**МИНИСТЕРСТВО ОБРАЗОВАНИЯ ПЕНЗЕНСКОЙ ОБЛАСТИ**

**ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ПРОФЕССИОНАЛЬНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ПЕНЗЕНСКОЙ ОБЛАСТИ**

**«ПЕНЗЕНСКИЙ КОЛЛЕДЖ ИНФОРМАЦИОННЫХ И ПРОМЫШЛЕННЫХ**

**ТЕХНОЛОГИЙ (ИТ-КОЛЛЕДЖ)»**

**Обеспечение качества функционирования КС**

Отчет о выполнении лабораторной работы №7

Выполнил: Лисичников Алексей

Обучающийся курса \_\_3\_\_\_\_ группы \_22ит35\_\_\_\_\_\_

По специальности\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Преподаватель

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

А.Ю. Сазонова

Пенза, 2024г.

**Лабораторная работа №7**

**Цель работы:**

1. Тестирование программных продуктов с целью выявления ошибок и некорректной работы.
2. Применение метрик кода

Название работы: «Система учета заказов и их выполнения в мебельном салоне»

Функциональные требования:

* реализация добавления данных;
* реализация удаления данных;
* реализация изменения данных;
* реализация соединение с базой данных;
* наличие регистрации и авторизации;
* реализация поиска данных внутри таблиц.

**Ход работы:**

**Ответы на вопросы:**

1. Изучить текст стандарта ГОСТ Р ИСО/МЭК 9126-93. Ответить на вопросы:

1.1. Назовите область применения стандарта ГОСТ Р ИСО/МЭК 9126-93.

Настоящий стандарт определяет шесть характеристик, которые с минимальным дублированием описывают качество программного обеспечения. Данные характеристики образуют основу для дальнейшего уточнения и описания качества программного обеспечения. Руководства описывают использование характеристик качества для оценки качества программного обеспечения.

1.2. Перечислите характеристики качества ПО в соответствии с данным стандартом.

* Функциональные возможности;
* Надежность;
* Практичность;
* Эффективность;
* Сопровождаемость;
* Мобильность.

1.3. Как стандарт определяет представления о качестве ПО?

Имеется несколько представлений о качестве, некоторые из которых обсуждаются ниже.

1.3.1 Представление пользователя

Определение качества по ИСО 8402 отражает представление пользователя так же, как и характеристики, определенные в настоящем стандарте.

Пользователи в основном проявляют заинтересованность в применении программного обеспечения, его производительности и результатах использования. Пользователи оценивают програмное обеспечение без изучения его внутренних аспектов или того, как программное обеспечение создавалось.

Пользователя могут интересовать следующие вопросы:

* Имеются ли требуемые функции в программном обеспечении?
* Насколько надежно программное обеспечение?
* Насколько эффективно программное обеспечение?
* Является ли программное обеспечение удобным для использования?
* Насколько просто переносится программное обеспечение в другую среду?

1.3.2 Представление разработчика

Процесс создания требует от пользователя и разработчика использования одних и тех же характеристик качества программного обеспечения, так как они применяются для установления требований и приемки. Когда разрабатывается программное обеспечение для продажи, в требованиях качества должны быть отражены предполагаемые потребности.

Так как разработчики отвечают за создание программного обеспечения, которое должно удовлетворять требованиям качества, они заинтересованы в качестве промежуточной продукции так же, как и в качестве конечной продукции. Для того чтобы оценить качество промежуточной продукции на каждой фазе цикла разработки, разработчики должны использовать различные метрики для одних и тех же характеристик, потому что одни и те же метрики неприменимы для всех фаз жизненного цикла. Например, пользователь понимает эффективность в терминах времени реакции, тогда как разработчик использует в проектной спецификации термины длины маршрута и времени ожидания и доступа. Метрики, применяемые для внешнего интерфейса продукции, заменимы метриками, применяемыми для ее структуры.

Представление пользователя должно также включать представление о характеристиках качества, требуемое тем, кто сопровождает программное обеспечение.

1.3.3 Представление руководителя

Руководитель может быть более заинтересован в общем качестве, чем в конкретной характеристике качества, и по этой причине будет нуждаться в определении важности значений, отражающих коммерческие требования для индивидуальных характеристик.

Руководителю может также потребоваться сопоставление повышения качества с критериями управляемости, такими как плановая задержка или перерасход стоимости, потому что он желает оптимизировать качество в пределах ограниченной стоимости, трудовых ресурсов и установленного времени.

1.4. Из каких этапов состоит процесс оценивания? В чем состоит каждый из этапов?

1. Установление требований к качеству;

2. Подготовка к оцениванию;

2.1. Выбор метрик (показателей) качества;

2.2. Определение уровней ранжирования;

2.3. Определение критерия оценки;

3. Процедура оценивания;

3.1. Измерение;

3.2. Ранжирование;

3.3. Оценка.

2. Изучить текст стандарта ГОСТ Р ИСО/МЭК 12119-2000. Ответить на вопросы:

* 1. Что устанавливает данный стандарт?

Стандарт устанавливает:

- требования к пакетам программ (требования к их качеству);

- инструкции по испытанию пакета программ на соответствие его установленным требованиям (инструкции по тестированию, в частности по тестированию третьей стороной).

2.2. Какие функциональные требования предъявляет стандарт к программам и данным?

2.2.1. Установка (инсталляция)

Если установка пакета может быть выполнена пользователем, то при ее проведении должна быть обеспечена возможность успешной установки программ в соответствии с информацией, содержащейся в руководстве по установке. Каждая из необходимых систем, указанных в описании продукта, должна быть пригодной для установки программ.

В процессе установки должно быть определено, могут ли установленные программы функционировать, например путем использования поставленных с программами контрольных примеров или самотестирования с выдачей соответствующих сообщений.

2.2.2. Реализация функций

Все функции, указанные в документации пользователя, должны выполняться в виде, заданном в документации пользователя, на соответствующих средствах, с соответствующими характеристиками и данными, в рамках граничных значений, заданных там же.

2.2.3. Правильность

Программы и данные должны соответствовать всем обязательным формулировкам, приведенным в описании продукта и документации пользователя. Функции должны выполняться методом, соответствующим рабочей задаче. В частности, программы и данные должны удовлетворять всем требованиям из любого нормативного документа, на который дана ссылка в описании продукта.

2.2.4. Непротиворечивость

Программы и данные не должны противоречить сами себе, а также описанию продукта и документации пользователя. Каждый термин везде должен иметь один и тот же смысл.

Управление работой программы со стороны пользователя и соответствующая реакция программы (например, сообщения, выходные экранные форматы и печатные отчеты) должны быть единообразно структурированы.

2.3. Какие требования предъявляются к протоколу тестирования?

Протоколы по каждому тесту должны содержать информацию, достаточную для повторения теста (Руководство И С О /М Э К 25 [6]). Данная информация должна включать:

- план тестирования или технические требования (спецификацию) к тестированию, содержащие контрольные примеры;

- все результаты, связанные с контрольными примерами, включая все ошибки, выявленные.

2.4. Какую структуру должен иметь отчет о тестировании?

В отчете о тестировании должны быть суммированы цели и результаты тестирования (описанные в протоколах тестирования для каждого теста). Отчет о тестировании должен иметь следующую структуру.

1. Обозначение продукта.

2. Вычислительные системы, использованные при тестировании (технические средства, программные средства и их конфигурация).

3. Использованные документы (включая их обозначения).

4. Результаты тестирования описания продукта, документации пользователя, программ и данных.

5. Перечень несоответствий требованиям.

6. Либо перечень несоответствий рекомендациям, либо перечень не учтенных в продукте рекомендаций, либо формулировка того, что продукт не был протестирован на соответствие рекомендациям.

7. Дата окончания тестирования.

2.5. Как происходит дополнительное тестирование?

Когда продукт, который уже был протестирован, тестируется повторно (с учетом результатов предыдущего тестирования), тогда:

* все измененные части документов, функций и данных должны быть протестированы как новый продукт;
* все неизмененные части, на которые могут влиять измененные части или изменения в необходимой системе (в соответствии с опытной оценкой тестировщика), должны быть протестированы как новый продукт;
* все другие части должны быть по крайней мере выборочно протестированы.

1. **Функциональное тестирование**

Тестовые наборы приведены в таблице 1.

Таблица 1 – Функциональное тестирование.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Вариант использования** | **Тест** | **Ожидаемый результат** | **Результат** |
| 1 | Вход в систему. | Вход осуществляется пользователем.  Логин: admin  Пароль: admin | Открытие приложения. | Тест выполнен, приложение открыто.  (рисунок 1) |
| 2 | Вход для несуществующего пользователя.  Логин: asasasd  Пароль: jkdladas | Вывод сообщения об ошибке. | Тест выполнен, выдано сообщение об ошибке, так как такого пользователя не существует.  (рисунок 2) |
| 3 | Изменение данных таблиц на примере изменения таблицы «Доставка». | Изменение на корректные данные:  ID: 11  ID клиента: 10  Цена доставки: 32000  Время ожидания: 20 дней | Изменение данных строки в таблице и вывод сообщения об успехе. | Тест выполнен.  (рисунок 3) |
| 4 | Изменение на некорректные данные:  ID: 11  ID клиента: фыв  Цена доставки: фыв  Время ожидания: фыв | Вывод сообщения об ошибке. | Тест выполнен, выдано сообщение об ошибке.  (рисунок 4) |
| 5 | Добавление данных в таблицу «Клиенты». | Ввод корректных данных:  Имя: Алексей  Фамилия: Алексеев  Отчество: Алексеевич  Год рождения: 2003  Адрес: ул Красная  Номер телефона: 89898989891 | Добавление введенных данных в таблицу «Клиенты». | Тест выполнен, данные были добавлены.  (рисунок 5) |
| 6 | Ввод некорректных данных:  Имя: фыв  Фамилия: фыв  Отчество: фыв  Год рождения: фыв  Адрес: фыв  Номер телефона: фыв | Вывод сообщения об ошибке. | Тест выполнен, выдано сообщение об ошибке, так как данные некорректные.  (рисунок 6) |
| 7 | Удаление данных. | Выбор строки таблицы. | Удаление выбранной строки. | Тест выполнен, строка была удалена.  (рисунки 7, 8) |
| 8 | Поиск данных в таблице. | Ввод существующих данных в поле для поиска. | Строка с нужными данными будет найдена. | Тест выполнен, строка с нужными данными была найдена.  (рисунок 9) |
| 9 | Ввод несуществующих данных в поле для поиска. | Строка найдена не будет. | Тест выполнен, строка с нужными данными не была найдена.  (рисунок 10) |

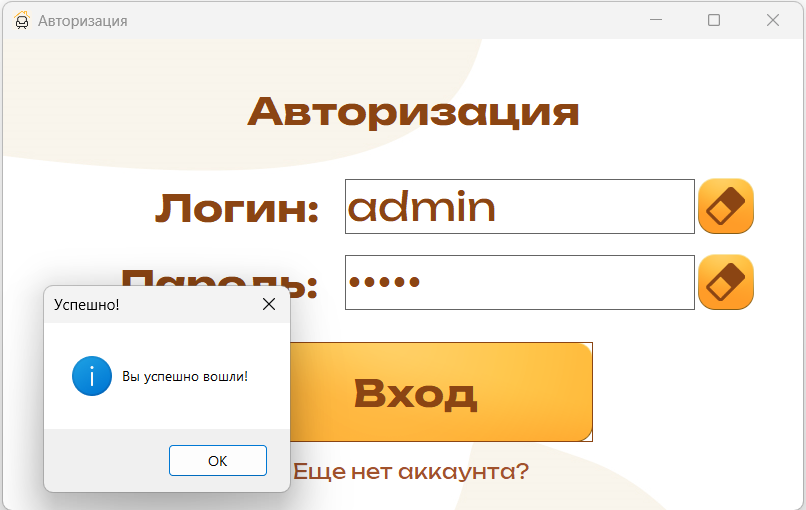


Рисунок 1 ­­­– Тестирование отправки вводимых корректных значений

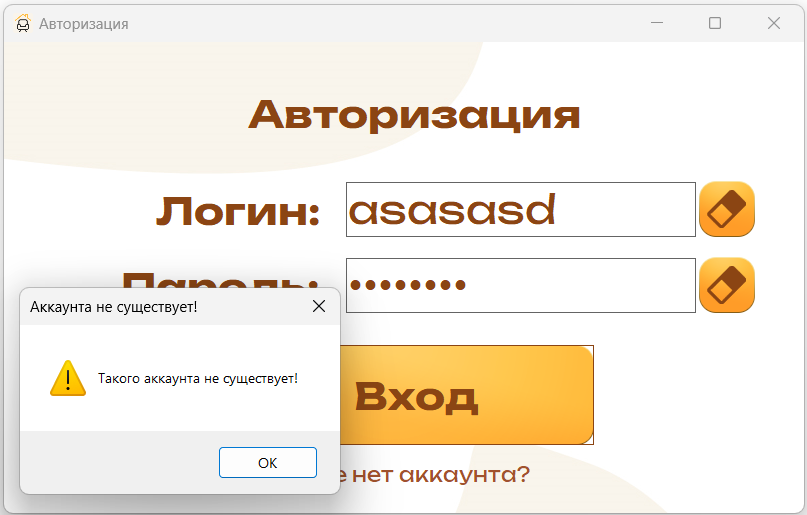


Рисунок 2 ­­­– Тестирование отправки вводимых некорректных значений

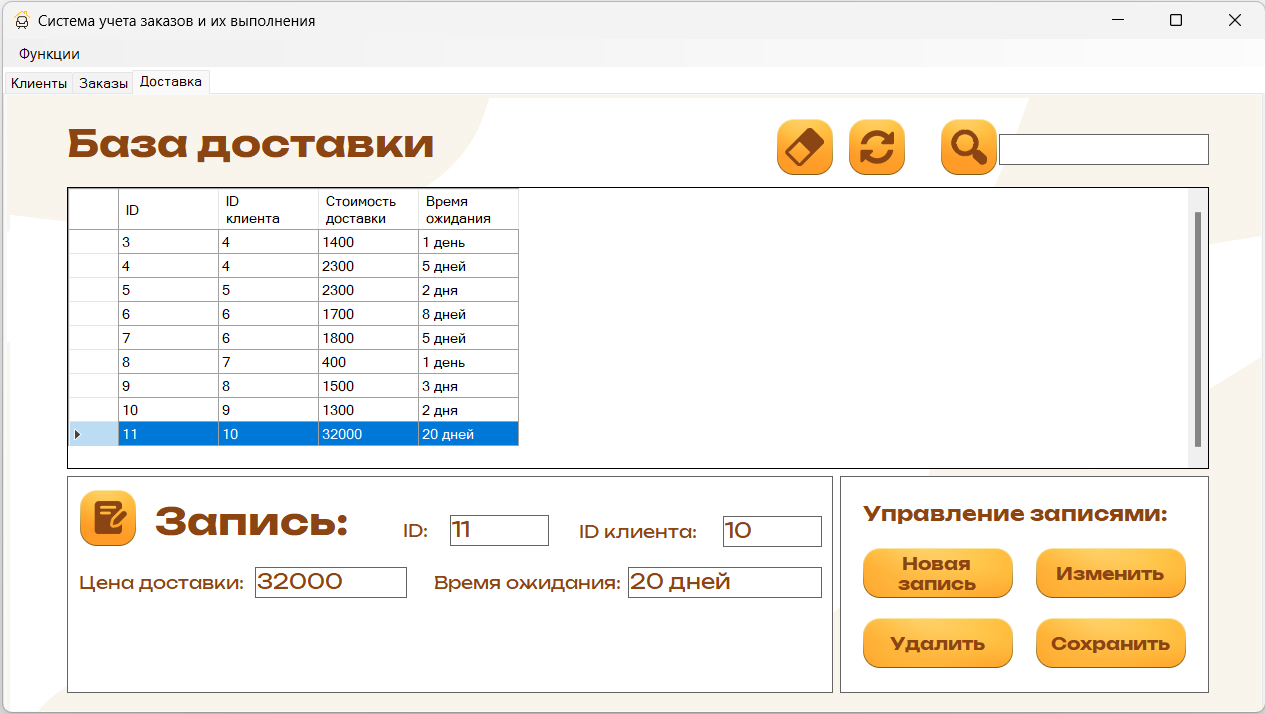


Рисунок 3 ­­­– Тестирование изменения данных на корректные значения

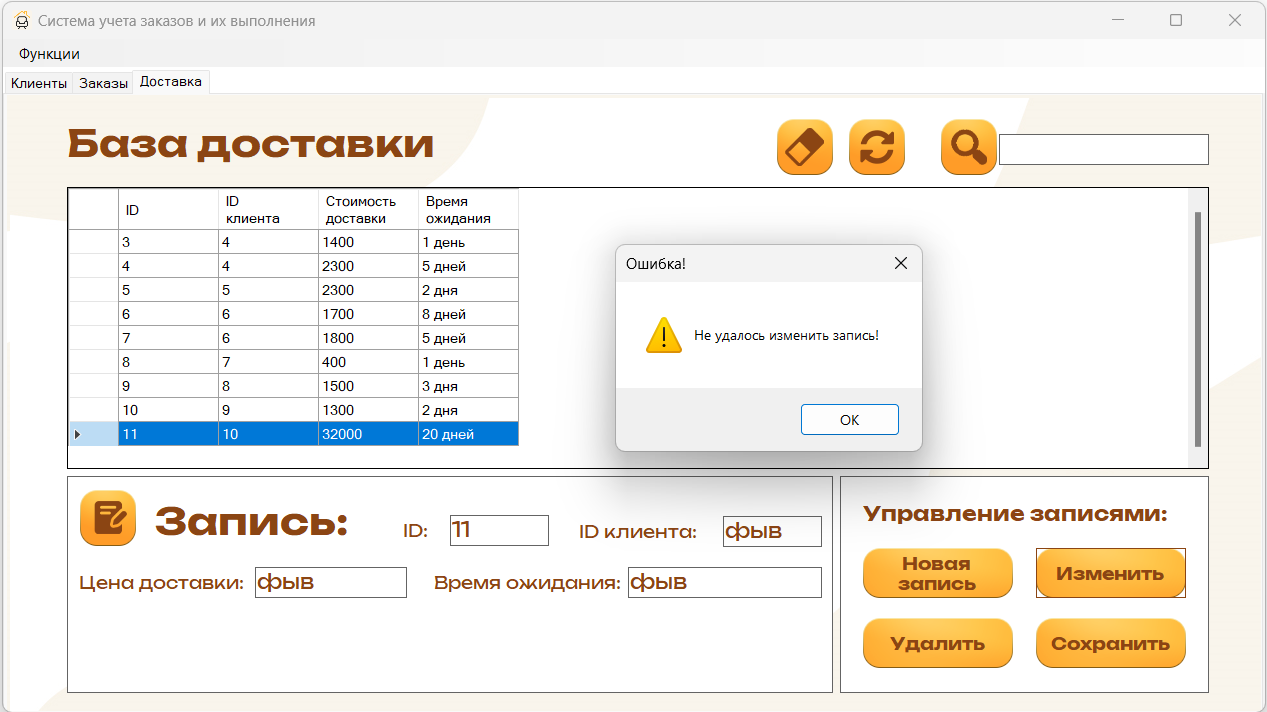


Рисунок 4 ­­­– Тестирование изменения данных на некорректные значения

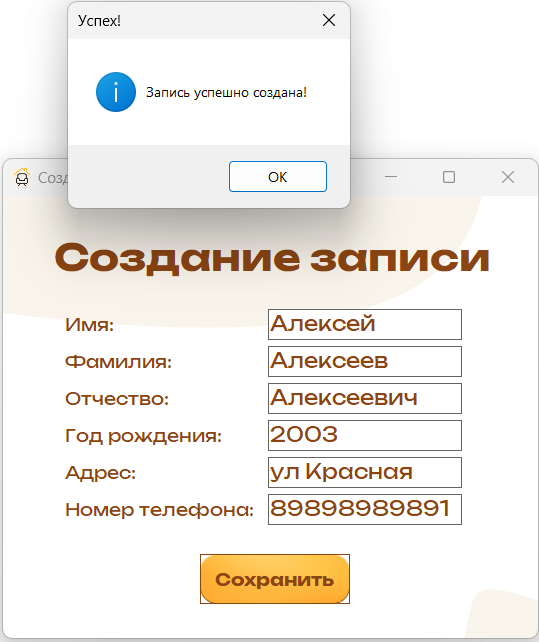


Рисунок 5 ­­­– Тестирование добавления корректных данных

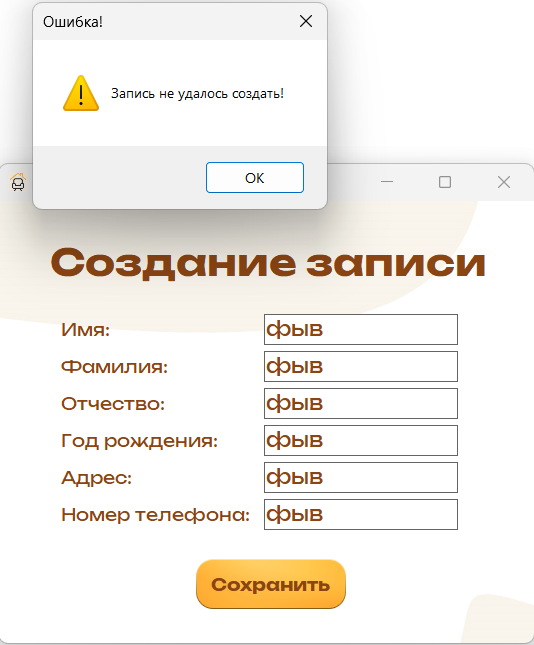


Рисунок 6 ­­­– Тестирование добавления некорректных данных

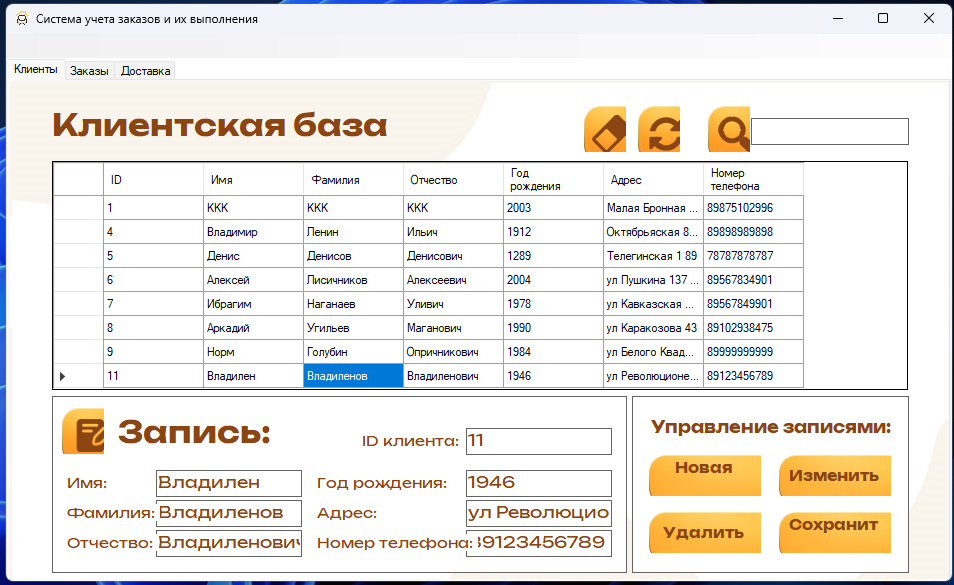


Рисунок 7 ­­­– Выбор записи для удаления

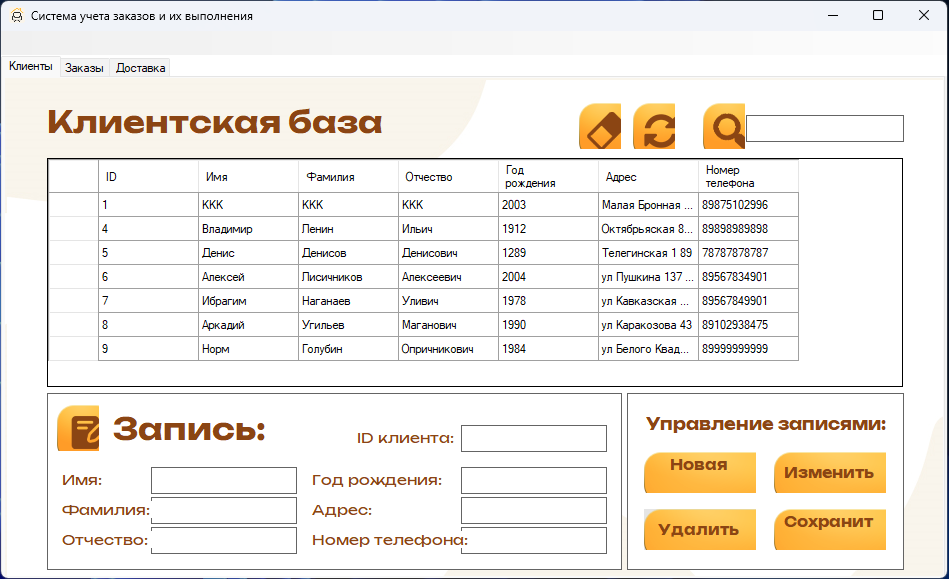


Рисунок 8 ­­­– Запись удалена



Рисунок 9 ­­­– Поиск существующей записи

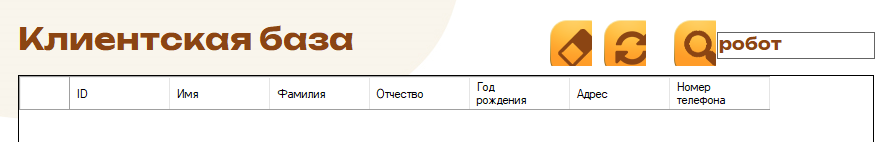


Рисунок 10 ­­­– Поиск несуществующей записи

Результаты функционального тестирования показали, что приложение работает корректно и надежно.

# Модульное тестирование

Модульное или unit-тестирование позволяет провести более глубокое исследование и анализ работоспособности отдельных модулей программного обеспечения. Существуют каркасы автономного тестирования, избавляющие пользователя от ручного воспроизведения тестов.

Тестированию подвергаются следующие классы:

* Form1 (основная программа);
* login (авторизация);
* signup (регистрация);
* Add\_Delivery (добавление данных в таблицу «Доставка»);
* Add\_Form (добавление данных в таблицу «Клиенты»);
* Add\_Zakaz (добавление данных в таблицу «Заказы»).

Результаты тестирования приведены на рисунке 11.

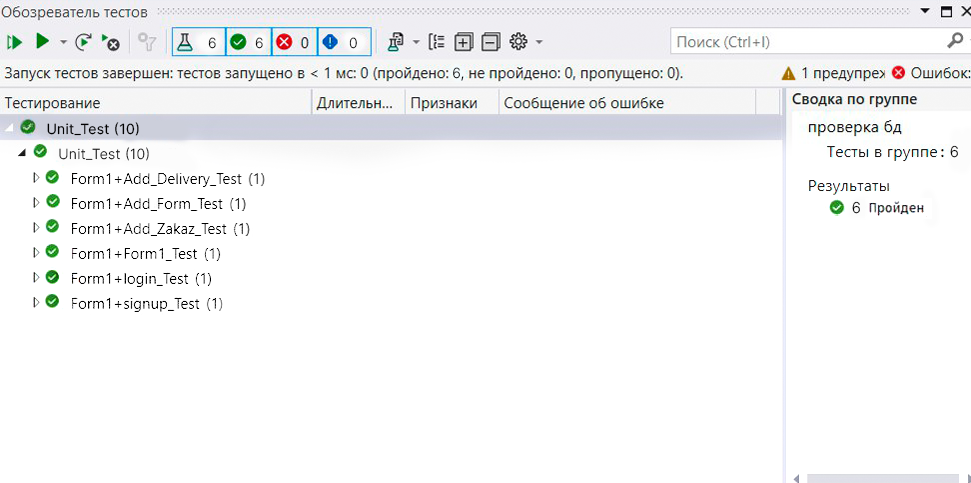


Рисунок 11 – Результаты модульного тестирования

# Метрики кода

Применение метрик позволяет руководителям проектов и предприятий изучить сложность разработанного или даже разрабатываемого проекта, оценить объем работ, стилистику разрабатываемой программы и усилия, потраченные каждым разработчиком для реализации того или иного решения [13].

Примером часто используемой метрики является комплексный показатель качества кода – Maintainability Index. Рассчитывается метрика по формуле:

****

где HV (Halstead Volume) – вычислительная сложность;

CC (Cyclomatic Complexity) – показывает структурную сложность кода, т.е. количество различных ветвей в коде;

LoC (Lines of Code) – количество строк кода.

Комплексный показатель качества кода может принимать значения   
от 0 до 100 и показывает относительную сложность поддержки кода. Чем больше значение этой метрики, тем легче поддерживать код.

Для определения вычислительной сложности программы сначала следует рассчитать словарьи длину программы, а затем вычислить на их основеHV. Словарь программы рассчитывается по формуле:

**,**

где  – число простых операторов и операций в программе;

 – число простых операндов (констант и переменных) в программе.

Длина программы рассчитывается по формуле:

,

где  – общее число простых операторов и операций в программе;

 – общее число простых операндов в программе.

Вычислительная сложность рассчитывается по формуле:

.

Структурную сложность программы можно рассчитать по формуле:

*СС = (количество операторов if) + (количество операторов цикла) +1.*

Если полученное значение СС больше 10, то это свидетельствует о структурной сложности программы, что затрудняет ее функциональное тестирование (методом «белого ящика») и требует упрощения   
программы [14].

Visual Studio позволяет вычислять метрики автоматически. Произведем вычисления для разрабатываемого программного обеспечения. Метрики исходного кода приложения библиотекаря представлены на рисунке 12.

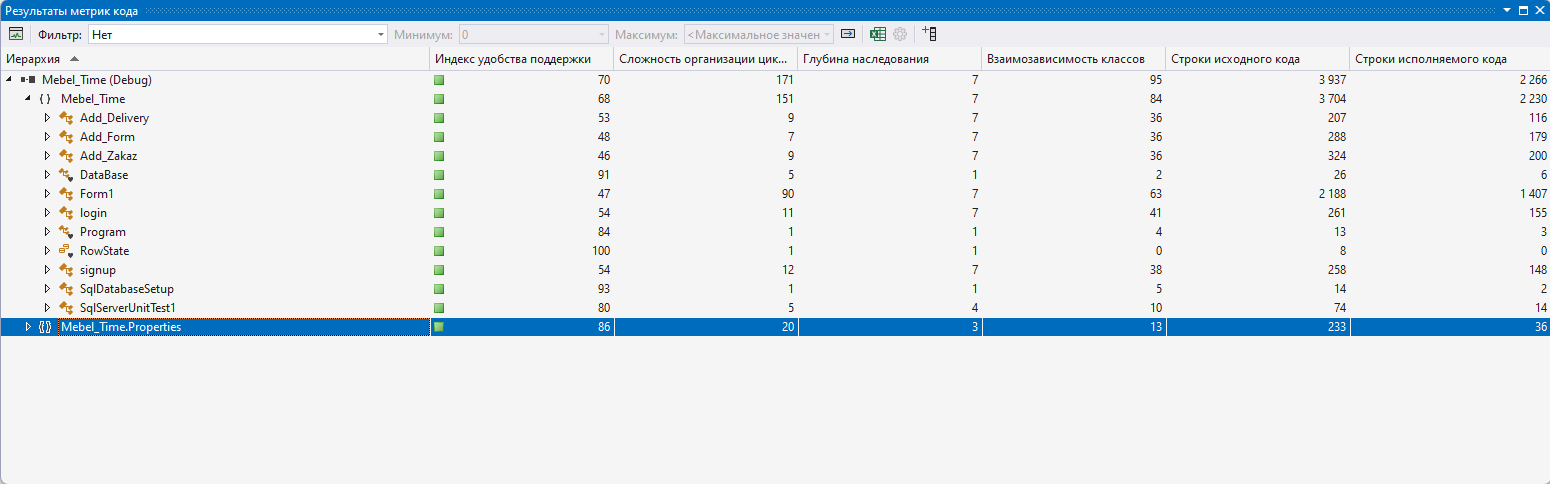


Рисунок 47 – Метрики исходного кода приложения библиотекаря

В данном проекте комплексный показатель кода (MI) находится на высоком уровне. Средняя отметка комплексного показателя составляет 70. Это свидетельствует об относительной простоте поддержки исходного кода.

Структурная сложность кода (СС) находится на среднем уровне.

В данном проекте большой глубиной наследования обладают   
классы-окна, описывающие графический интерфейс приложения, остальные классы обладают глубиной наследования не более 2. Это говорит о том, что человеку, не знакомому с данным проектом, будет легче понять, где определяются или переопределяются конкретные методы и поля класса.

Чем больше показатель связанности (объединения) классов, тем сложнее поддерживать данные классы. Как видно из результатов, показатель связанности относительно невысокий.

**Вывод:** научился тестировать программные продукты с целью выявления ошибок и некорректной работы, а также применять метрики кода.