблицы «Заказы»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Описание | Тип данного | Размер |
| ID\_zakaz | Первичный ключ.  Идентификатор заказа. | Численный | Целый тип |
| ID\_magaz | Внешний ключ.  Идентификатор магазина. | Численный | Целый тип |
| ID\_tovar | Внешний ключ.  Идентификатор товара. | Численный | Целый тип |
| data\_zakaza | Атрибут, содержащий дату заказа. | Дата | Краткий формат |
| FIO\_klienta | Атрибут, содержащий ФИО клиента. | Строковый | 70 |
| tel | Атрибут, содержащий телефон клиента. | Строковый | 11 |
| podtver\_zak | Атрибут, содержащий информацию об оплате заказа. | Численный | 1 |
| time\_zak | Атрибут, содержащий время заказа. | Время | Краткий формат |

Таблица 2.4 – Описание таблицы «Доставки»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Описание | Тип данного | Размер |
| ID\_dostavki | Первичный ключ.  Идентификатор доставки. | Численный | Целый тип |
| ID\_zakaz | Внешний ключ.  Идентификатор магазина. | Численный | Целый тип |
| data\_dostavki | Атрибут, содержащий дату доставки. | Дата | Краткий формат |
| time\_dostavki | Атрибут, содержащий время доставки. | Время | Краткий формат |
| adress | Атрибут, содержащий адрес доставки. | Строковый | 100 |
| FIO\_kur | Атрибут, содержащий ФИО курьера. | Строковый | 70 |

Таблица 2.5 – Описание таблицы «Пользователи»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Описание | Тип данного | Размер |
| password | Атрибут, содержащий пароль пользователя. | Строковый | 32 |
| login | Атрибут, содержащий логин пользователя. | Строковый | 20 |
| fio | Атрибут, содержащий ФИО пользователя. | Строковый | 70 |

# Структура программы

На рисунке 3.1 показана структура программы.



Рисунок 3.1 – Структура программы

Формы, используемые в основной программе, описаны в таблице 3.1.

Таблица 3.1 – Формы, используемые в программе

|  |  |
| --- | --- |
| Форма | Назначение |
| mainwindow | Форма авторизации пользователей |
| base\_view | Форма с рабочим пространством для работы с таблицами |
| del | Форма для добавления записей в таблицу |
| dobav | Форма для удаления записей из таблицы |
| obnova | Форма для изменения записей в таблице |
| regist | Форма регистрации пользователей |

Функции, используемые в форме mainwindow, описаны в таблице 3.2.

Таблица 3.2 – Функции, используемые в форме mainwindow

|  |  |
| --- | --- |
| Функция | Назначение |
| on\_pushButton\_regist\_clicked | Переход на форму для регистрации пользователя. |
| on\_pushButton\_entry\_clicked | Авторизация пользователя и переход на форму для работы с таблицами |

Функции, используемые в форме base\_view описаны в таблице 3.3.

Таблица 3.3 – Функции, используемые в форме base\_view

|  |  |
| --- | --- |
| Функция | Назначение |
| viewTab | Вывод записей таблиц базы данных на рабочее  пространство формы |
| on\_comboBox\_activated | Заполнение выпадающего списка названиями столбцов выбранной таблицы |
| sortTab | Сортировка таблиц по выбранному столбцу |
| on\_sortButton\_clicked | Активация функции сортировки таблицы |
| on\_dobavButton\_clicked | Переход на форму добавления записей |
| on\_udalButton\_clicked | Переход на форму удаления записей |
| on\_izmenButton\_clicked | Переход на форму изменения записей |

Продолжение таблицы 3.3

|  |  |
| --- | --- |
| on\_vigruzka\_clicked | Запись в файл по нажатию кнопки |
| on\_vivod\_clicked | Отправка на печать |
| on\_zagruzka\_clicked | Загрузка данных из файла по нажатию кнопки |

Функции, используемые в форме dobav, описаны в таблице 3.4.

Таблица 3.4 – Функции, используемые в форме dobav

|  |  |
| --- | --- |
| Функция | Назначение |
| on\_pushMagzDob\_clicked | Добавление записи в таблицу "Магазины" |
| on\_pushTovarDob\_clicked | Добавление записи в таблицу "Товары" |
| on\_pushZakazDob\_clicked | Добавление записи в таблицу "Заказы" |
| on\_pushDostDob\_clicked | Добавление записи в таблицу "Доставки" |

Функции, используемые в форме del, описаны в таблице 3.5.

Таблица 3.5 – Функции, используемые в форме del

|  |  |
| --- | --- |
| Функция | Назначение |
| id\_Table | Заполнение выпадающего списка с ID таблиц |
| DelTab | Удаление записи |
| on\_deleteButton\_clicked | Активация функции удаления |

Функции, используемые в форме obnova, описаны в таблице 3.6.

Таблица 3.6 – Функции, используемые в форме obnova

|  |  |
| --- | --- |
| Функция | Назначение |
| on\_pushObMagaz\_clicked | Изменение записи в таблице "Магазины" |
| on\_pushObTovar\_clicked | Изменение записи в таблице "Товары" |
| on\_pushObZakaz\_clicked | Изменение записи в таблице "Заказы" |
| on\_pushObDost\_clicked | Изменение записи в таблице "Доставки" |

Функции, используемые в форме obnova, описаны в таблице 3.7.

Таблица 3.7 – Функции, используемые в форме regist

|  |  |
| --- | --- |
| Функция | Назначение |
| on\_zaregist\_clicked | Регистрация пользователя по нажатию кнопки |

**4 Отладка программы**

Отладка – это процесс локализации и исправления ошибок. Практически любая программа перед началом отладки содержит ошибки. Отладка занимает значительную часть рабочего времени программиста, нередко большую, чем написание программы.

Локализацией называют процесс определения оператора программы, выполнение которого вызвало нарушение нормального вычислительного процесса.

Виды ошибок, которые можно обнаружить при отладке:

1. синтаксические;
2. ошибки компоновки;
3. ошибки выполнения.

Синтаксические ошибки – это ошибки, фиксируемые компилятором при выполнении синтаксического и частично сематического анализа программы.

Ошибки компоновки – это ошибки, обнаруженные компоновщиком (редактором связей), при объединении модулей программы.

Ошибки выполнения – это ошибки, обнаруженные ОС, аппаратными средствами или пользователем при выполнении программы.

Во время отладки данной программы были обнаружены синтаксические ошибки, ошибки компоновки, ошибки выполнения и логические ошибки проектирования.

Ошибка компоновки была допущена из-за отсутствия подключения заголовочных фыйлов dobav.h, del.h, obnova.h. к форме base\_view, для реализации функций добавления, удаления и изменения записей.

Логическая ошибка проектирования заключалась в неверной реализации алгоритма, впоследствии чего программа обращалась к несуществующему элементу запроса, так как выходила за диапазон значений индексов, считывая все записи подряд.

Фрагмент кода с ошибкой:

for(int i=0;i<query.count();i++){

names.push\_back(query.record().fieldName(i)

}

Исправленный фрагмент кода:

for(int i=0;i<query.record().count();i++){

names.push\_back(query.record().fieldName(i)

}

Логическая ошибка проектирования состояла в неверном составлении SQL запросов на выборку с позиционной вставкой. Она была допущена вследствие попытки подстановки названий таблиц в запросы на русском языке.

Для устранения данной ошибки были созданы контейнеры пар значений (ключ, значение) с соответствующими названиями таблиц из базы данных.

Ошибка выполнения была выявлена при пользовании программой и заключалась в выводе повторно выбранной таблицы на рабочее пространство формы без предварительного очищения.

Для устранения данной ошибки в фрагмент кода, реализующий вывод записей таблицы на рабочее пространство, была добавлена строка:

ui -> tableWidget -> clear();

Для отладки программы были задействованы следующие средства локализации ошибок:

1. печать в узлах;
2. слежение.

Локализация была реализована путем вывода сообщения в случае удачного или неудачного выполнения какой-либо функции. В больших функция, на стадии разработки также использовался вывод промежуточных значений, для понимания того, что программа выполняется правильно.

**5 Оптимизация программы**

Оптимизация представляет собой процесс модификации программы для улучшения её эффективности. Необходимость оптимизировать программу возникает потому, что, либо программа выполняется слишком долго, либо для нее требуется слишком большой объём памяти.

Оптимизация бывает двух видов:

1) по времени выполнения программы;

2) по используемой программной памяти.

При разработке программы была реализована оптимизация по времени посредством уменьшения времени компоновки программы. Для этого практически каждая реализованная процедура была реализована в отдельном модуле. Связи межмодульных интерфейсов были осуществлены подключением заголовочных файлов в исполняющие.

Помимо этого, удалось уменьшить затраты времени на выполнение, путем уменьшения количества обращений программы к базе данных, с помощью создания универсальных функций для выполнения SQL-запросов разных реляционных таблиц.

Также, все однотипные действия, выполняемые в программе более одного раза, были перенесены в уникальные слоты.

**6 Тестирование программы**

Тестирование – это набор процедур и действий, предназначенных для демонстрации правильности работы программы в заданных режимах и внешних условиях. Цель тестирования – выявить наличие ошибок или убедительно продемонстрировать их отсутствие.

Процесс тестирования делится на 3 этапа:

1. проверка в нормальных условиях;
2. проверка в экстремальных условиях;
3. проверка в исключительных ситуациях.

Каждый из трех этапов проверки должен гарантировать получение верных результатов при правильных входных данных и выдачу сообщений об ошибках при неправильных данных.

6.1 Тестирование в нормальных условиях

Проверка в нормальных условиях предполагает тестирование на основе дынных, которые характерны для реальных условий функционирования программы. Проверка в нормальных условиях должна показать, что программа выдает правильные результаты для характерных совокупностей данных.

Для проверки работы информационной системы в нормальных условиях были протестированы все формы разработанной программы.

На рисунке 6.1 показан ввод ФИО, логина и пароля при регистрации нового пользователя. Вводимые логин и пароль соответствуют установленному в системе формату.

После чего указанные данные записываются в базу данных информационной системы (рисунок 6.2).

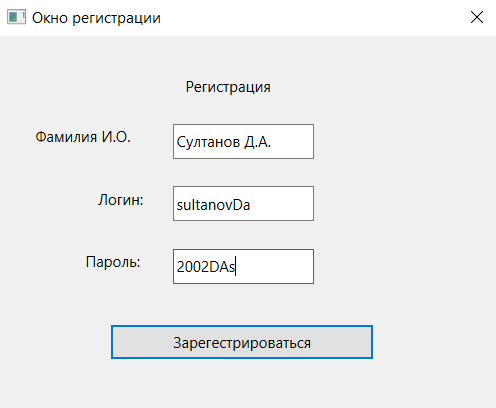


Рисунок 6.1 – Регистрация пользователя

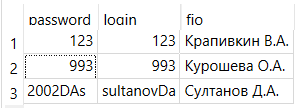


Рисунок 6.2 – Данные в таблице пользователи

На рисунке 6.3 представлен вход только что зарегистрированного пользователя.

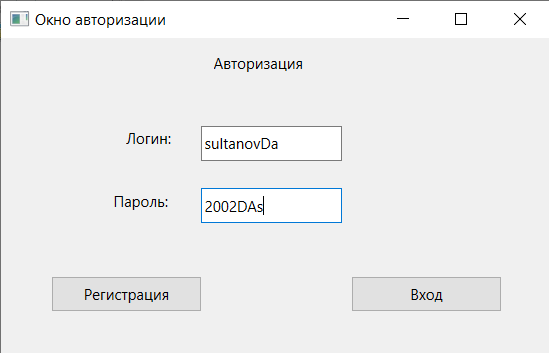


Рисунок 6.3 – Авторизация пользователя

По нажатию кнопки «Вход» открывается рабочее пространство для работы с реляционными таблицами базы данных (рисунок 6.4).

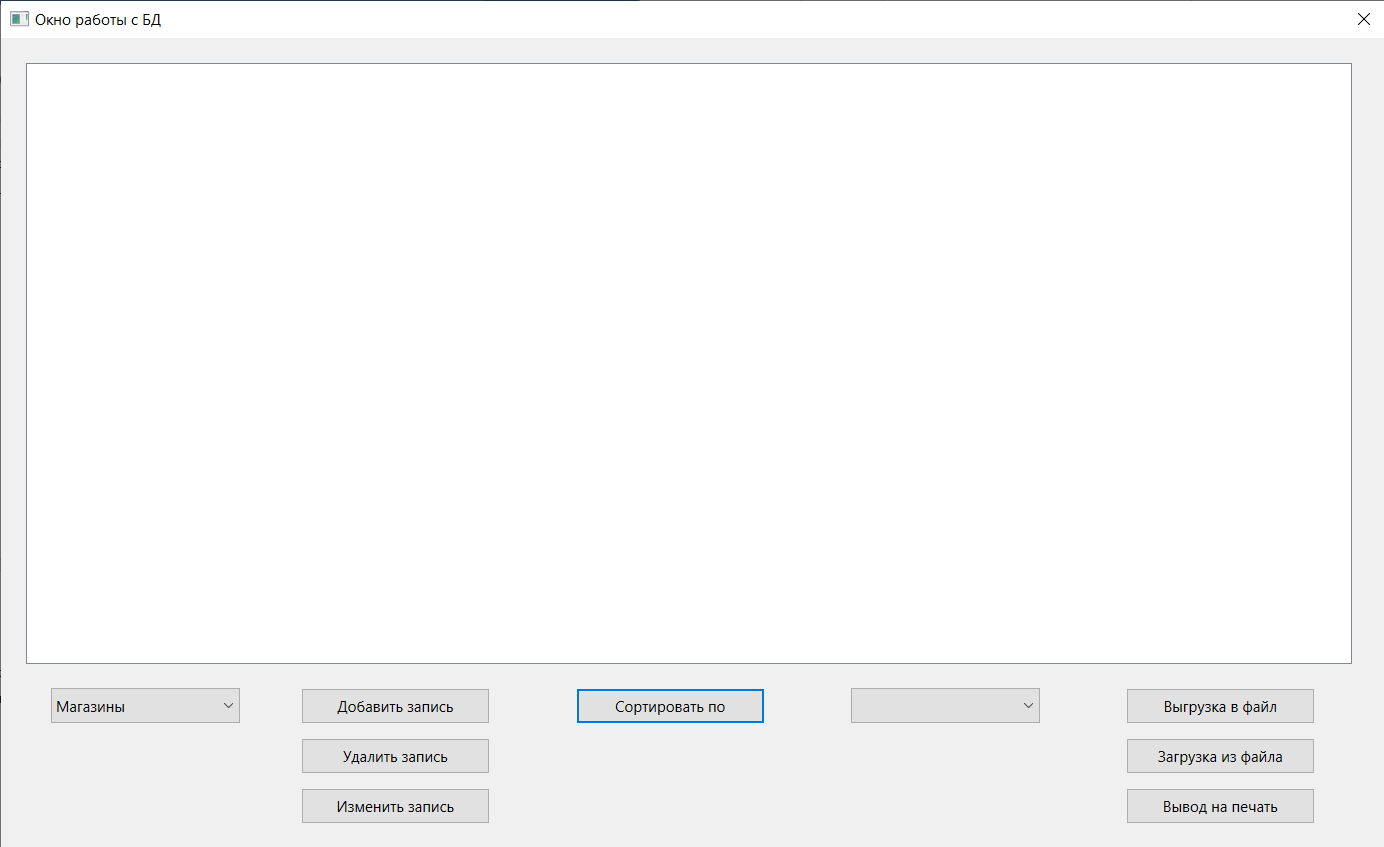


Рисунок 6.4 – Окно работы с таблицами базы данных

На открытой форме для работы с реляционными таблицами в выпадающем списке необходимо выбрать название таблицы для отображения на рабочем пространстве, что демонстрирует рисунок 6.5.

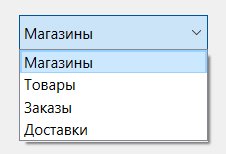


Рисунок 6.5 – Выбор таблицы для отображения

Результаты выбора таблицы из выпадающего списка для отображения показаны на рисунке 6.6.

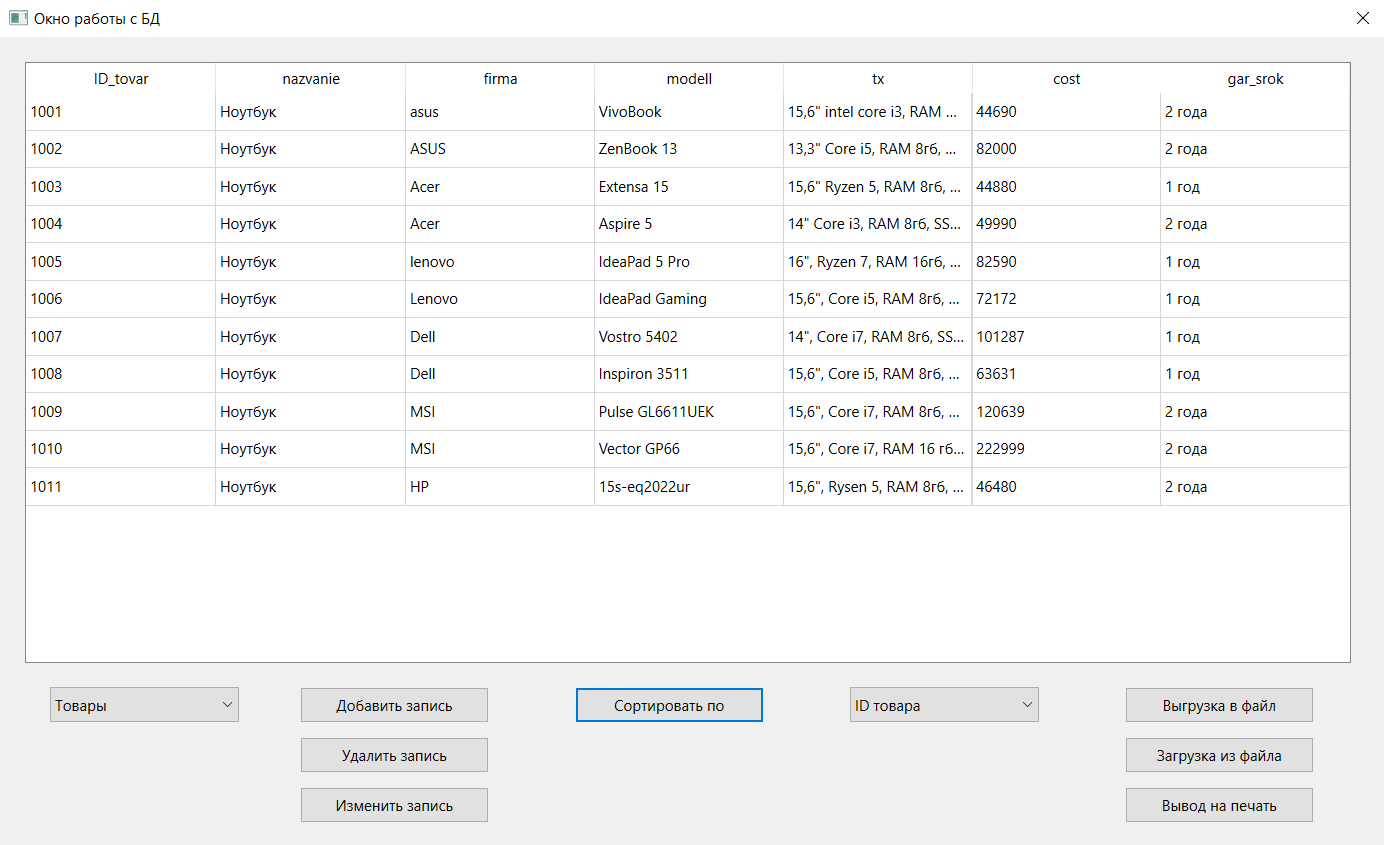


Рисунок 6.6 – Результаты выбора таблицы из выпадающего списка

Следующим этапом тестируется форма для добавления записи в реляционную таблицу базы данных. В форму были введены данные о товаре, не записанном ранее в базу данных. Результаты добавления продемонстрированы на рисунках 6.7 – 6.8.

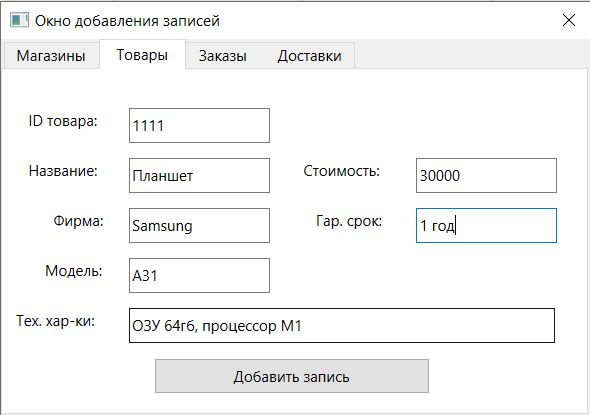


Рисунок 6.7 – Ввод данных для добавления записи

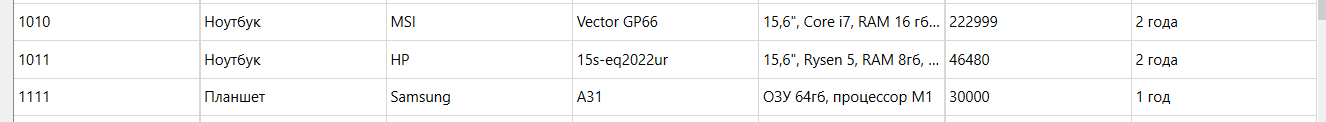


Рисунок 6.8 – Результаты добавления записи

Результаты тестирования для формы изменения записи в таблице продемонстрированы на рисунках 6.9 – 6.10.

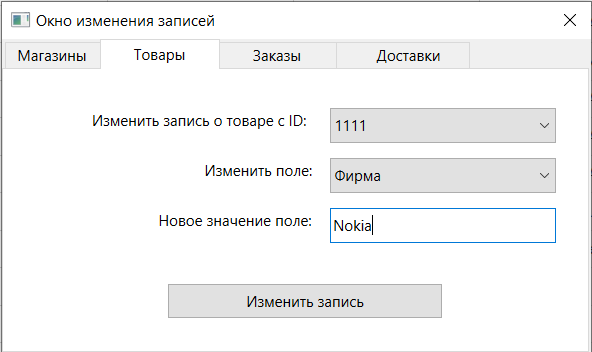


Рисунок 6.9 – Ввод данных для изменения данных

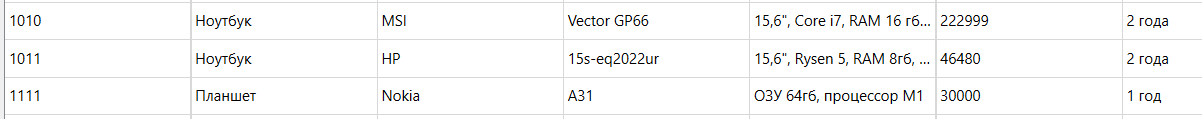


Рисунок 6.10 – Результаты изменения записи

Результаты тестирования для формы удаления записи из таблицы продемонстрированы на рисунках 6.11 – 6.12.

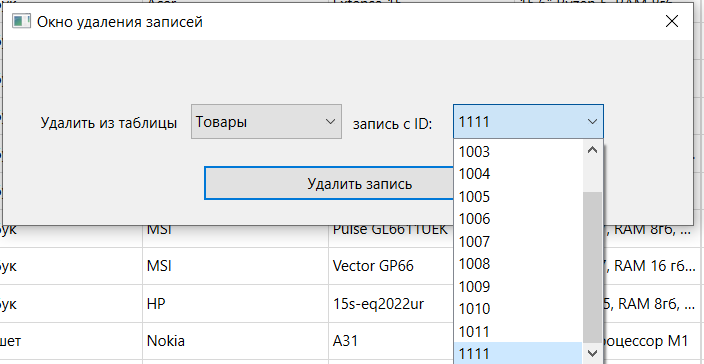


Рисунок 6.11 – Выбор записи для удаления



Рисунок 6.12 – Результаты удаления записи

Тестирование сортировки таблиц демонстрируют рисунки 6.13 -6.14.

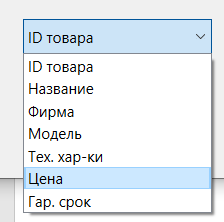


Рисунок 6.13 – Выбор столбца для реализации сортировки

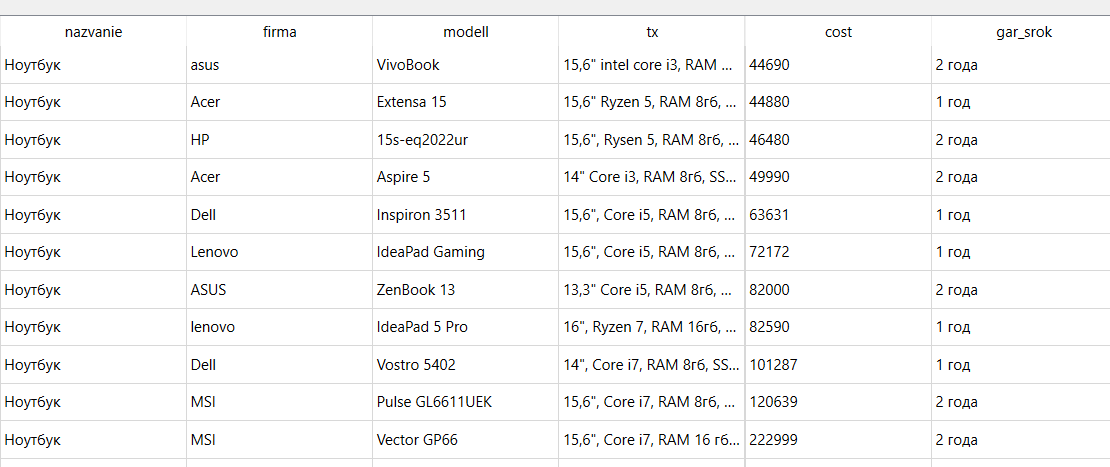


Рисунок 6.14 – Результаты сортировки записей

Результаты тестирование кнопки «Выгрузка в файл» продемонстрированы на рисунках 6.15 – 6.16.

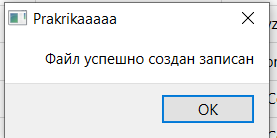


Рисунок 6.15 – Сообщение об успешной записи в файл

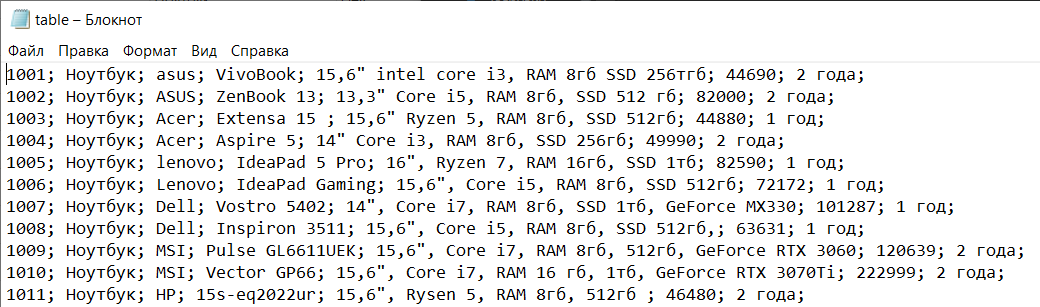


Рисунок 6.16 – Файл с выгруженной таблицей

Результаты тестирования отправки таблицы на печать по нажатию кнопки на форме показали корректную работу программы. Результаты тестирования данной функции показаны на рисунках 6.17 – 6.18.

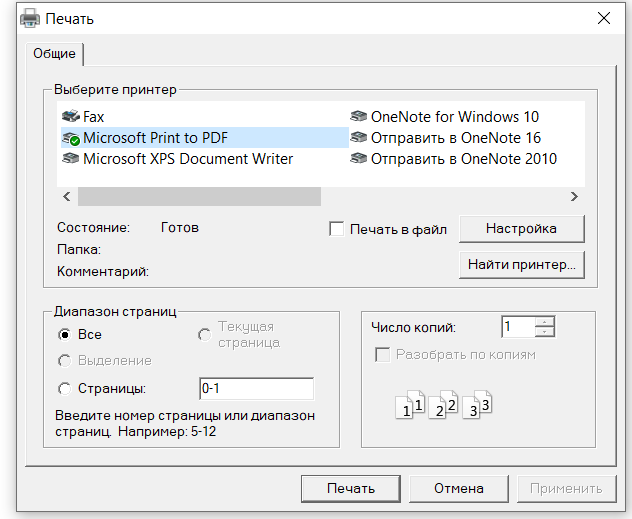


Рисунок 6.17 – Диалоговое окно выбора конфигурации принтера

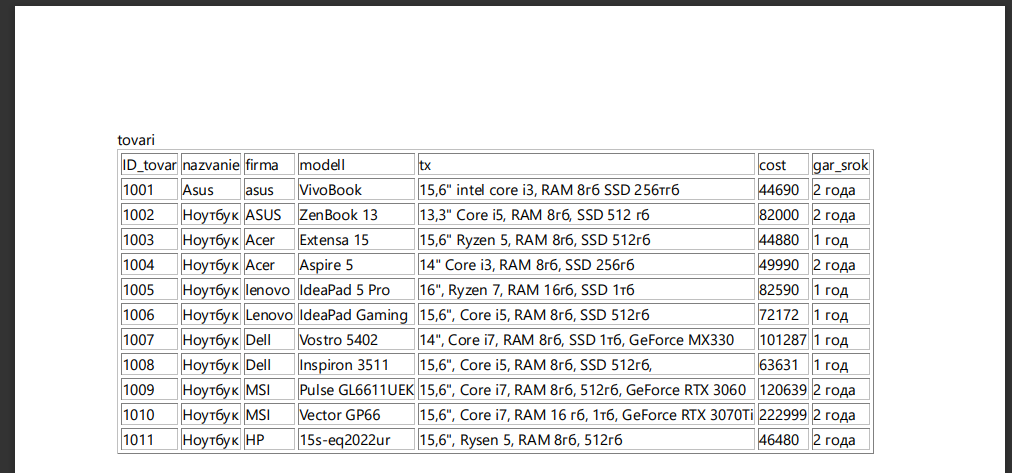


Рисунок 6.18 – Сформированный PDF файл

Тестирование в нормальных условиях всех форм разработанной информационной системы туристического агентства показали, что программа работает правильно.

6.2 Тестирование в экстремальных условиях

Проверка в экстремальных условиях должна идти сразу за проверкой программы в нормальных условиях. Тестовые данные этого этапа включают граничные значения области изменения выходных переменных, которые должны восприниматься программой, как правильные данные. Для цифровых данных в качестве экстремальных условий надо брать начальное и конечное значения допустимой области изменения переменной при одновременном изменении длин соответственно поля от минимума до максимума.

Разработанная программа предусматривает ввод пароля пользователя при регистрации длиной от 8 до 32 символов. На рисунке 6.19 показана попытка регистрации пользователя с паролем длиной 32 символа.

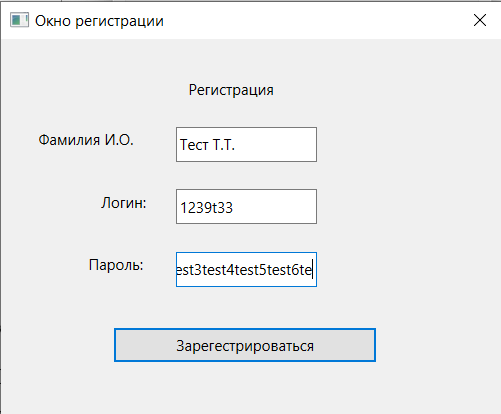


Рисунок 6.19 – Регистрация пользователя с максимальной длиной пароля

На рисунке 6.20 показана попытка регистрации пользователя с паролем длиной 8 символа.

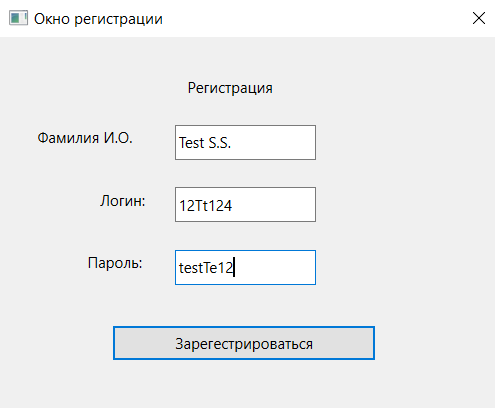


Рисунок 6.20 – Регистрация пользователя с минимальной длиной пароля

Для того, чтобы убедиться, что данные о новом пользователе были сохранены, был открыт файл реляционной базы данных и обновлена таблица с логинами и паролями пользователей. Добавленные пользователи показаны на рисунке 6.21.

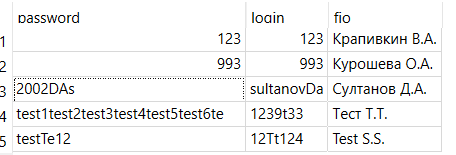


Рисунок 6.21 – Записи с данными новых пользователей

Тестирование программы в экстремальных условиях и проверка полученных результатов с ожидаемыми показали, что программа работает правильно.

6.3 Тестирование в исключительных ситуациях

Проверка в исключительных ситуациях – это последний этап тестирования программы. Он проводится с использованием данных, значения которых лежат за пределами допустимой области изменения, так как все программы разработаны в расчете на обработку какого-то ограниченного набора данных.

Наихудшая ситуация складывается тогда, когда программа воспринимает неверные данные, как верные и выдает неправильный, но правдоподобный ответ. Хорошая программа должна сама отвергать любые данные, которые она не в состоянии обработать правильно.

На рисунке 6.22 Представлена ситуация, когда пользователь осуществляет попытку входа с несуществующим в базе логином и паролем.

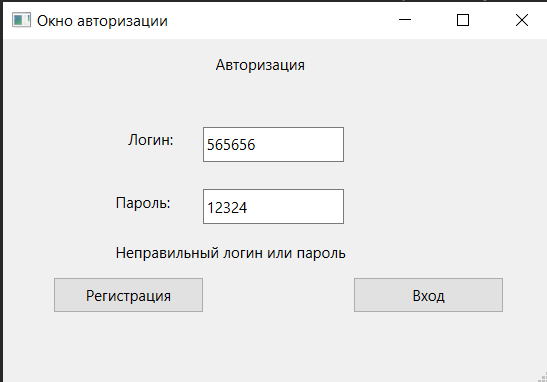


Рисунок 6.22 – Авторизация несуществующего пользователя

Программа сообщает о том, что логин или пароль введены неправильно, и не предоставляет доступ к главной форме.

Таким образом, просмотр результатов тестирования в исключительных ситуациях показал, что программа устойчива к вводу некорректных данных. Она отвергает такие данные и корректно продолжает свою работу

Вывод: Тестирование показало, что программа прошла тестирование в нормальных, экстремальных условиях, а также в исключительных ситуациях, следовательно, она работает правильно.

Листинг программы приведен в приложении А, а результаты выполнения – в приложении Б.

**7 Модульное тестирование**

Модульное тестирование (Unit Testing) – это тип тестирования программного обеспечения, при котором тестируются отдельные модули или компоненты программного обеспечения. Его цель заключается в том, чтобы проверить, что каждая единица программного кода работает должным образом. Данный вид тестирование выполняется разработчиками на этапе кодирования

приложения.

Модульные тесты изолируют часть кода и проверяют его работоспособность. Единицей для измерения может служить отдельная функция, метод, процедура, модуль или объект.

При проектировании информационной системы был разработан модульный тест для проверки логина и пароля при регистрации нового пользователя на регулярные выражения.

Результаты автоматического тестирования процедуры выведены в консоль при запуске программы.

Было предусмотрено 7 тестовых наборов входных пар значений (логин, пароль). Модуль теста корректно обработал набор данных и вывел в консоль результаты каждого теста, проведенного при запуске разработанной программы. Работа модульного теста при запуске программы показана на рисунке 7.1.