Частное учреждение образования

«Колледж бизнеса и права»

ПРОГРАММНОЕ СРЕДСВТВО «КАЛЬКУЛЯТОР ПРАВИЛЬНО ПИТАНИЯ»

Пояснительная записка

к курсовому проекту по учебному предмету «Компьютерное программирование и языки программирования»

КП Т.219005.401

Руководитель проекта ( Е.В. Багласова )

Обучающийся ( К.Ю. Дирко )

2024

**СОДЕРЖАНИЕ**

Изм.

Лист

№ докум.

Подпись

Датаа

Лист

3

КП Т.219005.401 ПЗ

Разраб.

Дирко К.Ю,

Провер.

Багласова Е.В.

Реценз.

Н. Контр.

Утверд.

*Приложение калькулятор правильного питания*

Лит.

Листов

КБП

55

у

[Введение 4](#_Toc170053280)

[1 Объектно-ориентированный анализ и проектирование приложения 5](#_Toc170053281)

[1.1 Сущность задачи 5](#_Toc170053282)

[1.2 Проектирование модели 5](#_Toc170053283)

[2 Вычислительная система 6](#_Toc170053284)

[2.1 Требования к аппаратным и операционным ресурсам 6](#_Toc170053285)

[2.2 Инструменты разработки 6](#_Toc170053286)

[3 Проектирование задачи 8](#_Toc170053287)

[3.1 Требования к приложению 8](#_Toc170053288)

[3.2 Концептуальный прототип 11](#_Toc170053289)

[3.3 Организация данных 14](#_Toc170053290)

[3.4 Функции и элементы управления 20](#_Toc170053291)

[3.5 Проектирование справочной системы приложения 23](#_Toc170053292)

[4 Описание программного средства 25](#_Toc170053293)

[4.1 Общие сведения 25](#_Toc170053294)

[4.2 Функциональное назначение 25](#_Toc170053295)

[4.3 Входные и выходные данные 26](#_Toc170053296)

[5 Методика испытаний 27](#_Toc170053297)

[5.1 Технические требования 27](#_Toc170053298)

[5.2 Функциональное тестирование 27](#_Toc170053299)

[6 Применение 31](#_Toc170053300)

[6.1 Назначение программы 31](#_Toc170053301)

[6.2 Условия применения 31](#_Toc170053302)

[6.3 Справочная система 31](#_Toc170053303)

[Заключение 33](#_Toc170053304)

[Список использованных источников 34](#_Toc170053305)

[Приложение А Тескт программы 35](#_Toc170053306)

[Приложение Б Результаты работы программы 49](#_Toc170053307)

# Введение

Цель данной программы — предоставить пользователям инструмент для учета потребляемых калорий и нутриентов, чтобы они могли лучше контролировать свой рацион, следить за прогрессом и достигать своих целей в области здоровья и фитнеса.

Такой калькулятор калорий может стать незаменимым помощником для людей, которые стремятся к здоровому образу жизни, занимаются спортом или имеют определенные цели в области здоровья и фитнеса. Он помогает осознать потребление калорий и принимать более информированные решения о своем питании, что в конечном итоге способствует улучшению качества жизни и достижению поставленных целей.

В первом разделе пояснительной записки идет речь об объектно-ориентированном анализе проекта, а именно о сущности задачи и проектировании модели.

Во втором разделе перечисляются характеристики системы, на которой ведется разработка программного средства и с помощью каких инструментов она ведется. Также он содержит обоснования почему были выбраны именно эти инструменты.

В третьем разделе речь идет о проектировании задачи. Оно включает в себя: требования к приложению, концептуальный прототип, организация данных, функции и элементы управления, проектирование справочной системы приложения.

Четвертый раздел пояснительной записки содержит информацию о функциях программного средства и его возможностях.

В пятом разделе описываются методики, по которым проводились испытания программы.

В шестом разделе повествует о назначении программы и ее условиях применения.

В заключении будут выводы, которые были сделаны в течение разработки и создания курсового проекта.

Приложение А будет содержать текст программы.

Приложение Б будет содержать результат работы программы.

Графическая часть представлена диаграммой вариантов использования, диаграммой классов и диаграммой деятельности.

# Объектно-ориентированный анализ и проектирование приложения

## Сущность задачи

Требуется разработать программное средство для ведения статистики правильного питания. Это программа для автоматизации рабочего места. Рабочие места начали автоматизироваться с увеличением популярности компьютера. Этот способ улучшения производительности помогает ускорить многие процессы и уменьшить расходы на них. Единственным отрицательным аспектом в данной сфере является то, что не везде есть компьютеры. Исходя из всего сказанного ранее, появляется вывод, что данная сфера нуждается в обновлении процессов работы и улучшении текущих средств.

Получение данных будет осуществляться из пользовательского интерфейса, который будет описан в пояснительной записке. Делая выводы из этого, можно поставить задачи для программного продукта, а именно:

* программа должна быть понятна конечному пользователю;
* программа должна легко и быстро настраиваться.

Популярных конкурентов у программы нет, так как эти программы утеряли свою актуальность.

## Проектирование модели

Диаграммы вариантов использования, классов и деятельности — это графические представления, которые используются для моделирования функциональной структуры объектов, их действий и взаимодействий. Они позволяют разработчикам и пользователям системы получить ясное представление о функциях системы и ее внутренней структуре.

Диаграмма вариантов использования (Use Case Diagram) отображает взаимодействие пользователей с системой, описывая, какие действия могут выполнять пользователи и как система реагирует на эти действия.

Диаграмма классов (Class Diagram) описывает структуру системы на уровне классов, показывая классы, их атрибуты, методы и взаимосвязи между ними.

Диаграмма деятельности (Activity Diagram) показывает динамические аспекты системы, описывая поток работ или действий, выполняемых в системе.

Перечисленные выше диаграммы представлены в графической части, диаграмма вариантов использования представлена на листе 1 графической части, диаграмма классов на листе 2 графической части и диаграмма деятельности представлена на листе 3 графической части.

# Вычислительная система

## Требования к аппаратным и операционным ресурсам

Приложение должно предоставлять пользовательский интерфейс, который легко понимается и прост в использовании. Он должен быть интуитивным, обеспечивая пользователям комфортное взаимодействие с функционалом приложения. Поддержка многопользовательского доступа должна быть предусмотрена, позволяя настраивать уровни привилегий в соответствии с ролями пользователей.

Написание проекта будет вестись на ноутбуке со следующими характеристиками:

* процессор AMD Ryzen 5 5500U, 2021 года выпуска, техпроцесс 7 нанометров, частота 2,1 ГГц и 4,0 ГГц в турбо режиме, 6 ядра, 12 потока, 64-х битный набор команд;
* оперативная память 8 ГБ DDR4 3200 Гц;
* клавиатура и мышка.

## Инструменты разработки

При разработке программного средства будут использованы следующие программы:

* операционная система MS Windows 10 Pro 64-х разрядная;
* редактор кода MS Visual Studio Community 2022;
* язык программирования C#;
* фреймворк MS Windows Forms.

MS Windows 10 Pro была выбрана из-за необходимости тестирования на ней в будущем. Также выбор был обусловлен тем, что она более лучше подходит для мощных систем и является более приятной для использования, чем операционные системы семейства Linux.

MS Visual Studio Community 2022 был выбран из-за удобства пользования разработчика. Также данный редактор кода абсолютно бесплатный и не требует постоянной подписки. Также он является одним из лучших решений для разработки программ на языке программирования С# на операционной системе MS Windows 11 Pro.

Язык C# был выбран основным в процессе создания программного средства в силу следующих причин:

* официальная документация по языку C# содержит множество примеров описания реализаций различных структур, что упрощает процесс проектирования;
* наличие в базе всех необходимых библиотек для работы с сетями и файловой системой.

Windows Forms – фреймворк программирования приложений, отвечающий за графический фреймворк пользователя и являющийся частью Microsoft .NET Framework. Данный фреймворк упрощает доступ к элементам интерфейса Microsoft Windows за счет создания обёртки для существующего Win32 API в управляемом коде.

# Проектирование задачи

## Требования к приложению

Для программы должны быть следующие аппаратные характеристики компьютера:

* процессор AMD Ryzen 3100, 2020 года выпуска, техпроцесс 7 нанометров, частота 3,6 ГГц и 3,9 ГГц в турбо режиме, 4 ядра, 8 потока, 64- х битный набор команд;
* оперативная память 4 ГБ DDR4 3200 Гц;
* операционная система MS Windows 10 Pro, 64-х разрядная.

Доступ к интернету не будет необходим при работе в программе.

## Концептуальный прототип

Данный раздел направлен на описание и уточнение внешнего вида программы и демонстрации прототипа приложения.

Интерфейс программного модуля для программного средства «Калькулятор правильного питания» построен таким образом, что в модуле присутствует три главных формы: одна для записи данных в таблицу, другая для записи данных в базу данных и третья для демонтсрвции графиков и отчетов.

Форма для записи данных в таблицы состоит из двух частей: панель управления и зона отобрадения сводки.

На рисунке 1 представлен интерфейс панели управления.

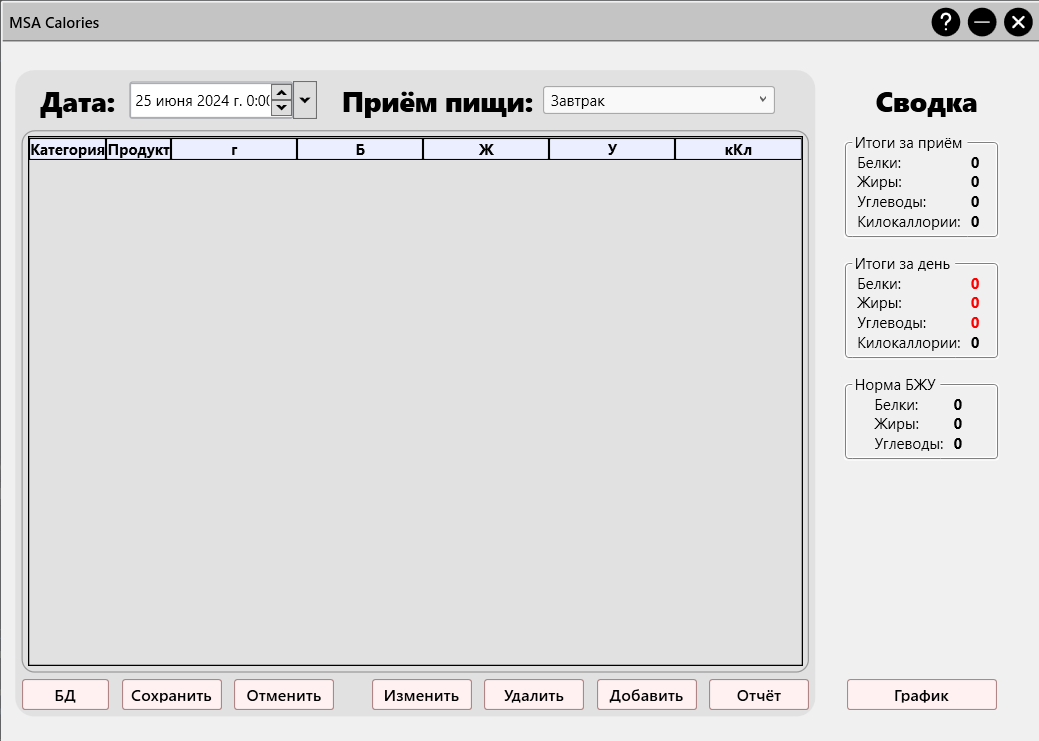


Рисунок – Интерфейс панели управления

Панель управления предназначена для выбора приема пищи и заполнения данных этого приема. Она включает в себя кнопки: «БД», «Сохранить», «Отменить», «Изменить», «Добавить», «Удалить», «Отчет», «График» и включает в себя выбор даты и выбор приема пищи. Названия кнопок соответствуют названиям таблиц базы данных.

Далее, в зоне отображения таблиц находится: сама таблица, реализованная за счет DataGridView, соответствующая выбранной кнопки.

В зоне сводка отображаются нормы БЖУк их норма и превышение.

Интерфейс зоны отображения таблиц представлен на рисунке 6.

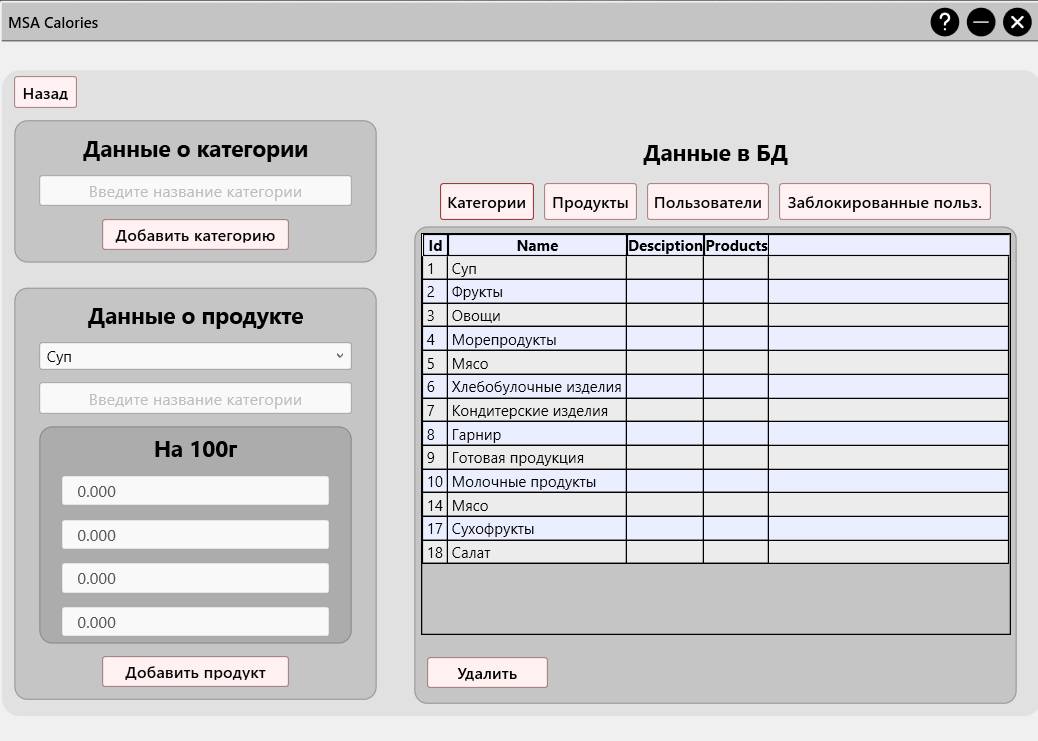


Рисунок 2 – Интерфейс зоны записи в БД

Кнопка «Добавить» открывает новую форму, для добавления, изменения, удаления записей. Она так же соответствует выбранной таблице.

На рисунке 7 изображен интерфейс кнопки «Добавить».

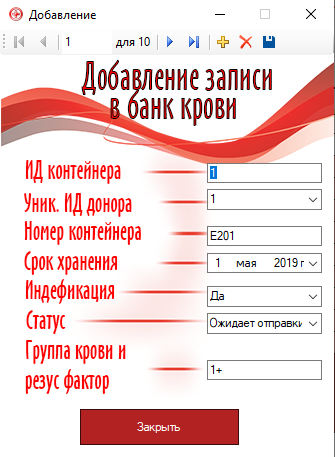


Рисунок 3 – Интерфейс кнопки «Добавить»

Кнопка «Обновить» обновляет данные в таблице для того, чтобы внесенные данные сразу в ней появились.

Далее, при нажатии кнопки «Учет» открывается новая форма, которая является второй главной формой.

Интерфейс данной формы представлен на рисунке 8.

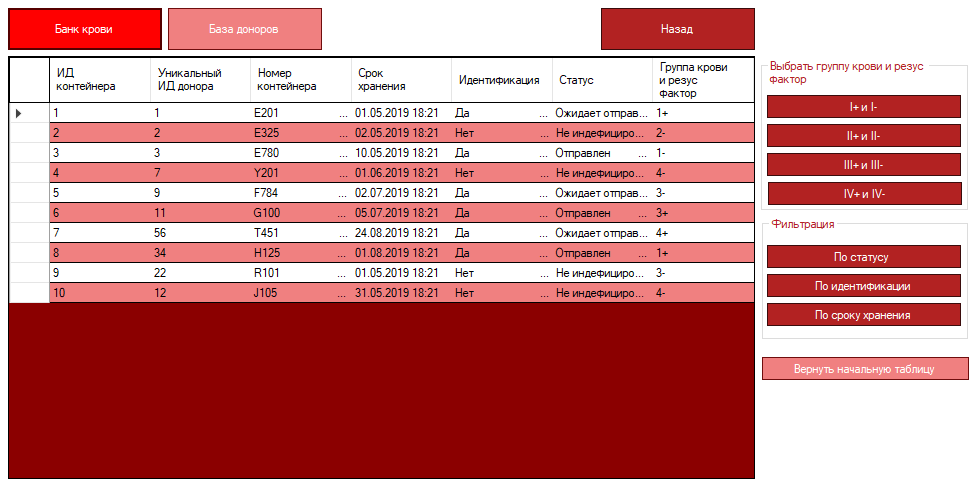


Рисунок 4 – Интерфейс формы «Система учета данных»

На форме находятся кнопки «Банк крови», «База доноров», «Назад». При нажатии на кнопки «Банк крови» или «База доноров» появляется одна из зон управления. На данной зоне находятся кнопки, при нажатии на которые выполняется хранимые процедуры, которые ранее были описаны. Так же присутствует кнопка «Вернуть начальную таблицу» которая возвращает изначальные данные таблицы.

При нажатии на кнопку «База доноров» появляется аналогичная зона. Меняется в ней только группа кнопок «Фильтрация».

Кнопка «Назад» закрывает данную форму, возвращаясь к форме «Система заполнения данных».

Оставшиеся кнопки на форме это: «О программе» и «Выход» представлены на рисунке 9.

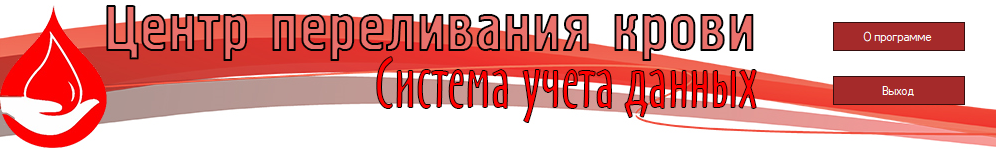


Рисунок 5 – Интерфейс кнопок «О программе» и «Выход»

При нажатии на кнопку «О программе» появляется новая форма, содержащая информацию о данном программном модуле, а также инициалы разработчика и руководителя.

Интерфейс формы «О программе» представлен на рисунке 10.

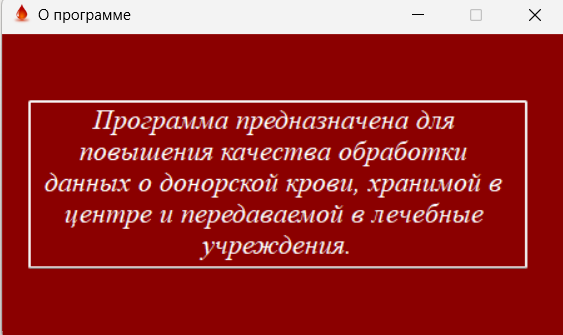


Рисунок 6 – Интерфейс формы «О программе»

И последнее – при нажатии кнопки «Выход», происходит выход из программы.

## Организация данных

Физическая структура данных представляет собой файловую систему. В главном каталоге находятся такие файлы как:

* Form1.cs – файл главной страницы;
* Form2.cs – файл с добавлением записи о доноре;
* Form3.cs – файл с добавлением записи в базу доноров;
* Form4.cs – файл с добавлением записи в отдел заготовки крови;
* Form5.cs – файл с добавлением записи в банк крови;
* Form6.cs – файл с добавлением записи в штат сотрудников;
* Form7.cs – файл с базой доноров;
* Form8.cs – файл со справкой для приложения;
* Program.cs – файл запуска приложения;
* Database1.mdf – файл базы данных;
* Database1DataSet.xsd – файл схемы описания XML-файла;

Все модули, которые представляют окно не связаны с файлами для определения функций, весь их код находиться в файлах заголовков. Но те файлы, которые не связаны с отображением окон, имеют файл заголовков с объявлением всех функций и классов и файл для определения их.

Для создания реляционной базы данных «ЦПК» было создано 5 таблиц, которые отображают полную информацию необходимую для автоматизации процесса обработки обращений доноров в центр переливания крови:

Таблица «Донор» предназначена для хранения и отображения данных о доноре, содержит следующие поля:

* «КодДонора» (ключевое поле) – тип данных счетчик, размер поля – длинное целое;
* «ФИО» – тип данных текстовый, (текстовый, размер поля – 255);
* «Пол» – тип данных текстовый, (текстовый, размер поля – 255);
* «АдресПроживания» – тип данных текстовый, (текстовый, размер поля – 255);
* «Возраст» – тип данных числовой, размер поля – длинное целое;
* «НомерТел» – тип данных числовой, размер поля – длинное целое;

Таблица «База доноров» предназначена для хранения и отображения данных о доноре сдавшему кровь, и содержит следующие поля:

* «КодДонораБаза» (ключевое поле) – тип данных счетчик, размер поля – длинное целое;
* «КодДонора» – тип данных числовой (SELECT Донор.КодДонора FROM Донор);
* «ОбщееКолВоСданКров» – тип данных текстовый, (текстовый, размер поля – 255);
* «ПослДатаСдачКров» – тип данных текстовый, (текстовый, размер поля – 255);
* «КатерДонора» – тип данных текстовый, (текстовый, размер поля – 255);
* «ПовторнСдача» – тип данных текстовый, (текстовый, размер поля – 255);
* «ГруппаКровРезФак» – тип данных текстовый, (текстовый, размер поля – 255);

Таблица «Заготовка крови» предназначена для хранения и отображения данных о количестве сданной крови и данных о доноре, и содержит следующие поля:

* «УнИДДонора» (ключевое поле) – тип данных счетчик, размер поля – длинное целое;
* «КодДонораБаза» – (SELECT База доноров.КодДонораБаза FROM База доноров);
* «КолВоСданнойКрови» – тип данных текстовый, (текстовый, размер поля – 255);
* «ДатаСдачКров» – тип данных дата/время;
* «Врач» – тип данных числовой (SELECT Врач.КодВрача FROM Врач);

Таблица «БанкКрови» предназначена для хранения и отображения данных о контейнерах крови и ведения их учета, и содержит следующие поля:

* «ИДКонт» (ключевое поле) – тип данных счетчик, размер поля – длинное целое;
* «УнИДДонора» – тип данных числовой (SELECT Заготовка крови. УнИДДонора FROM Заготовка крови);
* «НомКонт» – тип данных текстовый, (текстовый, размер поля – 255);
* «СрокХранения» – тип данных дата/время;
* «Индефикация» – тип данных текстовый, (текстовый, размер поля – 255);
* «Статус» – тип данных текстовый, (текстовый, размер поля – 255);
* «ГрупКровРезФак» – тип данных текстовый, (текстовый, размер поля – 255);

Таблица «Врач» предназначена для хранения и отображения данных о сотруднике центра, и содержит следующие поля:

«КодВрача» (ключевое поле) – тип данных счетчик, размер поля – длинное целое;

* «ФИО» – тип данных текстовый, (текстовый, размер поля – 255);
* «Пол» – тип данных текстовый, (текстовый, размер поля – 255);
* «Возраст» – тип данных текстовый, (текстовый, размер поля – 255);
* «Образование» – тип данных текстовый, (текстовый, размер поля – 255);
* «СтажРаботы» – тип данных числовой, размер поля – длинное целое;
* «НомерТел» – тип данных числовой, размер поля – длинное целое;

Все таблицы имеют атрибуты сущностей, при этом содержимое группы полей не относиться более чем к одной записи в таблице, то есть все значения атрибутов атомарны для данной предметной области и не нуждаются в дальнейшей детализации.

Таким образом, все отношения между сущностями находятся в третьей нормальной форме, и дальнейшая нормализация не имеет смысла.

На рисунке 11 изображена схема данных БД «ЦПК» в третьей нормальной форме.



Рисунок 7 – Схема данных базы данный «ЦПК»

Для реализации базы данных, была создана в рабочей среде MS Visual Studio 2022, таблицы. Для создания данных таблиц будет использоваться язык запросов SQL. Для работы с данными из нескольких таблиц установим связи между таблицами.

На рисунке 12 представлен SQL запрос на создание таблицы «Донор».

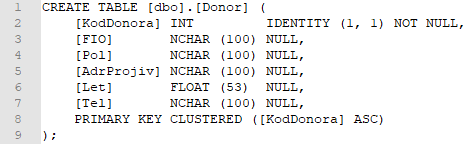


Рисунок 8 – SQL запрос на создание таблицы «Донор»

На рисунке 13 представлен SQL запрос на создание таблицы «База доноров».

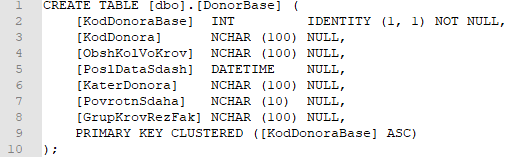


Рисунок 9 – SQL запрос на создание таблицы «База доноров»

На рисунке 14 представлен SQL запрос на создание таблицы «Отдел заготовки крови».

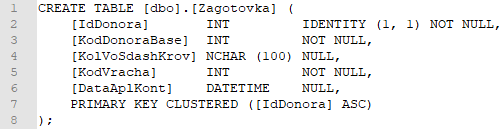


Рисунок 10 – SQL запрос на создание таблицы «Отдел заготовки крови»

На рисунке 15 представлен SQL запрос на создание таблицы «Банк крови».

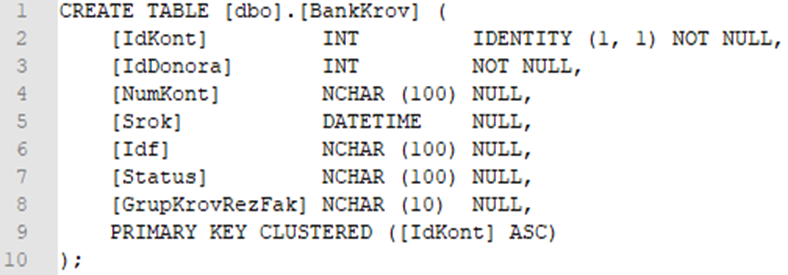


Рисунок 11 – SQL запрос на создание таблицы «Банк крови»

На рисунке 16 представлен SQL запрос на создание таблицы «Врач».

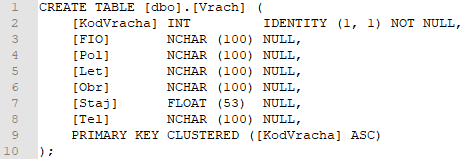


Рисунок 12 – SQL запрос на создание таблицы «Врач»

Все таблицы были созданы в соответствие с вышеперечисленной реляционной моделью базы данных.

В данном программном модуле должен присутствовать поиск, для более удобного и эффективного учета данных. Данная функция реализуется так же за счет SQL запросов. В отличие от другие СУБД, рабочая среда MS Visual Studio 2022 создает хранимую процедуру, которая в последствие реализуется за счет средств интерфейса.

В данном программном модуле хранимые процедуры будут выполнять функции:

* поиск данных по группе крови и резус фактору;
* сортировка по определенным критериям;

Данные хранимые процедуры будут использоваться в двух главных таблицах: «База доноров» и «Банк крови».

На рисунке 17 будет представлен SQL запрос на поиск контейнеров крови, в таблице «Банк крови», с группой крови и резус фактором – первая положительная и отрицательная.



Рисунок 13 – SQL запрос на поиск контейнеров крови

Если в поле «Группа крови и резус фактор», таблицы «Банк Крови», есть значения «1+» или «1-», то данная хранимая процедура вывод данные таблицы с соответствующими значениями.

Далее аналогичным образом формируются хранимые процедуры для остальных групп крови. Данные запросы применяются к обоим главным таблицам.

Так же для эффективности были созданы запросы сортировки. Для таблицы «Банк крови» сортировка происходит по полям: «Статус», «Идентификация», «Срок хранения». Для таблицы «База доноров» по полям: «Категория донора», «Последняя дата сдачи крови», «Общее количество сданной крови».

Как пример, на рисунке 18 будет представлен SQL запрос на сортировку таблицы «База донора» по полю «Категория донора».



Рисунок 14 – SQL запрос «Сортировка по категории донора»

## Функции и элементы управления

В рассмотренной ранее панели управления, формы «Система заполнения данных», для каждой кнопки имеется собственный код.

На примере кнопки «Банк крови» рассмотрим данный код, который представлен на рисунке 19.

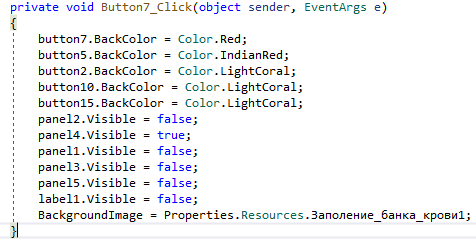


Рисунок 15 – Код кнопки «Банк крови»

Данный код реализует то, что при на нажатие на кнопку отображается панель, на которой находится соответствующая таблица, в данном случае таблица «Банк крови». При нажатии на кнопку, цвет её становится более яркий, данная функция реализуется за счет свойства BackColor. Так же меняется фон самой формы. Данные функции применяется для всех кнопок панели управления.

Переходим в зону отображения таблиц. Здесь, как говорилось ранее, находятся панели. Данные панели идентичны друг другу, кроме источника данных таблиц. На панели находится кнопка «Добавить» и «Обновить».

Код кнопки «Добавить» представлен на рисунке 20.

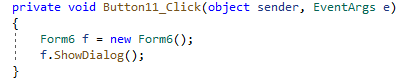


Рисунок 16 – Код кнопки «Добавить»

Данный код открывает форму, которая была описана ранее. Код для всех кнопок «Добавить» ничем не отличается друг от друга.

На форме «Добавление» так же присутствует код, позволяющий сохранять введенные данные в базу данных.

Данный код представлен на рисунке 21.

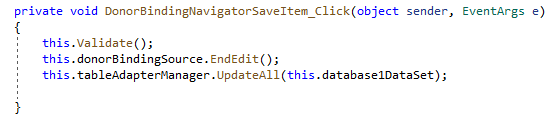


Рисунок 17 – Код формы «Добавление»

Последняя кнопка, из зоны отображение таблиц, это «Обновить».

Данная кнопка позволяет обновить таблицу для того, чтобы новые данные успешно прогрузились в таблицу. Код данной кнопки также идентичен остальным, кроме источника данных.

Код кнопки «Обновить» представлен на рисунке 22.

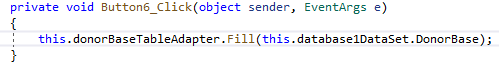


Рисунок 18 – Код кнопки «Обновить»

Далее, при нажатии на кнопку «Учет», аналогично кнопки «Добавить», открывает форма «Система учета данных». На данной форме находятся кнопки «Банк крови» и «База доноров».

Код кнопки «Банк крови» представлен на рисунке 23.

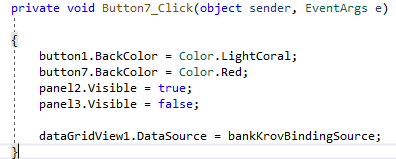


Рисунок 19 – Код кнопки «Банк крови» на форме «Система учета данных»

При нажатии на кнопку, она так же меняет свой цвет на более яркий, отображается нужную панель, и изначально загружает нужный источник данных.

На панели находятся группы кнопок, отвечающие за ранее описанные хранимые процедуры.

Пример кода кнопки «I+ и I-» представлен на рисунке 24.

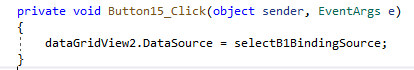


Рисунок 20 – Код кнопки «I+ и I-»

Запрос возвращает таблицу данных, поэтому данный код позволяет при нажатие на кнопку, поменять источник данных, что позволяет отобразить таблицу данных в одной и той же DataGridView. Таким же образом работают все кнопки.

Для смены источника данных в одном DataGridView, необходимо чтобы все источники данных загружались при открытии формы. Так же данный код делает обе панели невидимыми до нажатия кнопки.

Данный код представлен на рисунке 25.

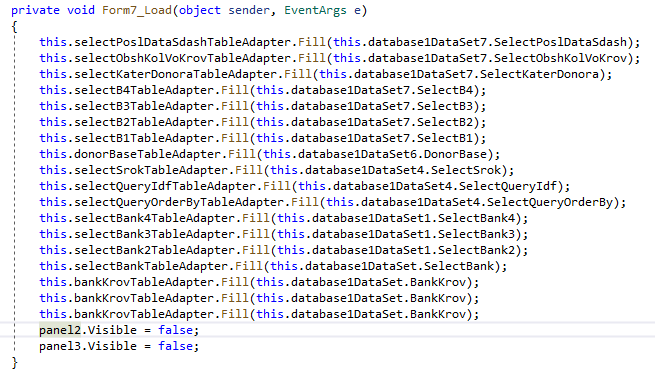


Рисунок 21 – Код загрузки всех источников данных для DataGridView

И последняя кнопка «Выход» производит закрытие программного модуля.

Код данной кнопки представлен на рисунке 26.

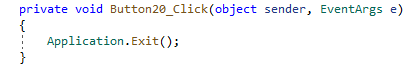


Рисунок 22 – Код кнопки «Выход»

Графический интерфейс создан, к нему подключена база данных, за счет программного кода реализованы такие функции, как:

– Добавление, изменение, удаление записей;

– Обновление данных базы данных;

– Поиск и сортировка данных;

Программный модуль соответствует поставленными нами целями.

## Проектирование справочной системы приложения

Справочная система будет представлена в виде формы, которая будет открываться при нажатии на кнопку «О программе». Она была создана простым путем написания справки в текстовом окне поля. В поле будет идти речь о том, для чего данная программа создана.

Справочная система представлена на рисунке 27.

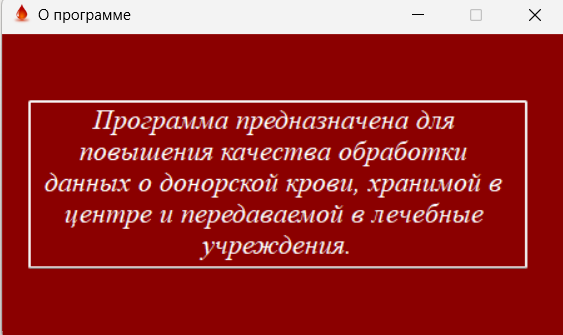


Рисунок 23 – Справочная система

# Описание программного средства

## Общие сведения

Как уже говорилось в первом разделе, программа будет автоматизацией рабочего места. Этот проект необходим медицинским работникам для упрощения работы.

Для запуска программы на данный момент потребуется операционная система MS Windows 10 64-х битная. Остальные библиотеки программа уже будет содержать для конечной версии программы.

Также для запуска программы понадобится следующее минимальное аппаратное обеспечение необходимое для функционирования данного курсового проекта:

* процессор AMD A12-9720P;
* оперативная память 4 ГБ DDR4;
* свободное место на диске 1 ГБ;
* встроенная видеокарта AMD Radeon R7 Graphics 2 ГБ.

Установщика программного средства не будет, в связи с тем, что программа сразу будет идти как исполняемый файл. Пользователь сможет скачать папку с программой и всеми необходимыми для нее файлами, уже находящиеся в папке.

## Функциональное назначение

Программа ведения статистики центра переливания крови предназначена для автоматизации рабочего места сотрудников центра переливания крови. Основные задачи, которые решает программа, включают сбор данных о донорах, регистрацию трансфузий, формирование отчетов, а также поиск и фильтрацию данных.

Для сбора данных о донорах программа позволяет вводить и хранить информацию, такую как ФИО, дата рождения, пол, контактные данные, группа крови, резус-фактор. Регистрация трансфузий включает учет данных о доноре, получателе, дате, времени и результатах трансфузии.

Программа позволяет формировать отчеты о количестве доноров и получателей, частоте трансфузий и активности доноров. Также реализована возможность быстрого поиска и фильтрации данных по различным параметрам, таким как, группа крови, резус-фактор, по статусу, по идентификации и по сроку хранения.

Средства защиты и разграничения доступа к данным включают локальное хранение данных на устройстве, на котором установлена программа, что исключает необходимость в интернет-соединении и минимизирует риск несанкционированного доступа через сеть. Для защиты конфиденциальной информации все данные могут быть зашифрованы с использованием алгоритмов шифрования, что предотвращает несанкционированный доступ к данным в случае физического доступа к устройству.

Программа работает без подключения к интернету, и сетевые функции не требуются для ее работы. Все данные обрабатываются и хранятся локально на устройстве пользователя, что обеспечивает высокую скорость работы программы и минимизирует риски, связанные с сетевой безопасностью и конфиденциальностью данных.

## Входные и выходные данные

Учет данных производится непосредственно в программе и не выводится наружу. Это значит, что все данные хранятся и обрабатываются внутри программы, без возможности экспорта данных или вставки сохраненных файлов. Такая организация работы повышает безопасность и конфиденциальность информации, минимизируя риски утечки данных.

# Методика испытаний

## Технические требования

Для программы должны быть следующие минимальные аппаратные характеристики компьютера:

* процессор AMD A12-9720P, 2017 года выпуска, техпроцесс 28 нанометра, частота 2,7 ГГц и 3,6 ГГц в турбо режиме, 4 ядра, 4 потока, 64-х битный набор команд;
* оперативная память 4 ГБ DDR4 1866 ГГц;
* свободное место на жёстком диске 1 ГБ;
* операционная система MS Windows 10, 64-x разрядная.

Доступ к интернету не будет необходим при работе в программе.

Программных средств для функционирования программе не нужно.

## Функциональное тестирование

Немало важную роль в разработке программного модуля играет тестирование. Процесс тестирования программных модулей состоит в проверке корректности обработки модулями поступающей информации и получающихся на выходе данных в соответствии с функциями, представленными в спецификациях требований.

Перед началом тестирования необходимо открыть приложение «kranan.exe».

Таблица – Тест-кейсы

|  |  |  |  |
| --- | --- | --- | --- |
| № | Модуль/  функция | Шаги воспроизведения | Ожидаемый результат |
| Фактический результат |
| 1 | Регистрация донора | 1 Выбрать пункт меню «Донор».  2 Нажать кнопку «Добавить».  3 Ввести данные донора (ФИО, пол, адрес проживания, возраст, номер телефона).  4 Нажать кнопку «Сохранить данные». | Донор успешно зарегистрирован и внесён в базу организации. |
| Совпадает с ожидаемым результатом.  Результат представлен в приложении Б на рисунке Б. 2. |

Продолжение таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название | Шаги воспроизведения | Ожидаемый результат |
| Фактический результат |
| 2 | Фильтрация данных | 1 Выбрать пункт меню «Учёт».  2 Выбрать пункт меню «Банк крови».  3 Выбрать фильтр «По статусу».  4 Фильтр применён. | Фильтрация успешно применена к данным. |
| Совпадает с ожидаемым результатом.  Результат представлен в приложении Б на рисунке Б. 3. |
| 3 | Удаление донора из базы данных | 1 Выбрать пункт меню «Донор».  2 Нажать кнопку «Добавить».  3 Выбрать донора, которого необходимо удалить.  4 Нажать кнопку «Удалить». | Донор успешно удалён с базы доноров. |
| Совпадает с ожидаемым результатом.  Результат представлен в приложении Б на рисунке Б. 4. |
| 4 | Добавление записи в банк крови | 1 Выбрать пункт меню «Банк крови».  2 Нажать кнопку «Добавить».  3 Ввести данные (уникальный ИД донора, номер контейнера, срок хранения, индефикация, статус, группа крови и резус фактор)  4 Нажать кнопку «Сохранить данные». | Запись о крови успешно добавлена в базу данных с банком крови. |
| Совпадает с ожидаемым результатом.  Результат представлен в приложении Б на рисунке Б. 5. |

Продолжение таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название | Шаги воспроизведения | Ожидаемый результат |
| Фактический результат |
| 5 | Добавление записи в штат сотрудников | 1 Выбрать пункт меню «Врач».  2 Нажать кнопку «Добавить».  3 Ввести данные (ФИО, пол, возраст, образование, стаж работы, номер телефона).  4. Нажать кнопку «Сохранить данные». | Запись о сотруднике успешно добавлена в базу данных врачей. |
| Совпадает с ожидаемым результатом.  Результат представлен в приложении Б на рисунке Б. 6. |
| 6 | Добавление записи в отдел заготовки крови. | 1 Выбрать пункт меню «Отдел заготовки крови».  2 Нажать кнопку «Добавить».  3 Ввести данные (уникальные ИД донора, код донора из базы, кол-во сданной крови, код врача, дата опломбировки контейнера).  4 Нажать кнопку «Сохранить данные». | Запись в отдел заготовки крови успешно добавлена в базу. |
| Совпадает с ожидаемым результатом.  Результат представлен в приложении Б на рисунке Б. 7. |
| 7 | Добавление записи в базу доноров | 1 Выбрать пункт меню «База доноров».  2 Нажать кнопку «Добавить».  3 Ввести данные (код донора, общее кол-во сданной крови, дата последней сдачи, | Запись успешно добавлена в базу доноров. |
| Совпадает с ожидаемым результатом.  Результат представлен в приложении Б на рисунке Б. 8. |

Продолжение таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название | Шаги воспроизведения | Ожидаемый результат |
| Фактический результат |
|  |  | категория донора, повторная сдача, группа крови и резус фактор).  4 Нажать кнопку «Сохранить данные». |  |
| 8 | Фильтрация по группе крови и резус фактору | 1 Выбрать пункт меню «Учёт».  2 Выбрать пункт меню «Банк крови».  3 Выбрать группу крови и резус фактор. | Фильтр по группе крови и резус фактору успешно применён. |
| Совпадает с ожидаемым результатом.  Результат представлен в приложении Б на рисунке Б. 9. |

# Применение

## Назначение программы

Программа ведения статистики центра переливания крови предназначена для автоматизации учета и анализа данных о донорах и получателях крови. Она используется в медицинских учреждениях, специализирующихся на переливании крови. Основные задачи программы включают сбор и хранение информации о донорах и получателях, регистрацию и учет трансфузий, а также формирование отчетов и статистики.

Программа работает в автономном режиме, без необходимости подключения к интернету, что обеспечивает безопасность и конфиденциальность данных.

Ограничения области применения включают отсутствие сетевых функций и невозможность импорта и экспорта данных из внешних файлов.

## Условия применения

Для корректной работы программы нет необходимости в сторонних программных продуктах или средствах. Программа является полностью автономной и может переноситься без особого труда на разные компьютеры.

Также для работы программы у компьютера должны следующие минимальные аппаратные характеристики:

* процессор AMD A12-9720P, 2017 года выпуска, техпроцесс 28 нанометров, частота 2,6 ГГц и 3,6 ГГц в турбо режиме, 4 ядра, 4 потока, 64-х битный набор команд;
* оперативная память 4 ГБ DDR4 1866 ГГц;
* свободное место на жёстком диске 1 ГБ;
* операционная система MS Windows 10, 64-х разрядная;
* клавиатура;
* мышь.

Данные аппаратные характеристики основаны на примере компьютера, на котором велось тестирование приложения.

## Справочная система

Справочная система будет представлена в виде формы, которая будет открываться при нажатии на кнопку «О программе». Она была создана простым путем написания справки в текстовом окне поля. В поле будет идти речь о том, для чего данная программа создана.

Справочная система представлена на рисунке 28.

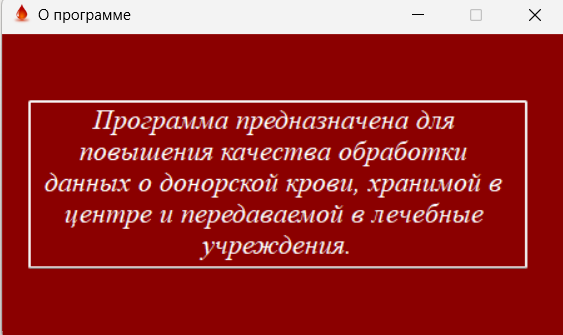


Рисунок 24 – Справочная форма

# Заключение

В заключение, можно сказать, что актуальность данной работы заключалась в недостатке автоматизации некоторых функций центра переливания крови – значительная часть работы производилась вручную, работниками центра.

По итогам создания курсового проекта на тему «Разработка приложения для ведения статистики центра переливая крови» было разработано программное средство «kranan.exe».

Основной и главной функцией программы является ведение статистики центра переливания крови.

Для достижения целей курсового проекта были решены следующие задачи:

* изучена предметная область;
* разработана физическая и логическая модель данных;
* разработано программное средство;
* описана область применения, созданного программное средство.

Интерфейс программы является минималистичным и интуитивно понятным.

В программном средстве все требования полностью реализованы и программное средство полностью отлаженным.

К особенностям программы можно отнести простоту интерфейса, удобство работы.

В процессе разработки программного продукта были закреплены и получены знания по уже известному материалу, отработаны навыки владения методами устойчивого программирования и эффективной разработки в MS Visual Studio Community 2022, с использованием языка программирования C#, Фреймворка Windows Form и языка запросов SQL.

Для проверки работоспособности модуля, было проведено тестирование.

# Список использованных источников

1. Багласова, Т.Г. Методические указания по оформлению курсовых проектов, дипломных проектов и отчетов для учащихся специальности 2-40 01 01 «Программное обеспечение информационных технологий» // Т.Г. Багласова, К.О. Якимович и др. – Минск : ЧУО «КБП» , 2023 . – 41 с.
2. Троелсен Э., Джепикс Ф. Язык программирования C# 7.0 и платформа .NET и .NET Core 2.0 / Э. Троелсен, Ф. Джепикс – М. : Вильямс, 2018. – 1400 с.
3. Шарп Д. С#. Программирование для профессионалов / Д. Шарп – СПб. : Питер, 2020. – 912 с.Мейерс, С. Эффективное использование С++. 55 верных способов улучшить структуру и код ваших программ / С. Мейерс. – М.: ДМК Пресс, 2006. – 300 с.
4. Обзор обновлений и новых функций Windows 11 [Электронный ресурс]. – Microsoft, 2020 . – Режим доступа: https://www.microsoft.com/ru-ru/windows/features. – Дата доступа: 20.06.2024.
5. Общие требования к тестовым документам: ГОСТ 2.105-95. – Введ. 01.01.1996. – Минск : Межгос. совет по стандартизации, метрологии и сертификации, 1995 . – 84 с
6. Объектно-ориентированный анализ и проектирование с примерами приложений / Гради Буч [и др.]. – 3-е изд. – М.: ООО «И.Д. Вильямс», 2008 . – 720 с.
7. Описание программы. Требования к содержанию, оформлению и контролю качества: ГОСТ 19.402-2000. – Введ. 01.09.2001. – Минск: Межгос. совет по стандартизации, метрологии и сертификации, 2000 . – 14 с.
8. Программа и методика испытаний. Требования к содержанию, оформлению и контролю качества: ГОСТ 19.301-2000. – Введ. 01.09.2001. – Минск: Межгос. совет по стандартизации, метрологии и сертификации, 2000. – 14 с.
9. Текст программы. Требования к содержанию, оформлению и контролю качества: ГОСТ 19.401-2000. – Введ. 01.09.2001. – Минск: Межгос. совет по стандартизации, метрологии и сертификации, 2000 . – 16 с.
10. Чарльз П. Visual C# 2019 и платформа .NET 4.8 для профессионалов. Полное руководство / П. Чарльз – М. : Диалектика, 2019. – 1536 с.
11. Макмиллан М. Программирование Windows Forms на C# / М. Макмиллан – СПб. : Питер, 2021. – 720 с.
12. Бен-Ганн И. SQL для профессионалов / И. Бен-Ганн – М. : Вильямс, 2019. – 864 с.
13. Гриффитс Р. Изучаем SQL / Р. Гриффитс – М. : Диалектика, 2020. – 624 с.

# Приложение А

**(обязательное)**

**Текст программы**

//Form1.cs

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 kranan

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "database1DataSet.Vrach". При необходимости она может быть перемещена или удалена.

this.vrachTableAdapter.Fill(this.database1DataSet.Vrach);

// TODO: данная строка кода позволяет загрузить данные в таблицу "database1DataSet.BankKrov". При необходимости она может быть перемещена или удалена.

this.bankKrovTableAdapter.Fill(this.database1DataSet.BankKrov);

// TODO: данная строка кода позволяет загрузить данные в таблицу "database1DataSet.Zagotovka". При необходимости она может быть перемещена или удалена.

this.zagotovkaTableAdapter.Fill(this.database1DataSet.Zagotovka);

// TODO: данная строка кода позволяет загрузить данные в таблицу "database1DataSet.DonorBase". При необходимости она может быть перемещена или удалена.

this.donorBaseTableAdapter.Fill(this.database1DataSet.DonorBase);

// TODO: данная строка кода позволяет загрузить данные в таблицу "database1DataSet.Donor". При необходимости она может быть перемещена или удалена.

this.donorTableAdapter.Fill(this.database1DataSet.Donor);

panel1.Visible = false;

panel2.Visible = false;

panel3.Visible = false;

panel4.Visible = false;

panel5.Visible = false;

label1.Visible = true;

}

private void Button1\_Click(object sender, EventArgs e)

{

Form2 f = new Form2();

f.ShowDialog();

}

private void Button2\_Click(object sender, EventArgs e)

{

button7.BackColor = Color.IndianRed;

button5.BackColor = Color.IndianRed;

button2.BackColor = Color.Red;

button10.BackColor = Color.LightCoral;

button15.BackColor = Color.LightCoral;

panel1.Visible = true;

panel2.Visible = false;

panel3.Visible = false;

panel4.Visible = false;

panel5.Visible = false;

label1.Visible = false;

BackgroundImage = Properties.Resources.Донор;

}

private void Button3\_Click(object sender, EventArgs e)

{

this.donorTableAdapter.Fill(this.database1DataSet.Donor);

}

private void Button5\_Click(object sender, EventArgs e)

{

button7.BackColor = Color.IndianRed;

button5.BackColor = Color.Red;

button2.BackColor = Color.LightCoral;

button10.BackColor = Color.LightCoral;

button15.BackColor = Color.LightCoral;

panel2.Visible = true;

panel1.Visible = false;

panel3.Visible = false;

panel4.Visible = false;

panel5.Visible = false;

label1.Visible = false;

BackgroundImage = Properties.Resources.Заполнение\_базы\_доноров;

}

private void Button4\_Click(object sender, EventArgs e)

{

Form3 f = new Form3();

f.ShowDialog();

}

private void Button6\_Click(object sender, EventArgs e)

{

this.donorBaseTableAdapter.Fill(this.database1DataSet.DonorBase);

}

private void Button10\_Click(object sender, EventArgs e)

{

button7.BackColor = Color.IndianRed;

button5.BackColor = Color.IndianRed;

button2.BackColor = Color.LightCoral;

button10.BackColor = Color.Red;

button15.BackColor = Color.LightCoral;

panel2.Visible = false;

panel4.Visible = false;

panel1.Visible = false;

panel3.Visible = true;

panel5.Visible = false;

label1.Visible = false;

BackgroundImage = Properties.Resources.Отдел\_заготовки\_крови;

}

private void Button8\_Click(object sender, EventArgs e)

{

Form4 f = new Form4();

f.ShowDialog();

}

private void Button9\_Click(object sender, EventArgs e)

{

this.zagotovkaTableAdapter.Fill(this.database1DataSet.Zagotovka);

}

private void Button7\_Click(object sender, EventArgs e)

{

button7.BackColor = Color.Red;

button5.BackColor = Color.IndianRed;

button2.BackColor = Color.LightCoral;

button10.BackColor = Color.LightCoral;

button15.BackColor = Color.LightCoral;

panel2.Visible = false;

panel4.Visible = true;

panel1.Visible = false;

panel3.Visible = false;

panel5.Visible = false;

label1.Visible = false;

BackgroundImage = Properties.Resources.Заполение\_банка\_крови1;

}

private void Button12\_Click(object sender, EventArgs e)

{

Form5 f = new Form5();

f.ShowDialog();

}

private void Button13\_Click(object sender, EventArgs e)

{

this.bankKrovTableAdapter.Fill(this.database1DataSet.BankKrov);

}

private void Button15\_Click(object sender, EventArgs e)

{

button7.BackColor = Color.IndianRed;

button5.BackColor = Color.IndianRed;

button2.BackColor = Color.LightCoral;

button10.BackColor = Color.LightCoral;

button15.BackColor = Color.Red;

panel2.Visible = false;

panel4.Visible = false;

panel1.Visible = false;

panel3.Visible = false;

panel5.Visible = true;

label1.Visible = false;

BackgroundImage = Properties.Resources.Штат\_сотрудников;

}

private void Button11\_Click(object sender, EventArgs e)

{

Form6 f = new Form6();

f.ShowDialog();

}

private void Button14\_Click(object sender, EventArgs e)

{

this.vrachTableAdapter.Fill(this.database1DataSet.Vrach);

}

private void Button17\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void Button18\_Click(object sender, EventArgs e)

{

Form7 f = new Form7();

f.ShowDialog();

}

private void Button16\_Click(object sender, EventArgs e)

{

Form8 f = new Form8();

f.ShowDialog();

}

}

}

//Form2.cs

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 kranan

{

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

}

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

{

this.Validate();

this.donorBindingSource.EndEdit();

this.tableAdapterManager.UpdateAll(this.database1DataSet);

}

private void Form2\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "database1DataSet.Donor". При необходимости она может быть перемещена или удалена.

this.donorTableAdapter.Fill(this.database1DataSet.Donor);

// TODO: данная строка кода позволяет загрузить данные в таблицу "database1DataSet.Donor". При необходимости она может быть перемещена или удалена.

this.donorTableAdapter.Fill(this.database1DataSet.Donor);

}

private void DonorBindingNavigatorSaveItem\_Click\_1(object sender, EventArgs e)

{

this.Validate();

this.donorBindingSource.EndEdit();

this.tableAdapterManager.UpdateAll(this.database1DataSet);

}

private void Button1\_Click(object sender, EventArgs e)

{

Close();

}

}

}

//Form3.cs

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 kranan

{

public partial class Form3 : Form

{

public Form3()

{

InitializeComponent();

}

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

{

this.Validate();

this.donorBaseBindingSource.EndEdit();

this.tableAdapterManager.UpdateAll(this.database1DataSet);

}

private void Form3\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "database1DataSet.Donor". При необходимости она может быть перемещена или удалена.

this.donorTableAdapter.Fill(this.database1DataSet.Donor);

// TODO: данная строка кода позволяет загрузить данные в таблицу "database1DataSet.DonorBase". При необходимости она может быть перемещена или удалена.

this.donorBaseTableAdapter.Fill(this.database1DataSet.DonorBase);

}

private void Button1\_Click(object sender, EventArgs e)

{

Close();

}

}

}

//Form4.cs

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 kranan

{

public partial class Form4 : Form

{

public Form4()

{

InitializeComponent();

}

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

{

this.Validate();

this.zagotovkaBindingSource.EndEdit();

this.tableAdapterManager.UpdateAll(this.database1DataSet);

}

private void Form4\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "database1DataSet.Vrach". При необходимости она может быть перемещена или удалена.

this.vrachTableAdapter.Fill(this.database1DataSet.Vrach);

// TODO: данная строка кода позволяет загрузить данные в таблицу "database1DataSet.DonorBase". При необходимости она может быть перемещена или удалена.

this.donorBaseTableAdapter.Fill(this.database1DataSet.DonorBase);

// TODO: данная строка кода позволяет загрузить данные в таблицу "database1DataSet.Zagotovka". При необходимости она может быть перемещена или удалена.

this.zagotovkaTableAdapter.Fill(this.database1DataSet.Zagotovka);

}

private void Button1\_Click(object sender, EventArgs e)

{

Close();

}

}

}

//Form5.cs

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 kranan

{

public partial class Form5 : Form

{

public Form5()

{

InitializeComponent();

}

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

{

this.Validate();

this.bankKrovBindingSource.EndEdit();

this.tableAdapterManager.UpdateAll(this.database1DataSet);

}

private void BankKrovBindingNavigatorSaveItem\_Click\_1(object sender, EventArgs e)

{

this.Validate();

this.bankKrovBindingSource.EndEdit();

this.tableAdapterManager.UpdateAll(this.database1DataSet);

}

private void Form5\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "database1DataSet.Zagotovka". При необходимости она может быть перемещена или удалена.

this.zagotovkaTableAdapter.Fill(this.database1DataSet.Zagotovka);

// TODO: данная строка кода позволяет загрузить данные в таблицу "database1DataSet.BankKrov". При необходимости она может быть перемещена или удалена.

this.bankKrovTableAdapter.Fill(this.database1DataSet.BankKrov);

}

private void Button1\_Click(object sender, EventArgs e)

{

Close();

}

}

}

//Form6.cs

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 kranan

{

public partial class Form6 : Form

{

public Form6()

{

InitializeComponent();

}

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

{

this.Validate();

this.vrachBindingSource.EndEdit();

this.tableAdapterManager.UpdateAll(this.database1DataSet);

}

private void Form6\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "database1DataSet.Vrach". При необходимости она может быть перемещена или удалена.

this.vrachTableAdapter.Fill(this.database1DataSet.Vrach);

}

private void Button1\_Click(object sender, EventArgs e)

{

Close();

}

}

}

//Form7.cs

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 kranan

{

public partial class Form7 : Form

{

public Form7()

{

InitializeComponent();

}

private void Button2\_Click(object sender, EventArgs e)

{

Close();

}

private void Button7\_Click(object sender, EventArgs e)

{

button1.BackColor = Color.LightCoral;

button7.BackColor = Color.Red;

panel2.Visible = true;

panel3.Visible = false;

dataGridView1.DataSource = bankKrovBindingSource;

}

private void Button1\_Click(object sender, EventArgs e)

{

dataGridView2.DataSource = donorBaseBindingSource;

button7.BackColor = Color.LightCoral;

button1.BackColor = Color.Red;

panel2.Visible = false;

panel3.Visible = true;

}

private void Form7\_Load(object sender, EventArgs e)

{

this.selectPoslDataSdashTableAdapter.Fill(this.database1DataSet7.SelectPoslDataSdash);

this.selectObshKolVoKrovTableAdapter.Fill(this.database1DataSet7.SelectObshKolVoKrov);

this.selectKaterDonoraTableAdapter.Fill(this.database1DataSet7.SelectKaterDonora);

this.selectB4TableAdapter.Fill(this.database1DataSet7.SelectB4);

this.selectB3TableAdapter.Fill(this.database1DataSet7.SelectB3);

this.selectB2TableAdapter.Fill(this.database1DataSet7.SelectB2);

this.selectB1TableAdapter.Fill(this.database1DataSet7.SelectB1);

this.donorBaseTableAdapter.Fill(this.database1DataSet6.DonorBase);

this.selectSrokTableAdapter.Fill(this.database1DataSet4.SelectSrok);

this.selectQueryIdfTableAdapter.Fill(this.database1DataSet4.SelectQueryIdf);

this.selectQueryOrderByTableAdapter.Fill(this.database1DataSet4.SelectQueryOrderBy);

this.selectBank4TableAdapter.Fill(this.database1DataSet1.SelectBank4);

this.selectBank3TableAdapter.Fill(this.database1DataSet1.SelectBank3);

this.selectBank2TableAdapter.Fill(this.database1DataSet1.SelectBank2);

this.selectBankTableAdapter.Fill(this.database1DataSet.SelectBank);

this.bankKrovTableAdapter.Fill(this.database1DataSet.BankKrov);

this.bankKrovTableAdapter.Fill(this.database1DataSet.BankKrov);

this.bankKrovTableAdapter.Fill(this.database1DataSet.BankKrov);

panel2.Visible = false;

panel3.Visible = false;

}

private void Button3\_Click(object sender, EventArgs e)

{

dataGridView1.DataSource = selectBankBindingSource;

}

private void Button4\_Click(object sender, EventArgs e)

{

dataGridView1.DataSource = selectBank2BindingSource;

}

private void Button5\_Click(object sender, EventArgs e)

{

dataGridView1.DataSource = selectBank3BindingSource;

}

private void Button6\_Click(object sender, EventArgs e)

{

dataGridView1.DataSource = selectBank4BindingSource;

}

private void Button8\_Click(object sender, EventArgs e)

{

dataGridView1.DataSource = selectQueryOrderByBindingSource;

}

private void Button9\_Click(object sender, EventArgs e)

{

dataGridView1.DataSource = selectQueryIdfBindingSource;

}

private void Button10\_Click(object sender, EventArgs e)

{

dataGridView1.DataSource = selectSrokBindingSource;

}

private void Button11\_Click(object sender, EventArgs e)

{

dataGridView1.DataSource = bankKrovBindingSource;

}

private void Button19\_Click(object sender, EventArgs e)

{

dataGridView2.DataSource = donorBaseBindingSource;

}

private void Button15\_Click(object sender, EventArgs e)

{

dataGridView2.DataSource = selectB1BindingSource;

}

private void Button14\_Click(object sender, EventArgs e)

{

dataGridView2.DataSource = selectB2BindingSource;

}

private void Button13\_Click(object sender, EventArgs e)

{

dataGridView2.DataSource = selectB3BindingSource;

}

private void Button12\_Click(object sender, EventArgs e)

{

dataGridView2.DataSource = selectB4BindingSource;

}

private void Button18\_Click(object sender, EventArgs e)

{

dataGridView2.DataSource = selectKaterDonoraBindingSource;

}

private void Button17\_Click(object sender, EventArgs e)

{

dataGridView2.DataSource = selectPoslDataSdashBindingSource;

}

private void Button16\_Click(object sender, EventArgs e)

{

dataGridView2.DataSource = selectObshKolVoKrovBindingSource;

}

private void Button20\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void Button21\_Click(object sender, EventArgs e)

{

Form8 f = new Form8();

f.ShowDialog();

}

}

}

//Form8.cs

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 kranan

{

public partial class Form8 : Form

{

public Form8()

{

InitializeComponent();

}

private void Button1\_Click(object sender, EventArgs e)

{

Close();

}

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

{

}

}

}

//Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace kranan

{

static class Program

{

/// <summary>

/// Главная точка входа для приложения.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

# Приложение Б

**(справочное)**

**Результат работы программы**

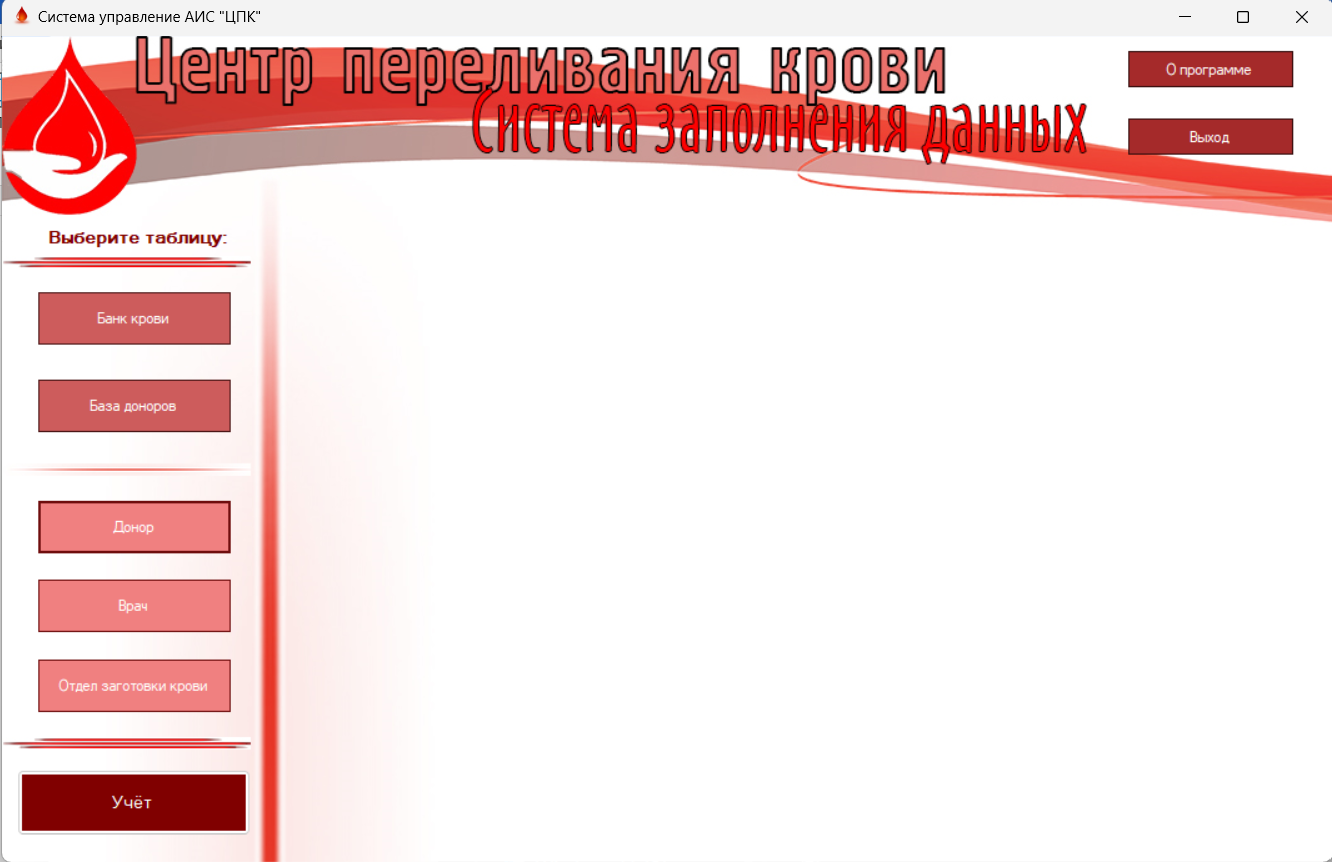


Рисунок Б. 1 – Результат функции «Запустить приложение»

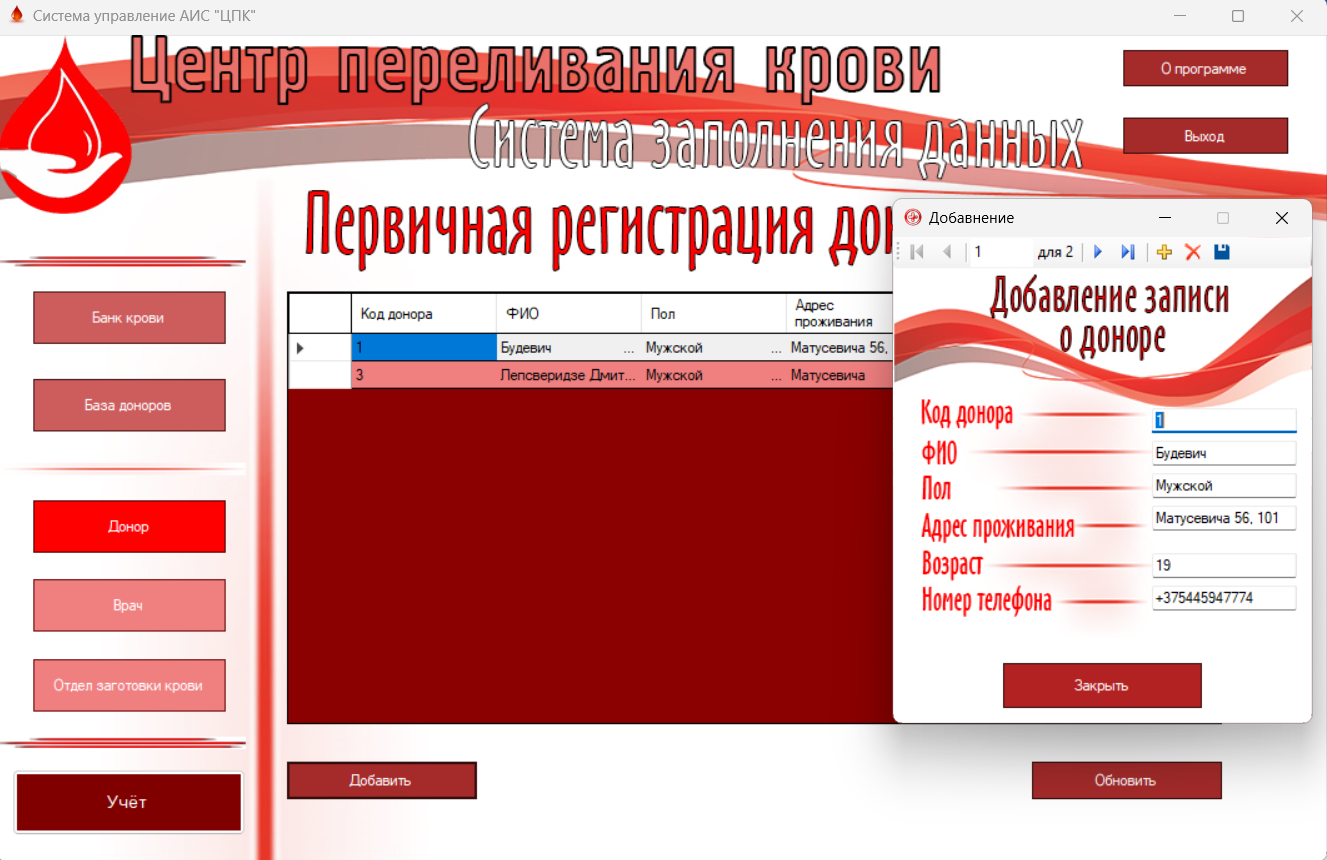


Рисунок Б. 2 – Результат функции «Регистрации донора»



Рисунок Б. 3 – Результат функции «Фильтрация данный по статусу»

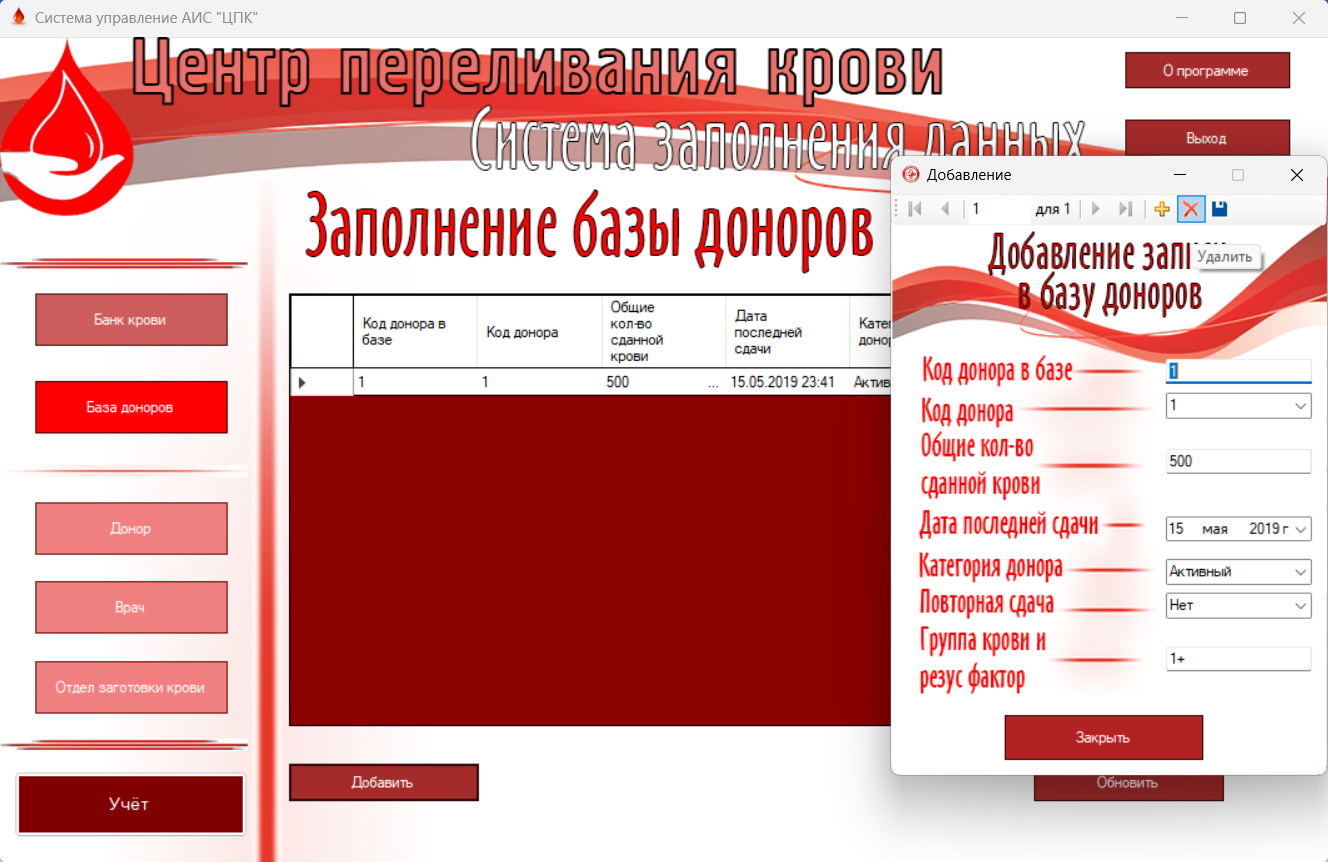


Рисунок Б. 4 – Результат функции «Удаление донора из базы данных»

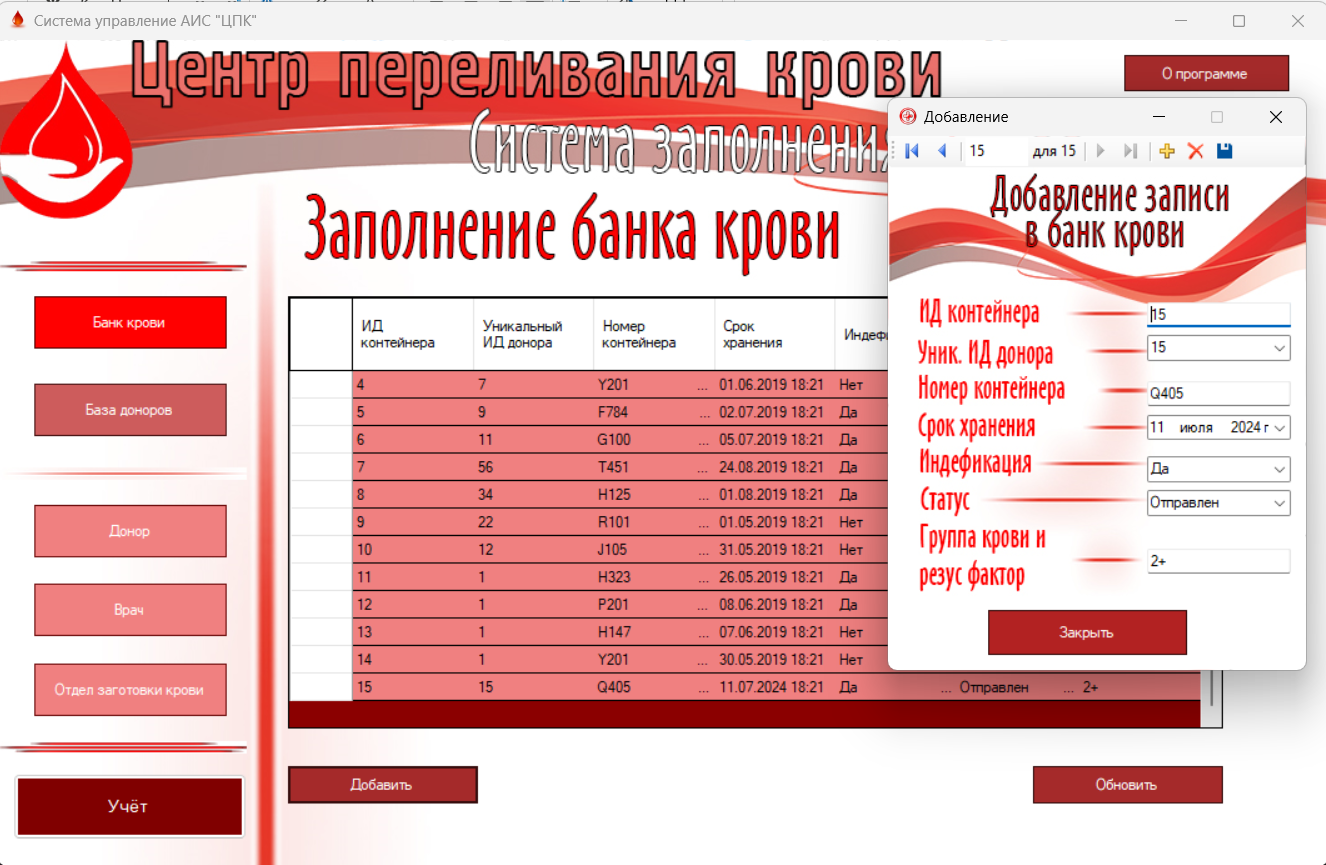


Рисунок Б. 5 – Результат функции «Добавление записи в банк крови»



Рисунок Б. 6 – Результат функции «Добавление записи в штат сотрудников»

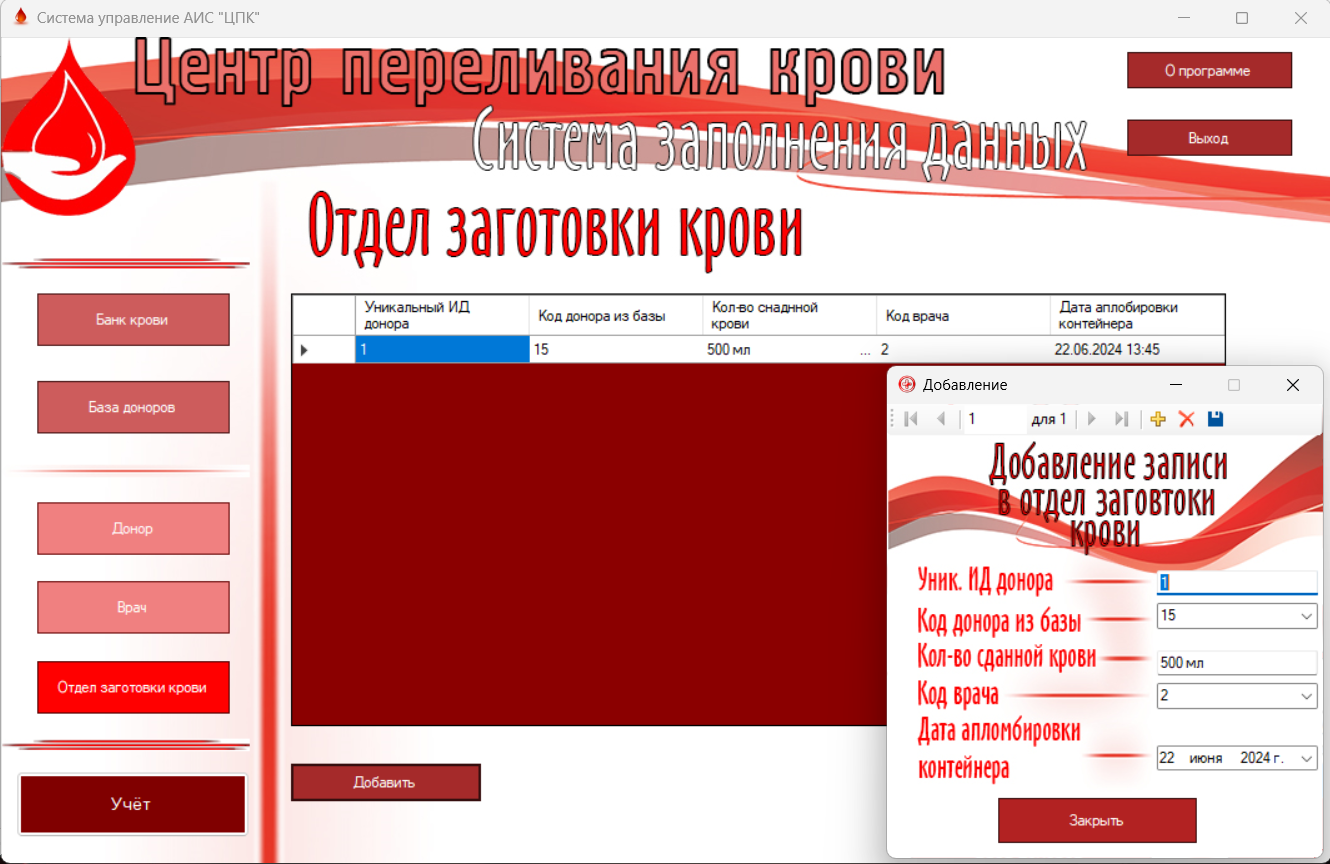


Рисунок Б. 7 – Результат функции «Добавление записи в отдел заготовки крови»

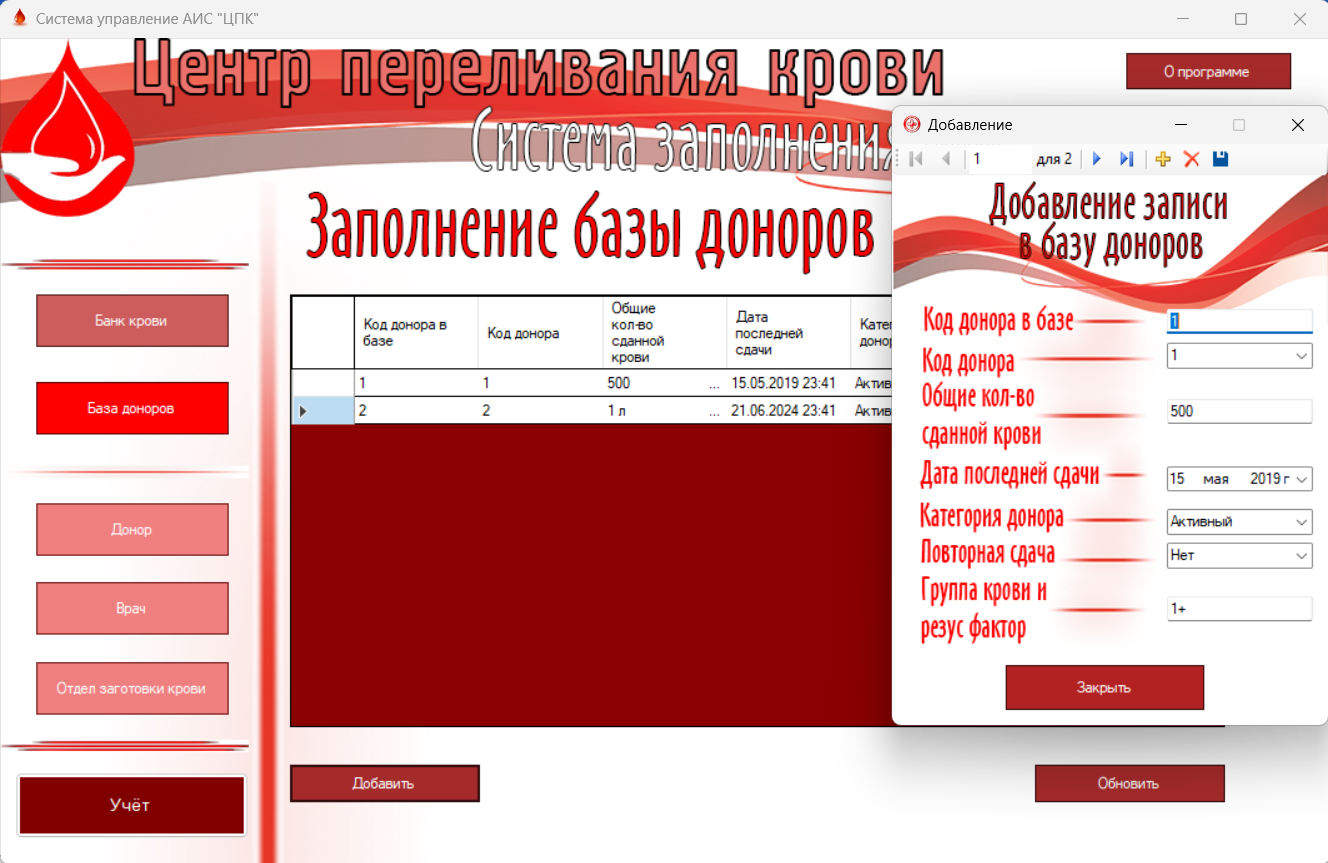


Рисунок Б. 8 – Результат функции «Добавление записи в базу доноров»



Рисунок Б. 9 – Результат функции «Фильтрация по группе крови и резус фактору»

**Удостоверяющий лист**

электронного документа – курсовой проект

Тема «Разработка приложения для ведения статистики центра переливая крови».

Обозначение КП Т.219013.401

Разработчик Дирко К.Ю. Руководитель Багласова Е.В.

(Ф.И.О.) (Ф.И.О.)

Подписи лиц, ответственных за разработку электронного документа

|  |  |  |
| --- | --- | --- |
| Состав электронного документа | Разработчик | Руководитель |
| Пояснительная записка (на бумажном носителе формата А4), Пояснительная записка(основная).docх |  |  |
| Папка с проектом «Курсовая» |  |  |
| Графическая часть.docx |  |  |
| Тип носителя: USB-флеш-накопитель |  |  |

**Этикетка**

для курсового проектов

**Курсовой проект**

Тема «Разработка приложения для ведения статистики центра переливая крови»

КП Т.219013.401.

Разработан

Утвержден

Разработчик Лепсверидзе Д.З.

Руководитель: Багласова Е.В.

Технические средства: Компьютер с процессором AMD A12-9720P от 2.6 ГГц, ОЗУ от 4 ГБ, 1 ГБ свободного места на диске, монитор, клавиатура, мышка

Программные средства: Операционная система MS Windows 10 Pro, MS Visual Studio 2022, текстовые редакторы Microsoft Word 2021, Notepad.exe

**Состав документа:**

Пояснительная записка – Пояснительная записка(основная).docх

Графическая часть – Графическая часть.docx

Папка с проектом – Курсовая

Сведение о защите информации: пароль отсутствует