Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Институт естественных и точных наук
Факультет математики, механики и компьютерных технологий
Кафедра прикладной математики и программирования
Направление подготовки Программная инженерия

РАБОТА ПРОВЕРЕНА
Рецензент,

_____/С.А.Загребина
«____»_____2017г.

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой, д.ф.-м.н., доцент
_____/А.А.Замышляева
«____»_____2017 г.

Информационная система мониторинга эффективности реабилитации пациентов после коронарного шунтирования

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ–09.03.04.2017.57.ПЗ ВКР

Консультант, ст. преп.
_____/М.Ю.Сартасова
«____»_____2017 г.

Руководитель работы, д.ф.-м.н., доцент
_____/А.В.Келлер
«____»_____2017 г.

Автор работы
Студент группы ЕТ-484
_____/ Г.И.Большакова
«____»_____2017 г.

Нормоконтролер, доцент
_____/Т.Ю. Оленчикова
«____»_____2017 г.

Челябинск 2017

АННОТАЦИЯ

Большакова Г. И. Разработка информационной системы мониторинга эффективности реабилитации пациентов после коронарного шунтирования. – Челябинск: ЮУрГУ, ЕТ-484, 54 с., 34 ил., 11 табл., библиогр. список – 21 наим., 1 прил.

Данная работа посвящена разработке информационной системы мониторинга эффективности реабилитации пациентов после коронарного шунтирования.

В работе выполнен обзор наиболее используемых методов анализа медицинских данных.

Разработана математическая модель, база данных и алгоритмы работы приложения. Реализован графический интерфейс пользователя. Разработано и отлажено приложение для мониторинга эффективности реабилитации пациентов после коронарного шунтирования.

Программа реализована на языке программирования C++, СУБД Access. В приложении приведён текст программы.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В последние годы, статистические методы в медицинских исследованиях представляют все больший научный интерес. Почти все принципы диагностики и лечения определяются статистическим анализом данных наблюдений, сделанных практикующими врачами. Поэтому анализ медицинских данных является одним из наиболее важных факторов, влияющих на уровень современной медицинской помощи [16].

Характерной особенностью внедрения компьютеров в медицинских учреждениях является применение статистических методов для обоснования гипотез, аргументированного формирования выборок и построения математических моделей различных диагностических методик.

Существует несколько важных факторов, которые усложняют разработку информационной системы:

- исследуемые признаки имеют свойство постоянно меняться по причине значительного числа факторов, не поддающихся контролю;
- существуют существенные затруднения в составлении выборок необходимого размера и структуры;
- проблемы в изучении методов статистического анализа работниками, которые не владеют специальными математическими знаниями.

Статистическая обработка медицинских исследований базируется на утверждении, что истинное для случайной выборки истинно и для генеральной совокупности (популяции), из которой эта выборка получена. Выбрать или набрать истинно случайную выборку из генеральной совокупности очень сложно, следовательно, необходимо стремиться к тому, чтобы выборка была репрезентативной по отношению к изучаемой популяции.

Многочисленные медицинские данные дают предпосылки программистам для предложения решений эффективных способов хранения, обработки и обмена данными. Важнейшей задачей является получение как можно большей информации из имеющихся данных.

Существует 2 основных пути применения анализа данных:

- описательный анализ, который чаще всего используется для группировки пациентов с похожими синдромами и нахождения важных характеристик каждого заболевания;
- прогностический анализ, который используется для выявления правил классификации, которые могут предсказать дальнейшее течение заболевания.

В большинстве случаев, принципы диагностики и лечения определяются статистическим анализом медицинских данных.

На данный момент в нашей стране функционирует несколько федеральных кардиологических центров, которые оснащены передовым оборудованием и центрами постоперационной реабилитации, но все еще не хватает информационных систем для мониторинга эффективности проведенного лечения

и построения прогноза на выздоровление. Кардиоцентры вынуждены самостоятельно вести статистику и проводить анализ данных.

В реабилитационном центре МБУЗ ГКБ № 2 данные фиксировались в бумажном виде, что существенно усложняло, а порой и делало невозможным, проведение какого-либо анализа с большими объемами данных.

Разработка данной медицинской информационной системы поможет увеличить уровень медицинского обслуживания за счет возможности анализа большего объема данных и предоставит возможность оперативного сбора и анализа медицинской статистики.

Целью данной работы является разработка информационной системы мониторинга эффективности реабилитации пациентов после коронарного шунтирования для МБУЗ ГКБ №2.

Для достижения поставленной цели необходимо решить следующие задачи:

- выполнить обзор существующих методов анализа данных;
- разработать математическую модель для выбранного метода анализа данных;
- разработать и реализовать алгоритмы работы приложения;
- разработать базу данных;
- реализовать пользовательский интерфейс;
- выполнить проверку работы приложения.

1 МЕТОДЫ СТАТИСТИЧЕСКОГО АНАЛИЗА КОЛИЧЕСТВЕННЫХ И КАЧЕСТВЕННЫХ МЕДИЦИНСКИХ ДАННЫХ ПАЦИЕНТОВ ГОРОДСКОГО ОТДЕЛЕНИЯ РЕАБИЛИТАЦИИ И КАРДИОЛОГИИ

1.1 Виды статистических данных в медицине

По причине того, что объем данных и выборок могут значительно отличаться, а данные, в свою очередь, могут являться не однородными, обнаруживается нужда в использовании методов статистического анализа, соответствующих поставленной задаче.

Статистические данные делятся на количественные (числовые непрерывные или дискретные) и качественные (категориальные порядковые или номинальные) переменные. При работе над заполнением БД крайне важно правильно указывать тип переменной, по причине того, что в дальнейшем от этого зависит сам процесс обработки [22, с. 5].

Количественные данные подразумевают, что переменная является каким-либо числовым значением. Они дают возможность как упорядочивать единицы, так и задавать интервал, который будет отделять одну единицу от другой. Количественные данные могут измеряться в различных шкалах:

- интервальная шкала, например, указание промежутка веса (от 55 до 70 кг.);
- относительная шкала, например, точный вес человека (60,48 кг.).

Качественные данные делятся на 3 группы: категориальные, номинальные и порядковые.

Категориальные данные используются для описания состояния объекта. Каждому объекту дается свой номер, который определяет принадлежность какой-либо категории. Нужно отметить тот факт, что для использования категориальных данных один объект исследования может принадлежать только к одной возможной категории для одного критерия.

Номинальные данные используются, когда категории не упорядочены. Они отражают условные коды неизмеримых категорий, например, коды групп препаратов[3]. Используемые коды состояния объекта никак не упорядочивают состояние. Например, по полю 1–женат(замужем), 2–не женат (не замужем).

Порядковые данные – данные, которые отражают степень выраженности какого-либо признака. Их можно ранжировать и использовать при проведении статистического анализа. Например, упорядочивание данных влияния какого-либо препарата на самочувствие: 1– в лучшую сторону, 2 – в худшую сторону, 3 – никак не повлияло.

Порядковые данные отличаются от дискретных тем, что в них нет пропорциональной шкалы для измерения выраженности признака [20].

Для обработки результатов в научных исследованиях зачастую пользуются двумя типами статистического анализа данных – первичный (запланированный) и вторичный (незапланированный) [3].

Первичный анализ данных служит для ознакомления и описания закономерностей, которые (гипотезы) были выдвинуты исследователем [4]. Этот вид анализа используется для оценки параметров выборки и проверки гипотез, выдвинутых еще до начала сбора данных, на этапе планирования исследования.

Вторичный анализ данных служит для прогнозирования возможных закономерностей и гипотез, которые еще могут быть выдвинуты в дальнейшем. Именно результаты вторичного анализа данных служат обоснованием для их выдвижения. Ввиду того, что интерпретация результатов порождает систематические различия между сопоставляемыми группами, проводить ее нужно осторожно.

Главной составляющей любого анализа служит описательная статистика, основная задача которой – предоставлять сжатую и насыщенную характеристику изучаемого объекта или явления в графическом и числовом виде.

Для наглядного представления и анализа результатов исследовательской выборки, экспериментальной и контрольной группы, показатели описательной статистика подразделяются на несколько групп.

О нормальности распределения признака автоматически сообщают указание в представлении данных меры центральной тенденции (среднее, медиана, мода). Когда распределение является нормальным, то все три показателя примерно равны, а если распределение является ассиметричным, то наблюдается существенное различие показателей.

Мода (*Mo*) является самым часто встречающимся значением в выборке, либо средним значением класса с наибольшей частотой. Используется для создания общего представления о распределении. В случаях, когда распределение имеет 2 моды, то это является свидетельством бимодального распределения.

Медиана (*Me, Md)* соответствует центральному значению в последовательном ряду всех полученных значений или среднему значению наиболее часто встречающихся значений выборки. Медиана вместе с квартилями используется для представления дискретных переменных или количественных непрерывных переменных с ненормальным распределением.

Среднее арифметическое (*M*) – это показатель центральной тенденции, полученной делением суммы всех значений данных на число этих данных. Среднее арифметическое используется для представления количественных переменных с нормальным распределением. Среднее значение, как мера центральной тенденции в описательной статистике количественных данных, имеет одно из двух представлений.

Первое в виде «$M \pm S$», или как в зарубежной традиции *M(S),* где *M* – среднее, а *S* - стандартное отклонение. При нормальном распределении в диапазон $M \pm S$ укладывается примерно 68,3% всех значений признака.

Второе представление результатов – в виде «$M \pm m$», где *m* – стандартная ошибка среднего, определяемая следующим образом:

$$m = \frac{s}{\sqrt{n}}.$$

Однако подобная форма представления данных в медицине является малоинформативной. В медицине объектами наблюдения выступают сложные системы, значительно различающиеся по своим свойствам, что определяет практическое отсутствие истинного значения параметра. В действительности, в медицине определяется не точное значение, а диапазон, в который укладывается большинство значений исследуемого признака, то есть ширина распределения. Поэтому оптимальным описанием ширины распределения в медицинских исследованиях в настоящее время принимается представление 95% доверительного интервала с указанием нижней (5%) и верхней (95%) границы.

Доверительный интервал представляет собой диапазон значений, который с определенной исследователем вероятностью включает в себя настоящее популяционное значение.

Наиболее адекватная непараметрическая характеристика ширины – это квантили. Квантили представляют собой частоту попадания значений переменной в определенные интервалы. Чаще всего используется разделение на 10 (по 10%) или на 4 интервала (25%, 50%, 75%). При разделении на четыре квантиля для представления оценки центральной тенденции, ширины и асимметрии распределения результатов достаточно трех чисел: нижний квартиль (25%), 50% квартиль, который соответствует медиане, и верхний квартиль (75%). Подобный метод предоставления данных является одним из наиболее компактных и удобных.

1.2 Корреляционный и регрессионный анализ

На этапе анализа данных изучается исследование между переменными. С этой целью применяются корреляционный анализ, который устанавливает факт наличия или отсутствия зависимости между переменными (выражается в виде числового значения), а также регрессионный анализ (для нахождения количественной зависимости между переменными, выраженной в виде уравнения и/или графика), и часто (в последнее время) – факторный анализ.

1.2.1 Корреляционный анализ

Корреляция – взаимосвязь между двумя или более переменными (во втором случае она называется множественной или совокупной) [4]. Целью данного анализа является выявления наличия или отсутствия этой взаимосвязи. В случае, когда имеются две переменных, значения которых измерены в шкале отношений, используется коэффициент линейной корреляции Пирсона *r*, который принимает значения от -1 до +1, причем если значение находится ближе к 1, то это свидетельствует о наличии сильной связи, а если ближе к 0, то слабой. Отрицательный коэффициент корреляции означает наличие противоположной связи: чем выше значение одной переменной, тем ниже значение другой. Важно заметить, что значение коэффициента близкое к +1 или -1 ничего не говорит о причинно-следственных отношениях между ними.

Термин «линейный» свидетельствует о том, что исследуется наличие линейной связи между переменными.

Для переменных, принадлежащих к порядковой шкале или к интервальной шкале, но не подчиняющихся нормальному распределению, расчитывается ранговая корреляция по Спирману. Коэффициент корреляции Спирмена улавливает тенденцию – изменения переменных в одном направлении, который обозначается r_s и определяется сравнением рангов – номеров значений сравниваемых переменных в их упорядочении. По сравнению с коэффициентом Пирсона коэффициент корреляции Спирмена является менее чувствительным.

1.2.2 Регрессионный анализ

В отличие от корреляционного анализа, регрессионный анализ – не только говорит о наличии зависимостей между независимой переменной и одной или несколькими зависимыми переменными, но и позволяет определить вид этой связи и дает возможность для прогнозирования значения одной (зависимой) переменной, отталкиваясь от значения другой (независимой) переменной. Независимые переменные – это регрессоры или предикторы, а зависимые переменные называются критериальными. Терминология зависимых и независимых переменных отражает лишь математическую зависимость, а не причинно-следственные отношения.

Существует несколько видов линейного и нелинейного регрессионного анализа. Для проведения линейного регрессионного анализа зависимая переменная должна иметь интервальную (порядковую) шкалу. Бинарная логистическая регрессия выявляет зависимость дихотомической переменной от некой другой переменной, относящейся к любой шкале. Мультиномиальная логистическая регрессия является подходящим методом для зависимой переменной, которая называется категориальной с тремя и более категориями. Порядковую регрессию можно использовать, когда зависимые переменные относятся к порядковой шкале.

Нелинейные связи между переменными, которые относятся к интервальной шкале можно анализировать с помощью нелинейной регрессии. Но как линейный, так и нелинейный регрессионный анализ позволяет обнаруживать математическую зависимость между несколькими переменными.

1.2.2.1 Бинарная логистическая регрессия

С помощью бинарной логистической регрессии можно исследовать зависимость дихотомических (бинарных, имеющих только 2 категориальных значения) переменных от независимых переменных, имеющих любой вид шкалы. Как правило, в случае с дихотомическими переменными речь идет о некотором событии, которое может произойти или не произойти. Бинарная логистическая регрессия в таком случае рассчитывает вероятность наступления события в зависимости от значений независимых переменных с выводом коэффициентов регрессии для каждой такой переменной и ее статистической значимости.

1.2.2.2 Оценка адекватности модели бинарной логистической регрессии

Точность результатов расчета логистической регрессии полностью зависима от выборки, с помощью которой рассчитывались коэффициенты в уравнении логистической регрессии. Таким образом, созданная модель требует проверки ее адекватности.

Самым простым способом оценки адекватности модели является проверка этой модели на исходных данных и сравнение полученных данных с предварительно используемыми исходами. Оценка выражается как процент наблюдений, верно предсказанными с помощью модели регрессии.

Проверка значимости отличия коэффициентов от 0 проводится при помощи статистики Вальда, которая использует распределение Хи-квадрат и представляет собой квадрат отношения соответствующего коэффициента к его стандартной ошибке [22].

1.2.2.3 Мультиномиальная логистическая регрессия

Этот метод является вариантом логистической регрессии, при которой зависимая переменная не является дихотомической, а имеет больше двух категорий. В то время как, при бинарной логистической регрессии независимая переменная может иметь интервальную шкалу, то мультиномиальная логистическая регрессия подходит только для категориальных независимых переменных, причем имеет значение, относятся ли они к шкале наименований или к порядковой шкале. Возможно задание в качестве ковариат переменных, которые имеют неправильную шкалу.

Для построения мультиномиальной логистической регрессии формируется n недублированных логитов для $n+1$ возможных значений независимой переменной, причем одна категория используется как эталонная, ее коэффициенты принимаются равными 0:

$$g_1 = \ln\frac{p_1}{p_n} = b_{10} + b_{11} + \ldots + b_{1|n-2|},$$

$$g_2 = \ln\frac{p_2}{p_n} = b_{20} + b_{21} + \ldots + b_{2|n-2|}, \quad \cdots \quad g_n = 0.$$

Нахождение коэффициентов $b_{10}, b_{11}, b_{20}, u\, b_{21}$ (называемых параметрическими оценками) является основной задачей мультиномиальной логистической регрессии.

1.2.2.4 Регрессия Кокса

Регрессия Кокса (модель пропорциональных рисков) - математическое представление и графическое построение в виде коэффициентов регрессионного уравнения, экспонент коэффициентов (отношения шансов) риска наступления события как функции, зависящей от времени, и оценки влияния каждой из независимых переменных на этот риск.

Риск наступления события измеряет правдоподобие наступления события в ближайшем будущем для тех, кто еще находится в группе риска. Риск

наступления события равен предельному значению условий вероятности наступления события во временном промежутке [*t, t + dt*] для объектов, еще оставшихся в группе риска на момент времени t, деленному на длину временного интервала *dt*.

Метод Кокса не рассматривает зависимость риска от времени. Последнее происходит из предположения о пропорциональности рисков. Чтобы ослабить это предположение, используются ковариванты, зависящие от времени.

1.3 Снижение размерности

В медицинских исследованиях существует большое количество связей, и в их числе присутствуют статистически не значимые. Более того, невозможно предположить сколько из них будут значимы для поставленной цели исследования. В большинстве случаев, их оказывается 5-25% от всех возможных связей.

Между многими показателями зависимости выражены в разной степени. Если их использовать в качестве исходного подмножества признаков, можно поставить и решить задачу конструирования на их основе более сложных признаков с помощью факторного анализа. Число полученных комплексных признаков (индексов) будет значительно меньше, чем исходное число. В результате мы получим новые признаки, которые в совокупности несут в себе гораздо больший объем информации, чем каждый из них в отдельности. В итоге можно отфильтровать случайную составляющую и получить более достоверную информацию о структуре исходных признаков и исследуемых групп пациентов.

1.3.1 Факторный анализ

Факторный анализ - это процедура, с помощью которой большое количество переменных сводят к меньшему числу независимых влияющих величин, называемых факторами (факторными комплексами, компонентами). При этом в один фактор объединяются переменные, сильно коррелирующие между собой. Переменные из разных факторов слабо коррелируют между собой. Таким образом, целью факторного анализа является нахождение таких комплексных факторов, которые как можно более полно объясняют наблюдаемые связи между переменными, имеющимися в наличии.

Сильная зависимость между двумя разными переменными дает понять об избыточности двух пунктов исследования. Ее между переменными можно обнаружить с помощью диаграммы рассеяния. Полученная линия регрессии путем аппроксимации (подгонки) дает графическое представление зависимости. Переменная будет включить в себя наиболее существенные черты обеих переменных, если определить новую переменную на основе линии регрессии, изображенной на этой диаграмме. Новый фактор в действительности является линейной комбинацией двух исходных переменных.

1.4 Классификация как важная задача статистического анализа

Важной задачей статистического анализа данных является классификация. Принято выделять три подобласти теории классификации: группировка, дискриминация (дискриминантный анализ) и кластеризация (кластерный анализ).

1.4.1 Группировка

Часто запланированная группировка служит подходом к классификации объектов, при этом сам исследователь разделяет на группы. На этом принципе основаны когортные исследования.

Когортное исследование – это обсервационное исследование, в котором выделенную группу людей (когорту) наблюдают в течение некоторого времени. Исходы у испытуемых в разных подргруппах данной когорты сравниваются. В проспективном когортном исследовании когорты составляют в настоящем и наблюдают их в будущем. В ретроспективном (или историческом) когортном исследовании когорту подбирают по архивным записям и прослеживают их исходы с того момента по настоящее время.

Когортные подразделяются на фиксированные и динамические. В фиксированной когорте число пациентов остается постоянным, в случае, если пациенты покидают фиксированную когорту, то их не заменяют. В динамических когортах возможно, как исключение, так и включение новых пациентов в когорту.

Главным недостатком когортных исследований является их высокая стоимость, вызванная как необходимостью формирования больших выборок, так и длительностью наблюдения за больными.

1.4.2 Дискриминантный анализ

В дискриминантном анализе классы (группы) предполагаются уже заданными, а задача заключается в том, чтобы вновь появляющийся объект отнести к одному их этих классов на основании значения некой переменной [1]. Основная идея дискриминантного анализа заключается в том, чтобы определить, отличаются ли разные совокупности по среднему какой-либо переменной (или линейной комбинации переменных), и затем использовать эту переменную, чтобы предсказать для новых членов их принадлежность к той или иной группе. Таким образом, априорная классификация (предсказание для новых объектов) строится на основании значения весов классификации, построенных с помощью функции классификации на основе апостериорной (на основе имеющихся данных) классификации.

И группировка, и дискриминантный анализ имеют общий недостаток: оба этих метода привносят структуру классов извне, вместо определения реальной структуры.

1.4.3 Кластерный анализ

Кластерный анализ является методом поиска закономерностей группирования, как объектов исследования, так и признаков в отдельные локальные подмножества (кластеры).

Задача кластерного анализа заключается в выделении по эмпирическим данным резко различающихся групп (кластеров) объектов, которые схожи между собой внутри каждой из групп. С помощью кластерного анализа можно производить группировку объектов исследования в кластеры, группировку признаков в кластеры (редукция числа переменных), одновременную группировку объектов исследования и признаков.

Группировка объектов исследования в кластеры применяются в тех случаях, когда предполагается, что имеющаяся выборка гетерогенна, но причина гетерогенности при этом неизвестна. Результатом применения процедуры кластеризации может быть формирование нескольких подгрупп (кластеров) объектов исследования, в каждой из которых содержатся сходные наблюдения. Дальнейший анализ подгрупп может выявить некоторые объективные признаки, по которым эти подгруппы различаются.

1.5 Системы хранения данных в городском отделении реабилитации и кардиологии МБУЗ ГКБ № 2

До настоящего время специалисты городского отделения реабилитации и кардиологии МБУЗ ГКБ № 2 хранили информацию о пациентах посредством печатных анкет большого объема, которые содержали различные назначения, рекомендации, данные по анализам пациентов.

Анкета состоит из нескольких блоков, каждый из который выделен под определенное посещение центра больным. Информацию из анкет можно разбить на несколько групп:

1) персональные данные;
2) данные, относящиеся непосредственно к операции;
3) анализы;
4) медикаментозная терапия.

1.6 Основные требования к проектированию медицинской информационной системы

Медицинская информационная система (ИС) позволяет автоматизировать деятельность сотрудников центра. В ней объединены: система поддержки принятия медицинских решений, электронные медицинские записи о пациентах, данные медицинских исследований в цифровой форме, данные мониторинга состояния пациента с медицинских приборов.

ИС учреждений и организаций здравоохранения должны включать в себя взаимодействующие функциональные подсистемы, обеспечивающие

информационное обслуживание администрации и специалистов структурных подразделений учреждения (организации).

Основу взаимодействия функциональных подсистем должна составлять подсистема электронного документооборота, общая для всех подсистем ИС.

Прикладное программное обеспечение (приложения) и информационное обеспечение (базы данных) функциональных подсистем ИС должны быть построены по модульному принципу, т. е. включать в себя модули и компоненты, которые могут быть заменены или модернизированы без необходимости переработки всей подсистемы или ИС в целом.

Под программным обеспечением в настоящем документе понимают совокупность программ системы обработки информации и программных документов, необходимых для эксплуатации этих программ.

Разработка функционала ИС должна также учитывать требования ГОСТа Р 52636-2006, письма МЗ РФ от 10.08.2012 № 18-1/10/2-1336 и приказа МЗ РФ 11.11.2013 г. № 18-1/1010 для автоматизации процесса сбора статистических показателей.

Функциональные подсистемы ИС должны обеспечивать реализацию следующих функций (согласно классификации по п. 3.1.):

- ведение медицинской документации («электронных историй болезни»);
- формирование структурно-экономических описаний (паспортов) центра;
- учет пациентов и ведение реестра выполненных медицинских услуг;
- ведение нормативно-справочной информации (НСИ);
- оперативное планирование и учет ресурсов медицинской помощи (коечный фонд, медицинский персонал, сложная медицинская аппаратура, кабинеты приема, запасы аптечных товаров);
- представление государственной медицинской статистической отчетности;
- ведение базы данных (БД) зарегистрированных диагнозов для формирования статистики заболеваний;
- формирование сведений о наличии лекарств, доступных пациентам, и ведение учета лекарств, представленных пациентам по льготам.

Под ведением БД понимают деятельность по обновлению, восстановлению и перестройке структуры БД с целью обеспечения ее целостности, сохранности и эффективности использования.

В зависимости от предметной области информационные системы могут значительно различаться по своей архитектуре, своим функциям и назначению. Но можно выделить несколько свойств, которые являются общими [5].

1. Информационные системы предназначены для сбора, хранения и обработки информации, поэтому в основе любой из них лежит среда хранения и доступа к данным.

2. Информационные системы ориентированы на конечного пользователя, не обладающего высокой квалификацией в области вычислительной техники. Поэтому клиентские приложения информационной системы должны обладать простым, удобным, легко осваиваемым интерфейсом, который предоставляет

конечному пользователю все необходимые для работы функции и в то же время не дает ему возможность выполнять неправильных действий.

Следовательно, при проектировании информационной системы необходимо решить две главные задачи – разработать базу данных для хранения информации, а также интерфейс пользователя клиентских приложений.

Система управления СУБД является важной частью любой информационной системы. Тип используемой СУБД обычно определяется объемом информационной системы. Малые могут применить локальные СУБД, в других же информационных системах потребуется мощная клиент-серверная СУБД, поддерживающая многопользовательскую работу. Сейчас больше всего распространены реляционные СУБД.

Также определены требования к разрабатываемой информационной системе, выбрана СУБД, и выбрана среда разработки.

1.7 Выводы по разделу

В данной главе рассмотрены существующие методы анализа медицинских данных, системы хранения данных в городском отделении реабилитации и кардиологии МБУЗ ГКБ № 2 и основные требования к проектированию медицинской информационной системы.

После анализа предметной области основным математическим методом для данной работы был выбран дисперсионный анализ.

2 МАТЕМАТИЧЕСКАЯ МОДЕЛЬ АНАЛИЗА МЕДИЦИНСКИХ ДАННЫХ ПАЦИЕНТОВ ГОРОДСКОГО ОТДЕЛЕНИЯ РЕАБИЛИТАЦИИ И КАРДИОЛОГИИ МБУЗ ГКБ № 2

2.1 Дисперсионный анализ

Для анализа медицинских данных пациентов городского отделения реабилитации и кардиологии будем использовать метод дисперсионного анализа, основной целью которого является исследование значимости различия между средними (для групп или переменных). Эта проверка проводится с помощью разбиения суммы квадратов на компоненты, т.е. с помощью разбиения общей дисперсии (вариации) на части, одна из которых обусловлена случайной ошибкой (то есть внутригрупповой изменчивостью), а вторая связана с различием средних значений. Последняя компонента дисперсии затем используется для анализа статистической значимости различия между средними значениями.

Для описания данных и для оценки статистической значимости результатов проведенных исследований используют статистические методы, которые зачастую называют критериями значимости. Данные критерии строятся по следующему алгоритму:

1. формируется нулевая гипотеза – предположение о том, что исследуемые факторы не влияют на исследуемую величину и полученные различия случайны;
2. вычисляется вероятность получения наблюдаемых различий при условии справедливости нулевой гипотезы;
3. при малой полученной вероятности (пункт 2) отвергается нулевая гипотеза, и делается вывод: результаты эксперимента статистически значимы.

То есть, необходимо решить вопрос о случайности выявленных различий, от этого зависит принятие решения о том, является ли выявленные различия свидетельством различного состояния и/или свидетельством эффекта от вмешательства. Количественную характеристику случайности представляет теория вероятностей в виде *p*-значения. Чем это значение больше, тем больше вероятность отсутствия различий в пользу нулевой гипотезы, и чем оно меньше, тем больше вероятность наличия различий в пользу альтернативной гипотезы.

Во время реабилитации, пациентам рекомендуют заниматься физическими тренировками, так как малоподвижный образ жизни отрицательно сказывается на работе сердца и, поэтому, целью исследования было установить соотношение между состоянием пациента и частотой физических тренировок. Первая группа состояла из пациентов, состояние которых улучшилось. Во вторую группу вошли пациенты, состояние которых не изменилось. Пациенты третьей группы почувствовали ухудшение состояния.

Первоначально рассчитываем среднее количество тренировок для каждой группы:

$$\acute{x}_i = \sum ¿ kol \quad ,$$

где *sum* – общая сумма тренировок в группе, *kol* – количество людей в каждой группе;

стандартное отклонение для каждой группы:

$$s_i = \frac{\sum_{j=0}^{kol} \left(\acute{x}_i - x_{ij}\right)^2}{kol-1} \ ,$$

где x_{ij} –количество тренировок каждого человека в группе

С помощью дисперсионного анализа проверим гипотезу H_0 : можно ли считать эти различия случайными.

Вычислим сначала внутригрупповую дисперсию как среднюю дисперсий всех групп:

$$s^2_{вну} = \frac{1}{3}\left(s_1^2 + s_2^2 + s_3^2\right)$$

Вычислим межгрупповую дисперсию. Среднее трех выборочных средних равно

$$\acute{x} = \frac{1}{3}\left(\acute{x}_1 + \acute{x}_2 + \acute{x}_3\right),$$

следовательно, стандартное отклонение равно

$$s_{\acute{x}} = \sqrt{\frac{\left(\acute{x}_1 - \acute{x}\right)^2 + \left(\acute{x}_2 - \acute{x}\right)^2 + \left(\acute{x}_3 - \acute{x}\right)^2}{m-1}} \ ,$$

где *m*–количество групп;

межгрупповая дисперсия равна

$$s^2_{меж} = kol * s^2_{\acute{x}}$$

Эти два вида дисперсий нужны для того, чтобы с их помощью делать расчет статистического критерия F, который как раз и позволяет дать научно обоснованный ответ на вопрос: являются ли различия между группами достоверными и насколько достоверными.

Теперь перейдем непосредственно к вычислению критерия *F*. Суть *F*-критерия заключается в том, что он сравнивает две дисперсии: межгрупповую и внутригрупповую, поэтому их соотношение называют *F* – отношением:

$$F = \frac{Дисперсия\ совокупности,\ оцененная\ по\ выборочным\ средним}{Дисперсия\ совокупности,\ оцененная\ по\ выборочным\ дисперсиям} = \frac{s^2_{меж}}{s^2_{вну}}.$$

Критическое значение *F* однозначно определяется уровнем значимости (максимально приемлемая вероятность отвергнуть верную нулевую гипотезу, обозначается α, обычно α = 0,05) и еще двумя параметрами: внутригрупповое число степеней свободы и межгрупповое число степеней свободы (обозначаются *v*).

Межгрупповое число степеней свободы:

$$v_{меж} = m\text{-}1$$

Внутригрупповое число степеней свободы:

$$v_{вну} = m(kol\text{-}1).$$

Далее по таблице критических значений критерия F следует определить чему равно F и сделать вывод относительно нашей гипотезы H_0.
Если

$$F_{эмп} \geq F_{табл} \ ,$$

то гипотеза H_0 не подтверждена и различия в среднем количестве тренировок статистически значимы, иначе – гипотеза подтверждена.

2.2 Выводы по разделу

В результате была разработана математическая модель дисперсионного анализа, которая позволит исследовать значимость различия между средними (для групп или переменных) .

3 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ ПАЦИЕНТОВ ГОРОДСКОГО ОТДЕЛЕНИЯ РЕАБИЛИТАЦИИ И КАРДИОЛОГИИ МБУЗ ГКБ № 2

При проектировании базы данных необходимо выполнить 3 основных этапа:

1. Концептуальное проектирование – формулирование, анализ и выделение основных требований к данным.

2. Логическое проектирование – преобразование концептуальных требований в структуры данных.

3. Физическое проектирование – определение особенностей хранения данных, методов доступа и т.д. и реализация этих особенностей на основе одной из существующих СУБД.

Перед началом проектирования базы данных необходимо сформировать понятия о предметах, фактах и событиях, которыми будет оперировать данная система.

Для того чтобы привести основные понятия предметной области к той или иной модели данных, необходимо заменить их информационными представлениями. Одним из наиболее удобных инструментов унифицированного представления данных, независимого от реализующего его программного обеспечения, является модель "сущность-связь" (Entity-Relationship Model), или ER - модель. Она является наиболее известным представителем класса семантических (концептуальных, инфологических) моделей предметной области и предназначена для логического представления данных. Она определяет значения данных в контексте их взаимосвязи с другими данными. ER – модель обычно представляется в графической форме.

Основными элементами ER – моделей являются объекты (сущности), атрибуты объектов и связи между объектами. В свою очередь, связь между сущностями характеризуется типом связи и классом принадлежности. Класс может быть обязательным и необязательным. Если каждый экземпляр сущности участвует в связи, то класс принадлежности – обязательный, иначе – необязательный.

3.1 Выделение сущностей предметной области

В качестве основных для представления пациента в базе данных выделены следующие сущности: Пациент, Клиническое состояние, Эхокардиография и Лабораторное исследование.

Сущность Пациент должна обладать следующими свойствами: идентификатор пациента; номер пациента в исследовании, фамилия, имя, отчество, должность, рабочий телефон, домашний телефон, e-mail, почтовый адрес, дата поступления в стационар, дата выписки из стационара, дата рождения, дата повторного осмотра, возраст, пол, семейное положение, трудоспособность, инвалидность, место жительства, дата выкопировки данных, статус пациента, причина смерти, участие

в программах реабилитации, продолжительность реабилитации, специальность врача, рекомендация тренировок, тренировки, количество госпитализаций по причине с-с заболеваний, развитие ИМ, улучшение состояния, курение, прием терапии, продолжительность болезни, стенокардия напряжения, стенокардия покоя, безболевая ИМ, ИМ в анамнезе, кол-во ИМ, дата последнего ИМ, вмешательства на коронарных артериях, кол-во вмешательств, ангиопластика, стентирование, КШ, другие операции, желудочковая тахикардия, злокачественная ЖЭС, тромб в полости левого желудочка, аневризма левого желудочка, гипертрофия левого желудочка, фракция выброса левого желудочка, нагрузочный тест, гипертония, фибрилляция, риск тромбоэмболитических осложнений, церебральный атеросклероз, транзиторная ишемическая атака, инсульт, облитерирующий атеросклероз, хроническая сердечная недостаточность, функциональный класс сердечной недостаточности, дыхательная система, нарушение углеводного обмена, нарушение гликемии натощак, нарушение толерантности к глюкозе, сахарный диабет, тип сахарного диабета, инсулинопотребность, щитовидная железа, система пищеварения, другие патологии системы пищеварения, мочевыделительная система, другие патологии мочевыделительной системы, костно-мышечная система, другие патологии костно-мышечной системы, локализация остеоартроза, онкологические заболевания, локализация онкологического заболевания, стадия онкологического заболевания, количество баллов риска смертности, процент риска смертности, Отягощенный семейный анамнез развития сердечно-сосудистых заболеваний, курение табака, индекс курения, употребление алкоголя, артериальная гипертония, продолжительность артериальной гипертонии, нарушение углеводного обмена, нарушение липидного обмена, физическая активность, избыточная масса, недостаточная масса, ИМТ, искуственное кровообращение, время ИК, время пережатия аорты, дата операции, вид кардиоплегии, вид КШ, количество наложенных шунтов, реваскуляризация, гемотрансфузия, гибридная операция, вид гибридной операции, манипуляции в послеоперационном периоде, временная ЭКС, фармокологическая кардиоверсия, электрическая дефибрилляция, ЧПЭС, установка постоянного ЭКС, ремедиастернотомия, использование VAC системы, дренирование плевральной полости в связи с гидротораксом, дренирование плевральной полости в связи с пневмотораксом, дренирование полости перикарда, использование в/аортального баллонного контр-пульсатора, коронарография и шунтография после КШ, экстренное стентирование после КШ, прочие манипуляции.

Пациенты проходят лабораторные исследования и эхокардиографию, а так же у них оценивается клиническое состояние.

Лабораторные исследования обладают следующими свойствами: идентификатор, идентификатор пациента, дата исследования, гемоглобин, лейкоциты, СОЭ, глюкоза натощак, глюкоза через 2 часа после еды, венозная глюкоза натощак, креатинин, калий, СРБ, СКФ-epi, клиренс креатинина (Кокрофт-Голт), МНО, фибриноген, общий холестерин, ХС ЛНП, ХС ЛВП, Триглицериды.

Эхокардиография обладает следующими свойствами: идентификатор, идентификатор пациента, левое предсердие, объем ЛП, индекс объема ЛП, КДР, КСР, ТМЖП, ТЗС ЛЖ, ФВ, ФУ, масса миокарда, индекс массы миокарда, МГД на аортальном клапане, МГД на митральном клапане, дата осмотра.

Клиническое состояние обладает следующими свойствами: идентификатор, идентификатор пациента, рост, вес, объем талии, ИМТ, САД покой, ДАД покой, ЧСС покой, дата осмотра, дистанция теста 6 минутной ходьбы, ЧСС исходная, ЧСС после теста, САД исходная, САД после теста, ДАД исходная, ДАД после теста, должный уровень физической работоспособности, фактический уровень физической работоспособности, ФК сердечной недостаточности по NYHA, нарушение ритма сердца, нарушение проводимости сердца, другие нарушения ритма сердца.

3.2 ER-Модели сущностей

Отношение между Клиническим состоянием и Пациентом изображено на рисунке 3.1. У Пациента может быть множество клинических состояний, а Клиническое состояние относится только к одному Пациенту.

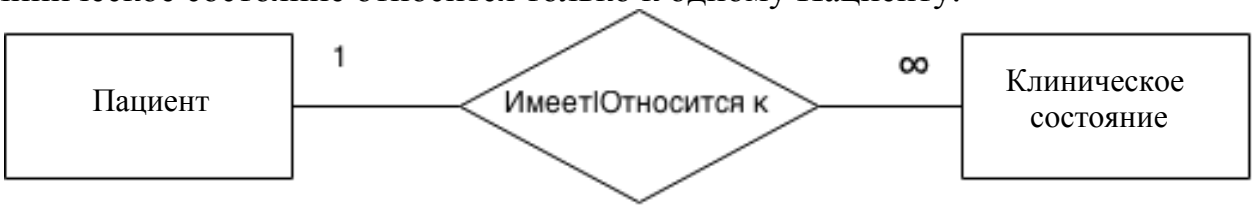

Рисунок 3.1 – Отношение Пациент - Клиническое состояние

Связь Пациента и Лабораторного исследования можно изобразить, как показано на рисунке 3.2.

Рисунок 3.2 – Отношение Пациент - Лабораторное исследование

Отношение Пациента и Эхокардиографии представлено на рисунке 3.3.

Рисунок 3.3 – Отношение Пациент - Эхокардиография

3.3 Представление сущностей в виде элементов реляционной базы данных

Для преобразования ER-диаграмм к схеме базы данных, сущности и отдельные отношения были преобразованы в таблицы.

Таблица 3.1 хранит основную информацию о пациентах.

Таблица 3.1

Основная информация

Имя	Тип	Размер	Комментарий
IdPatient	Длинное целое	4	идентификатор пациента
NumberPat	Текстовый	20	номер пациента в исследовании
LastName	Текстовый	50	фамилия
FirstName	Текстовый	50	имя
PaternalName	Текстовый	50	отчество
Job	Текстовый	100	должность
WorkPhone	Текстовый	15	рабочий телефон
HomePhone	Текстовый	15	домашний телефон
Email	Текстовый	100	адрес электронной почты
Post	Текстовый	255	почтовый адрес
DateStart	Дата/время	8	дата поступления в стационар
DateEnd	Дата/время	8	дата выписки из стационара
BirthDate	Дата/время	8	дата рождения
DatePovt	Дата/время	8	дата повторного осмотра

В таблице 3.2 хранятся выкопированные данные.

Таблица 3.2

Выкопированные данные

Имя	Тип	Размер	Комментарий
IdPatient	Длинное целое	4	идентификатор пациента
Age	Целое	2	возраст
Sex	Текстовый	3	пол
MaritalStatus	Текстовый	25	семейное положение
LaborAbility	Текстовый	25	трудоспособность
Disability	Текстовый	25	инвалидность
Location	Текстовый	50	место жительства
DateZapoln	Дата/время	8	дата заполнения

Таблица 3.3 хранит факторы риска на момент госпитализации.

Таблица 3.3

Факторы риска

Имя	Тип	Размер	Комментарий
IdPatient	Длинное целое	4	идентификатор пациента
Anamnez	Текстовый	15	отягощенный семейный анамнез развития сердечно-сосудистых заболеваний
Tabak	Текстовый	15	курение табака,
IndexKur	Одинарное с плавающей точкой	8	индекс курения
Alkogol	Текстовый	15	употребление алкоголя
ArtGip	Текстовый	4	артериальная гипертония
AG	Одинарное с плавающей точкой	4	продолжительность артериальной гипертонии
UglObm	Текстовый	5	нарушение углеводного обмена
LipObm	Текстовый	5	нарушение липидного обмена
FisAct	Текстовый	15	физическая активность
IzbMassa	Текстовый	4	избыточная масса
NedMassa	Текстовый	4	недостаточная масса
IMT	Одинарное с плавающей точкой	4	ИМТ

В таблице 3.4 хранятся особенности течения ишемической болезни сердца до КШ.

Таблица 3.4

Особенности ИБ до КШ

Имя	Тип	Размер	Комментарий
IdPatient	Длинное целое	4	идентификатор пациента
ProdBolez	Одинарное с плавающей точкой	4	продолжительность болезни
StenokNapr	Текстовый	255	стенокардия напряжения
StenokPok	Текстовый	255	стенокардия покоя
BezbolIsh	Текстовый	255	безболевая ИМ
IMAnam	Текстовый	255	ИМ в анамнезе
KolIM	Целое	3	кол-во ИМ
DatePoslIM	Дата/время	8	дата последнего ИМ

Окончание таблицы 3.4

Имя	Тип	Размер	Комментарий
VmeshKorArt	Текстовый	255	вмешательства на коронарных артериях
KolVm	Целое	3	кол-во вмешательств
Angio	Текстовый	255	ангиопластика
Stent	Текстовый	255	стентирование
KSh	Текстовый	255	КШ
drugoe	Текстовый	255	другие операции
GelTahAnam	Текстовый	255	желудочковая тахикардия
ZlokGES	Текстовый	255	злокачественная ЖЭС
TrombLevG	Текстовый	255	тромб в полости левого желудочка
AnevrLevG	Текстовый	255	аневризма левого желудочка
GipertLevG	Текстовый	255	гипертрофия левого желудочка
FrakVibLevGDo	Текстовый	255	фракция выброса левого желудочка
NagrTest	Текстовый	255	нагрузочный тест

Таблица 3.5 хранит коморбидные состояния до КШ.

Таблица 3.5

Коморбидные состояния до КШ

Имя	Тип	Размер	Комментарий
IdPatient	Длинное целое	4	идентификатор пациента
Gipert	Текстовый	25	гипертония
FibrTrep	Текстовый	25	фибрилляция
RiskTrombOsl	Одинарное с плавающей точкой	4	риск тромбоэмболитических осложнений
CerAter	Текстовый	25	церебральный атеросклероз
TranzIsh	Текстовый	25	транзиторная ишемическая атака
Insult	Текстовый	25	инсульт
AterNiz	Текстовый	25	облитерирующий атеросклероз
HronSerdNed	Текстовый	25	хроническая сердечная недостаточность
FunkKlSerNed	Текстовый	25	функциональный класс сердечной недостаточности

DihatS	Текстовый	25	дыхательная система

Окончание таблицы 3.5

Имя	Тип	Размер	Комментарий
UglObm	Текстовый	25	нарушение углеводного обмена
GlikNat	Текстовый	25	нарушение гликемии натощак
TolGluk	Текстовый	25	нарушение толерантности к глюкозе
SakarD	Текстовый	25	сахарный диабет
TipDiab	Текстовый	25	тип сахарного диабета
Insulpotr	Текстовый	25	инсулинопотребность
Shit	Текстовый	25	щитовидная железа
Pish	Текстовый	25	система пищеварения
PishDrugoe	Текстовый	25	другие патологии системы пищеварения
Moch	Текстовый	25	мочевыделительная система
MochDrugoe	Текстовый	25	другие патологии мочевыделительной системы
Kost	Текстовый	25	костно-мышечная система
OstLoc	Текстовый	25	локализация остеоартроза
KostDrugoe	Текстовый	25	другие патологии костно-мышечной системы
Onko	Текстовый	25	онкологические заболевания
Local	Текстовый	25	локализация онкологического заболевания
Stad	Текстовый	25	стадия онкологического заболевания

В таблице 3.6 хранятся оценки риска смертности по шкале EUROSCORE II.

Таблица 3.6

Оценка риска смертности

Имя	Тип	Размер	Комментарий
IdPatient	Длинное целое	4	идентификатор пациента
Ball	Одинарное с плавающей точкой	4	Балл
Proc	Одинарное с плавающей точкой	4	Процент

Таблица 3.7 хранит данные о проведении КШ.

Таблица 3.7

Коронарное шунтирование

Имя	Тип	Размер	Комментарий
IdPatient	Длинное целое	4	идентификатор пациента
IskKrovoob	Текстовый	25	искусственное кровообращение
TimeIK	Одинарное с плавающей точкой	4	время ИК
TimePerAort	Одинарное с плавающей точкой	4	время пережатия аорты
DateOper	Дата/время	8	дата операции
Kardiople	Текстовый	255	вид кардиоплегии
VidKS	Текстовый	255	вид КШ
KolS	Текстовый	25	количество наложенных шунтов
Rev	Текстовый	25	реваскуляризация
Gem	Текстовый	25	гемотрансфузия
Gibr	Текстовый	4	гибридная операции
GO	Текстовый	255	вид гибридной операции
ManPosle	Текстовый	4	манипуляции в послеоперационном периоде
VrEKS	Текстовый	4	временная ЭКС
Farm	Текстовый	4	фармокологическая кардиоверсия
Def	Текстовый	4	электрическая дефибрилляция
HPS	Текстовый	4	ЧПЭС
PostEKS	Текстовый	4	установка постоянного ЭКС
Rem	Текстовый	4	ремедиастернотомия
VAC	Текстовый	4	использование VAC системы
DrPer	Текстовый	4	дренирование полости перикарда
DrGid	Текстовый	4	дренирование плевральной полости в связи с гидротораксом
DrPnev	Текстовый	4	дренирование плевральной полости в связи с пневмотораксом
KonP	Текстовый	4	использование в/аортального баллонного контр-пульсатора
KorKS	Текстовый	4	коронарография и шунтография после КШ

Stent	Текстовый	4	экстренное стентирование после КШ

Окончание таблицы 3.7

Имя	Тип	Размер	Комментарий
Dr	Текстовый	4	прочие манипуляции

В таблице 3.8 хранятся данные о реабилитации.

Таблица 3.8

Реабилитация

Имя	Тип	Размер	Комментарий
IdPatient	Длинное целое	4	идентификатор пациента
Stat	Текстовый	25	статус пациента
PrSm	Текстовый	50	причина смерти
PrReab	Текстовый	255	участие в программах реабилитации
Prod	Одинарное с плавающей точкой	4	продолжительность реабилитации
Who	Текстовый	150	специальность врача
TrH	Текстовый	25	рекомендация тренировок
Tr	Длинное целое	4	тренировки
KolGosp	Длинное целое	4	количество госпитализаций по причине с-с заболеваний
OIM	Текстовый	10	развитие ИМ
Sost	Текстовый	25	улучшение состояния
Tab	Текстовый	25	курение
Ter	Текстовый	25	прием терапии

Таблица 3.9 хранит данные о клиническом состоянии.

Таблица 3.9

Клиническое состояние

Имя	Тип	Размер	Комментарий
IdKlSost	Длинное целое	4	идентификатор клинического состояния
IdPatient	Длинное целое	4	идентификатор пациента
Rost	Одинарное с плавающей точкой	4	рост
Ves	Одинарное с плавающей точкой	4	вес

Tal	Одинарное с плавающей точкой	4	объем талии
IMT	Одинарное с плавающей точкой	4	ИМТ

Окончание таблицы 3.9

Имя	Тип	Размер	Комментарий
SAD	Одинарное с плавающей точкой	4	САД покой
DAD	Одинарное с плавающей точкой	4	ДАД покой
HSS	Одинарное с плавающей точкой	4	ЧСС покой
DataOsm	Дата/время	8	дата осмотра
Dist	Одинарное с плавающей точкой	4	дистанция теста 6 минутной ходьбы
HSSIsh	Одинарное с плавающей точкой	4	ЧСС исходная
HSSPos	Одинарное с плавающей точкой	4	ЧСС после теста
SADIsh	Одинарное с плавающей точкой	4	САД исходная
SADPos	Одинарное с плавающей точкой	4	САД после теста
DADIsh	Одинарное с плавающей точкой	4	ДАД исходная
DADPos	Одинарное с плавающей точкой	4	ДАД после теста
DolUrFP	Текстовый	255	должный уровень физической работоспособности
FaktUrFP	Текстовый	255	фактический уровень физической работоспособности
FKSN	Одинарное с плавающей точкой	4	ФК сердечной недостаточности по NYHA
NarRitm	Текстовый	255	нарушение ритма сердца
NarProv	Текстовый	255	нарушение проводимости сердца
NarRitmDr	Текстовый	255	другие нарушения ритма сердца

Таблица 3.10 хранит данные о лабораторных исследованиях.

Таблица 3.10

Лабораторные исследования

Имя	Тип	Размер	Комментарий
IdLab	Длинное целое	4	идентификатор лабораторного исследования

Окончание таблицы 3.10

Имя	Тип	Размер	Комментарий
IdPatient	Длинное целое	4	идентификатор пациента
DataLab	Дата/время	8	дата лабораторного исследования
Gem	Одинарное с плавающей точкой	4	гемоглобин
Ley	Одинарное с плавающей точкой	4	лейкоциты
SOA	Одинарное с плавающей точкой	4	СОЭ
GlNat	Одинарное с плавающей точкой	4	глюкоза натощак
GlAf	Одинарное с плавающей точкой	4	глюкоза через 2 часа после еды
GlNatVen	Одинарное с плавающей точкой	4	венозная глюкоза натощак
Krea	Одинарное с плавающей точкой	4	креатинин
Kaliy	Одинарное с плавающей точкой	4	калий
SRB	Одинарное с плавающей точкой	4	СРБ
SKF	Одинарное с плавающей точкой	4	СКФ-epi
Klir	Одинарное с плавающей точкой	4	клиренс креатинина (Кокрофт-Голт)
MNO	Одинарное с плавающей точкой	4	МНО
Fibr	Одинарное с плавающей точкой	4	фибриноген
ObHol	Одинарное с плавающей точкой	4	общий холестерин
HSLNP	Одинарное с плавающей точкой	4	ХС ЛНП

HSLVP	Одинарное с плавающей точкой	4	ХС ЛВП
Trigl	Одинарное с плавающей точкой	4	триглицериды

В таблице 3.11 хранятся данные об эхокардиографии.

Таблица 3.11

Эхокардиография

Имя	Тип	Размер	Комментарий
IdEho	Длинное целое	4	идентификатор эхокардиографии
IdPatient	Длинное целое	4	идентификатор пациента
LevPr	Текстовый	150	левое предсердие
VLevPr	Текстовый	150	объем ЛП
IndVLevPr	Текстовый	150	индекс объема ЛП
KDR	Текстовый	150	КДР
KSR	Текстовый	150	КСР
TMGP	Текстовый	150	ТМЖП
TZS	Текстовый	150	ТЗС ЛЖ
FV	Текстовый	150	ФВ
FU	Текстовый	150	ФУ
MMiok	Текстовый	150	масса миокарда
IndMMiok	Текстовый	150	индекс массы миокарда
MGDAor	Текстовый	150	МГД на аортальном клапане
MGDMitr	Текстовый	150	МГД на митральном клапане
DataOsm	Дата/время	8	дата осмотра

Результат приведения ER-диаграммы к схеме базы данных показан на рисунке 3.4. Наполнение базы данных производится через информационную систему.

3.4 Выводы по разделу

В данном разделе была разработана база данных.

Первоначально было выполнено концептуальное проектирование, в ходе которого сначала были выделены сущности и связи в системе, а потом на основе них была построена ER-диаграмма.

Далее был осуществлен переход к даталогическому проектированию – построена реляционная схема базы данных и было приведено описание всех таблиц и атрибутов, находящихся в ней.

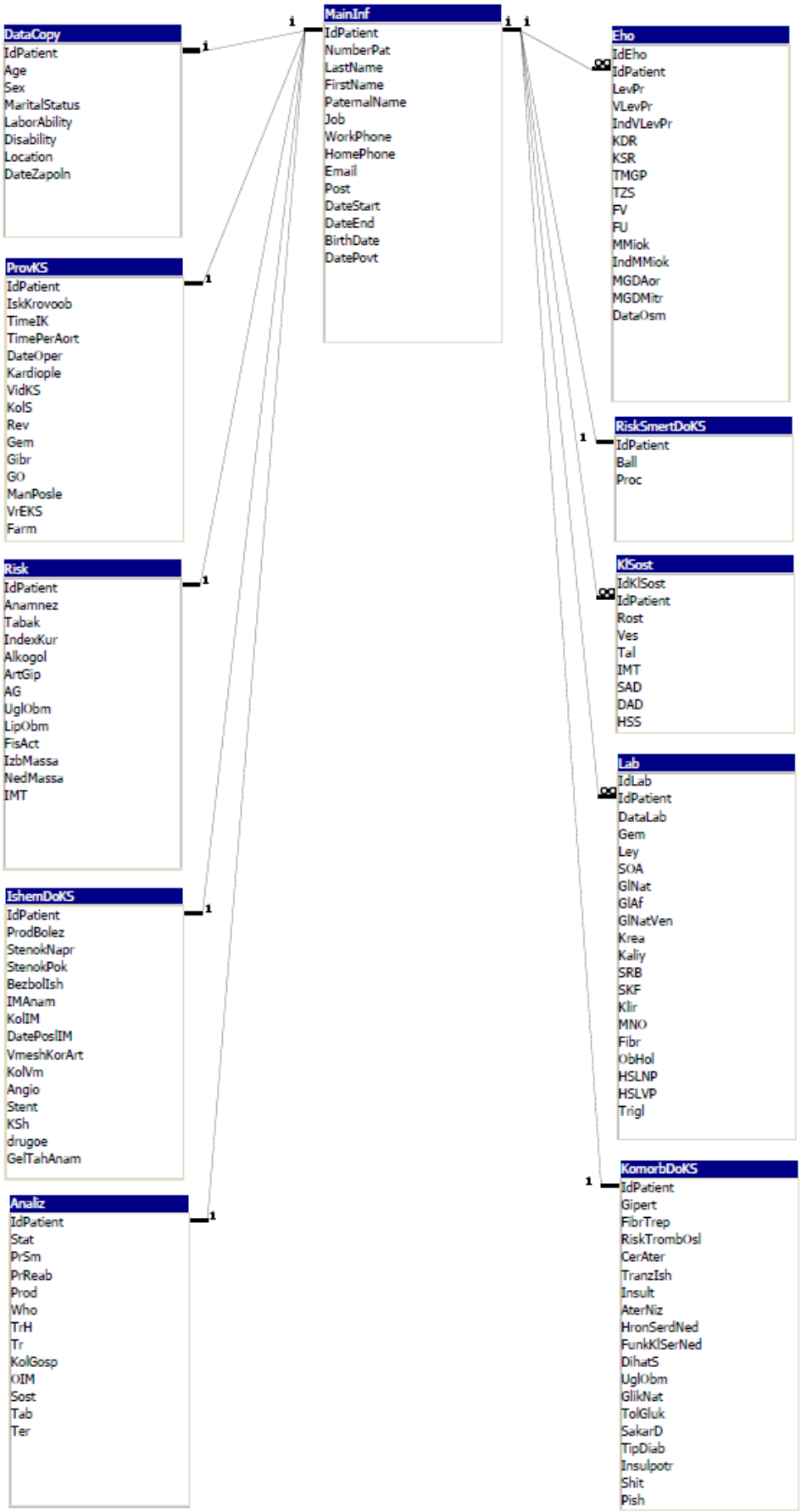

Рисунок 3.4 – Схема БД

4 РАЗРАБОТКА ПРОГРАММЫ

4.1 Разработка алгоритмов

Для решения задачи создания информационной системы мониторинга эффективности реабилитации пациентов после коронарного шунтирования следует выделить отдельные модули, которые должны реализовывать определенную часть функций и выполнять следующие действия:

- собирать информацию о пациентах;
- организовывать хранение данных;
- производить автоматический дисперсионный анализ;
- осуществлять управление пациентами.

Приложение разрабатывается для операционной системы Windows и должно использовать принцип обработки сообщений. Основной алгоритм работы приложения изображен на рисунке 4.1.

4.1.1 Алгоритм обработки событий главного окна

После двойного щелчка по ярлыку приложения происходит запуск приложения. Затем система ожидает события от пользователя.

Если выбран пункт меню «Дисперсионный анализ», то происходит вызов окна «Дисперсионный анализ», если выбран пункт «Общий анализ», то происходит вызов окна «Общий анализ», в случае, если выбран пункт меню «О программе» происходит вызов окна «О программе».

В случае, когда нажата кнопка «Добавить», происходит вызов окна «Добавление пациента», если нажата кнопка «Изменить», то вызывается окно «Изменение данных», а если нажата кнопка «Найти», то открывается окно «Поиск пациента».

Если нажать кнопку «Сбросить фильтр», то фильтр, примененный к данным ранее, будет сброшен. При нажатии кнопки «Обновить» – таблица с данными будет обновлена. Если нажать кнопку «Удалить», то будет выведено окно для подтверждение действия.

После любого действия происходит передача сообщения стандартному обработчику Windows. Алгоритм обработки событий главного окна приведён на рисунке 4.2.

4.1.2 Алгоритм дисперсионного анализа

После нажатия на пункт меню «Дисперсионный анализ», происходит открытие окна «Дисперсионный анализ», где рассчитываются все необходимые показатели. Алгоритм дисперсионного анализа приведен на рисунке 4.3.

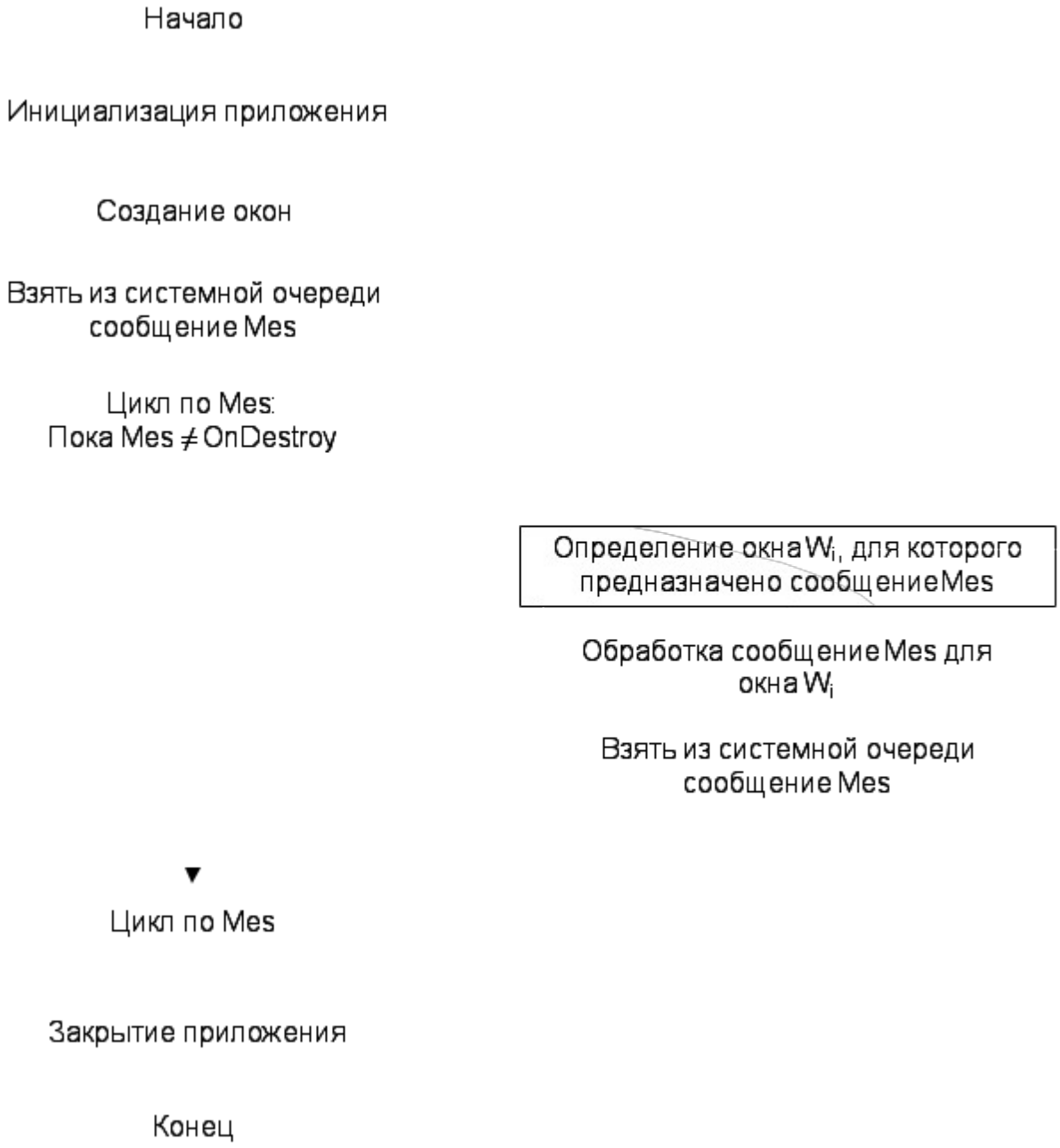

Рисунок 4.1 – Основной алгоритм

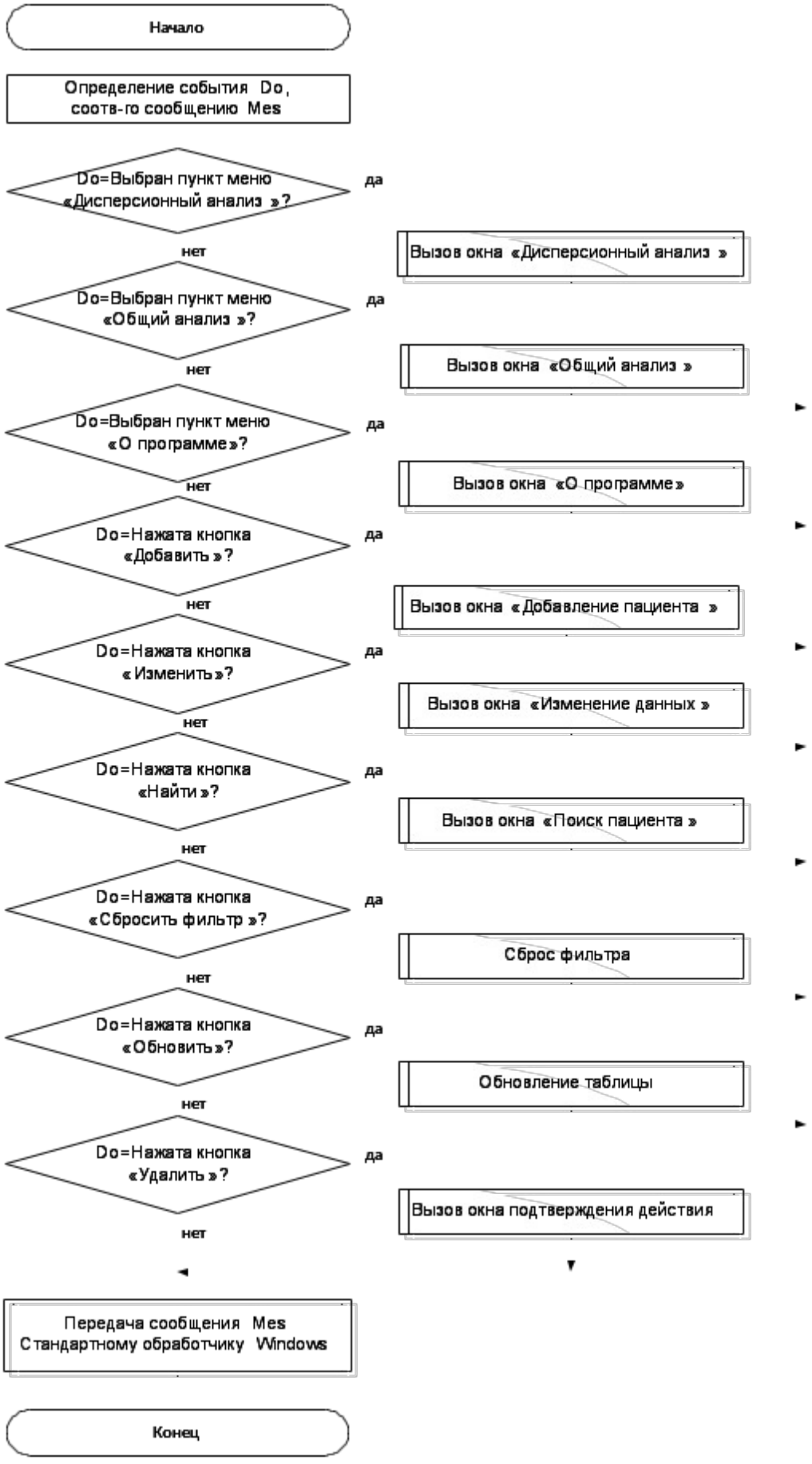

Рисунок 4.2 – Алгоритм обработки событий главного окна

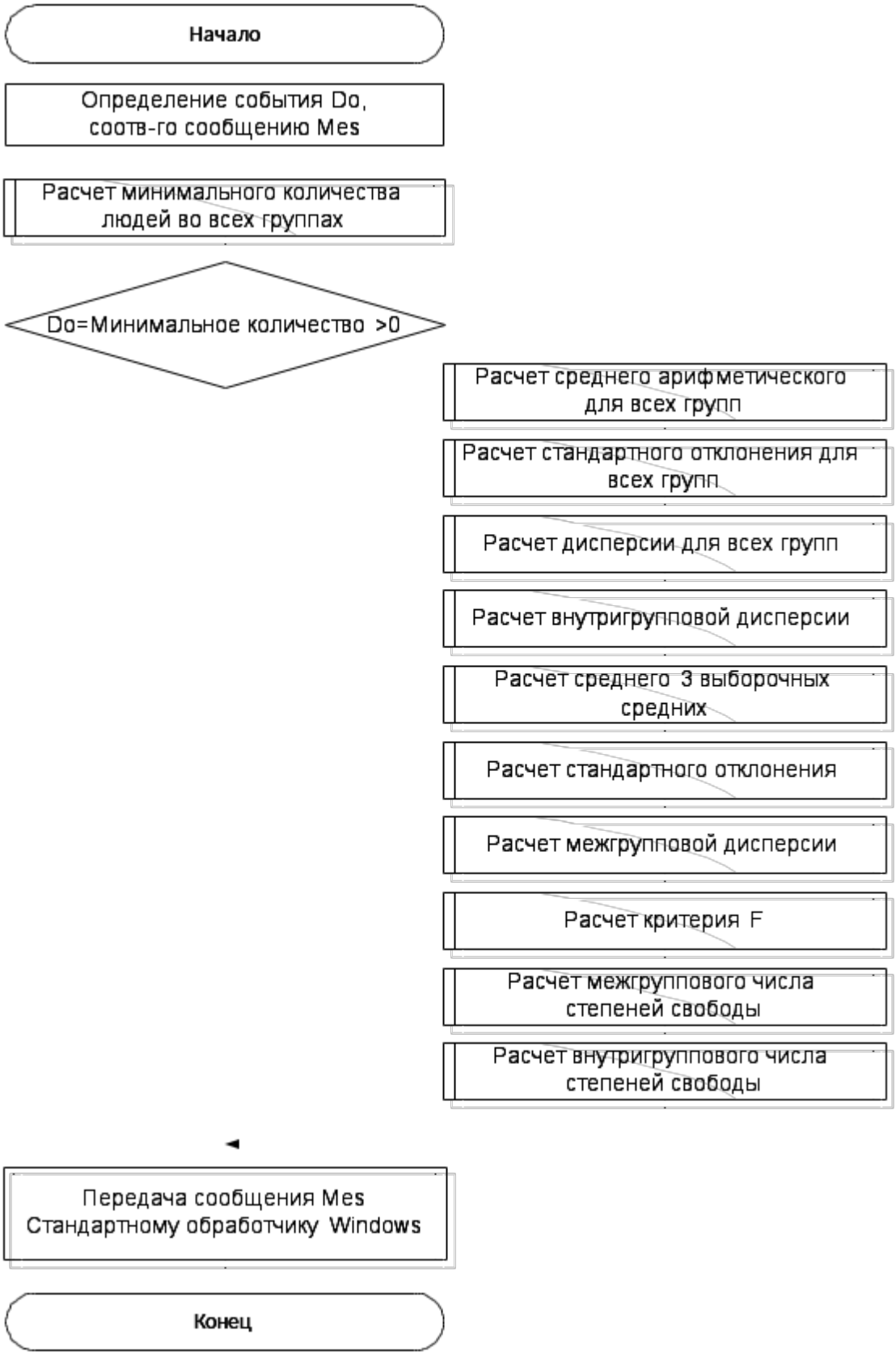

Рисунок 4.3 – Алгоритм дисперсионного анализа

4.2 Разработка пользовательского интерфейса

После запуска приложения открывается форма с таблицей пациентов (рисунок 4.4).

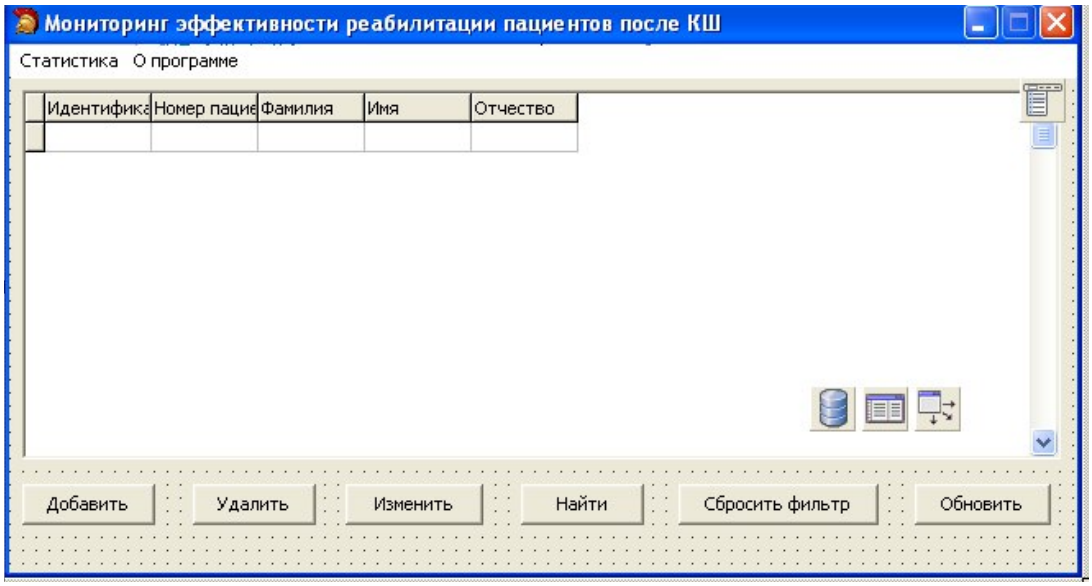

Рисунок 4.4 – Интерфейс главного окна программы

Нажав на кнопку «Добавить», происходит вызов окна «Добавление пациента», где врач может добавить нового пациента в базу данных, нажав на кнопку «Сохранить изменения» (рисунок 4.5).

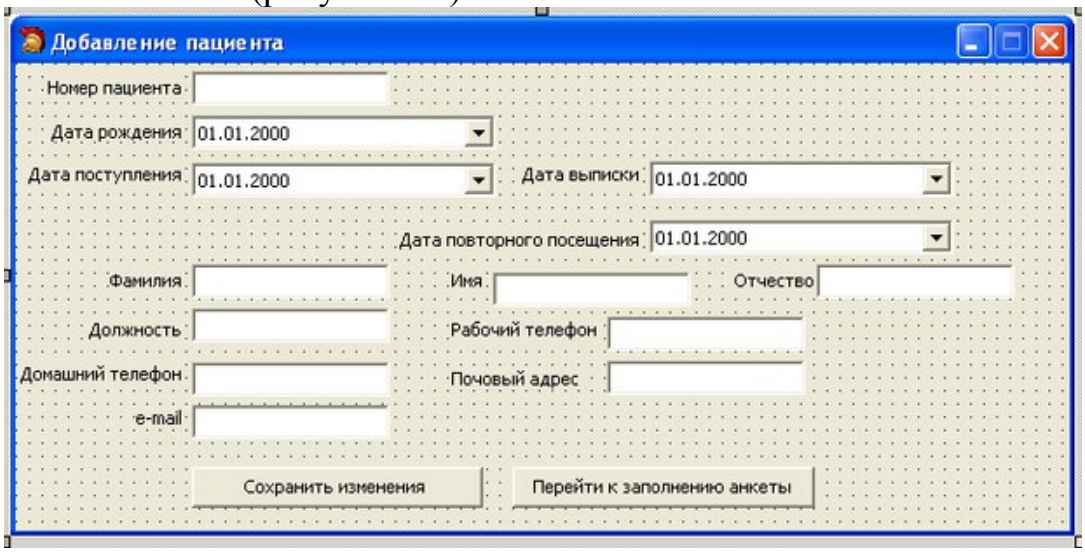

Рисунок 4.5 – Окно добавления пациента

Если нажата кнопка «Изменить», то откроется окно «Изменение данных», которое аналогично приведенному на рисунке 4.5, но в него будут выгружены ранее введенные данные, которые врач сможет изменить. Нажав на кнопку «Сохранить изменения», все внесенные изменения будет сохранены. Если пользователь нажмет кнопку «Перейти к заполнению анкеты», то откроется окно «Заполнение анкеты», приведенное на рисунке 4.6. В данном окне врач выбирает, какой пункт анкеты он хочет заполнить.

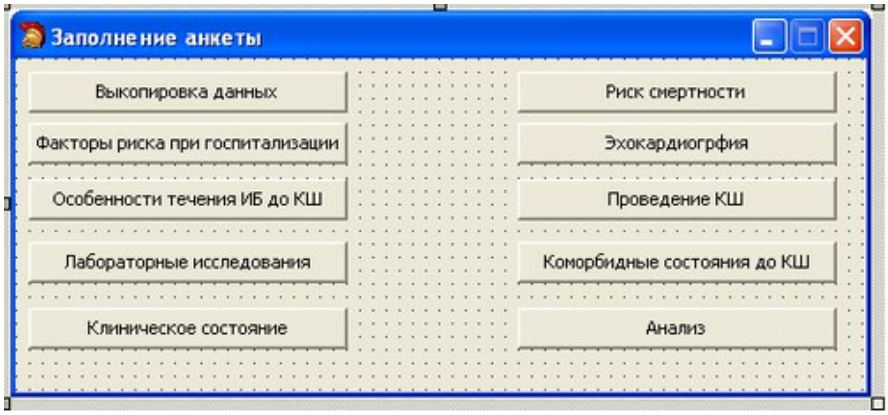

Рисунок 4.6 – Окно «Заполнение анкеты»

При нажатии на какую-либо форму открывается соответствующее окно. Например, при нажатии на кнопку «Выкопировка данных» открывается окно, приведенное на рисунке 4.7.

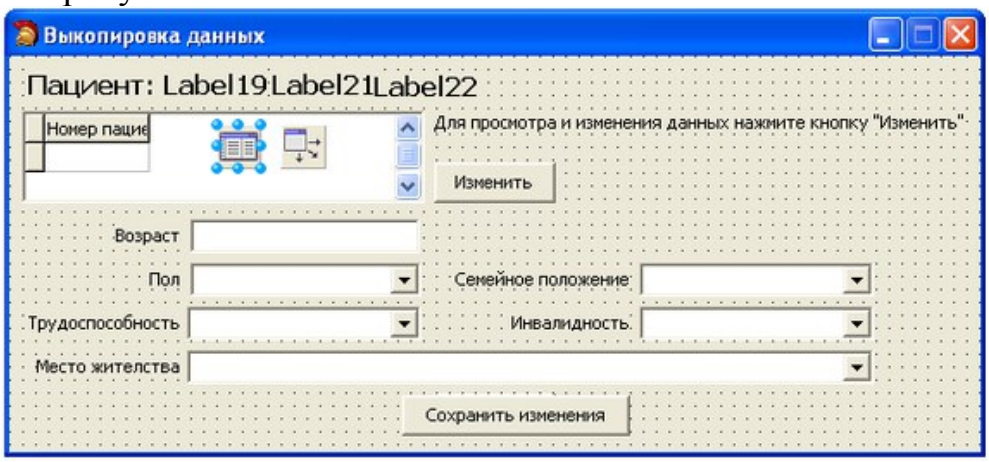

Рисунок 4.7 – Окно «Выкопировка данных»

В данное окно врач заносит данные пациента. Для удобства использования ФИО пациента, анкета которого изменяется в данный момент, выводится правее надписи «Пациент», на местах Label19, Label21, Label22. Если значения какого-либо поля заранее могут быть предопределены, то используется компонент GroupBox, в котором прописаны возможные варианты ответа. Если врач ранее вносил данные пациента в этом окне, то нажав на кнопку «Изменить» он может их выгрузить в форму для дальнейшего просмотра и редактирования. Нажав на кнопку «Сохранить изменения», все внесенные изменения будут сохранены.

Аналогичным образом разработаны все пункты анкеты, представленные на рисунке 4.6.

Нажав на кнопку «Найти» на главной форме, откроется окно «Поиск пациента» (рисунок 4.8).

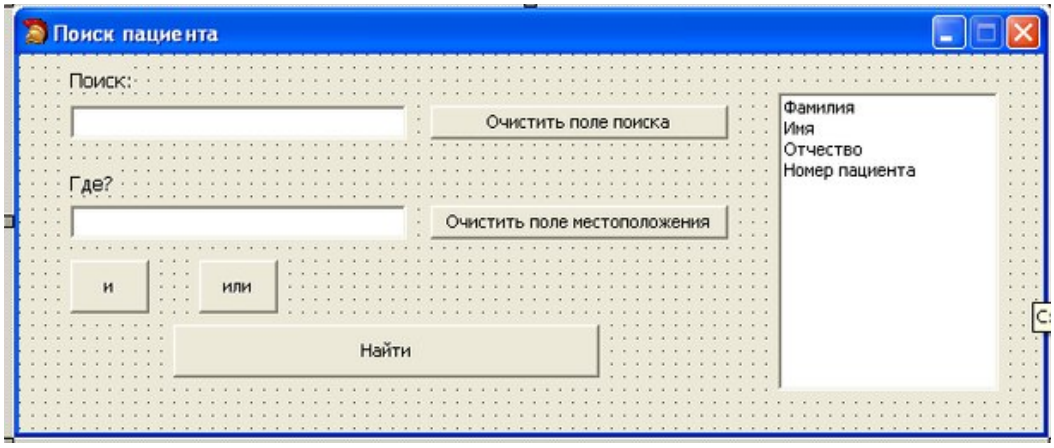

Рисунок 4.8 – Окно поиска пациентов

В верхнем поле ввода врач может вписать фамилию, имя, отчество или номер пациента, после чего в правом поле он должен выбрать соответствующий параметр. После этого он может нажать кнопку «Найти» и в таблице на главной странице останутся пациенты, соответствующие запросу. Врач может искать пациентов по нескольким полям, используя кнопки «И» и «ИЛИ», тем самым создавая сложный запрос. Врач имеет возможность переключаться между окном поиска и главным окном программы.

Нажав на главной форме на кнопку «Сбросить фильтр», фильтр, который был применен к данным через окно поиска пациента, будет сброшен.

Если врач нажмет на кнопку «Удалить», то выведется сообщение о подтверждении действия, положительно ответив на которое, выбранный пациент будет удален.

Нажав на кнопку «Обновить», таблица, расположенная на главной форме будет обновлена.

Если врач выберет пункт меню «Статистика», то дальше у него будет возможность выбора из двух пунктов «Общий анализ» и «Дисперсионный анализ». Нажав на кнопку «Общий анализ» откроется соответствующее окно (рисунок 4.9), где врач может просмотреть такую информацию, как количество проведенных операций, количество живых и мертвых пациентов по итогам телефонного опроса, узнать количество смертей по причине сердечно-сосудистых заболеваний и их процентное соотношение к общему числу смертей.

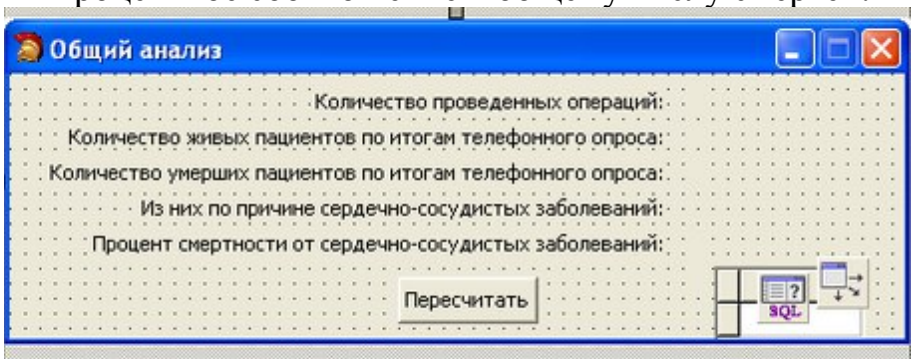

Рисунок 4.9 – Окно «Общий анализ»

Выбрав пункт «Дисперсионный анализ», то откроется соответствующее окно со всеми необходимыми показателями (рисунок 4.10).

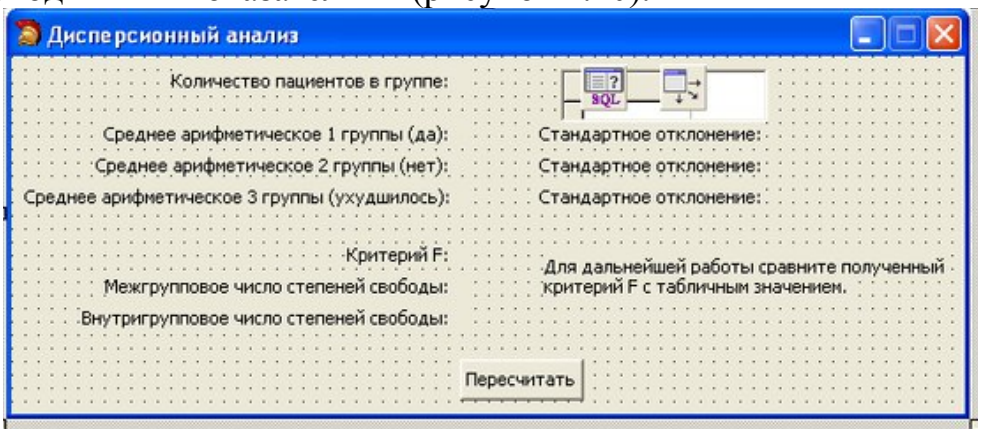

Рисунок 4.10 – Окно «Дисперсионный анализ»

В окнах «Дисперсионный анализ» и «Общий анализ» есть кнопка «Пересчитать», нажав на которую произойдет пересчет всех показателей, учитывая последние внесенные изменения.

Выбрав в главном окне пункт меню «О программе», откроется окно «О программе», где можно будет просмотреть общую информацию о программе (рисунок 4.11) .

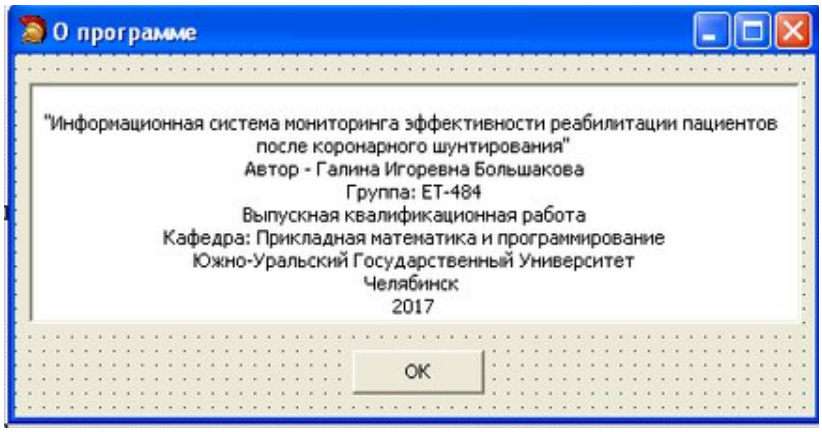

Рисунок 4.11 – Окно «О программе»

4.3 Программный эксперимент

После разработки и реализации алгоритмов и интерфейса над полученной программой был произведен опыт, предназначенный для экспериментального подтверждения работоспособности спроектированных программных средств.

В папке с приложением лежит файл с расширением .exe. Запуск приложения производится двойным нажатием левой кнопки мыши по данному файлу (рисунок 4.12).

После запуска приложения открывается окно для ввода имени пользователя и пароля к базе данных (рисунок 4.13).

После успешного ввода данных для подключения к БД открывается главное окно приложения (рисунок 4.14).

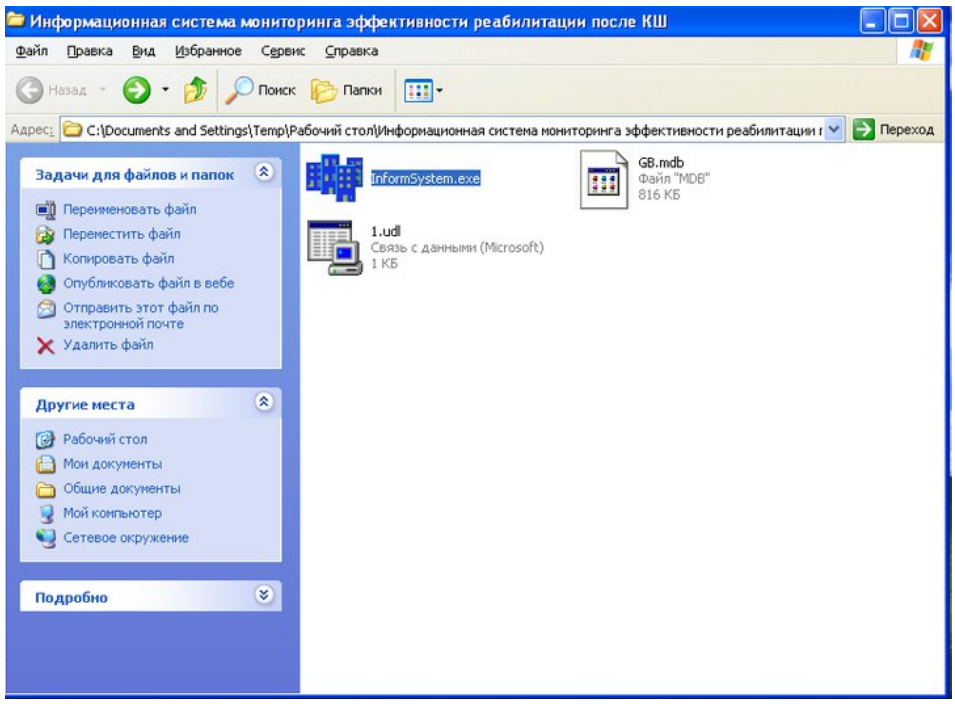

Рисунок 4.12 – Папка с приложением

Рисунок 4.13 – Подключение к базе данных

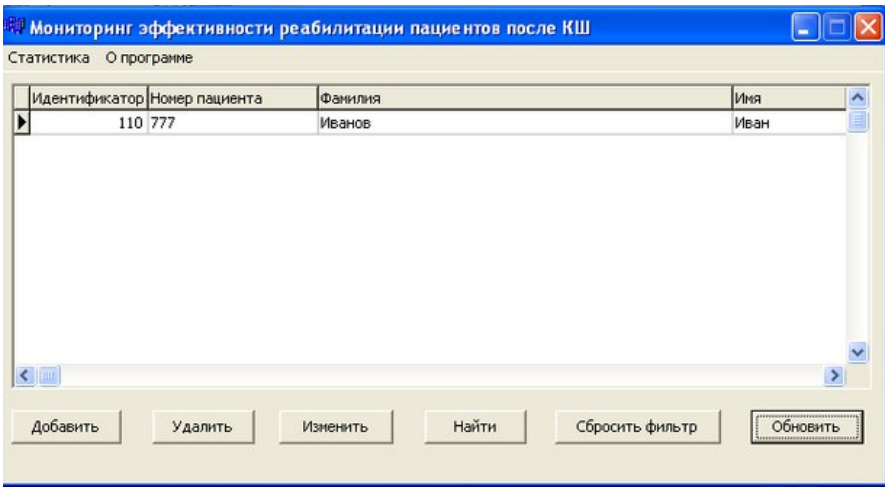

Рисунок 4.14 – Главное окно приложения

После выбора пункта меню «О программе», открывается окно с основной информацией (рисунок 4.15).

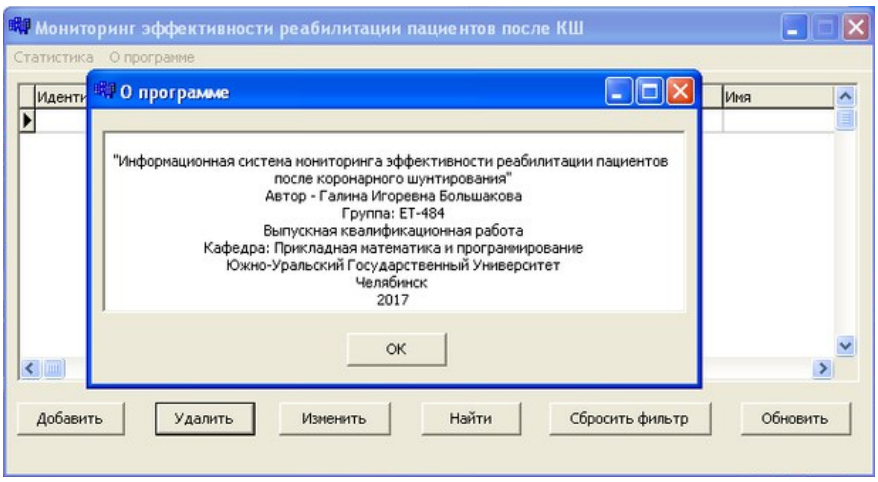

Рисунок 4.15 – Окно «О программе»

При выборе в пустой базе пункта меню «Статистика», а затем пункта «Общий анализ» и нажатия на кнопку «Пересчитать», система выведет только общее число пациентов (рисунок 4.16), а если выбрать пункт «Дисперсионный анализ», а затем нажать кнопку «Пересчитать», то выводится надпись, сообщающая, что расчет необходимых критериев выполнен не был, так как минимальное число людей в группе равно нулю (рисунок 4.17). При выполнении аналогичных действий на заполненной базе, выводятся результаты всех расчетов, предусмотренных программой (рисунок 4.18 и рисунок 4.19).

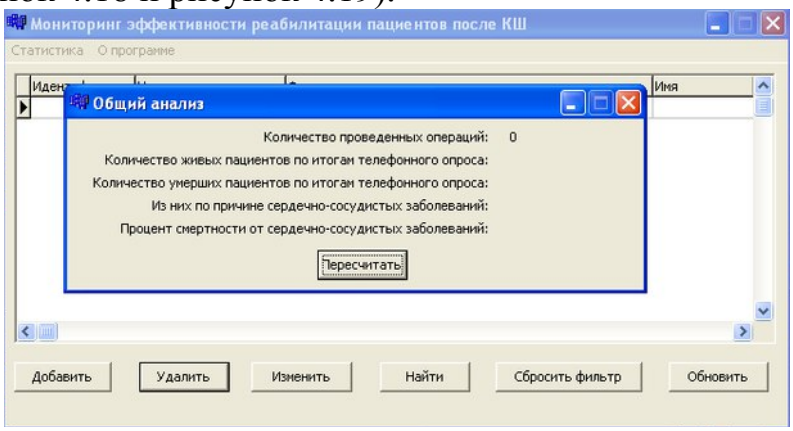

Рисунок 4.16 – Общий анализ на пустой базе

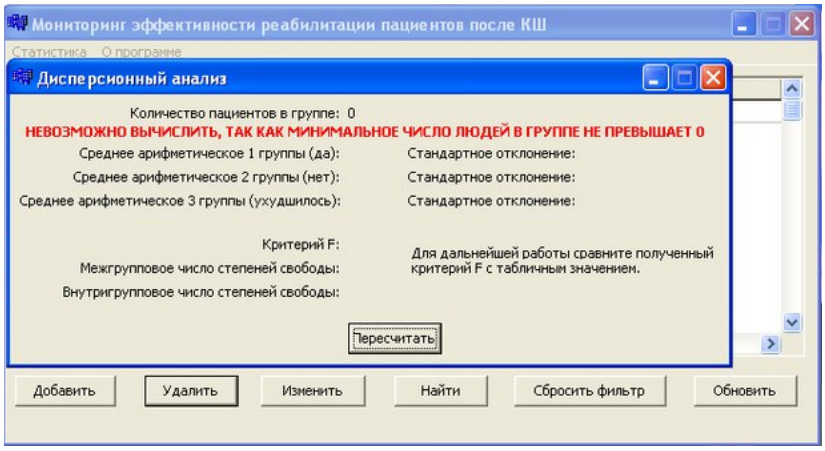

Рисунок 4.17 – Дисперсионный анализ на пустой базе

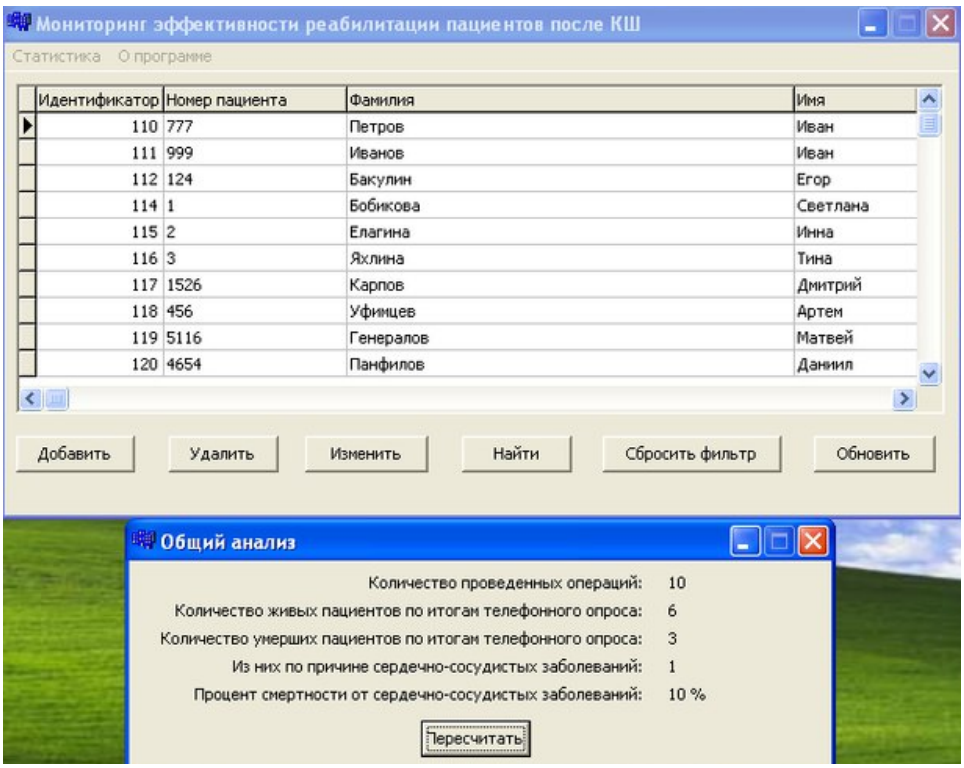

Рисунок 4.18 – Общий анализ заполненной базы

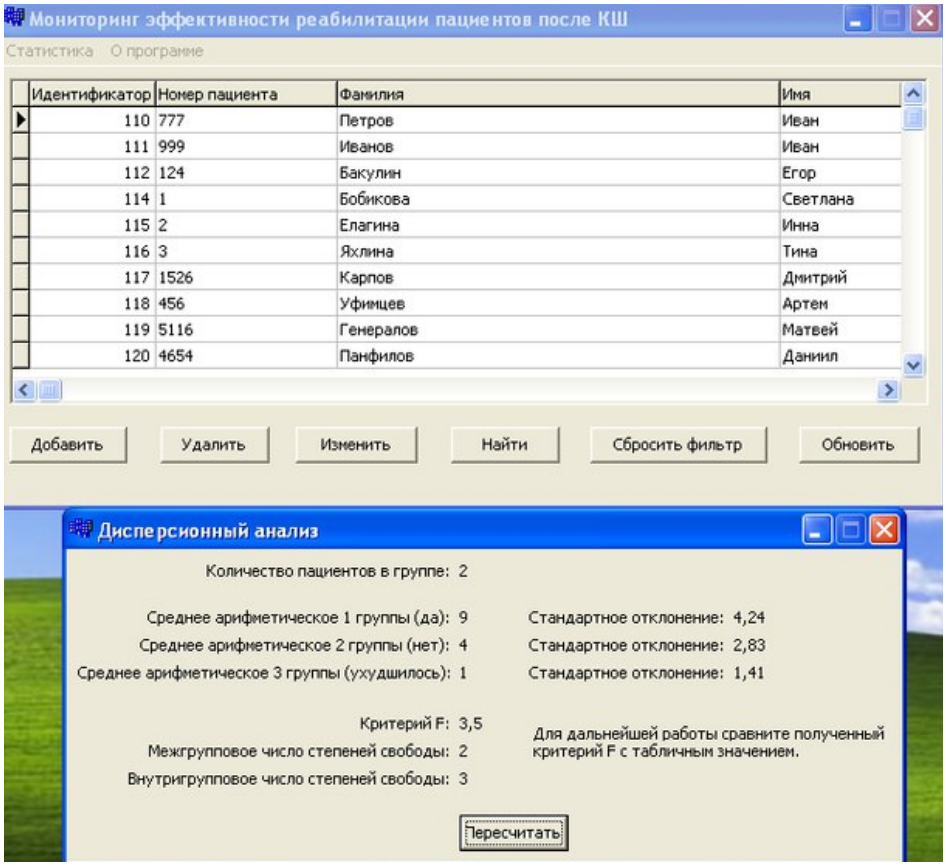

Рисунок 4.19 – Дисперсионный анализ заполненной базы

При нажатии на кнопку «Добавить», открывается окно добавления пациента, где после нажатия кнопки «Сохранить» пациент добавляется в базу (рисунок

4.20). При нажатии на кнопку «Обновить» в главном окне программы, происходит присвоение пациенту идентификатора.

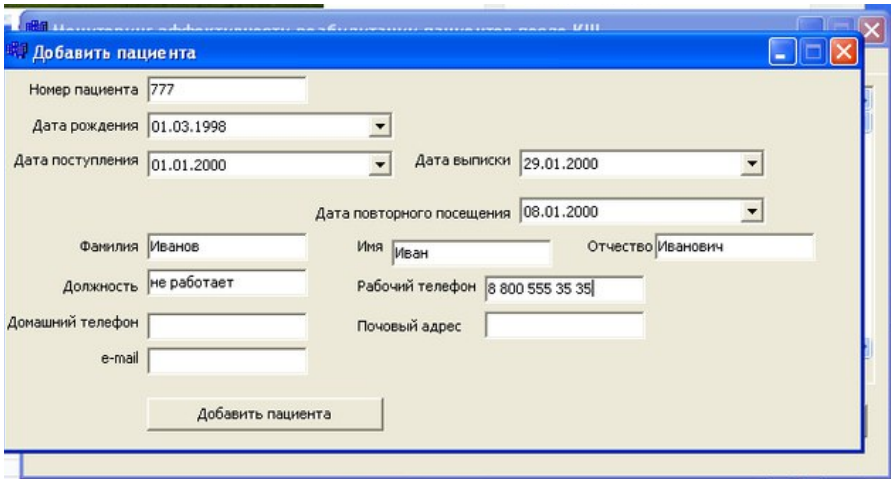

Рисунок 4.20 – Окно «Добавить пациента»

Если выбрать пациента в таблице и нажать на кнопку «Изменить» на главной форме приложения, открывается окно изменения основных данных пациента, где после нажатия на кнопку «Сохранить изменения», изменения сохраняются (рисунок 4.21).

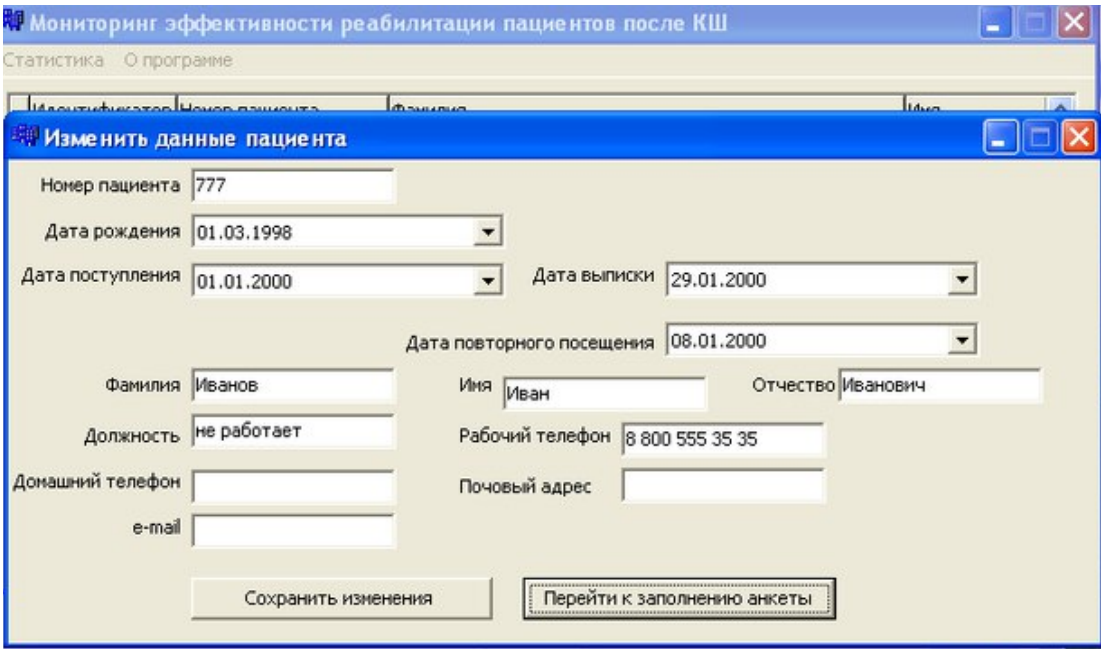

Рисунок 4.21 – Изменение основных данных пациента

При нажатии кнопки «Перейти к заполнению анкеты», открывается окно с пунктами анкеты (рисунок 4.22).

При нажатии на пункт «Выкопировка данных», открывается соответствующее окно. После заполнения всех необходимых полей и нажатия на кнопку «Сохранить» данные сохраняются в базу данных (рисунок 4.23). При нажатии на кнопку «Изменить», данные выгружаются в окно, где, если их изменить и нажать

на кнопку «Сохранить изменения», данные в базе будут обновлены (рисунок 4.24).

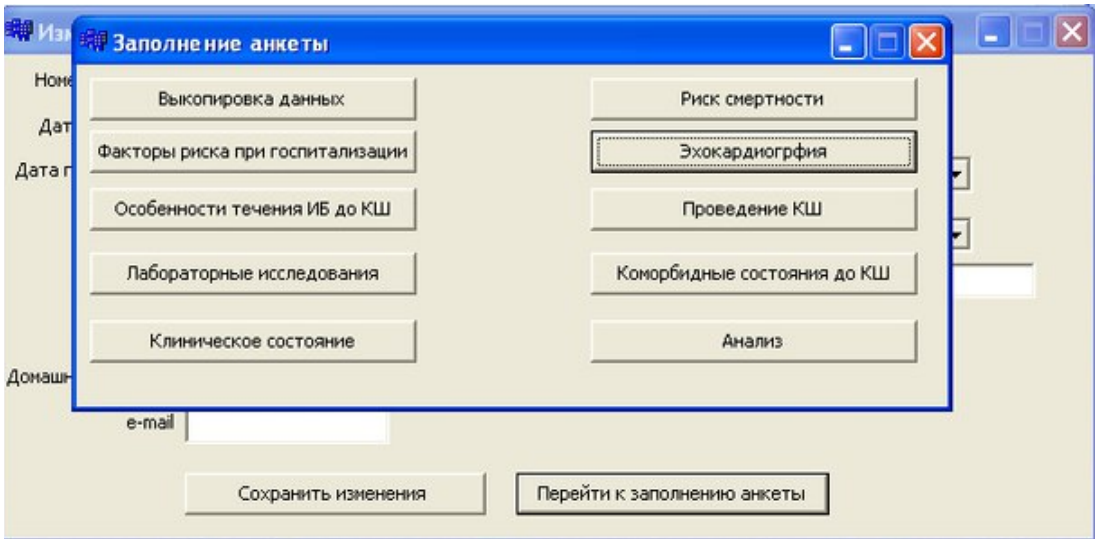

Рисунок 4.22 – Окно «Заполнение анкеты»

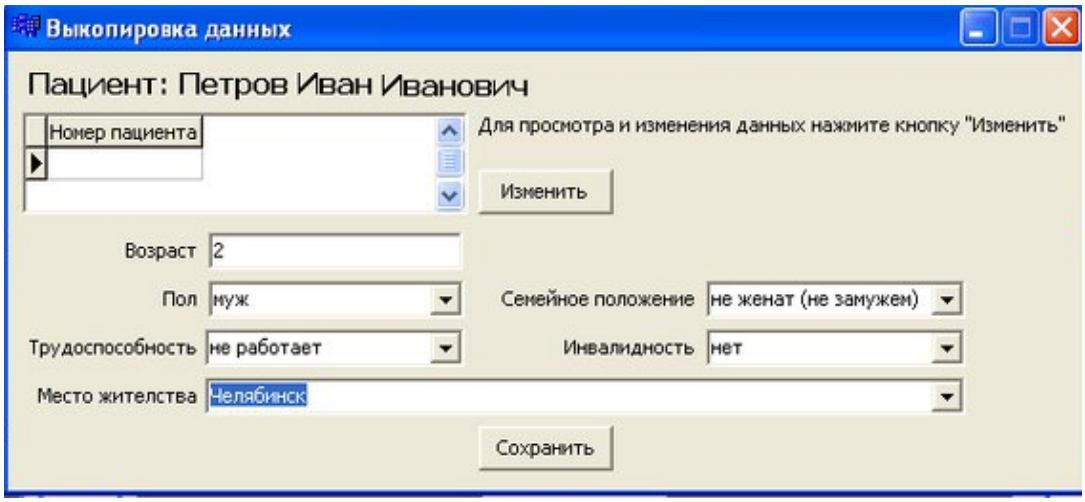

Рисунок 4.23 – Окно «Выкопировка данных»

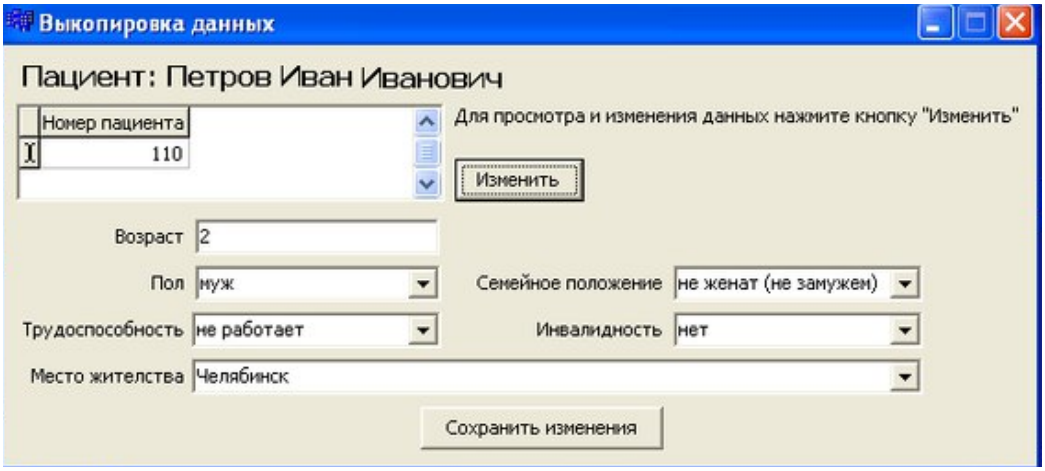

Рисунок 4.24 – Изменение выкопированных данных

Аналогичным образом была проверена работоспособность следующих кнопок окна «Заполнение анкеты»:

1) факторы риска при госпитализации;
2) особенности течения ИБ до КШ;
3) риск смертности;
4) проведение КШ;
5) коморбидные состояния до КШ;
6) анализ.

Проверка работы кнопок «Клиническое состояние», «Лабораторные исследования» и «Эхокардиография» проводилась следующим образом:

- заполнялись необходимые данные, и нажималась кнопка «Сохранить». Результат: данные успешно вносились в БД (рисунок 4.25);

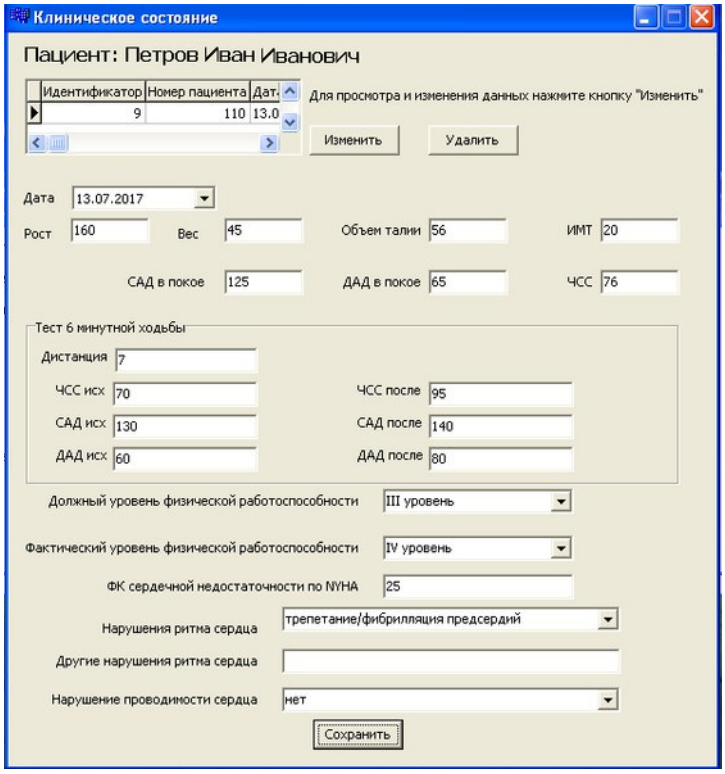

Рисунок 4.25 – Окно «Клиническое состояние»

- нажималась кнопка «Изменить». Результат: данные выгружались в окно, а при нажатии «Сохранить изменения» все внесенные изменения сохранялись (рисунок 4.26);

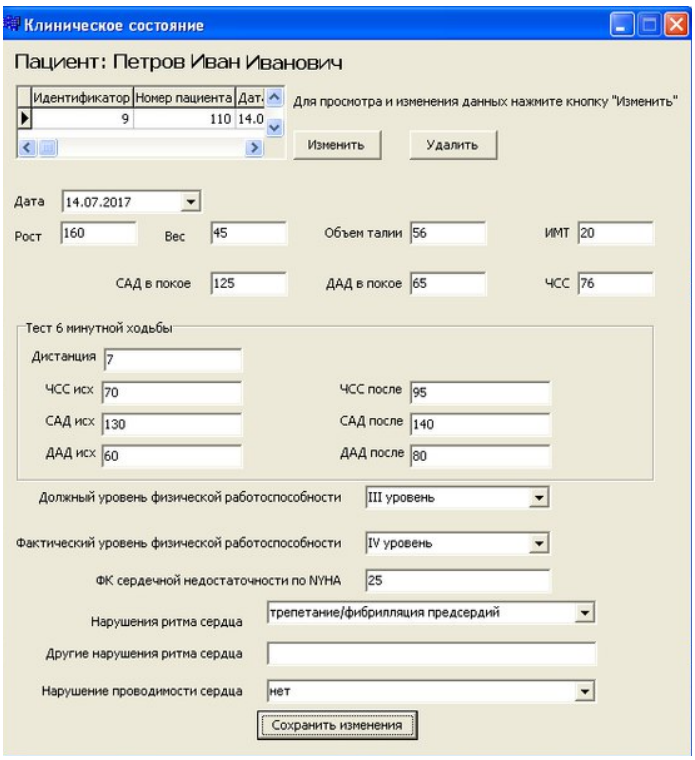

Рисунок 4.26 – Изменение данных клинического состояния

- нажималась кнопка «Удалить». Результат: вывод предупреждения об удалении, после положительного ответа запись удалялась из базы, а после отрицательного ответа – ничего не происходило (рисунок 4.27).

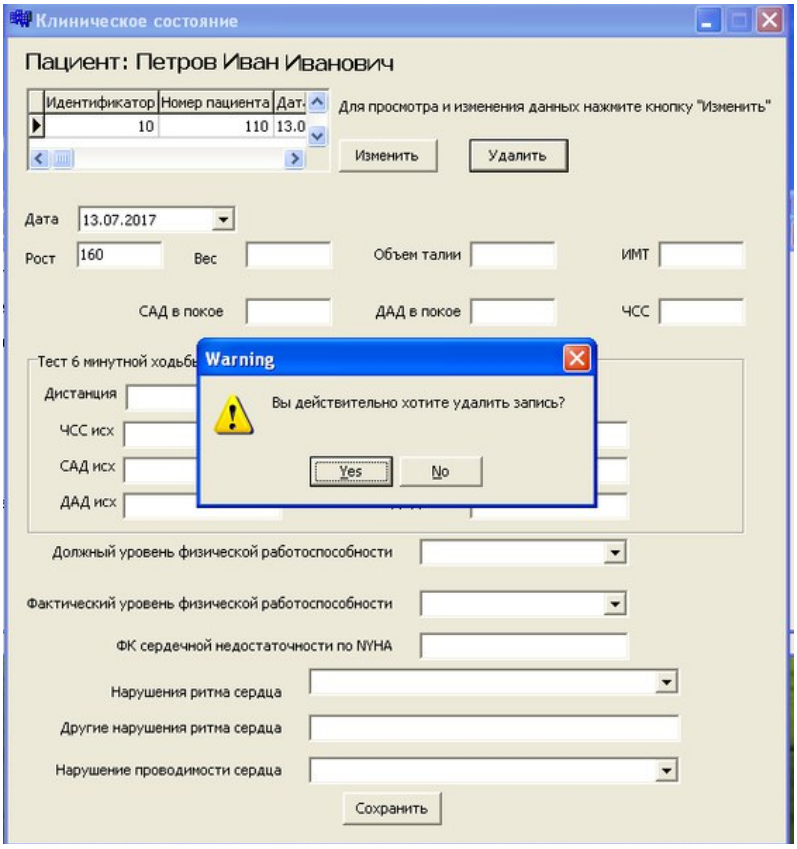

Рисунок 4.27 – Предупреждение об удалении записи

При нажатии кнопки «Удалить» на главном окне, выводится предупреждение об удалении, положительно ответив на которое, запись о пациенте удаляется из базы, а после отрицательного ответа – ничего не происходит (рисунок 4.28).

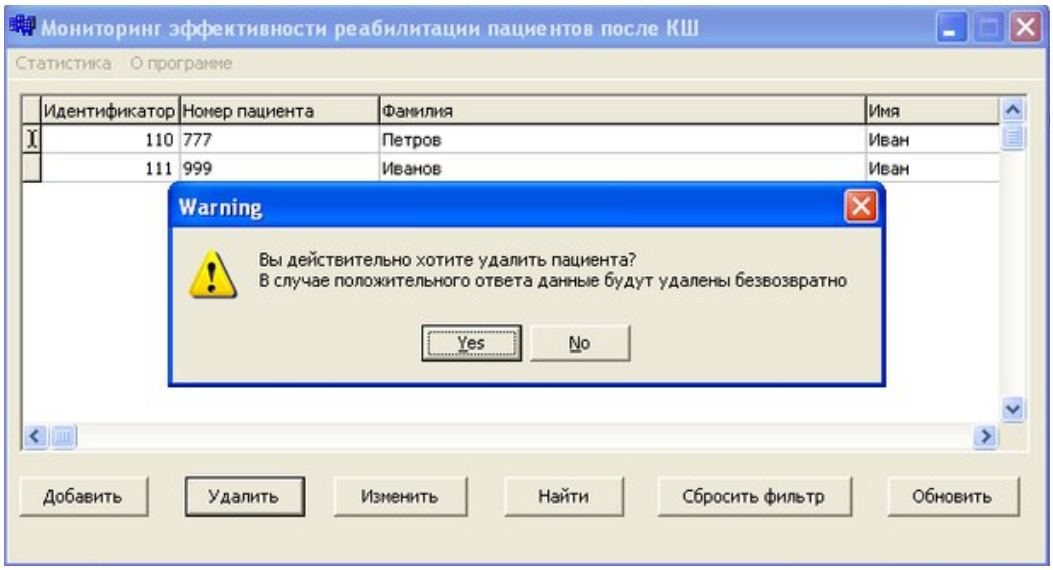

Рисунок 4.28 – Удаление всей записи о пациенте

При нажатии на кнопку «Найти» открывается окно поиска пациента. Была проверена работоспособность на поиск простого запроса (рисунок 4.29) и сложного запроса (рисунок 4.30).

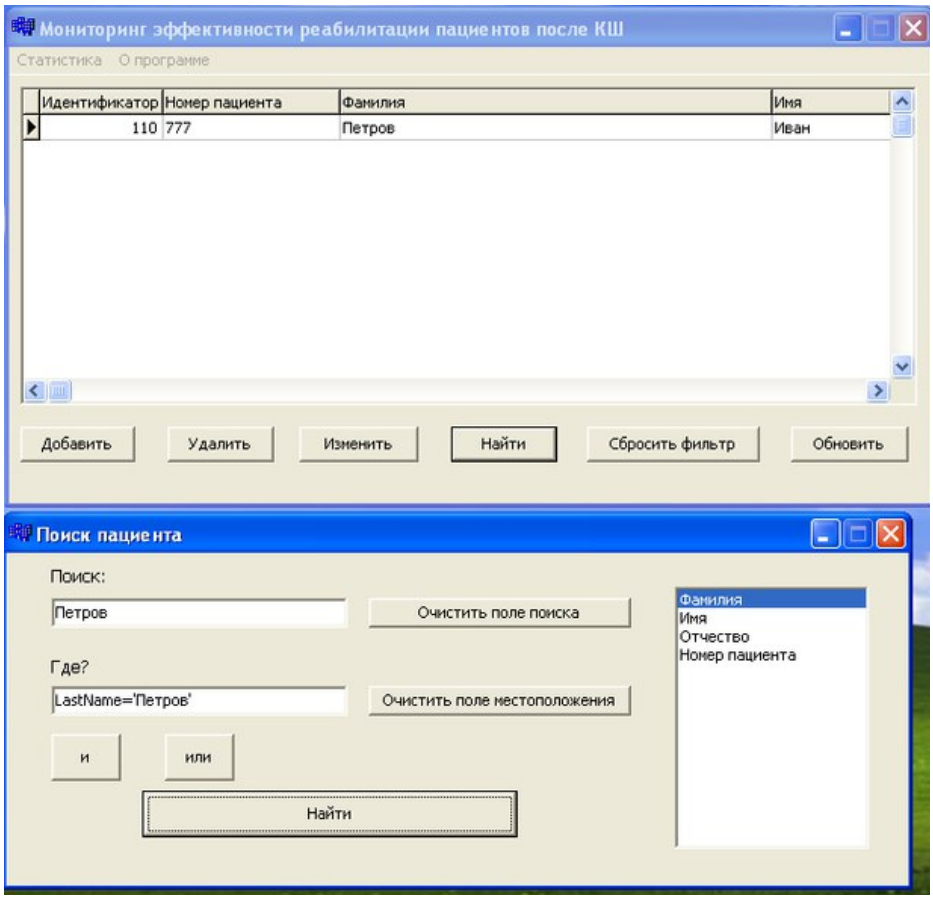

Рисунок 4.29 – Простой запрос поиска

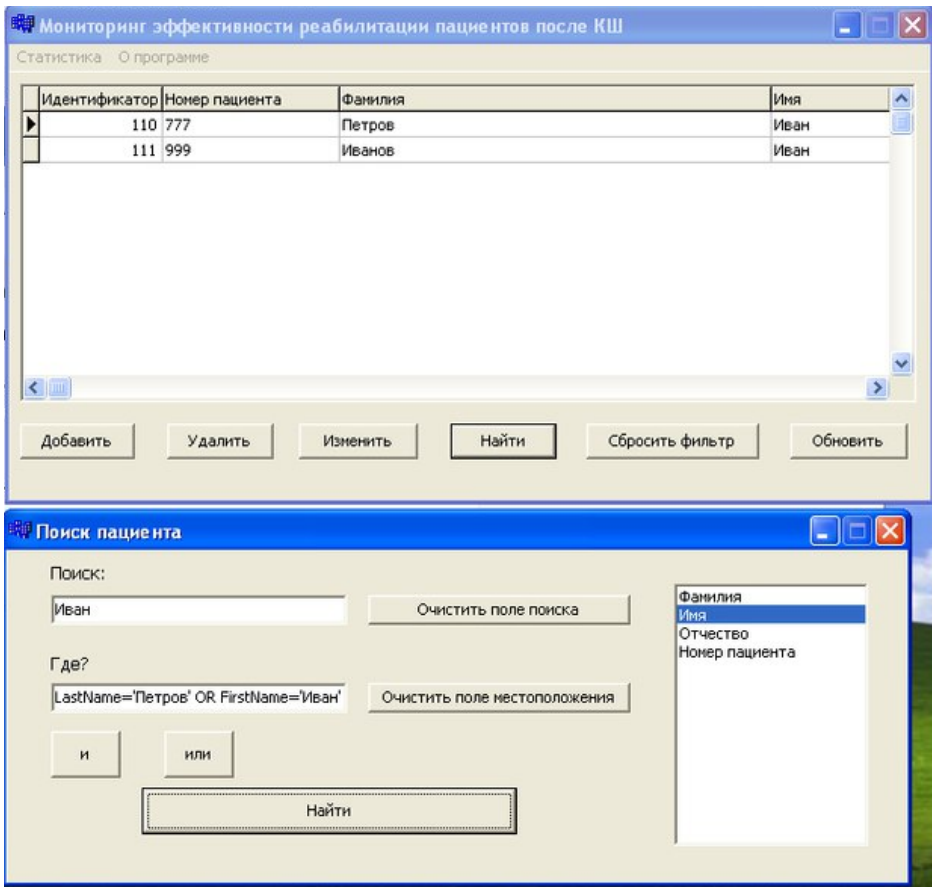

Рисунок 4.30 – Сложный запрос поиска

При нажатии на кнопку «Сбросить фильтр», примененная фильтрация сбрасывается.

4.4 Выводы по разделу

В результате разработаны алгоритмы работы различных окон приложения, а также работы всего приложения целиком.

Также был разработан пользовательский интерфейс. Приложение состоит из множества различных окон. В одном окне находится основная информация о пациентах, в другом реализовано редактирование основной информация, третье окно позволяет выбрать пункт анкеты, само же заполнение анкеты происходит в окнах с четвертого по тринадцатое, анализ можно просмотреть в четырнадцатом и пятнадцатом окне, а поиск пациентов реализован в шестнадцатом окне, общую информацию о программе можно просмотреть в семнадцатом окне.

По результатам проведенного программного эксперимента была подтверждена работоспособность разработанной системы. Обнаруженные в ходе промежуточных экспериментов ошибки устранялись в течение всего периода разработки.

ЗАКЛЮЧЕНИЕ

Данная работа посвящена разработке информационной системы мониторинга эффективности реабилитации пациентов после коронарного шунтирования.

В результате работы получен законченный программный продукт, который внедрен и используется в работе городского отделения реабилитации и кардиологии МБУЗ ГКБ №2.

Разработанная информационная система позволяет:

- хранить, изменять и просматривать данные о пациентах;
- просматривать статистику.

В ходе работы над проектом решены следующие задачи:

1) выполнен обзор существующих методов анализа данных;
2) разработана математическая модель для проведения дисперсионного анализа;
3) разработаны и реализованы алгоритмы работы приложения;
4) разработана база данных;
5) реализован пользовательский интерфейс;
6) выполнена проверка работы приложения.

В дальнейшем планируется реализовать и другие методы статистического анализа данных, создать возможность динамического выбора анализируемых критериев.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Айвазян, С.А. Интегральные индикаторы качества жизни населения: их построение и использование в социально-экономическом управлении и межрегиональных сопоставлениях.- М.: РАН, Центральный экономико-математический институт, 2000. - 32 с.
2. Бахвалов, Н.С. Численные методы. – М.: Наука, 1975. – 632 с.
3. Гмурман, В. Е. Теория вероятностей и математическая статистики: учебное пособие / В. Е. Гмурман. — М.: Юрайт, 2011.— 479 с.
4. Зубов, Н.Н. Математические методы и модели в фармацевтической науке и практике / Н.Н. Зубов, С.З. Умаров, С.А. Бунин. – СПб.: Изд-во Политехи, ун-та, 2008. – 249 с.
5. Избачков, Ю.. Информационные системы / Ю. Избачков, В. Петров. – СПб.: Питер, 2006.
6. Кант, В.И. Математические методы и моделирование в здравоохранении. - М.: Медицина, 1987. - 224 с.
7. Кирьянов, Б. Ф. Математические модели в здравоохранении: учебное пособие / Б.Ф. Кирьянов, М.С. Токмачов; НовГУ им. Ярослава Мудрого. – Великий Новгород, 2009. – 279 с.
8. Ланг, Т.А. Описание статистики в медицине. Руководство для авторов, редакторов и рецензентов / Т.А.Ланг, М.Сесик. - М. : Практическая медицина. – 2011. – 477с.
9. Леонов, В.П. Применение разведочного анализа для оценки исходных данных / Леонов В.П. // Электронный журнал Биометрика. – 2005. Режим доступа: http://www.biometrica.tomsk.ru/urolitiaz.htm/ свободный. .— Загл. с экрана. — Яз. рус.
10. Мандель, И.Д. Кластерный анализ / И.Д. Мандель. - М. : Финансы и статистика, 1988. – 176с.
11. Методы статистической обработки медицинских данных: Методические рекомендации для ораторов и аспирантов медицинских учебных заведений, научных работников / сост.: А.Г. Кочетов, О.В. Лянг., В.П. Масенко, И.В. Жиров, С.Н. Наконечников, С.Н. Терещенко. – М.: РКНПК, 2012. – 42 с.
12. Сайт «Статистика» [Электронный ресурс] : Первичный анализ данных. – Режим доступа: http://www.statistica.ru/glossary/general/pervichnyy-analiz-dannykh/, свободный - Загл. с экрана. — Яз. рус.
13. Banks J. Interpreting Simulation Software Checklists. – OR/MS Today, 1996, Num. 23. – P. 74 ¸ 78.
14. Brown, L.D. Interval estimation for a binomial proportion / L.D. Brown, T.T. Cai, A. Dasgupta // Statistical science. – 2001. – №2. – P. 101–133.
15. Garcia-Perez, M.A. On the confidence interval for the binomial parameter / M.A. Garcia-Perez // Quality and quantity. – 2005. – N 39. – P. 467–481.

16. Kaplan, R.M. New health promotion Indicators: the general health policy model / Health Promotion. – V. 3, № 1, 1996. – P. 35 ¸ 49.
17. MySQL.RU: Одобрено лучшими российскими программистами [Электронный ресурс] — Электрон. дан. — Режим доступа: http://www.mysql.ru/docs/mysql-man-4.0-ru/introduction.html#what-is, свободный. — Загл. с экрана. — Яз. рус.
18. Tandy, R.D. Technical Note: The Initial Stages of Statistical Data Analysis / R.D. Tandy // Journal of Athletic Training. – 1998. – P.69-71.
19. Tzoulaki, I. Risk of cardiovascular disease and all cause mortality among patients with type 2 diabetes prescribed oral antidiabetes drugs: retrospective cohort study using UK general practice research database / I. Tzoulaki, M. Molokhia, V. Curcin et al. // British Medical Journal. – 2009. - №339. - b4731.
20. Wilson, E.B. Probable inference, the law of succession, and statistical inference / E. B. Wilson // Journal of American Statistical Association. – 1927. – №22. – P. 209-212.
21. World Health Organization: Statistics of World Health Organization, 2001.

ТЕКСТ ПРОГРАММЫ

About.cpp – форма с информацией о программе

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "About.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TAboutForm *AboutForm;
//---------------------------------------------------------------------------
__fastcall TAboutForm::TAboutForm(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TAboutForm::Button1Click(TObject *Sender)
{
AboutForm->Close();
}
//---------------------------------------------------------------------------
```

About.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef AboutH
#define AboutH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
//---------------------------------------------------------------------------
class TAboutForm : public TForm
{
__published:      // IDE-managed Components
      TMemo *Memo1;
      TButton *Button1;
      void __fastcall Button1Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
      __fastcall TAboutForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TAboutForm *AboutForm;
//---------------------------------------------------------------------------
#endif
```

AddPatient1.cpp – форма добавления пациента

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include "AddPatient1.h"
#include "Change.h"
#include "Main1.h"
//---------------------------------------------------------------------------
```

```
#pragma package(smart_init)
#pragma resource "*.dfm"
TAddPatientForm *AddPatientForm;
//---------------------------------------------------------------------------
__fastcall TAddPatientForm::TAddPatientForm(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TAddPatientForm::Button1Click(TObject *Sender)
{
MainForm->MainTable->Active=true;
 MainForm->MainTable->Insert();
 MainForm->MainTable->FieldByName("NumberPat")->AsString= Edit1->Text;
 MainForm->MainTable->FieldByName("FirstName")->AsString= Edit5->Text;
 MainForm->MainTable->FieldByName("PaternalName")->AsString= Edit6->Text;
 MainForm->MainTable->FieldByName("Job")->AsString= Edit7->Text;
 MainForm->MainTable->FieldByName("WorkPhone")->AsString= Edit8->Text;
 MainForm->MainTable->FieldByName("HomePhone")->AsString= Edit9->Text;
 MainForm->MainTable->FieldByName("Email")->AsString= Edit11->Text;
 MainForm->MainTable->FieldByName("Post")->AsString= Edit10->Text;
 MainForm->MainTable->FieldByName("DateStart")->AsDateTime= DateTimePicker1->Date;
 MainForm->MainTable->FieldByName("DateEnd")->AsDateTime= DateTimePicker2->Date;
 MainForm->MainTable->FieldByName("BirthDate")->AsDateTime= DateTimePicker3->Date;
 MainForm->MainTable->FieldByName("DatePovt")->AsDateTime= DateTimePicker4->Date;
 MainForm->MainTable->FieldByName("LastName")->AsString= Edit4->Text;
 MainForm->MainTable->Post();
 MainForm->MainTable->Refresh();
  }
void __fastcall TAddPatientForm::Button2Click(TObject *Sender)
{
ChangeForm->ShowModal();
}
//---------------------------------------------------------------------------

void __fastcall TAddPatientForm::Button3Click(TObject *Sender)
{
 MainForm->MainTable->Edit();
 MainForm->MainTable->FieldByName("NumberPat")->AsString= Edit1->Text;
 MainForm->MainTable->FieldByName("FirstName")->AsString= Edit5->Text;
 MainForm->MainTable->FieldByName("PaternalName")->AsString= Edit6->Text;
 MainForm->MainTable->FieldByName("Job")->AsString= Edit7->Text;
 MainForm->MainTable->FieldByName("WorkPhone")->AsString= Edit8->Text;
 MainForm->MainTable->FieldByName("HomePhone")->AsString= Edit9->Text;
 MainForm->MainTable->FieldByName("Email")->AsString= Edit11->Text;
 MainForm->MainTable->FieldByName("Post")->AsString= Edit10->Text;
 MainForm->MainTable->FieldByName("DateStart")->AsDateTime= DateTimePicker1->Date;
 MainForm->MainTable->FieldByName("DateEnd")->AsDateTime= DateTimePicker2->Date;
 MainForm->MainTable->FieldByName("BirthDate")->AsDateTime= DateTimePicker3->Date;
 MainForm->MainTable->FieldByName("DatePovt")->AsDateTime= DateTimePicker4->Date;
 MainForm->MainTable->FieldByName("LastName")->AsString= Edit4->Text;
 MainForm->MainTable->Post();
}
//---------------------------------------------------------------------------
```

AddPatient1.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef AddPatient1H
#define AddPatient1H
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
```

```
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <DB.hpp>
#include <DBTables.hpp>
#include <DBGrids.hpp>
#include <Grids.hpp>
#include <DBCtrls.hpp>
#include <ExtCtrls.hpp>
#include <ComCtrls.hpp>
//---------------------------------------------------------------------------
class TAddPatientForm : public TForm
{
__published:      // IDE-managed Components
      TButton *Button1;
      TLabel *Label1;
      TLabel *Label2;
      TLabel *Label3;
      TLabel *Label4;
      TLabel *Label5;
      TLabel *Label6;
      TLabel *Label7;
      TLabel *Label8;
      TLabel *Label9;
      TLabel *Label10;
      TLabel *Label11;
      TEdit *Edit1;
      TEdit *Edit4;
      TEdit *Edit5;
      TEdit *Edit6;
      TEdit *Edit7;
      TEdit *Edit8;
      TEdit *Edit9;
      TEdit *Edit10;
      TEdit *Edit11;
      TDateTimePicker *DateTimePicker1;
      TDateTimePicker *DateTimePicker2;
      TButton *Button2;
      TButton *Button3;
      TLabel *Label12;
      TLabel *Label13;
      TDateTimePicker *DateTimePicker3;
      TDateTimePicker *DateTimePicker4;
      void __fastcall Button1Click(TObject *Sender);
      void __fastcall Button2Click(TObject *Sender);
      void __fastcall Button3Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
      __fastcall TAddPatientForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TAddPatientForm *AddPatientForm;
//---------------------------------------------------------------------------
#endif
```

Analiz.cpp – форма ввода данных для анализа

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop
#include "Analiz.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TAnalizForm *AnalizForm;
```

```
//---------------------------------------------------------------------------
__fastcall TAnalizForm::TAnalizForm(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TAnalizForm::Button1Click(TObject *Sender)
{
AnalizForm->AnalizTable->Insert();
AnalizForm->AnalizTable->FieldByName("Prod")->AsString= Edit1->Text;
AnalizForm->AnalizTable->FieldByName("KolGosp")->AsString= Edit2->Text;
AnalizForm->AnalizTable->FieldByName("Stat")->AsString= ComboBox1->Text;
AnalizForm->AnalizTable->FieldByName("PrSm")->AsString= ComboBox2->Text;
AnalizForm->AnalizTable->FieldByName("PrReab")->AsString= ComboBox5->Text;
AnalizForm->AnalizTable->FieldByName("Who")->AsString= ComboBox3->Text;
AnalizForm->AnalizTable->FieldByName("TrH")->AsString= ComboBox4->Text;
AnalizForm->AnalizTable->FieldByName("Tr")->AsString= ComboBox10->Text;
AnalizForm->AnalizTable->FieldByName("OIM")->AsString= ComboBox11->Text;
AnalizForm->AnalizTable->FieldByName("Sost")->AsString= ComboBox12->Text;
AnalizForm->AnalizTable->FieldByName("Tab")->AsString= ComboBox9->Text;
AnalizForm->AnalizTable->FieldByName("Ter")->AsString= ComboBox8->Text;
AnalizForm->AnalizTable->Post();
}
//---------------------------------------------------------------------------
void __fastcall TAnalizForm::ComboBox1Select(TObject *Sender)
{
 if (ComboBox1->ItemIndex==1)
      AnalizForm->ComboBox2->Enabled=true;
else{
      AnalizForm->ComboBox2->ItemIndex=-1;
      AnalizForm->ComboBox2->Enabled=false;
      }
}
//---------------------------------------------------------------------------
void __fastcall TAnalizForm::ComboBox5Select(TObject *Sender)
{
       if       ((ComboBox5->ItemIndex==4)||(ComboBox5->ItemIndex==5)||(ComboBox5-
>ItemIndex==6)||(ComboBox5->ItemIndex==7))
      AnalizForm->Edit1->Enabled=true;
else{
      AnalizForm->Edit1->Clear();
      AnalizForm->Edit1->Enabled=false;
      }
}
//---------------------------------------------------------------------------
void __fastcall TAnalizForm::Button2Click(TObject *Sender)
{
Button1->Visible=false;
Button3->Visible=true;
AnalizTable->Edit()    ;
Edit1->Text=AnalizTable->FieldByName("Prod")->AsString ;
Edit2->Text=AnalizTable->FieldByName("KolGosp")->AsString;
ComboBox1->Text=AnalizTable->FieldByName("Stat")->AsString;
ComboBox2->Text=AnalizTable->FieldByName("PrSm")->AsString;
ComboBox5->Text=AnalizTable->FieldByName("PrReab")->AsString;
ComboBox3->Text=AnalizTable->FieldByName("Who")->AsString;
ComboBox4->Text=AnalizTable->FieldByName("TrH")->AsString;
ComboBox10->Text=AnalizTable->FieldByName("Tr")->AsString;
ComboBox11->Text=AnalizTable->FieldByName("OIM")->AsString;
ComboBox12->Text=AnalizTable->FieldByName("Sost")->AsString;
ComboBox9->Text=AnalizTable->FieldByName("Tab")->AsString;
ComboBox8->Text=AnalizTable->FieldByName("Ter")->AsString;
```

```
}
//---------------------------------------------------------------------------
void __fastcall TAnalizForm::Button3Click(TObject *Sender)
{
AnalizForm->AnalizTable->Edit();
AnalizForm->AnalizTable->FieldByName("Prod")->AsString= Edit1->Text;
AnalizForm->AnalizTable->FieldByName("KolGosp")->AsString= Edit2->Text;
AnalizForm->AnalizTable->FieldByName("Stat")->AsString= ComboBox1->Text;
AnalizForm->AnalizTable->FieldByName("PrSm")->AsString= ComboBox2->Text;
AnalizForm->AnalizTable->FieldByName("PrReab")->AsString= ComboBox5->Text;
AnalizForm->AnalizTable->FieldByName("Who")->AsString= ComboBox3->Text;
AnalizForm->AnalizTable->FieldByName("TrH")->AsString= ComboBox4->Text;
AnalizForm->AnalizTable->FieldByName("Tr")->AsString= ComboBox10->Text;
AnalizForm->AnalizTable->FieldByName("OIM")->AsString= ComboBox11->Text;
AnalizForm->AnalizTable->FieldByName("Sost")->AsString= ComboBox12->Text;
AnalizForm->AnalizTable->FieldByName("Tab")->AsString= ComboBox9->Text;
AnalizForm->AnalizTable->FieldByName("Ter")->AsString= ComboBox8->Text;
AnalizForm->AnalizTable->Post();
AnalizForm->Close();
}
//---------------------------------------------------------------------------
```

Analiz.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef AnalizH
#define AnalizH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <DB.hpp>
#include <DBCtrls.hpp>
#include <DBGrids.hpp>
#include <DBTables.hpp>
#include <ExtCtrls.hpp>
#include <Grids.hpp>
//---------------------------------------------------------------------------
class TAnalizForm : public TForm
{
__published:      // IDE-managed Components
      TLabel *Label19;
      TLabel *Label20;
      TLabel *Label21;
      TLabel *Label22;
      TDBGrid *DBGrid1;
      TTable *AnalizTable;
      TDataSource *AnalizDataSource;
      TButton *Button1;
      TLabel *Label1;
      TComboBox *ComboBox1;
      TLabel *Label2;
      TComboBox *ComboBox2;
      TLabel *Label3;
      TComboBox *ComboBox3;
      TLabel *Label4;
      TComboBox *ComboBox4;
      TLabel *Label5;
      TComboBox *ComboBox5;
      TLabel *Label6;
      TLabel *Label7;
      TLabel *Label8;
```

```
        TComboBox *ComboBox8;
        TLabel *Label9;
        TComboBox *ComboBox9;
        TLabel *Label10;
        TComboBox *ComboBox10;
        TLabel *Label11;
        TComboBox *ComboBox11;
        TLabel *Label12;
        TComboBox *ComboBox12;
        TEdit *Edit1;
        TEdit *Edit2;
        TButton *Button2;
        TButton *Button3;
        TLabel *Label13;
        void __fastcall Button1Click(TObject *Sender);
        void __fastcall ComboBox1Select(TObject *Sender);
        void __fastcall ComboBox5Select(TObject *Sender);
        void __fastcall Button2Click(TObject *Sender);
        void __fastcall Button3Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
        __fastcall TAnalizForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TAnalizForm *AnalizForm;
//---------------------------------------------------------------------------
#endif
```

Change.cpp – форма выбора пункта меню

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop
#include "Change.h"
#include "CopyData.h"
#include "Risk.h"
#include "IshemDoKS.h"
#include "Lab.h"
#include "KlinSost.h"
#include "RiskSmert.h"
#include "Main1.h"
#include "Eho.h"
#include "ProvKS.h"
#include "KomSost.h"
#include "Analiz.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TChangeForm *ChangeForm;
//---------------------------------------------------------------------------
__fastcall TChangeForm::TChangeForm(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TChangeForm::Button1Click(TObject *Sender)
{
CopyDataForm->Label19->Caption=MainForm->MainTable->FieldByName("LastName")-
>AsString;
CopyDataForm->Label21->Left=CopyDataForm->Label19->Width+5+CopyDataForm->Label19-
>Left;
CopyDataForm->Label21->Caption=MainForm->MainTable->FieldByName("FirstName")-
>AsString;
```

```
CopyDataForm->Label22->Left=CopyDataForm->Label21->Width+5+CopyDataForm->Label21-
>Left;
CopyDataForm->Label22->Caption=MainForm->MainTable->FieldByName("PaternalName")-
>AsString;
CopyDataForm->Buttom1->Visible=true;
CopyDataForm->Button3->Visible=false;
CopyDataForm->Edit1->Clear();
CopyDataForm->ComboBox1->Text="";
CopyDataForm->ComboBox2->Text="";
CopyDataForm->ComboBox3->Text="";
CopyDataForm->ComboBox4->Text="";
CopyDataForm->ComboBox5->Text="";
CopyDataForm->ShowModal();
}
//---------------------------------------------------------------------------
void __fastcall TChangeForm::Button2Click(TObject *Sender)
{
RiskForm->Label19->Caption=MainForm->MainTable->FieldByName("LastName")->AsString;
RiskForm->Label21->Left=RiskForm->Label19->Width+5+RiskForm->Label19->Left;
RiskForm->Label21->Caption=MainForm->MainTable->FieldByName("FirstName")-
>AsString;
RiskForm->Label22->Left=RiskForm->Label21->Width+5+RiskForm->Label21->Left;
RiskForm->Label22->Caption=MainForm->MainTable->FieldByName("PaternalName")-
>AsString;
RiskForm->Button1->Visible=true;
RiskForm->Button3->Visible=false;
RiskForm->ComboBox1->Text="";
RiskForm->ComboBox2->Text="";
RiskForm->Edit1->Clear();
RiskForm->ComboBox3->Text="";
RiskForm->ComboBox4->Text="";
RiskForm->Edit2->Clear();
RiskForm->ComboBox5->Text="";
RiskForm->ComboBox6->Text="";
RiskForm->ComboBox7->Text="";
RiskForm->ComboBox8->Text="";
RiskForm->ComboBox9->Text="";
RiskForm->Edit4->Clear();
RiskForm->ShowModal();
}
//---------------------------------------------------------------------------
void __fastcall TChangeForm::Button3Click(TObject *Sender)
{
IshemDoKSForm->Button1->Visible=true;
IshemDoKSForm->Button3->Visible=false;
IshemDoKSForm->Label21->Caption=MainForm->MainTable->FieldByName("LastName")-
>AsString;
IshemDoKSForm->Label23->Left=IshemDoKSForm->Label21->Width+5+IshemDoKSForm-
>Label21->Left;
IshemDoKSForm->Label23->Caption=MainForm->MainTable->FieldByName("FirstName")-
>AsString;
IshemDoKSForm->Label24->Left=IshemDoKSForm->Label23->Width+5+IshemDoKSForm-
>Label23->Left;
IshemDoKSForm->Label24->Caption=MainForm->MainTable->FieldByName("PaternalName")-
>AsString;
IshemDoKSForm->Edit1->Clear();
IshemDoKSForm->ComboBox1->Text="";
IshemDoKSForm->ComboBox2->Text="";
IshemDoKSForm->ComboBox3->Text="";
IshemDoKSForm->ComboBox4->Text="";
IshemDoKSForm->Edit2->Clear();
IshemDoKSForm->DateTimePicker1->Date=Date();
```

```
IshemDoKSForm->ComboBox5->Text="";
IshemDoKSForm->Edit3->Clear();
IshemDoKSForm->ComboBox6->Text="";
IshemDoKSForm->ComboBox7->Text="";
IshemDoKSForm->ComboBox8->Text="";
IshemDoKSForm->ComboBox9->Text="";
IshemDoKSForm->ComboBox12->Text="";
IshemDoKSForm->ComboBox13->Text="";
IshemDoKSForm->ComboBox14->Text="";
IshemDoKSForm->ComboBox15->Text="";
IshemDoKSForm->ComboBox16->Text="";
IshemDoKSForm->ComboBox10->Text="";
IshemDoKSForm->ComboBox11->Text="";
IshemDoKSForm->ShowModal();
}
//---------------------------------------------------------------------------
void __fastcall TChangeForm::Button4Click(TObject *Sender)
{
LabForm->Label19->Caption=MainForm->MainTable->FieldByName("LastName")->AsString;
LabForm->Label21->Left=LabForm->Label19->Width+5+LabForm->Label19->Left;
LabForm->Label21->Caption=MainForm->MainTable->FieldByName("FirstName")->AsString;
LabForm->Label22->Left=LabForm->Label21->Width+5+LabForm->Label21->Left;
LabForm->Label22->Caption=MainForm->MainTable->FieldByName("PaternalName")-
>AsString;
LabForm->Button1->Visible=true;
LabForm->Button3->Visible=false;
LabForm->Edit1->Clear();
LabForm->Edit2->Clear();
LabForm->Edit3->Clear();
LabForm->Edit4->Clear();
LabForm->Edit5->Clear();
LabForm->Edit6->Clear();
LabForm->Edit7->Clear();
LabForm->Edit8->Clear();
LabForm->Edit9->Clear();
LabForm->Edit10->Clear();
LabForm->Edit11->Clear();
LabForm->Edit12->Clear();
LabForm->Edit13->Clear();
LabForm->Edit14->Clear();
LabForm->Edit15->Clear();
LabForm->Edit16->Clear();
LabForm->Edit17->Clear();
LabForm->DateTimePicker1->Date=Date();
LabForm->ShowModal();
}
//---------------------------------------------------------------------------

void __fastcall TChangeForm::Button5Click(TObject *Sender)
{
KlinSostForm->Label20->Caption=MainForm->MainTable->FieldByName("LastName")-
>AsString;
KlinSostForm->Label24->Left=KlinSostForm->Label20->Width+5+KlinSostForm->Label20-
>Left;
KlinSostForm->Label24->Caption=MainForm->MainTable->FieldByName("FirstName")-
>AsString;
KlinSostForm->Label25->Left=KlinSostForm->Label24->Width+5+KlinSostForm->Label24-
>Left;
KlinSostForm->Label25->Caption=MainForm->MainTable->FieldByName("PaternalName")-
>AsString;
KlinSostForm->Button1->Visible=true;
KlinSostForm->Button3->Visible=false;
```

```
KlinSostForm->Edit1->Clear();
KlinSostForm->Edit2->Clear();
KlinSostForm->Edit3->Clear();
KlinSostForm->Edit4->Clear();
KlinSostForm->Edit5->Clear();
KlinSostForm->Edit6->Clear();
KlinSostForm->Edit7->Clear();
KlinSostForm->Edit8->Clear();
KlinSostForm->Edit9->Clear();
KlinSostForm->Edit10->Clear();
KlinSostForm->Edit11->Clear();
KlinSostForm->Edit12->Clear();
KlinSostForm->Edit13->Clear();
KlinSostForm->Edit14->Clear();
KlinSostForm->Edit15->Clear();
KlinSostForm->Edit16->Clear();
KlinSostForm->ComboBox1->Text="";
KlinSostForm->ComboBox2->Text="";
KlinSostForm->ComboBox6->Text="";
KlinSostForm->ComboBox4->Text="";
KlinSostForm->DateTimePicker1->Date=Date();
KlinSostForm->ShowModal();
}
//---------------------------------------------------------------------------

void __fastcall TChangeForm::Button6Click(TObject *Sender)
{
RiskSmertForm->Label19->Caption=MainForm->MainTable->FieldByName("LastName")-
>AsString;
RiskSmertForm->Label21->Left=RiskSmertForm->Label19->Width+5+RiskSmertForm-
>Label19->Left;
RiskSmertForm->Label21->Caption=MainForm->MainTable->FieldByName("FirstName")-
>AsString;
RiskSmertForm->Label22->Left=RiskSmertForm->Label21->Width+5+RiskSmertForm-
>Label21->Left;
RiskSmertForm->Label22->Caption=MainForm->MainTable->FieldByName("PaternalName")-
>AsString;
RiskSmertForm->Button1->Visible=true;
RiskSmertForm->Button3->Visible=false;
RiskSmertForm->Edit1->Clear();
RiskSmertForm->Edit2->Clear();
RiskSmertForm->ShowModal();
}
//---------------------------------------------------------------------------

void __fastcall TChangeForm::Button7Click(TObject *Sender)
{
EhoForm->Label19->Caption=MainForm->MainTable->FieldByName("LastName")->AsString;
EhoForm->Label21->Left=EhoForm->Label19->Width+5+EhoForm->Label19->Left;
EhoForm->Label21->Caption=MainForm->MainTable->FieldByName("FirstName")->AsString;
EhoForm->Label22->Left=EhoForm->Label21->Width+5+EhoForm->Label21->Left;
EhoForm->Label22->Caption=MainForm->MainTable->FieldByName("PaternalName")-
>AsString;
EhoForm->Button1->Visible=true;
EhoForm->Button3->Visible=false;
EhoForm->Edit1->Clear();
EhoForm->Edit2->Clear();
EhoForm->Edit3->Clear();
EhoForm->Edit4->Clear();
EhoForm->Edit5->Clear();
EhoForm->Edit6->Clear();
EhoForm->Edit7->Clear();
```

```
EhoForm->Edit8->Clear();
EhoForm->Edit9->Clear();
EhoForm->Edit10->Clear();
EhoForm->Edit11->Clear();
EhoForm->Edit12->Clear();
EhoForm->Edit13->Clear();
EhoForm->DateTimePicker1->Date=Date();
EhoForm->ShowModal();
}
//---------------------------------------------------------------------------

void __fastcall TChangeForm::Button8Click(TObject *Sender)
{
ProvKSForm->Label27->Caption=MainForm->MainTable->FieldByName("LastName")-
>AsString;
ProvKSForm->Label29->Left=ProvKSForm->Label27->Width+5+ProvKSForm->Label27->Left;
ProvKSForm->Label29->Caption=MainForm->MainTable->FieldByName("FirstName")-
>AsString;
ProvKSForm->Label33->Left=ProvKSForm->Label29->Width+5+ProvKSForm->Label29->Left;
ProvKSForm->Label33->Caption=MainForm->MainTable->FieldByName("PaternalName")-
>AsString;
ProvKSForm->Button1->Visible=true;
ProvKSForm->Button3->Visible=false;
ProvKSForm->Edit1->Clear();
ProvKSForm->Edit2->Clear();
ProvKSForm->ComboBox1->Text="";
ProvKSForm->ComboBox2->Text="";
ProvKSForm->ComboBox3->Text="";
ProvKSForm->ComboBox4->Text="";
ProvKSForm->ComboBox5->Text="";
ProvKSForm->ComboBox6->Text="";
ProvKSForm->ComboBox7->Text="";
ProvKSForm->ComboBox8->Text="";
ProvKSForm->ComboBox9->Text="";
ProvKSForm->ComboBox10->Text="";
ProvKSForm->ComboBox11->Text="";
ProvKSForm->ComboBox12->Text="";
ProvKSForm->ComboBox13->Text="";
ProvKSForm->ComboBox14->Text="";
ProvKSForm->ComboBox15->Text="";
ProvKSForm->ComboBox16->Text="";
ProvKSForm->ComboBox17->Text="";
ProvKSForm->ComboBox18->Text="";
ProvKSForm->ComboBox19->Text="";
ProvKSForm->ComboBox20->Text="";
ProvKSForm->ComboBox21->Text="";
ProvKSForm->ComboBox22->Text="";
ProvKSForm->ComboBox23->Text="";
ProvKSForm->DateTimePicker1->Date=Date();
ProvKSForm->ShowModal();
}
//---------------------------------------------------------------------------

void __fastcall TChangeForm::Button9Click(TObject *Sender)
{
 KomSostForm->Button1->Visible=true;
 KomSostForm->Button3->Visible=false;
 KomSostForm->Edit1->Clear();
KomSostForm->Edit2->Clear();
KomSostForm->Edit3->Clear();
KomSostForm->Edit4->Clear();
KomSostForm->Edit5->Clear();
```

```
KomSostForm->Edit6->Clear();
KomSostForm->Edit7->Clear();
KomSostForm->ComboBox1->Text="";
KomSostForm->ComboBox2->Text="";
KomSostForm->ComboBox3->Text="";
KomSostForm->ComboBox4->Text="";
KomSostForm->ComboBox5->Text="";
KomSostForm->ComboBox6->Text="";
KomSostForm->ComboBox7->Text="";
KomSostForm->ComboBox8->Text="";
KomSostForm->ComboBox9->Text="";
KomSostForm->ComboBox10->Text="";
KomSostForm->ComboBox11->Text="";
KomSostForm->ComboBox12->Text="";
KomSostForm->ComboBox13->Text="";
KomSostForm->ComboBox14->Text="";
KomSostForm->ComboBox15->Text="";
KomSostForm->ComboBox16->Text="";
KomSostForm->ComboBox17->Text="";
KomSostForm->ComboBox18->Text="";
KomSostForm->ComboBox19->Text="";
KomSostForm->ComboBox20->Text="";
        KomSostForm->Label19->Caption=MainForm->MainTable->FieldByName("LastName")-
>AsString;
     KomSostForm->Label21->Left=KomSostForm->Label19->Width+5+KomSostForm->Label19-
>Left;
       KomSostForm->Label21->Caption=MainForm->MainTable->FieldByName("FirstName")-
>AsString;
     KomSostForm->Label33->Left=KomSostForm->Label21->Width+5+KomSostForm->Label21-
>Left;
    KomSostForm->Label33->Caption=MainForm->MainTable->FieldByName("PaternalName")-
>AsString;
 KomSostForm->ShowModal();

}
//---------------------------------------------------------------------------

void __fastcall TChangeForm::Button10Click(TObject *Sender)
{
AnalizForm->Edit1->Clear();
AnalizForm->Edit2->Clear();
AnalizForm->ComboBox1->Text="";
AnalizForm->ComboBox2->Text="";
AnalizForm->ComboBox5->Text="";
AnalizForm->ComboBox3->Text="";
AnalizForm->ComboBox4->Text="";
AnalizForm->ComboBox10->Text="";
AnalizForm->ComboBox11->Text="";
AnalizForm->ComboBox12->Text="";
AnalizForm->ComboBox9->Text="";
AnalizForm->ComboBox8->Text="";
AnalizForm->Button3->Visible=false;
AnalizForm->Button1->Visible=true;
AnalizForm->Label19->Caption=MainForm->MainTable->FieldByName("LastName")-
>AsString;
AnalizForm->Label21->Left=AnalizForm->Label19->Width+5+AnalizForm->Label19->Left;
AnalizForm->Label21->Caption=MainForm->MainTable->FieldByName("FirstName")-
>AsString;
AnalizForm->Label22->Left=AnalizForm->Label21->Width+5+AnalizForm->Label21->Left;
AnalizForm->Label22->Caption=MainForm->MainTable->FieldByName("PaternalName")-
>AsString;
AnalizForm->ShowModal();
```

```
}
//---------------------------------------------------------------------------
```

Change.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef ChangeH
#define ChangeH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
//---------------------------------------------------------------------------
class TChangeForm : public TForm
{
__published:      // IDE-managed Components
      TButton *Button1;
      TButton *Button2;
      TButton *Button3;
      TButton *Button4;
      TButton *Button5;
      TButton *Button6;
      TButton *Button7;
      TButton *Button8;
      TButton *Button9;
      TButton *Button10;
      void __fastcall Button1Click(TObject *Sender);
      void __fastcall Button2Click(TObject *Sender);
      void __fastcall Button3Click(TObject *Sender);
      void __fastcall Button4Click(TObject *Sender);
      void __fastcall Button5Click(TObject *Sender);
      void __fastcall Button6Click(TObject *Sender);
      void __fastcall Button7Click(TObject *Sender);
      void __fastcall Button8Click(TObject *Sender);
      void __fastcall Button9Click(TObject *Sender);
      void __fastcall Button10Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
      __fastcall TChangeForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TChangeForm *ChangeForm;
//---------------------------------------------------------------------------
#endif
```

CopyData.cpp – форма выкопировки данных

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "CopyData.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TCopyDataForm *CopyDataForm;
//---------------------------------------------------------------------------
__fastcall TCopyDataForm::TCopyDataForm(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------
```

```
void __fastcall TCopyDataForm::Buttom1Click(TObject *Sender)
{
CopyDataForm->DataCopyTable->Insert();
CopyDataForm->DataCopyTable->FieldByName("Age")->AsString= Edit1->Text;
CopyDataForm->DataCopyTable->FieldByName("Sex")->AsString= ComboBox1->Text;
CopyDataForm->DataCopyTable->FieldByName("MaritalStatus")->AsString=     ComboBox2-
>Text;
CopyDataForm->DataCopyTable->FieldByName("LaborAbility")->AsString=      ComboBox3-
>Text;
CopyDataForm->DataCopyTable->FieldByName("Disability")->AsString= ComboBox4->Text;
CopyDataForm->DataCopyTable->FieldByName("Location")->AsString= ComboBox5->Text;
CopyDataForm->DataCopyTable->Post();
CopyDataForm->Close();
}
//---------------------------------------------------------------------------
void __fastcall TCopyDataForm::Button3Click(TObject *Sender)
{
CopyDataForm->DataCopyTable->Edit();
CopyDataForm->DataCopyTable->FieldByName("Age")->AsString= Edit1->Text;
CopyDataForm->DataCopyTable->FieldByName("Sex")->AsString= ComboBox1->Text;
CopyDataForm->DataCopyTable->FieldByName("MaritalStatus")->AsString=     ComboBox2-
>Text;
CopyDataForm->DataCopyTable->FieldByName("LaborAbility")->AsString=      ComboBox3-
>Text;
CopyDataForm->DataCopyTable->FieldByName("Disability")->AsString= ComboBox4->Text;
CopyDataForm->DataCopyTable->FieldByName("Location")->AsString= ComboBox5->Text;
CopyDataForm->DataCopyTable->Post();
CopyDataForm->Close();
}
//---------------------------------------------------------------------------

void __fastcall TCopyDataForm::Button2Click(TObject *Sender)
{
Buttom1->Visible=false;
Button3->Visible=true;
CopyDataForm->DataCopyTable->Edit();
Edit1->Text=DataCopyTable->FieldByName("Age")->AsString;
ComboBox1->Text=DataCopyTable->FieldByName("Sex")->AsString;
ComboBox2->Text=DataCopyTable->FieldByName("MaritalStatus")->AsString;
ComboBox3->Text=DataCopyTable->FieldByName("LaborAbility")->AsString;
ComboBox4->Text=DataCopyTable->FieldByName("Disability")->AsString;
ComboBox5->Text=DataCopyTable->FieldByName("Location")->AsString;
}
//---------------------------------------------------------------------------
```

CopyData.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef CopyDataH
#define CopyDataH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <ComCtrls.hpp>
#include <DB.hpp>
#include <DBCtrls.hpp>
#include <DBGrids.hpp>
#include <DBTables.hpp>
#include <ExtCtrls.hpp>
#include <Grids.hpp>
//---------------------------------------------------------------------------
```

```
class TCopyDataForm : public TForm
{
__published:      // IDE-managed Components
      TDBGrid *DBGrid1;
      TTable *DataCopyTable;
      TDataSource *CopyDataSource1;
      TLabel *Label2;
      TEdit *Edit1;
      TComboBox *ComboBox1;
      TLabel *Label3;
      TComboBox *ComboBox2;
      TLabel *Label4;
      TComboBox *ComboBox3;
      TLabel *Label5;
      TComboBox *ComboBox4;
      TLabel *Label6;
      TComboBox *ComboBox5;
      TLabel *Label7;
      TButton *Buttom1;
      TLabel *Label19;
      TLabel *Label20;
      TLabel *Label21;
      TLabel *Label22;
      TLabel *Label32;
      TButton *Button2;
      TButton *Button3;
      void __fastcall Buttom1Click(TObject *Sender);
      void __fastcall Button3Click(TObject *Sender);
      void __fastcall Button2Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
      __fastcall TCopyDataForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TCopyDataForm *CopyDataForm;
//---------------------------------------------------------------------------
#endif
```

Disp.cpp – форма с дисперсионным анализом

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop
#include <math.h>
#include "Disp.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TDispForm *DispForm;
//---------------------------------------------------------------------------
__fastcall TDispForm::TDispForm(TComponent* Owner)
      : TForm(Owner)
{

}
//---------------------------------------------------------------------------
void __fastcall TDispForm::Button1Click(TObject *Sender)
{
double gr=3; //Количество групп
//Считаем количество людей
double kolgr1=0, kolgr2=0, kolgr3=0;     //количество людей в каждой из групп
пациентов
Query1->Close();
```

```
Query1->SQL->Clear();
Query1->SQL->Add("Select count(*) as Kolvo from Analiz Where Sost='да'");
Query1->Open();
kolgr1=Query1->FieldByName("Kolvo")->AsInteger;

Query1->Close();
Query1->SQL->Clear();
Query1->SQL->Add("Select count(*) as Kolvo1 from Analiz Where Sost='нет'");
Query1->Open();
kolgr2=Query1->FieldByName("Kolvo1")->AsInteger;

Query1->Close();
Query1->SQL->Clear();
Query1->SQL->Add("Select count(*) as Kolvo2 from Analiz Where Sost='ухудшилось'");
Query1->Open();
kolgr3=Query1->FieldByName("Kolvo2")->AsInteger;

//Выбираем группу с минимальным количеством людей
double min=kolgr1;
if (min>kolgr2) min=kolgr2;
if (min>kolgr3) min=kolgr3;
Label4->Caption=min;
if (min>0) {


/***** 1 ГРУППА *****/
/*Считаем среднее арифметическое 1 группы  */
//1 группа сумма ряда
Query1->Close();
Query1->SQL->Clear();
Query1->SQL->Add("Select Tr From Analiz Where Sost='да'");
Query1->Open();
Query1->First();
int ch=1; //счетчик шагов
double summ=0; // сумма ряда
while(ch<=min)
{
      ch++;
      summ=summ+Query1->FieldByName("Tr")->Value;
      Query1->Next();
}

 //1 группа среднее арифметическое
double srar1;
srar1=(floor(summ/min*100+0.5))/100;
Label6->Caption=srar1;
 /*Стандартное отклонение*/
//сумма разниц квадратов 1 группы
Query1->First();
double summkv1=0;
double razn=0;
int ch1=1;
while(ch1<=min)
{
      ch1++;
      razn=srar1-Query1->FieldByName("Tr")->Value;
      summkv1=summkv1+razn*razn;
      Query1->Next();
}

//дисперсия 1 группы
double disp1;
```

```
disp1=summkv1/(min-1);

//стандартное отклонение 1 группы
double st1;
st1=(floor(sqrt(disp1)*100+0.5))/100;
Label9->Caption=st1;

/***** 2 ГРУППА *****/
/*Считаем среднее арифметическое 2 группы  */
//2 группа сумма ряда
Query1->Close();
Query1->SQL->Clear();
Query1->SQL->Add("Select Tr From Analiz Where Sost='нет'");
Query1->Open();
Query1->First();
int ch12=1; //счетчик шагов
double summ2=0; // сумма ряда
while(ch12<=min)
{
      ch12++;
      summ2=summ2+Query1->FieldByName("Tr")->Value;
      Query1->Next();
}
 //2 группа среднее арифметическое
double srar2;
srar2=(floor(summ2/min*100+0.5))/100;
Label11->Caption=srar2;
 /*Стандартное отклонение*/
//сумма разниц квадратов 2 группы
Query1->First();
double summkv2=0;
double razn2=0;
int ch2=1;
while(ch2<=min)
{
      ch2++;
      razn2=srar2-Query1->FieldByName("Tr")->Value;
      summkv2=summkv2+razn2*razn2;
      Query1->Next();
}
//дисперсия 2 группы
double disp2;
disp2=summkv2/(min-1);
//стандартное отклонение 2 группы
double st2;
st2=(floor(sqrt(disp2)*100+0.5))/100;
Label14->Caption=st2;

/***** 3 ГРУППА *****/
/*Считаем среднее арифметическое 3 группы  */
//3 группа сумма ряда
Query1->Close();
Query1->SQL->Clear();
Query1->SQL->Add("Select Tr From Analiz Where Sost='ухудшилось'");
Query1->Open();
Query1->First();
int ch13=1; //счетчик шагов
double summ3=0; // сумма ряда
while(ch13<=min)
{
      ch13++;
      summ3=summ3+Query1->FieldByName("Tr")->Value;
```

```
      Query1->Next();
}
//3 группа среднее арифметическое
double srar3;
srar3=(floor(summ3/min*100+0.5))/100;
Label16->Caption=srar3;
 /*Стандартное отклонение*/
//сумма разниц квадратов 3 группы
Query1->First();
double summkv3=0;
double razn3=0;
int ch3=1;
while(ch3<=min)
{
      ch3++;
      razn3=srar3-Query1->FieldByName("Tr")->Value;

      summkv3=summkv3+razn3*razn3;
      Query1->Next();
}
//дисперсия 3 группы
double disp3;
disp3=summkv3/(min-1);
//стандартное отклонение 3 группы
double st3;
st3=(floor(sqrt(disp3)*100+0.5))/100;
Label19->Caption=st3;

/***** ВНУТРИГРУППОВАЯ ДИСПЕРСИЯ *****/
double Svn;
Svn=(disp1+disp2+disp3)/gr;

/***** СРЕДНЕЕ 3 ВЫБОРОЧНЫХ СРЕДНИХ*****/
double Srar;
Srar=(srar1+srar2+srar3)/gr;

/***** СТАНДАРТНОЕ ОТКЛОНЕНИЕ *****/
double SSmeg;
SSmeg=sqrt(((srar1-Srar)*(srar1-Srar)+            (srar2-Srar)*(srar2-Srar)+(srar3-
Srar)*(srar3-Srar))/(gr-1));

/***** МЕЖГРУППОВАЯ ДИСПЕРСИЯ *****/
double Smeg;
Smeg=min*SSmeg*SSmeg;

/***** КРИТЕРИЙ F *****/
double F;
F=(floor(Smeg/Svn*100+0.5))/100;
Label24->Caption=F;

/***** МЕЖГРУППОВОЕ ЧИСЛО СТЕПЕНЕЙ СВОБОДЫ *****/
double Vmeg;
Vmeg=gr-1;
Label25->Caption=Vmeg;

/***** ВНУТРИГРУППОВОЕ ЧИСЛО СТЕПЕНЕЙ СВОБОДЫ *****/
double Vvn;
Vvn=gr*(min-1);
Label26->Caption=Vvn;
}
else{
      Label21->Visible=true;
```

```
      Label21->Caption="НЕВОЗМОЖНО ВЫЧИСЛИТЬ, ТАК КАК МИНИМАЛЬНОЕ ЧИСЛО ЛЮДЕЙ В
ГРУППЕ НЕ ПРЕВЫШАЕТ 0";
}
}
//---------------------------------------------------------------------------
```

Disp.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef DispH
#define DispH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <DB.hpp>
#include <DBGrids.hpp>
#include <DBTables.hpp>
#include <Grids.hpp>
#include <DBCtrls.hpp>
#include <ExtCtrls.hpp>
//---------------------------------------------------------------------------
class TDispForm : public TForm
{
__published:      // IDE-managed Components
      TDBGrid *DBGrid1;
      TQuery *Query1;
      TDataSource *DataSource1;
      TLabel *Label4;
      TLabel *Label6;
      TLabel *Label9;
      TLabel *Label11;
      TLabel *Label14;
      TLabel *Label16;
      TLabel *Label19;
      TLabel *Label24;
      TLabel *Label25;
      TLabel *Label26;
      TLabel *Label1;
      TLabel *Label2;
      TLabel *Label3;
      TLabel *Label8;
      TLabel *Label10;
      TLabel *Label5;
      TLabel *Label12;
      TLabel *Label13;
      TLabel *Label7;
      TLabel *Label15;
      TLabel *Label17;
      TLabel *Label18;
      TLabel *Label20;
      TLabel *Label21;
      TButton *Button1;
      void __fastcall Button1Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
      __fastcall TDispForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TDispForm *DispForm;
//---------------------------------------------------------------------------
#endif
```

Eho.cpp – форма эхокардиографии

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "Eho.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TEhoForm *EhoForm;
//---------------------------------------------------------------------------
__fastcall TEhoForm::TEhoForm(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TEhoForm::Button1Click(TObject *Sender)
{
EhoForm->EhoTable->Insert();
EhoForm->EhoTable->FieldByName("LevPr")->AsString= Edit1->Text;
EhoForm->EhoTable->FieldByName("VLevPr")->AsString= Edit2->Text;
EhoForm->EhoTable->FieldByName("IndVLevPr")->AsString= Edit3->Text;
EhoForm->EhoTable->FieldByName("KDR")->AsString= Edit4->Text;
EhoForm->EhoTable->FieldByName("KSR")->AsString= Edit5->Text;
EhoForm->EhoTable->FieldByName("TMGP")->AsString= Edit6->Text;
EhoForm->EhoTable->FieldByName("TZS")->AsString= Edit7->Text;
EhoForm->EhoTable->FieldByName("FV")->AsString= Edit8->Text;
EhoForm->EhoTable->FieldByName("FU")->AsString= Edit9->Text;
EhoForm->EhoTable->FieldByName("MMiok")->AsString= Edit10->Text;
EhoForm->EhoTable->FieldByName("IndMMiok")->AsString= Edit11->Text;
EhoForm->EhoTable->FieldByName("MGDAor")->AsString= Edit12->Text;
EhoForm->EhoTable->FieldByName("MGDMitr")->AsString= Edit13->Text;
EhoForm->EhoTable->FieldByName("DataOsm")->AsDateTime= DateTimePicker1->Date;
EhoForm->EhoTable->Post();
EhoTable->Refresh();
}
//---------------------------------------------------------------------------
void __fastcall TEhoForm::Button2Click(TObject *Sender)
{
Button1->Visible=false;
Button3->Visible=true;
EhoTable->Edit();
Edit1->Text=EhoTable->FieldByName("LevPr")->AsString;
Edit2->Text=EhoTable->FieldByName("VLevPr")->AsString;
Edit3->Text=EhoTable->FieldByName("IndVLevPr")->AsString;
Edit4->Text=EhoTable->FieldByName("KDR")->AsString;
Edit5->Text=EhoTable->FieldByName("KSR")->AsString;
Edit6->Text=EhoTable->FieldByName("TMGP")->AsString;
Edit7->Text=EhoTable->FieldByName("TZS")->AsString;
Edit8->Text=EhoTable->FieldByName("FV")->AsString;
Edit9->Text=EhoTable->FieldByName("FU")->AsString;
Edit10->Text=EhoTable->FieldByName("MMiok")->AsString;
Edit11->Text=EhoTable->FieldByName("IndMMiok")->AsString;
Edit12->Text=EhoTable->FieldByName("MGDAor")->AsString;
Edit13->Text=EhoTable->FieldByName("MGDMitr")->AsString;
DateTimePicker1->Date=EhoTable->FieldByName("DataOsm")->AsDateTime;
}
//---------------------------------------------------------------------------

void __fastcall TEhoForm::Button3Click(TObject *Sender)
{
```

```
EhoForm->EhoTable->Edit();
EhoForm->EhoTable->FieldByName("LevPr")->AsString= Edit1->Text;
EhoForm->EhoTable->FieldByName("VLevPr")->AsString= Edit2->Text;
EhoForm->EhoTable->FieldByName("IndVLevPr")->AsString= Edit3->Text;
EhoForm->EhoTable->FieldByName("KDR")->AsString= Edit4->Text;
EhoForm->EhoTable->FieldByName("KSR")->AsString= Edit5->Text;
EhoForm->EhoTable->FieldByName("TMGP")->AsString= Edit6->Text;
EhoForm->EhoTable->FieldByName("TZS")->AsString= Edit7->Text;
EhoForm->EhoTable->FieldByName("FV")->AsString= Edit8->Text;
EhoForm->EhoTable->FieldByName("FU")->AsString= Edit9->Text;
EhoForm->EhoTable->FieldByName("MMiok")->AsString= Edit10->Text;
EhoForm->EhoTable->FieldByName("IndMMiok")->AsString= Edit11->Text;
EhoForm->EhoTable->FieldByName("MGDAor")->AsString= Edit12->Text;
EhoForm->EhoTable->FieldByName("MGDMitr")->AsString= Edit13->Text;
EhoForm->EhoTable->FieldByName("DataOsm")->AsDateTime= DateTimePicker1->Date;
EhoForm->EhoTable->Post();
EhoTable->Refresh();
}
//---------------------------------------------------------------------------

void __fastcall TEhoForm::Button4Click(TObject *Sender)
{
EhoTable->Delete();
EhoTable->Refresh();
}
//---------------------------------------------------------------------------
```

Eho.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef EhoH
#define EhoH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <ComCtrls.hpp>
#include <DB.hpp>
#include <DBCtrls.hpp>
#include <DBGrids.hpp>
#include <DBTables.hpp>
#include <ExtCtrls.hpp>
#include <Grids.hpp>
//---------------------------------------------------------------------------
class TEhoForm : public TForm
{
__published:      // IDE-managed Components
      TLabel *Label1;
      TLabel *Label2;
      TLabel *Label3;
      TLabel *Label5;
      TLabel *Label6;
      TLabel *Label7;
      TLabel *Label8;
      TLabel *Label9;
      TLabel *Label10;
      TLabel *Label11;
      TLabel *Label12;
      TLabel *Label13;
      TLabel *Label18;
      TEdit *Edit1;
      TEdit *Edit2;
```

```
        TEdit *Edit3;
        TEdit *Edit4;
        TEdit *Edit5;
        TEdit *Edit6;
        TEdit *Edit7;
        TEdit *Edit8;
        TEdit *Edit9;
        TEdit *Edit10;
        TEdit *Edit11;
        TEdit *Edit12;
        TEdit *Edit13;
        TDBGrid *DBGrid1;
        TDateTimePicker *DateTimePicker1;
        TTable *EhoTable;
        TDataSource *EhoDataSource;
        TLabel *Label19;
        TLabel *Label20;
        TLabel *Label21;
        TLabel *Label22;
        TButton *Button1;
        TLabel *Label4;
        TButton *Button2;
        TLabel *Label32;
        TButton *Button3;
        TButton *Button4;
        void __fastcall Button1Click(TObject *Sender);
        void __fastcall Button2Click(TObject *Sender);
        void __fastcall Button3Click(TObject *Sender);
        void __fastcall Button4Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
        __fastcall TEhoForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TEhoForm *EhoForm;
//---------------------------------------------------------------------------
#endif
```

Found.cpp – форма поиска пациента

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop
#include "Main1.h"
#include "Found.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TFoundForm *FoundForm;
//---------------------------------------------------------------------------
__fastcall TFoundForm::TFoundForm(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TFoundForm::clearfbtnClick(TObject *Sender)
{
Edit1->Clear();
}
//---------------------------------------------------------------------------

void __fastcall TFoundForm::clearpbtnClick(TObject *Sender)
{
```

```
Edit2->Clear();
}
//---------------------------------------------------------------------------

void __fastcall TFoundForm::andbtnClick(TObject *Sender)
{
Edit2->Text = Edit2->Text + " AND ";
}
//---------------------------------------------------------------------------

void __fastcall TFoundForm::orbtnClick(TObject *Sender)
{
 Edit2->Text = Edit2->Text + " OR ";
}
//---------------------------------------------------------------------------

void __fastcall TFoundForm::findbtnClick(TObject *Sender)
{
    MainForm->MainTable->Active=true;
      MainForm->MainTable->Filter = Edit2->Text;
      MainForm->MainTable->Filtered = true;
}
//---------------------------------------------------------------------------

void __fastcall TFoundForm::ListBox1Click(TObject *Sender)
{
      if( ListBox1->ItemIndex == 0)
            Edit2->Text = Edit2->Text + "LastName" + "='" + Edit1->Text + "'";
      if( ListBox1->ItemIndex == 1)
            Edit2->Text = Edit2->Text + "FirstName" + "='" + Edit1->Text + "'";
      if( ListBox1->ItemIndex == 2)
            Edit2->Text = Edit2->Text + "PaternalName" + "='" + Edit1->Text + "'";
      if( ListBox1->ItemIndex == 3)
            Edit2->Text = Edit2->Text + "NumberPat" + "='" + Edit1->Text + "'";
}
//---------------------------------------------------------------------------
```

Found.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef FoundH
#define FoundH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
//---------------------------------------------------------------------------
class TFoundForm : public TForm
{
__published:      // IDE-managed Components
      TLabel *Label1;
      TLabel *Label2;
      TEdit *Edit1;
      TButton *clearfbtn;
      TEdit *Edit2;
      TButton *clearpbtn;
      TButton *andbtn;
      TButton *findbtn;
      TListBox *ListBox1;
      TButton *orbtn;
      void __fastcall clearfbtnClick(TObject *Sender);
      void __fastcall clearpbtnClick(TObject *Sender);
```

```
      void __fastcall andbtnClick(TObject *Sender);
      void __fastcall orbtnClick(TObject *Sender);
      void __fastcall findbtnClick(TObject *Sender);
      void __fastcall ListBox1Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
      __fastcall TFoundForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TFoundForm *FoundForm;
//---------------------------------------------------------------------------
#endif
```

IshemDoKS.cpp – форма особенностей течения ИБ до КШ

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "IshemDoKS.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TIshemDoKSForm *IshemDoKSForm;
//---------------------------------------------------------------------------
__fastcall TIshemDoKSForm::TIshemDoKSForm(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TIshemDoKSForm::Button1Click(TObject *Sender)
{
IshemDoKSForm->IshemDoKSTable->Insert();
IshemDoKSForm->IshemDoKSTable->FieldByName("ProdBolez")->AsString= Edit1->Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("StenokNapr")->AsString=      ComboBox1-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("StenokPok")->AsString=       ComboBox2-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("BezbolIsh")->AsString=       ComboBox3-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("IMAnam")->AsString= ComboBox4->Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("KolIM")->AsString= Edit2->Text;
if (ComboBox4->ItemIndex==1){
      IshemDoKSForm->IshemDoKSTable->FieldByName("DatePoslIM")->AsDateTime=
DateTimePicker1->Date;
}
IshemDoKSForm->IshemDoKSTable->FieldByName("VmeshKorArt")->AsString=     ComboBox5-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("KolVm")->AsString= Edit3->Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("Angio")->AsString= ComboBox6->Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("Stent")->AsString= ComboBox7->Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("KSh")->AsString= ComboBox8->Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("drugoe")->AsString= ComboBox9->Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("TrombLevG")->AsString=      ComboBox12-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("AnevrLevG")->AsString=      ComboBox13-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("GipertLevG")->AsString=     ComboBox14-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("FrakVibLevGDo")->AsString=  ComboBox15-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("NagrTest")->AsString=       ComboBox16-
>Text;
```

```
IshemDoKSForm->IshemDoKSTable->FieldByName("GelTahAnam")->AsString=     ComboBox10-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("ZlokGES")->AsString= ComboBox11->Text;
IshemDoKSForm->IshemDoKSTable->Post();
IshemDoKSForm->Close();
}
//---------------------------------------------------------------------------
void __fastcall TIshemDoKSForm::ComboBox4Select(TObject *Sender)
{
if (ComboBox4->ItemIndex==1){
      IshemDoKSForm->Edit2->Enabled=true;
      IshemDoKSForm->DateTimePicker1->Enabled=true;
}
else{
      IshemDoKSForm->Edit2->Clear();
      IshemDoKSForm->Edit2->Enabled=false;
      IshemDoKSForm->DateTimePicker1->Enabled=false;

      }
}
//---------------------------------------------------------------------------
void __fastcall TIshemDoKSForm::ComboBox5Select(TObject *Sender)
{
 if (ComboBox5->ItemIndex==1){
      IshemDoKSForm->Edit3->Enabled=true;
}
else{
      IshemDoKSForm->Edit3->Clear();
      IshemDoKSForm->Edit3->Enabled=false;
      }
}
//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void __fastcall TIshemDoKSForm::Button3Click(TObject *Sender)
{
IshemDoKSForm->IshemDoKSTable->Edit();
IshemDoKSForm->IshemDoKSTable->FieldByName("ProdBolez")->AsString= Edit1->Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("StenokNapr")->AsString=      ComboBox1-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("StenokPok")->AsString=       ComboBox2-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("BezbolIsh")->AsString=       ComboBox3-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("IMAnam")->AsString= ComboBox4->Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("KolIM")->AsString= Edit2->Text;
if (ComboBox4->Text=="да"){
      IshemDoKSForm->IshemDoKSTable->FieldByName("DatePoslIM")->AsDateTime=
DateTimePicker1->Date;
}
else IshemDoKSTable->FieldByName("DatePoslIM")->Clear();
IshemDoKSForm->IshemDoKSTable->FieldByName("VmeshKorArt")->AsString=     ComboBox5-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("KolVm")->AsString= Edit3->Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("Angio")->AsString= ComboBox6->Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("Stent")->AsString= ComboBox7->Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("KSh")->AsString= ComboBox8->Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("drugoe")->AsString= ComboBox9->Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("TrombLevG")->AsString=      ComboBox12-
>Text;
```

```
IshemDoKSForm->IshemDoKSTable->FieldByName("AnevrLevG")->AsString=      ComboBox13-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("GipertLevG")->AsString=     ComboBox14-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("FrakVibLevGDo")->AsString=  ComboBox15-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("NagrTest")->AsString=       ComboBox16-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("GelTahAnam")->AsString=     ComboBox10-
>Text;
IshemDoKSForm->IshemDoKSTable->FieldByName("ZlokGES")->AsString= ComboBox11->Text;
IshemDoKSForm->IshemDoKSTable->Post();
IshemDoKSForm->Close();
}
//---------------------------------------------------------------------------

void __fastcall TIshemDoKSForm::Button2Click(TObject *Sender)
{
Button1->Visible=false;
Button3->Visible=true;
IshemDoKSTable->Edit();
Edit1->Text=IshemDoKSTable->FieldByName("ProdBolez")->AsString;
ComboBox1->Text=IshemDoKSTable->FieldByName("StenokNapr")->AsString;
ComboBox2->Text=IshemDoKSTable->FieldByName("StenokPok")->AsString;
ComboBox3->Text=IshemDoKSTable->FieldByName("BezbolIsh")->AsString;
ComboBox4->Text=IshemDoKSTable->FieldByName("IMAnam")->AsString;
Edit2->Text=IshemDoKSTable->FieldByName("KolIM")->AsString;
if (ComboBox4->Text=="да"){
      DateTimePicker1->Date=IshemDoKSTable->FieldByName("DatePoslIM")->AsDateTime;
      IshemDoKSForm->Edit2->Enabled=true;
      IshemDoKSForm->DateTimePicker1->Enabled=true;
}

ComboBox5->Text=IshemDoKSTable->FieldByName("VmeshKorArt")->AsString;
Edit3->Text=IshemDoKSTable->FieldByName("KolVm")->AsString;
ComboBox6->Text=IshemDoKSTable->FieldByName("Angio")->AsString;
ComboBox7->Text=IshemDoKSTable->FieldByName("Stent")->AsString;
ComboBox8->Text=IshemDoKSTable->FieldByName("KSh")->AsString;
ComboBox9->Text=IshemDoKSTable->FieldByName("drugoe")->AsString;
ComboBox12->Text=IshemDoKSTable->FieldByName("TrombLevG")->AsString;
ComboBox13->Text=IshemDoKSTable->FieldByName("AnevrLevG")->AsString;
ComboBox14->Text=IshemDoKSTable->FieldByName("GipertLevG")->AsString;
ComboBox15->Text=IshemDoKSTable->FieldByName("FrakVibLevGDo")->AsString;
ComboBox16->Text=IshemDoKSTable->FieldByName("NagrTest")->AsString;
ComboBox10->Text=IshemDoKSTable->FieldByName("GelTahAnam")->AsString;
ComboBox11->Text=IshemDoKSTable->FieldByName("ZlokGES")->AsString;
}
//---------------------------------------------------------------------------
```

IshemDoKS.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef IshemDoKSH
#define IshemDoKSH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <ComCtrls.hpp>
#include <DB.hpp>
#include <DBCtrls.hpp>
#include <DBGrids.hpp>
```

```
#include <DBTables.hpp>
#include <ExtCtrls.hpp>
#include <Grids.hpp>
//---------------------------------------------------------------------------
class TIshemDoKSForm : public TForm
{
__published:      // IDE-managed Components
      TLabel *Label1;
      TEdit *Edit1;
      TLabel *Label2;
      TComboBox *ComboBox1;
      TLabel *Label3;
      TComboBox *ComboBox2;
      TLabel *Label4;
      TComboBox *ComboBox3;
      TLabel *Label5;
      TComboBox *ComboBox4;
      TLabel *Label6;
      TEdit *Edit2;
      TLabel *Label7;
      TDateTimePicker *DateTimePicker1;
      TLabel *Label8;
      TComboBox *ComboBox5;
      TLabel *Label9;
      TEdit *Edit3;
      TLabel *Label10;
      TComboBox *ComboBox6;
      TLabel *Label11;
      TComboBox *ComboBox7;
      TLabel *Label12;
      TComboBox *ComboBox8;
      TLabel *Label13;
      TComboBox *ComboBox9;
      TLabel *Label14;
      TComboBox *ComboBox10;
      TLabel *Label15;
      TComboBox *ComboBox11;
      TLabel *Label16;
      TComboBox *ComboBox12;
      TLabel *Label17;
      TComboBox *ComboBox13;
      TLabel *Label18;
      TComboBox *ComboBox14;
      TLabel *Label19;
      TComboBox *ComboBox15;
      TLabel *Label20;
      TComboBox *ComboBox16;
      TDBGrid *DBGrid1;
      TTable *IshemDoKSTable;
      TDataSource *IshemDoKSDataSource1;
      TButton *Button1;
      TLabel *Label32;
      TButton *Button2;
      TLabel *Label21;
      TLabel *Label22;
      TLabel *Label23;
      TLabel *Label24;
      TButton *Button3;
      void __fastcall Button1Click(TObject *Sender);
      void __fastcall ComboBox4Select(TObject *Sender);
      void __fastcall ComboBox5Select(TObject *Sender);
      void __fastcall Button3Click(TObject *Sender);
```

```
      void __fastcall Button2Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
      __fastcall TIshemDoKSForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TIshemDoKSForm *IshemDoKSForm;
//---------------------------------------------------------------------------
#endif
```

KlinSost.cpp – форма с клиническим состоянием

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "KlinSost.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TKlinSostForm *KlinSostForm;
//---------------------------------------------------------------------------
__fastcall TKlinSostForm::TKlinSostForm(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TKlinSostForm::Button1Click(TObject *Sender)
{
KlinSostForm->KlinSostTable->Insert();
KlinSostForm->KlinSostTable->FieldByName("Rost")->AsString= Edit1->Text;
KlinSostForm->KlinSostTable->FieldByName("Ves")->AsString= Edit2->Text;
KlinSostForm->KlinSostTable->FieldByName("Tal")->AsString= Edit3->Text;
KlinSostForm->KlinSostTable->FieldByName("IMT")->AsString= Edit4->Text;
KlinSostForm->KlinSostTable->FieldByName("SAD")->AsString= Edit5->Text;
KlinSostForm->KlinSostTable->FieldByName("DAD")->AsString= Edit6->Text;
KlinSostForm->KlinSostTable->FieldByName("HSS")->AsString= Edit7->Text;
KlinSostForm->KlinSostTable->FieldByName("Dist")->AsString= Edit8->Text;
KlinSostForm->KlinSostTable->FieldByName("HSSIsh")->AsString= Edit9->Text;
KlinSostForm->KlinSostTable->FieldByName("HSSPos")->AsString= Edit10->Text;
KlinSostForm->KlinSostTable->FieldByName("SADPos")->AsString= Edit11->Text;
KlinSostForm->KlinSostTable->FieldByName("SADIsh")->AsString= Edit12->Text;
KlinSostForm->KlinSostTable->FieldByName("DADPos")->AsString= Edit13->Text;
KlinSostForm->KlinSostTable->FieldByName("DADIsh")->AsString= Edit14->Text;
KlinSostForm->KlinSostTable->FieldByName("FKSN")->AsString= Edit15->Text;
KlinSostForm->KlinSostTable->FieldByName("NarRitmDr")->AsString= Edit16->Text;
KlinSostForm->KlinSostTable->FieldByName("DolUrFP")->AsString= ComboBox1->Text;
KlinSostForm->KlinSostTable->FieldByName("FaktUrFP")->AsString= ComboBox2->Text;
KlinSostForm->KlinSostTable->FieldByName("NarRitm")->AsString= ComboBox6->Text;
KlinSostForm->KlinSostTable->FieldByName("NarProv")->AsString= ComboBox4->Text;
KlinSostForm->KlinSostTable->FieldByName("DataOsm")->AsDateTime=   DateTimePicker1-
>Date;
KlinSostForm->KlinSostTable->Post();
KlinSostTable->Refresh();
}
//---------------------------------------------------------------------------

//---------------------------------------------------------------------------
void __fastcall TKlinSostForm::ComboBox6Select(TObject *Sender)
{
if (ComboBox6->ItemIndex==5)
      KlinSostForm->Edit16->Enabled=true;
else{
```

```
      KlinSostForm->Edit16->Clear();
      KlinSostForm->Edit16->Enabled=false;
      }
}
//---------------------------------------------------------------------------
void __fastcall TKlinSostForm::Button3Click(TObject *Sender)
{
KlinSostForm->KlinSostTable->Edit();
KlinSostForm->KlinSostTable->FieldByName("Rost")->AsString= Edit1->Text;
KlinSostForm->KlinSostTable->FieldByName("Ves")->AsString= Edit2->Text;
KlinSostForm->KlinSostTable->FieldByName("Tal")->AsString= Edit3->Text;
KlinSostForm->KlinSostTable->FieldByName("IMT")->AsString= Edit4->Text;
KlinSostForm->KlinSostTable->FieldByName("SAD")->AsString= Edit5->Text;
KlinSostForm->KlinSostTable->FieldByName("DAD")->AsString= Edit6->Text;
KlinSostForm->KlinSostTable->FieldByName("HSS")->AsString= Edit7->Text;
KlinSostForm->KlinSostTable->FieldByName("Dist")->AsString= Edit8->Text;
KlinSostForm->KlinSostTable->FieldByName("HSSIsh")->AsString= Edit9->Text;
KlinSostForm->KlinSostTable->FieldByName("HSSPos")->AsString= Edit10->Text;
KlinSostForm->KlinSostTable->FieldByName("SADPos")->AsString= Edit11->Text;
KlinSostForm->KlinSostTable->FieldByName("SADIsh")->AsString= Edit12->Text;
KlinSostForm->KlinSostTable->FieldByName("DADPos")->AsString= Edit13->Text;
KlinSostForm->KlinSostTable->FieldByName("DADIsh")->AsString= Edit14->Text;
KlinSostForm->KlinSostTable->FieldByName("FKSN")->AsString= Edit15->Text;
KlinSostForm->KlinSostTable->FieldByName("NarRitmDr")->AsString= Edit16->Text;
KlinSostForm->KlinSostTable->FieldByName("DolUrFP")->AsString= ComboBox1->Text;
KlinSostForm->KlinSostTable->FieldByName("FaktUrFP")->AsString= ComboBox2->Text;
KlinSostForm->KlinSostTable->FieldByName("NarRitm")->AsString= ComboBox6->Text;
KlinSostForm->KlinSostTable->FieldByName("NarProv")->AsString= ComboBox4->Text;
KlinSostForm->KlinSostTable->FieldByName("DataOsm")->AsDateTime=   DateTimePicker1-
>Date;
KlinSostForm->KlinSostTable->Post();
KlinSostTable->Refresh();
}
//---------------------------------------------------------------------------

void __fastcall TKlinSostForm::Button2Click(TObject *Sender)
{
Button1->Visible=false;
Button3->Visible=true;
KlinSostTable->Edit();
Edit1->Text=KlinSostTable->FieldByName("Rost")->AsString;
Edit2->Text=KlinSostTable->FieldByName("Ves")->AsString;
Edit3->Text=KlinSostTable->FieldByName("Tal")->AsString;
Edit4->Text=KlinSostTable->FieldByName("IMT")->AsString;
Edit5->Text=KlinSostTable->FieldByName("SAD")->AsString;
Edit6->Text=KlinSostTable->FieldByName("DAD")->AsString;
Edit7->Text=KlinSostTable->FieldByName("HSS")->AsString;
Edit8->Text=KlinSostTable->FieldByName("Dist")->AsString;
Edit9->Text=KlinSostTable->FieldByName("HSSIsh")->AsString;
Edit10->Text=KlinSostTable->FieldByName("HSSPos")->AsString;
Edit11->Text=KlinSostTable->FieldByName("SADPos")->AsString;
Edit12->Text=KlinSostTable->FieldByName("SADIsh")->AsString;
Edit13->Text=KlinSostTable->FieldByName("DADPos")->AsString;
Edit14->Text=KlinSostTable->FieldByName("DADIsh")->AsString;
Edit15->Text=KlinSostTable->FieldByName("FKSN")->AsString;
Edit16->Text=KlinSostTable->FieldByName("NarRitmDr")->AsString;
ComboBox1->Text=KlinSostTable->FieldByName("DolUrFP")->AsString;
ComboBox2->Text=KlinSostTable->FieldByName("FaktUrFP")->AsString;
ComboBox6->Text=KlinSostTable->FieldByName("NarRitm")->AsString;
ComboBox4->Text=KlinSostTable->FieldByName("NarProv")->AsString;
DateTimePicker1->Date=KlinSostTable->FieldByName("DataOsm")->AsDateTime;
}
```

```
//---------------------------------------------------------------------------

void __fastcall TKlinSostForm::Button4Click(TObject *Sender)
{
KlinSostTable->Delete();
KlinSostTable->Refresh();
}
//---------------------------------------------------------------------------
```

KlinSost.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef KlinSostH
#define KlinSostH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <ComCtrls.hpp>
#include <DB.hpp>
#include <DBCtrls.hpp>
#include <DBGrids.hpp>
#include <DBTables.hpp>
#include <ExtCtrls.hpp>
#include <Grids.hpp>
//---------------------------------------------------------------------------
class TKlinSostForm : public TForm
{
__published:      // IDE-managed Components
      TLabel *Label1;
      TLabel *Label2;
      TLabel *Label3;
      TLabel *Label4;
      TLabel *Label18;
      TEdit *Edit1;
      TEdit *Edit2;
      TEdit *Edit3;
      TEdit *Edit4;
      TDBGrid *DBGrid1;
      TButton *Button1;
      TDateTimePicker *DateTimePicker1;
      TTable *KlinSostTable;
      TDataSource *KlinSostDataSource;
      TLabel *Label5;
      TLabel *Label6;
      TLabel *Label7;
      TEdit *Edit5;
      TEdit *Edit6;
      TEdit *Edit7;
      TGroupBox *GroupBox1;
      TLabel *Label8;
      TEdit *Edit8;
      TLabel *Label9;
      TEdit *Edit9;
      TEdit *Edit10;
      TLabel *Label10;
      TEdit *Edit11;
      TLabel *Label11;
      TEdit *Edit12;
      TLabel *Label12;
      TEdit *Edit13;
      TLabel *Label13;
```

```
        TEdit *Edit14;
        TLabel *Label14;
        TLabel *Label15;
        TComboBox *ComboBox1;
        TLabel *Label16;
        TComboBox *ComboBox2;
        TLabel *Label17;
        TEdit *Edit15;
        TComboBox *ComboBox6;
        TLabel *Label21;
        TEdit *Edit16;
        TLabel *Label22;
        TComboBox *ComboBox4;
        TLabel *Label19;
        TLabel *Label20;
        TLabel *Label23;
        TLabel *Label24;
        TLabel *Label25;
        TButton *Button2;
        TLabel *Label32;
        TButton *Button3;
        TButton *Button4;
        void __fastcall Button1Click(TObject *Sender);
        void __fastcall ComboBox6Select(TObject *Sender);
        void __fastcall Button3Click(TObject *Sender);
        void __fastcall Button2Click(TObject *Sender);
        void __fastcall Button4Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
        __fastcall TKlinSostForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TKlinSostForm *KlinSostForm;
//---------------------------------------------------------------------------
#endif
```

KomSost.cpp – форма коморбидных состояний

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "KomSost.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TKomSostForm *KomSostForm;
//---------------------------------------------------------------------------
__fastcall TKomSostForm::TKomSostForm(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TKomSostForm::Button1Click(TObject *Sender)
{
KomSostForm->KomSostTable->Insert();
KomSostForm->KomSostTable->FieldByName("RiskTrombOsl")->AsString= Edit1->Text;
KomSostForm->KomSostTable->FieldByName("PishDrugoe")->AsString= Edit2->Text;
KomSostForm->KomSostTable->FieldByName("MochDrugoe")->AsString= Edit3->Text;
KomSostForm->KomSostTable->FieldByName("OstLoc")->AsString= Edit4->Text;
KomSostForm->KomSostTable->FieldByName("KostDrugoe")->AsString= Edit5->Text;
KomSostForm->KomSostTable->FieldByName("Local")->AsString= Edit6->Text;
KomSostForm->KomSostTable->FieldByName("Stad")->AsString= Edit7->Text;
```

```
KomSostForm->KomSostTable->FieldByName("Gipert")->AsString= ComboBox1->Text;
KomSostForm->KomSostTable->FieldByName("FibrTrep")->AsString= ComboBox2->Text;
KomSostForm->KomSostTable->FieldByName("CerAter")->AsString= ComboBox3->Text;
KomSostForm->KomSostTable->FieldByName("TranzIsh")->AsString= ComboBox4->Text;
KomSostForm->KomSostTable->FieldByName("Insult")->AsString= ComboBox5->Text;
KomSostForm->KomSostTable->FieldByName("AterNiz")->AsString= ComboBox6->Text;
KomSostForm->KomSostTable->FieldByName("HronSerdNed")->AsString= ComboBox7->Text;
KomSostForm->KomSostTable->FieldByName("FunkKlSerNed")->AsString= ComboBox8->Text;
KomSostForm->KomSostTable->FieldByName("DihatS")->AsString= ComboBox9->Text;
KomSostForm->KomSostTable->FieldByName("UglObm")->AsString= ComboBox10->Text;
KomSostForm->KomSostTable->FieldByName("GlikNat")->AsString= ComboBox11->Text;
KomSostForm->KomSostTable->FieldByName("TolGluk")->AsString= ComboBox12->Text;
KomSostForm->KomSostTable->FieldByName("SakarD")->AsString= ComboBox13->Text;
KomSostForm->KomSostTable->FieldByName("TipDiab")->AsString= ComboBox14->Text;
KomSostForm->KomSostTable->FieldByName("Insulpotr")->AsString= ComboBox15->Text;
KomSostForm->KomSostTable->FieldByName("Shit")->AsString= ComboBox16->Text;
KomSostForm->KomSostTable->FieldByName("Pish")->AsString= ComboBox17->Text;
KomSostForm->KomSostTable->FieldByName("Moch")->AsString= ComboBox18->Text;
KomSostForm->KomSostTable->FieldByName("Kost")->AsString= ComboBox19->Text;
KomSostForm->KomSostTable->FieldByName("Onko")->AsString= ComboBox20->Text;
KomSostForm->KomSostTable->Post();
KomSostForm->Close();
}
//---------------------------------------------------------------------------

void __fastcall TKomSostForm::ComboBox13Select(TObject *Sender)
{
if (ComboBox13->ItemIndex==1)
      KomSostForm->ComboBox14->Enabled=true;
else{
      KomSostForm->ComboBox14->ItemIndex=-1;
      KomSostForm->ComboBox14->Enabled=false;
      KomSostForm->ComboBox15->ItemIndex=-1;
      KomSostForm->ComboBox15->Enabled=false;
      }
}
//---------------------------------------------------------------------------

void __fastcall TKomSostForm::ComboBox14Select(TObject *Sender)
{
 if (ComboBox14->ItemIndex==1)
      KomSostForm->ComboBox15->Enabled=true;
 else{
      KomSostForm->ComboBox15->ItemIndex=-1;
      KomSostForm->ComboBox15->Enabled=false;
      }
}
//---------------------------------------------------------------------------

void __fastcall TKomSostForm::ComboBox17Select(TObject *Sender)
{
 if (ComboBox17->ItemIndex==7)
      KomSostForm->Edit2->Enabled=true;
 else{
      KomSostForm->Edit2->Clear();
      KomSostForm->Edit2->Enabled=false;
      }
}
//---------------------------------------------------------------------------

void __fastcall TKomSostForm::ComboBox18Select(TObject *Sender)
{
```

```
 if (ComboBox18->ItemIndex==5)
      KomSostForm->Edit3->Enabled=true;
 else{
      KomSostForm->Edit3->Clear();
      KomSostForm->Edit3->Enabled=false;
      }
}
//---------------------------------------------------------------------------

void __fastcall TKomSostForm::ComboBox19Select(TObject *Sender)
{
if (ComboBox19->ItemIndex==1)
      KomSostForm->Edit4->Enabled=true;
else if (ComboBox19->ItemIndex==4) {
        KomSostForm->Edit4->Clear();
            KomSostForm->Edit4->Enabled=false;
            KomSostForm->Edit5->Enabled=true;
             }else{
                  KomSostForm->Edit4->Clear();
                  KomSostForm->Edit4->Enabled=false;
                  KomSostForm->Edit5->Clear();
                  KomSostForm->Edit5->Enabled=false;
      }
}
//---------------------------------------------------------------------------

void __fastcall TKomSostForm::ComboBox20Select(TObject *Sender)
{
 if (ComboBox20->ItemIndex==1){
      KomSostForm->Edit6->Enabled=true;
      KomSostForm->Edit7->Enabled=true;
 }else{
      KomSostForm->Edit6->Clear();
      KomSostForm->Edit6->Enabled=false;
      KomSostForm->Edit7->Clear();
      KomSostForm->Edit7->Enabled=false;
      }
}
//---------------------------------------------------------------------------

void __fastcall TKomSostForm::Button2Click(TObject *Sender)
{
Button1->Visible=false;
Button3->Visible=true;
KomSostTable->Edit();
Edit1->Text=KomSostTable->FieldByName("RiskTrombOsl")->AsString;
Edit2->Text=KomSostTable->FieldByName("PishDrugoe")->AsString;
Edit3->Text=KomSostTable->FieldByName("MochDrugoe")->AsString;
Edit4->Text=KomSostTable->FieldByName("OstLoc")->AsString;
Edit5->Text=KomSostTable->FieldByName("KostDrugoe")->AsString;
Edit6->Text=KomSostTable->FieldByName("Local")->AsString;
Edit7->Text=KomSostTable->FieldByName("Stad")->AsString;
ComboBox1->Text=KomSostTable->FieldByName("Gipert")->AsString;
ComboBox2->Text=KomSostTable->FieldByName("FibrTrep")->AsString;
ComboBox3->Text=KomSostTable->FieldByName("CerAter")->AsString;
ComboBox4->Text=KomSostTable->FieldByName("TranzIsh")->AsString;
ComboBox5->Text=KomSostTable->FieldByName("Insult")->AsString;
ComboBox6->Text=KomSostTable->FieldByName("AterNiz")->AsString;
ComboBox7->Text=KomSostTable->FieldByName("HronSerdNed")->AsString;
ComboBox8->Text=KomSostTable->FieldByName("FunkKlSerNed")->AsString;
ComboBox9->Text=KomSostTable->FieldByName("DihatS")->AsString;
ComboBox10->Text=KomSostTable->FieldByName("UglObm")->AsString;
```

```
ComboBox11->Text=KomSostTable->FieldByName("GlikNat")->AsString;
ComboBox12->Text=KomSostTable->FieldByName("TolGluk")->AsString;
ComboBox13->Text=KomSostTable->FieldByName("SakarD")->AsString;
ComboBox14->Text=KomSostTable->FieldByName("TipDiab")->AsString;
ComboBox15->Text=KomSostTable->FieldByName("Insulpotr")->AsString;
ComboBox16->Text=KomSostTable->FieldByName("Shit")->AsString;
ComboBox17->Text=KomSostTable->FieldByName("Pish")->AsString;
ComboBox18->Text=KomSostTable->FieldByName("Moch")->AsString;
ComboBox19->Text=KomSostTable->FieldByName("Kost")->AsString;
ComboBox20->Text=KomSostTable->FieldByName("Onko")->AsString;

}
//---------------------------------------------------------------------------

void __fastcall TKomSostForm::Button3Click(TObject *Sender)
{
KomSostForm->KomSostTable->Edit();
KomSostForm->KomSostTable->FieldByName("RiskTrombOsl")->AsString= Edit1->Text;
KomSostForm->KomSostTable->FieldByName("PishDrugoe")->AsString= Edit2->Text;
KomSostForm->KomSostTable->FieldByName("MochDrugoe")->AsString= Edit3->Text;
KomSostForm->KomSostTable->FieldByName("OstLoc")->AsString= Edit4->Text;
KomSostForm->KomSostTable->FieldByName("KostDrugoe")->AsString= Edit5->Text;
KomSostForm->KomSostTable->FieldByName("Local")->AsString= Edit6->Text;
KomSostForm->KomSostTable->FieldByName("Stad")->AsString= Edit7->Text;
KomSostForm->KomSostTable->FieldByName("Gipert")->AsString= ComboBox1->Text;
KomSostForm->KomSostTable->FieldByName("FibrTrep")->AsString= ComboBox2->Text;
KomSostForm->KomSostTable->FieldByName("CerAter")->AsString= ComboBox3->Text;
KomSostForm->KomSostTable->FieldByName("TranzIsh")->AsString= ComboBox4->Text;
KomSostForm->KomSostTable->FieldByName("Insult")->AsString= ComboBox5->Text;
KomSostForm->KomSostTable->FieldByName("AterNiz")->AsString= ComboBox6->Text;
KomSostForm->KomSostTable->FieldByName("HronSerdNed")->AsString= ComboBox7->Text;
KomSostForm->KomSostTable->FieldByName("FunkKlSerNed")->AsString= ComboBox8->Text;
KomSostForm->KomSostTable->FieldByName("DihatS")->AsString= ComboBox9->Text;
KomSostForm->KomSostTable->FieldByName("UglObm")->AsString= ComboBox10->Text;
KomSostForm->KomSostTable->FieldByName("GlikNat")->AsString= ComboBox11->Text;
KomSostForm->KomSostTable->FieldByName("TolGluk")->AsString= ComboBox12->Text;
KomSostForm->KomSostTable->FieldByName("SakarD")->AsString= ComboBox13->Text;
KomSostForm->KomSostTable->FieldByName("TipDiab")->AsString= ComboBox14->Text;
KomSostForm->KomSostTable->FieldByName("Insulpotr")->AsString= ComboBox15->Text;
KomSostForm->KomSostTable->FieldByName("Shit")->AsString= ComboBox16->Text;
KomSostForm->KomSostTable->FieldByName("Pish")->AsString= ComboBox17->Text;
KomSostForm->KomSostTable->FieldByName("Moch")->AsString= ComboBox18->Text;
KomSostForm->KomSostTable->FieldByName("Kost")->AsString= ComboBox19->Text;
KomSostForm->KomSostTable->FieldByName("Onko")->AsString= ComboBox20->Text;
KomSostForm->KomSostTable->Post();
KomSostForm->Close();

}
//---------------------------------------------------------------------------
```

KomSost.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef KomSostH
#define KomSostH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <DB.hpp>
#include <DBCtrls.hpp>
```

```
#include <DBGrids.hpp>
#include <DBTables.hpp>
#include <ExtCtrls.hpp>
#include <Grids.hpp>
//---------------------------------------------------------------------------
class TKomSostForm : public TForm
{
__published:      // IDE-managed Components
      TLabel *Label19;
      TLabel *Label20;
      TLabel *Label21;
      TLabel *Label22;
      TDBGrid *DBGrid1;
      TTable *KomSostTable;
      TDataSource *KomSostDataSource;
      TGroupBox *GroupBox1;
      TLabel *Label1;
      TComboBox *ComboBox1;
      TComboBox *ComboBox2;
      TLabel *Label2;
      TLabel *Label4;
      TEdit *Edit1;
      TComboBox *ComboBox3;
      TLabel *Label3;
      TLabel *Label5;
      TComboBox *ComboBox4;
      TLabel *Label6;
      TComboBox *ComboBox5;
      TLabel *Label7;
      TComboBox *ComboBox6;
      TComboBox *ComboBox7;
      TLabel *Label8;
      TLabel *Label9;
      TComboBox *ComboBox8;
      TLabel *Label10;
      TComboBox *ComboBox9;
      TGroupBox *GroupBox2;
      TComboBox *ComboBox10;
      TLabel *Label11;
      TLabel *Label12;
      TComboBox *ComboBox11;
      TLabel *Label13;
      TComboBox *ComboBox12;
      TLabel *Label14;
      TComboBox *ComboBox13;
      TLabel *Label15;
      TComboBox *ComboBox14;
      TLabel *Label16;
      TComboBox *ComboBox15;
      TGroupBox *GroupBox3;
      TLabel *Label18;
      TComboBox *ComboBox17;
      TEdit *Edit2;
      TLabel *Label24;
      TGroupBox *GroupBox4;
      TLabel *Label23;
      TLabel *Label25;
      TEdit *Edit3;
      TComboBox *ComboBox18;
      TGroupBox *GroupBox5;
      TLabel *Label26;
      TLabel *Label27;
```

```
        TEdit *Edit4;
        TComboBox *ComboBox19;
        TEdit *Edit5;
        TLabel *Label28;
        TLabel *Label29;
        TComboBox *ComboBox20;
        TLabel *Label30;
        TEdit *Edit6;
        TLabel *Label31;
        TEdit *Edit7;
        TComboBox *ComboBox16;
        TLabel *Label17;
        TButton *Button1;
        TButton *Button2;
        TButton *Button3;
        TLabel *Label32;
        TLabel *Label33;
        void __fastcall Button1Click(TObject *Sender);
        void __fastcall ComboBox13Select(TObject *Sender);
        void __fastcall ComboBox14Select(TObject *Sender);
        void __fastcall ComboBox17Select(TObject *Sender);
        void __fastcall ComboBox18Select(TObject *Sender);
        void __fastcall ComboBox19Select(TObject *Sender);
        void __fastcall ComboBox20Select(TObject *Sender);
        void __fastcall Button2Click(TObject *Sender);
        void __fastcall Button3Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
        __fastcall TKomSostForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TKomSostForm *KomSostForm;
//---------------------------------------------------------------------------
#endif
```

Lab.cpp – форма с лабораторными исследованиями

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "Lab.h"
#include "Main1.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TLabForm *LabForm;
//---------------------------------------------------------------------------
__fastcall TLabForm::TLabForm(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TLabForm::Button1Click(TObject *Sender)
{
LabForm->LabTable->Insert();
LabForm->LabTable->FieldByName("Gem")->AsString= Edit1->Text;
LabForm->LabTable->FieldByName("Ley")->AsString= Edit2->Text;
LabForm->LabTable->FieldByName("SOA")->AsString= Edit3->Text;
LabForm->LabTable->FieldByName("GlNat")->AsString= Edit4->Text;
LabForm->LabTable->FieldByName("GlAf")->AsString= Edit5->Text;
LabForm->LabTable->FieldByName("GlNatVen")->AsString= Edit6->Text;
LabForm->LabTable->FieldByName("Krea")->AsString= Edit7->Text;
```

```
LabForm->LabTable->FieldByName("Kaliy")->AsString= Edit8->Text;
LabForm->LabTable->FieldByName("SRB")->AsString= Edit9->Text;
LabForm->LabTable->FieldByName("SKF")->AsString= Edit10->Text;
LabForm->LabTable->FieldByName("Klir")->AsString= Edit11->Text;
LabForm->LabTable->FieldByName("MNO")->AsString= Edit12->Text;
LabForm->LabTable->FieldByName("Fibr")->AsString= Edit13->Text;
LabForm->LabTable->FieldByName("ObHol")->AsString= Edit14->Text;
LabForm->LabTable->FieldByName("HSLVP")->AsString= Edit15->Text;
LabForm->LabTable->FieldByName("Trigl")->AsString= Edit16->Text;
LabForm->LabTable->FieldByName("HSLNP")->AsString= Edit17->Text;
LabForm->LabTable->FieldByName("DataLab")->AsDateTime= DateTimePicker1->Date;
LabForm->LabTable->Post();
LabTable->Refresh();
}
//---------------------------------------------------------------------------



//---------------------------------------------------------------------------

void __fastcall TLabForm::Button3Click(TObject *Sender)
{
LabForm->LabTable->Edit();
LabForm->LabTable->FieldByName("Gem")->AsString= Edit1->Text;
LabForm->LabTable->FieldByName("Ley")->AsString= Edit2->Text;
LabForm->LabTable->FieldByName("SOA")->AsString= Edit3->Text;
LabForm->LabTable->FieldByName("GlNat")->AsString= Edit4->Text;
LabForm->LabTable->FieldByName("GlAf")->AsString= Edit5->Text;
LabForm->LabTable->FieldByName("GlNatVen")->AsString= Edit6->Text;
LabForm->LabTable->FieldByName("Krea")->AsString= Edit7->Text;
LabForm->LabTable->FieldByName("Kaliy")->AsString= Edit8->Text;
LabForm->LabTable->FieldByName("SRB")->AsString= Edit9->Text;
LabForm->LabTable->FieldByName("SKF")->AsString= Edit10->Text;
LabForm->LabTable->FieldByName("Klir")->AsString= Edit11->Text;
LabForm->LabTable->FieldByName("MNO")->AsString= Edit12->Text;
LabForm->LabTable->FieldByName("Fibr")->AsString= Edit13->Text;
LabForm->LabTable->FieldByName("ObHol")->AsString= Edit14->Text;
LabForm->LabTable->FieldByName("HSLVP")->AsString= Edit15->Text;
LabForm->LabTable->FieldByName("Trigl")->AsString= Edit16->Text;
LabForm->LabTable->FieldByName("HSLNP")->AsString= Edit17->Text;
LabForm->LabTable->FieldByName("DataLab")->AsDateTime= DateTimePicker1->Date;
LabForm->LabTable->Post();
LabTable->Refresh();
}
//---------------------------------------------------------------------------

void __fastcall TLabForm::Button2Click(TObject *Sender)
{
Button1->Visible=false;
Button3->Visible=true;
LabTable->Edit();
Edit1->Text=LabTable->FieldByName("Gem")->AsString;
Edit2->Text=LabTable->FieldByName("Ley")->AsString;
Edit3->Text=LabTable->FieldByName("SOA")->AsString;
Edit4->Text=LabTable->FieldByName("GlNat")->AsString;
Edit5->Text=LabTable->FieldByName("GlAf")->AsString;
Edit6->Text=LabTable->FieldByName("GlNatVen")->AsString;
Edit7->Text=LabTable->FieldByName("Krea")->AsString;
Edit8->Text=LabTable->FieldByName("Kaliy")->AsString;
Edit9->Text=LabTable->FieldByName("SRB")->AsString;
Edit10->Text=LabTable->FieldByName("SKF")->AsString;
Edit11->Text=LabTable->FieldByName("Klir")->AsString;
```

```
Edit12->Text=LabTable->FieldByName("MNO")->AsString;
Edit13->Text=LabTable->FieldByName("Fibr")->AsString;
Edit14->Text=LabTable->FieldByName("ObHol")->AsString;
Edit15->Text=LabTable->FieldByName("HSLVP")->AsString;
Edit16->Text=LabTable->FieldByName("Trigl")->AsString;
Edit17->Text=LabTable->FieldByName("HSLNP")->AsString;
DateTimePicker1->Date=LabTable->FieldByName("DataLab")->AsDateTime;
}
//---------------------------------------------------------------------------

void __fastcall TLabForm::Button4Click(TObject *Sender)
{
 LabTable->Delete();
 LabTable->Refresh();
}
//---------------------------------------------------------------------------
```

Lab.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef LabH
#define LabH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <DB.hpp>
#include <DBCtrls.hpp>
#include <DBGrids.hpp>
#include <DBTables.hpp>
#include <ExtCtrls.hpp>
#include <Grids.hpp>
#include <ComCtrls.hpp>
//---------------------------------------------------------------------------
class TLabForm : public TForm
{
__published:      // IDE-managed Components
      TEdit *Edit1;
      TEdit *Edit2;
      TEdit *Edit3;
      TEdit *Edit4;
      TEdit *Edit5;
      TEdit *Edit6;
      TEdit *Edit7;
      TEdit *Edit8;
      TEdit *Edit9;
      TEdit *Edit10;
      TEdit *Edit11;
      TEdit *Edit12;
      TEdit *Edit13;
      TDBGrid *DBGrid1;
      TButton *Button1;
      TLabel *Label1;
      TLabel *Label2;
      TLabel *Label3;
      TLabel *Label4;
      TLabel *Label5;
      TLabel *Label6;
      TLabel *Label7;
      TLabel *Label8;
      TLabel *Label9;
      TLabel *Label10;
```

```
        TLabel *Label11;
        TLabel *Label12;
        TLabel *Label13;
        TGroupBox *GroupBox1;
        TTable *LabTable;
        TDataSource *LabDataSource;
        TEdit *Edit14;
        TEdit *Edit15;
        TEdit *Edit16;
        TEdit *Edit17;
        TLabel *Label14;
        TLabel *Label15;
        TLabel *Label16;
        TLabel *Label17;
        TDateTimePicker *DateTimePicker1;
        TLabel *Label18;
        TLabel *Label19;
        TLabel *Label20;
        TLabel *Label21;
        TLabel *Label22;
        TLabel *Label32;
        TButton *Button2;
        TButton *Button3;
        TButton *Button4;
        void __fastcall Button1Click(TObject *Sender);
        void __fastcall Button3Click(TObject *Sender);
        void __fastcall Button2Click(TObject *Sender);
        void __fastcall Button4Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
        __fastcall TLabForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TLabForm *LabForm;
//---------------------------------------------------------------------------
#endif
```

Main1.cpp – форма с описанием программы

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop
#include "Found.h"
#include "Main1.h"
#include "AddPatient1.h"
#include "CopyData.h"
#include "Disp.h"
#include "ObAn.h"
#include "Analiz.h"
#include "Eho.h"
#include "IshemDoKS.h"
#include "KlinSost.h"
#include "KomSost.h"
#include "Lab.h"
#include "ProvKS.h"
#include "Risk.h"
#include "RiskSmert.h"
#include "About.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TMainForm *MainForm;
//---------------------------------------------------------------------------
```

```
__fastcall TMainForm::TMainForm(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TMainForm::N5Click(TObject *Sender)
{
AddPatientForm->ShowModal();
}
//---------------------------------------------------------------------------

void __fastcall TMainForm::Button1Click(TObject *Sender)
{
AddPatientForm->Edit1->Clear();
AddPatientForm->Edit5->Clear();
AddPatientForm->Edit6->Clear();
AddPatientForm->Edit7->Clear();
AddPatientForm->Edit8->Clear();
AddPatientForm->Edit9->Clear();
AddPatientForm->Edit10->Clear();
AddPatientForm->Edit11->Clear();
AddPatientForm->Edit4->Clear();
AddPatientForm->Button3->Visible=false;
AddPatientForm->Button2->Visible=false;
AddPatientForm->Button1->Visible=true;
AddPatientForm->Caption="Добавить пациента";
AddPatientForm->ShowModal();
}
//---------------------------------------------------------------------------

void __fastcall TMainForm::Button2Click(TObject *Sender)
{
if (MessageDlg("Вы действительно хотите удалить пациента?\n"\
      "В  случае  положительного  ответа  данные  будут  удалены  безвозвратно",
mtWarning, TMsgDlgButtons() << mbYes << mbNo,0) == mrYes){
             if  (CopyDataForm->DataCopyTable->FieldByName("IdPatient")->AsInteger!
=0)CopyDataForm->DataCopyTable->Delete();if                    (RiskForm->RiskTable-
>FieldByName("IdPatient")->AsInteger!=0)RiskForm->RiskTable->Delete();
            if  (IshemDoKSForm->IshemDoKSTable->FieldByName("IdPatient")->AsInteger!
=0)IshemDoKSForm->IshemDoKSTable->Delete();
            if  (RiskSmertForm->RiskSmertTable->FieldByName("IdPatient")->AsInteger!
=0)RiskSmertForm->RiskSmertTable->Delete();
                 if   (ProvKSForm->ProvKSTable->FieldByName("IdPatient")->AsInteger!
=0)ProvKSForm->ProvKSTable->Delete();
               if   (KomSostForm->KomSostTable->FieldByName("IdPatient")->AsInteger!
=0)KomSostForm->KomSostTable->Delete();
                if   (AnalizForm->AnalizTable->FieldByName("IdPatient")->AsInteger!
=0)AnalizForm->AnalizTable->Delete();

             if  (LabForm->LabTable->FieldByName("IdPatient")->AsInteger!=0)LabForm-
>LabTable->Delete();

              if  (KlinSostForm->KlinSostTable->FieldByName("IdPatient")->AsInteger!
=0)KlinSostForm->KlinSostTable->Delete();
            if  (EhoForm->EhoTable->FieldByName("IdPatient")->AsInteger!=0)EhoForm-
>EhoTable->Delete();
          MainForm->MainTable->Delete();
         MainForm->MainTable->Refresh();
}
}
//---------------------------------------------------------------------------
```

```
void __fastcall TMainForm::Button4Click(TObject *Sender)
{
FoundForm->Show();
}
//---------------------------------------------------------------------------

void __fastcall TMainForm::Button5Click(TObject *Sender)
{
MainForm->MainTable->Filtered = false;
}
//---------------------------------------------------------------------------

void __fastcall TMainForm::Button3Click(TObject *Sender)

{
AddPatientForm->Edit1->Clear();
AddPatientForm->Edit5->Clear();
AddPatientForm->Edit6->Clear();
AddPatientForm->Edit7->Clear();
AddPatientForm->Edit8->Clear();
AddPatientForm->Edit9->Clear();
AddPatientForm->Edit10->Clear();
AddPatientForm->Edit11->Clear();
AddPatientForm->Edit4->Clear();
AddPatientForm->Button1->Visible=false;
AddPatientForm->Button2->Visible=true;
AddPatientForm->Button3->Visible=true;
AddPatientForm->Caption="Изменить данные пациента";
AddPatientForm->Show();
MainTable->Edit();
AddPatientForm->Edit1->Text=MainForm->MainTable->FieldByName("NumberPat")-
>AsString;
AddPatientForm->Edit5->Text=MainForm->MainTable->FieldByName("FirstName")-
>AsString;
AddPatientForm->Edit6->Text=MainForm->MainTable->FieldByName("PaternalName")-
>AsString;
AddPatientForm->Edit7->Text=MainForm->MainTable->FieldByName("Job")->AsString;
AddPatientForm->Edit8->Text=MainForm->MainTable->FieldByName("WorkPhone")-
>AsString;
AddPatientForm->Edit9->Text=MainForm->MainTable->FieldByName("HomePhone")-
>AsString;
AddPatientForm->Edit11->Text=MainForm->MainTable->FieldByName("Email")->AsString;
AddPatientForm->Edit10->Text=MainForm->MainTable->FieldByName("Post")->AsString;
AddPatientForm->DateTimePicker1->Date=MainForm->MainTable-
>FieldByName("DateStart")->AsDateTime;
AddPatientForm->DateTimePicker2->Date=MainForm->MainTable->FieldByName("DateEnd")-
>AsDateTime;
AddPatientForm->DateTimePicker3->Date=MainForm->MainTable-
>FieldByName("BirthDate")->AsDateTime;
AddPatientForm->DateTimePicker4->Date=MainForm->MainTable-
>FieldByName("DatePovt")->AsDateTime;
AddPatientForm->Edit4->Text=MainForm->MainTable->FieldByName("LastName")-
>AsString;
}
//---------------------------------------------------------------------------

void __fastcall TMainForm::Button6Click(TObject *Sender)
{
 MainTable->Active=false;
 MainTable->Active=true;
}
//---------------------------------------------------------------------------
```

```
void __fastcall TMainForm::N2Click(TObject *Sender)
{
DispForm->Show();
}
//---------------------------------------------------------------------------

void __fastcall TMainForm::N4Click(TObject *Sender)
{

ObAnForm->Show();
}

void __fastcall TMainForm::N3Click(TObject *Sender)
{
AboutForm->Show();
}
//---------------------------------------------------------------------------
```

Main1.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef Main1H
#define Main1H
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <Menus.hpp>
#include <DB.hpp>
#include <DBCtrls.hpp>
#include <DBGrids.hpp>
#include <DBTables.hpp>
#include <ExtCtrls.hpp>
#include <Grids.hpp>
//---------------------------------------------------------------------------
class TMainForm : public TForm
{
__published:      // IDE-managed Components
      TMainMenu *MainMenu1;
      TMenuItem *N1;
      TDBGrid *DBGrid1;
      TDatabase *Database1;
      TTable *MainTable;
      TDataSource *MainDataSource1;
      TButton *Button1;
      TButton *Button2;
      TButton *Button3;
      TButton *Button4;
      TButton *Button5;
      TButton *Button6;
      TMenuItem *N2;
      TMenuItem *N4;
      TMenuItem *N3;
      void __fastcall N5Click(TObject *Sender);
      void __fastcall Button1Click(TObject *Sender);
      void __fastcall Button2Click(TObject *Sender);
      void __fastcall Button4Click(TObject *Sender);
      void __fastcall Button5Click(TObject *Sender);
      void __fastcall Button3Click(TObject *Sender);
      void __fastcall Button6Click(TObject *Sender);
      void __fastcall N2Click(TObject *Sender);
```

```
      void __fastcall N4Click(TObject *Sender);
      void __fastcall N3Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
      __fastcall TMainForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TMainForm *MainForm;
//---------------------------------------------------------------------------
#endif
```

ObAn.cpp – форма общего анализа

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include <math.h>
#include "ObAn.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TObAnForm *ObAnForm;
//---------------------------------------------------------------------------
__fastcall TObAnForm::TObAnForm(TComponent* Owner)
      : TForm(Owner)
{

}
//---------------------------------------------------------------------------
void __fastcall TObAnForm::Button1Click(TObject *Sender)
{
 //Общее количество
double kol;
Query2->Close();
Query2->SQL->Clear();
Query2->SQL->Add("Select count(*) as Kolvo from MainInf ");
Query2->Open();
kol=Query2->FieldByName("Kolvo")->AsInteger;
Label2->Caption=kol;

if (kol>0) {


//Количество живых
double kollive;
Query2->Close();
Query2->SQL->Clear();
Query2->SQL->Add("Select count(*) as Kolvo from Analiz Where Stat='жив' ");
Query2->Open();
kollive=Query2->FieldByName("Kolvo")->AsInteger;
Label4->Caption=kollive;

//Количество мертвых
double koldead;
Query2->Close();
Query2->SQL->Clear();
Query2->SQL->Add("Select count(*) as Kolvo from Analiz Where Stat='мертв' ");
Query2->Open();
koldead=Query2->FieldByName("Kolvo")->AsInteger;
Label8->Caption=koldead;

//Количество умерших от сердечно-сосудистых заболеваний
```

```
double koldeadss;
Query2->Close();
Query2->SQL->Clear();
Query2->SQL->Add("Select count(*) as Kolvo from Analiz Where Stat='мертв' and
PrSm='сердечно-сосудистые' ");
Query2->Open();
koldeadss=Query2->FieldByName("Kolvo")->AsInteger;
Label9->Caption=koldeadss;

//Процент смертности от сердечно-сосудистых заболеваний
double prsm;
prsm=(floor(koldeadss/kol*10000+0.5))/100;
Label10->Caption=prsm;
Label11->Left=Label10->Left+Label10->Width+3;
Label11->Caption="%";
}
}
//---------------------------------------------------------------------------
```

ObAn.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef ObAnH
#define ObAnH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <DB.hpp>
#include <DBGrids.hpp>
#include <DBTables.hpp>
#include <Grids.hpp>
//---------------------------------------------------------------------------
class TObAnForm : public TForm
{
__published:      // IDE-managed Components
      TLabel *Label1;
      TLabel *Label2;
      TLabel *Label3;
      TLabel *Label4;
      TLabel *Label5;
      TLabel *Label6;
      TLabel *Label7;
      TLabel *Label8;
      TLabel *Label9;
      TLabel *Label10;
      TDBGrid *DBGrid1;
      TQuery *Query2;
      TDataSource *DataSource1;
      TLabel *Label11;
      TButton *Button1;
      void __fastcall Button1Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
      __fastcall TObAnForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TObAnForm *ObAnForm;
//---------------------------------------------------------------------------
#endif
```

ProvKS.cpp – форма проведения КШ

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop
#include "ProvKS.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TProvKSForm *ProvKSForm;

//---------------------------------------------------------------------------
__fastcall TProvKSForm::TProvKSForm(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void __fastcall TProvKSForm::Button1Click(TObject *Sender)
{
ProvKSForm->ProvKSTable->Insert();
ProvKSForm->ProvKSTable->FieldByName("TimeIK")->AsString= Edit1->Text;
ProvKSForm->ProvKSTable->FieldByName("TimePerAort")->AsString= Edit2->Text;
ProvKSForm->ProvKSTable->FieldByName("IskKrovoob")->AsString= ComboBox1->Text;
ProvKSForm->ProvKSTable->FieldByName("Kardiople")->AsString= ComboBox2->Text;
ProvKSForm->ProvKSTable->FieldByName("VidKS")->AsString= ComboBox3->Text;
ProvKSForm->ProvKSTable->FieldByName("KolS")->AsString= ComboBox4->Text;
ProvKSForm->ProvKSTable->FieldByName("Rev")->AsString= ComboBox5->Text;
ProvKSForm->ProvKSTable->FieldByName("Gem")->AsString= ComboBox6->Text;
ProvKSForm->ProvKSTable->FieldByName("Gibr")->AsString= ComboBox7->Text;
ProvKSForm->ProvKSTable->FieldByName("GO")->AsString= ComboBox8->Text;
ProvKSForm->ProvKSTable->FieldByName("ManPosle")->AsString= ComboBox9->Text;
ProvKSForm->ProvKSTable->FieldByName("VrEKS")->AsString= ComboBox10->Text;
ProvKSForm->ProvKSTable->FieldByName("Def")->AsString= ComboBox11->Text;
ProvKSForm->ProvKSTable->FieldByName("Farm")->AsString= ComboBox12->Text;
ProvKSForm->ProvKSTable->FieldByName("Rem")->AsString= ComboBox13->Text;
ProvKSForm->ProvKSTable->FieldByName("PostEKS")->AsString= ComboBox14->Text;
ProvKSForm->ProvKSTable->FieldByName("HPS")->AsString= ComboBox15->Text;
ProvKSForm->ProvKSTable->FieldByName("VAC")->AsString= ComboBox16->Text;
ProvKSForm->ProvKSTable->FieldByName("Stent")->AsString= ComboBox17->Text;
ProvKSForm->ProvKSTable->FieldByName("Dr")->AsString= ComboBox18->Text;
ProvKSForm->ProvKSTable->FieldByName("KorKS")->AsString= ComboBox19->Text;
ProvKSForm->ProvKSTable->FieldByName("KonP")->AsString= ComboBox20->Text;
ProvKSForm->ProvKSTable->FieldByName("DrPnev")->AsString= ComboBox21->Text;
ProvKSForm->ProvKSTable->FieldByName("DrGid")->AsString= ComboBox22->Text;
ProvKSForm->ProvKSTable->FieldByName("DrPer")->AsString= ComboBox23->Text;
ProvKSForm->ProvKSTable->FieldByName("DateOper")->AsDateTime=     DateTimePicker1-
>Date;
ProvKSForm->ProvKSTable->Post();
ProvKSForm->Close();
}
//---------------------------------------------------------------------------

void __fastcall TProvKSForm::ComboBox1Select(TObject *Sender)
{
if (ComboBox1->ItemIndex==1){
      ProvKSForm->Edit1->Enabled=true;
      ProvKSForm->Edit2->Enabled=true;
}else{
      ProvKSForm->Edit1->Clear();
      ProvKSForm->Edit1->Enabled=false;
```

```
      ProvKSForm->Edit2->Clear();
      ProvKSForm->Edit2->Enabled=false;
      }
}
//---------------------------------------------------------------------------

void __fastcall TProvKSForm::ComboBox7Select(TObject *Sender)
{
 if (ComboBox7->ItemIndex==1)
      ProvKSForm->ComboBox8->Enabled=true;
else{
      ProvKSForm->ComboBox8->ItemIndex=-1;;
      ProvKSForm->ComboBox8->Enabled=false;
      }
}
//---------------------------------------------------------------------------

void __fastcall TProvKSForm::ComboBox9Select(TObject *Sender)
{
if (ComboBox9->ItemIndex==1)
      ProvKSForm->GroupBox1->Enabled=true;
else{
      ProvKSForm->ComboBox10->ItemIndex=-1;
      ProvKSForm->ComboBox11->ItemIndex=-1;
      ProvKSForm->ComboBox12->ItemIndex=-1;
      ProvKSForm->ComboBox13->ItemIndex=-1;
      ProvKSForm->ComboBox14->ItemIndex=-1;
      ProvKSForm->ComboBox15->ItemIndex=-1;
      ProvKSForm->ComboBox16->ItemIndex=-1;
      ProvKSForm->ComboBox17->ItemIndex=-1;
      ProvKSForm->ComboBox18->ItemIndex=-1;
      ProvKSForm->ComboBox19->ItemIndex=-1;
      ProvKSForm->ComboBox20->ItemIndex=-1;
      ProvKSForm->ComboBox21->ItemIndex=-1;
      ProvKSForm->ComboBox22->ItemIndex=-1;
      ProvKSForm->ComboBox23->ItemIndex=-1;
      ProvKSForm->GroupBox1->Enabled=false;
      }
}
//---------------------------------------------------------------------------

void __fastcall TProvKSForm::Button2Click(TObject *Sender)
{
Button1->Visible=false;
Button3->Visible=true;
ProvKSTable->Edit();
Edit1->Text=ProvKSTable->FieldByName("TimeIK")->AsString;
Edit2->Text=ProvKSTable->FieldByName("TimePerAort")->AsString;
ComboBox1->Text=ProvKSTable->FieldByName("IskKrovoob")->AsString;
ComboBox2->Text=ProvKSTable->FieldByName("Kardiople")->AsString;
ComboBox3->Text=ProvKSTable->FieldByName("VidKS")->AsString;
ComboBox4->Text=ProvKSTable->FieldByName("KolS")->AsString;
ComboBox5->Text=ProvKSTable->FieldByName("Rev")->AsString;
ComboBox6->Text=ProvKSTable->FieldByName("Gem")->AsString;
ComboBox7->Text=ProvKSTable->FieldByName("Gibr")->AsString;
ComboBox8->Text=ProvKSTable->FieldByName("GO")->AsString;
ComboBox9->Text=ProvKSTable->FieldByName("ManPosle")->AsString;
ComboBox10->Text=ProvKSTable->FieldByName("VrEKS")->AsString;
ComboBox11->Text=ProvKSTable->FieldByName("Def")->AsString;
ComboBox12->Text=ProvKSTable->FieldByName("Farm")->AsString;
ComboBox13->Text=ProvKSTable->FieldByName("Rem")->AsString;
ComboBox14->Text=ProvKSTable->FieldByName("PostEKS")->AsString;
```

```
ComboBox15->Text=ProvKSTable->FieldByName("HPS")->AsString;
ComboBox16->Text=ProvKSTable->FieldByName("VAC")->AsString;
ComboBox17->Text=ProvKSTable->FieldByName("Stent")->AsString;
ComboBox18->Text=ProvKSTable->FieldByName("Dr")->AsString;
ComboBox19->Text=ProvKSTable->FieldByName("KorKS")->AsString;
ComboBox20->Text=ProvKSTable->FieldByName("KonP")->AsString;
ComboBox21->Text=ProvKSTable->FieldByName("DrPnev")->AsString;
ComboBox22->Text=ProvKSTable->FieldByName("DrGid")->AsString;
ComboBox23->Text=ProvKSTable->FieldByName("DrPer")->AsString;
DateTimePicker1->Date=ProvKSTable->FieldByName("DateOper")->AsDateTime;

}
//---------------------------------------------------------------------------

void __fastcall TProvKSForm::Button3Click(TObject *Sender)
{
ProvKSForm->ProvKSTable->Edit();
ProvKSForm->ProvKSTable->FieldByName("TimeIK")->AsString= Edit1->Text;
ProvKSForm->ProvKSTable->FieldByName("TimePerAort")->AsString= Edit2->Text;
ProvKSForm->ProvKSTable->FieldByName("IskKrovoob")->AsString= ComboBox1->Text;
ProvKSForm->ProvKSTable->FieldByName("Kardiople")->AsString= ComboBox2->Text;
ProvKSForm->ProvKSTable->FieldByName("VidKS")->AsString= ComboBox3->Text;
ProvKSForm->ProvKSTable->FieldByName("KolS")->AsString= ComboBox4->Text;
ProvKSForm->ProvKSTable->FieldByName("Rev")->AsString= ComboBox5->Text;
ProvKSForm->ProvKSTable->FieldByName("Gem")->AsString= ComboBox6->Text;
ProvKSForm->ProvKSTable->FieldByName("Gibr")->AsString= ComboBox7->Text;
ProvKSForm->ProvKSTable->FieldByName("GO")->AsString= ComboBox8->Text;
ProvKSForm->ProvKSTable->FieldByName("ManPosle")->AsString= ComboBox9->Text;
ProvKSForm->ProvKSTable->FieldByName("VrEKS")->AsString= ComboBox10->Text;
ProvKSForm->ProvKSTable->FieldByName("Def")->AsString= ComboBox11->Text;
ProvKSForm->ProvKSTable->FieldByName("Farm")->AsString= ComboBox12->Text;
ProvKSForm->ProvKSTable->FieldByName("Rem")->AsString= ComboBox13->Text;
ProvKSForm->ProvKSTable->FieldByName("PostEKS")->AsString= ComboBox14->Text;
ProvKSForm->ProvKSTable->FieldByName("HPS")->AsString= ComboBox15->Text;
ProvKSForm->ProvKSTable->FieldByName("VAC")->AsString= ComboBox16->Text;
ProvKSForm->ProvKSTable->FieldByName("Stent")->AsString= ComboBox17->Text;
ProvKSForm->ProvKSTable->FieldByName("Dr")->AsString= ComboBox18->Text;
ProvKSForm->ProvKSTable->FieldByName("KorKS")->AsString= ComboBox19->Text;
ProvKSForm->ProvKSTable->FieldByName("KonP")->AsString= ComboBox20->Text;
ProvKSForm->ProvKSTable->FieldByName("DrPnev")->AsString= ComboBox21->Text;
ProvKSForm->ProvKSTable->FieldByName("DrGid")->AsString= ComboBox22->Text;
ProvKSForm->ProvKSTable->FieldByName("DrPer")->AsString= ComboBox23->Text;
ProvKSForm->ProvKSTable->FieldByName("DateOper")->AsDateTime=      DateTimePicker1-
>Date;
ProvKSForm->ProvKSTable->Post();
ProvKSForm->Close();
}
//---------------------------------------------------------------------------
```

ProvKS.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef ProvKSH
#define ProvKSH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <ComCtrls.hpp>
#include <DB.hpp>
#include <DBCtrls.hpp>
#include <DBGrids.hpp>
```

```
#include <DBTables.hpp>
#include <ExtCtrls.hpp>
#include <Grids.hpp>
//---------------------------------------------------------------------------
class TProvKSForm : public TForm
{
__published:      // IDE-managed Components
      TDBGrid *DBGrid1;
      TButton *Button1;
      TTable *ProvKSTable;
      TDataSource *ProvKSDataSource1;
      TLabel *Label1;
      TDateTimePicker *DateTimePicker1;
      TLabel *Label2;
      TComboBox *ComboBox1;
      TLabel *Label3;
      TLabel *Label4;
      TEdit *Edit1;
      TEdit *Edit2;
      TLabel *Label5;
      TComboBox *ComboBox2;
      TLabel *Label6;
      TComboBox *ComboBox3;
      TLabel *Label7;
      TComboBox *ComboBox4;
      TLabel *Label8;
      TComboBox *ComboBox5;
      TLabel *Label9;
      TComboBox *ComboBox6;
      TLabel *Label10;
      TComboBox *ComboBox7;
      TLabel *Label11;
      TComboBox *ComboBox8;
      TLabel *Label12;
      TComboBox *ComboBox9;
      TGroupBox *GroupBox1;
      TLabel *Label13;
      TComboBox *ComboBox10;
      TLabel *Label14;
      TComboBox *ComboBox11;
      TLabel *Label15;
      TLabel *Label16;
      TLabel *Label17;
      TLabel *Label18;
      TLabel *Label19;
      TComboBox *ComboBox12;
      TComboBox *ComboBox13;
      TComboBox *ComboBox14;
      TComboBox *ComboBox15;
      TComboBox *ComboBox16;
      TComboBox *ComboBox17;
      TComboBox *ComboBox18;
      TComboBox *ComboBox19;
      TComboBox *ComboBox20;
      TComboBox *ComboBox21;
      TComboBox *ComboBox22;
      TComboBox *ComboBox23;
      TLabel *Label20;
      TLabel *Label21;
      TLabel *Label22;
      TLabel *Label23;
      TLabel *Label24;
```

```
      TLabel *Label25;
      TLabel *Label26;
      TLabel *Label27;
      TLabel *Label28;
      TLabel *Label29;
      TLabel *Label33;
      TButton *Button2;
      TLabel *Label32;
      TButton *Button3;
      void __fastcall Button1Click(TObject *Sender);
      void __fastcall ComboBox1Select(TObject *Sender);
      void __fastcall ComboBox7Select(TObject *Sender);
      void __fastcall ComboBox9Select(TObject *Sender);
      void __fastcall Button2Click(TObject *Sender);
      void __fastcall Button3Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
      __fastcall TProvKSForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TProvKSForm *ProvKSForm;
//---------------------------------------------------------------------------
#endif
```

Risk.cpp – форма с рисками

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "Risk.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TRiskForm *RiskForm;
//---------------------------------------------------------------------------
__fastcall TRiskForm::TRiskForm(TComponent* Owner)
      : TForm(Owner)
{

}


//---------------------------------------------------------------------------
void __fastcall TRiskForm::ComboBox2Select(TObject *Sender)
{
if (ComboBox2->ItemIndex==2)
      RiskForm->Edit1->Enabled=true;
else{
      RiskForm->Edit1->Clear();
      RiskForm->Edit1->Enabled=false;
      }
}
//---------------------------------------------------------------------------
void __fastcall TRiskForm::ComboBox4Select(TObject *Sender)
{
      if (ComboBox4->ItemIndex==1)RiskForm->Edit2->Enabled=true;
      else{
      RiskForm->Edit2->Clear();
      RiskForm->Edit2->Enabled=false;
      }
}
//---------------------------------------------------------------------------
```

```
void __fastcall TRiskForm::ComboBox8Select(TObject *Sender)
{
      if (ComboBox8->ItemIndex==1) RiskForm->Edit4->Enabled=true;
      else{
      RiskForm->Edit4->Clear();
      RiskForm->Edit4->Enabled=false;
      }
 }
//---------------------------------------------------------------------------
void __fastcall TRiskForm::ComboBox9Select(TObject *Sender)
{
      if (ComboBox9->ItemIndex==1)RiskForm->Edit4->Enabled=true;
      else{
      RiskForm->Edit4->Clear();
      RiskForm->Edit4->Enabled=false;
      }
}
//---------------------------------------------------------------------------
void __fastcall TRiskForm::Button1Click(TObject *Sender)
{
RiskForm->RiskTable->Insert();
RiskForm->RiskTable->FieldByName("Anamnez")->AsString= ComboBox1->Text;
RiskForm->RiskTable->FieldByName("Tabak")->AsString= ComboBox2->Text;
RiskForm->RiskTable->FieldByName("IndexKur")->AsString= Edit1->Text;
RiskForm->RiskTable->FieldByName("Alkogol")->AsString= ComboBox3->Text;
RiskForm->RiskTable->FieldByName("ArtGip")->AsString= ComboBox4->Text;
RiskForm->RiskTable->FieldByName("AG")->AsString= Edit2->Text;
RiskForm->RiskTable->FieldByName("UglObm")->AsString= ComboBox5->Text;
RiskForm->RiskTable->FieldByName("LipObm")->AsString= ComboBox6->Text;
RiskForm->RiskTable->FieldByName("FisAct")->AsString= ComboBox7->Text;
RiskForm->RiskTable->FieldByName("IzbMassa")->AsString= ComboBox8->Text;
RiskForm->RiskTable->FieldByName("NedMassa")->AsString= ComboBox9->Text;
RiskForm->RiskTable->FieldByName("IMT")->AsString= Edit4->Text;
RiskForm->RiskTable->Post();
RiskForm->Close();
}
//---------------------------------------------------------------------------
void __fastcall TRiskForm::Button2Click(TObject *Sender)
{
Button1->Visible=false;
Button3->Visible=true;
RiskForm->RiskTable->Edit();
ComboBox1->Text=RiskTable->FieldByName("Anamnez")->AsString;
ComboBox2->Text=RiskTable->FieldByName("Tabak")->AsString;
Edit1->Text=RiskTable->FieldByName("IndexKur")->AsString;
ComboBox3->Text=RiskTable->FieldByName("Alkogol")->AsString;
ComboBox4->Text=RiskTable->FieldByName("ArtGip")->AsString;
Edit2->Text=RiskTable->FieldByName("AG")->AsString;
ComboBox5->Text=RiskTable->FieldByName("UglObm")->AsString;
ComboBox6->Text=RiskTable->FieldByName("LipObm")->AsString;
ComboBox7->Text=RiskTable->FieldByName("FisAct")->AsString;
ComboBox8->Text=RiskTable->FieldByName("IzbMassa")->AsString;
ComboBox9->Text=RiskTable->FieldByName("NedMassa")->AsString;
Edit4->Text=RiskTable->FieldByName("IMT")->AsString;
}
//---------------------------------------------------------------------------

void __fastcall TRiskForm::Button3Click(TObject *Sender)
{
RiskForm->RiskTable->Edit();
RiskForm->RiskTable->FieldByName("Anamnez")->AsString= ComboBox1->Text;
RiskForm->RiskTable->FieldByName("Tabak")->AsString= ComboBox2->Text;
```

```
RiskForm->RiskTable->FieldByName("IndexKur")->AsString= Edit1->Text;
RiskForm->RiskTable->FieldByName("Alkogol")->AsString= ComboBox3->Text;
RiskForm->RiskTable->FieldByName("ArtGip")->AsString= ComboBox4->Text;
RiskForm->RiskTable->FieldByName("AG")->AsString= Edit2->Text;
RiskForm->RiskTable->FieldByName("UglObm")->AsString= ComboBox5->Text;
RiskForm->RiskTable->FieldByName("LipObm")->AsString= ComboBox6->Text;
RiskForm->RiskTable->FieldByName("FisAct")->AsString= ComboBox7->Text;
RiskForm->RiskTable->FieldByName("IzbMassa")->AsString= ComboBox8->Text;
RiskForm->RiskTable->FieldByName("NedMassa")->AsString= ComboBox9->Text;
RiskForm->RiskTable->FieldByName("IMT")->AsString= Edit4->Text;
RiskForm->RiskTable->Post();
RiskForm->Close();
}
//---------------------------------------------------------------------------
```

Risk.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef RiskH
#define RiskH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <DB.hpp>
#include <DBCtrls.hpp>
#include <DBGrids.hpp>
#include <DBTables.hpp>
#include <ExtCtrls.hpp>
#include <Grids.hpp>
//---------------------------------------------------------------------------
class TRiskForm : public TForm
{
__published:      // IDE-managed Components
      TDBGrid *DBGrid1;
      TTable *RiskTable;
      TDataSource *RiskDataSource1;
      TLabel *Label1;
      TComboBox *ComboBox1;
      TLabel *Label2;
      TComboBox *ComboBox2;
      TLabel *Label3;
      TEdit *Edit1;
      TLabel *Label4;
      TComboBox *ComboBox3;
      TLabel *Label5;
      TComboBox *ComboBox4;
      TLabel *Label6;
      TEdit *Edit2;
      TLabel *Label7;
      TComboBox *ComboBox5;
      TLabel *Label8;
      TComboBox *ComboBox6;
      TLabel *Label9;
      TComboBox *ComboBox7;
      TLabel *Label10;
      TComboBox *ComboBox8;
      TLabel *Label12;
      TComboBox *ComboBox9;
      TLabel *Label13;
      TEdit *Edit4;
      TButton *Button1;
```

```
        TLabel *Label19;
        TLabel *Label20;
        TLabel *Label21;
        TLabel *Label22;
        TLabel *Label32;
        TButton *Button2;
        TButton *Button3;
        void __fastcall ComboBox2Select(TObject *Sender);
        void __fastcall ComboBox4Select(TObject *Sender);
        void __fastcall ComboBox8Select(TObject *Sender);
        void __fastcall ComboBox9Select(TObject *Sender);
        void __fastcall Button1Click(TObject *Sender);
        void __fastcall Button2Click(TObject *Sender);
        void __fastcall Button3Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
        __fastcall TRiskForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TRiskForm *RiskForm;
//---------------------------------------------------------------------------
#endif
```

RiskSmert.cpp – форма с риском смерти

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "RiskSmert.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TRiskSmertForm *RiskSmertForm;
//---------------------------------------------------------------------------
__fastcall TRiskSmertForm::TRiskSmertForm(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TRiskSmertForm::Button1Click(TObject *Sender)
{
RiskSmertForm->RiskSmertTable->Insert();
RiskSmertForm->RiskSmertTable->FieldByName("Ball")->AsString= Edit1->Text;
RiskSmertForm->RiskSmertTable->FieldByName("Proc")->AsString= Edit2->Text;
RiskSmertForm->RiskSmertTable->Post();
RiskSmertForm->Close();
}
//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void __fastcall TRiskSmertForm::Button2Click(TObject *Sender)
{
Button1->Visible=false;
Button3->Visible=true;
RiskSmertTable->Edit();
Edit1->Text=RiskSmertTable->FieldByName("Ball")->AsString;
Edit2->Text=RiskSmertTable->FieldByName("Proc")->AsString;
}
//---------------------------------------------------------------------------

void __fastcall TRiskSmertForm::Button3Click(TObject *Sender)
```

```
{
RiskSmertForm->RiskSmertTable->Edit();
RiskSmertForm->RiskSmertTable->FieldByName("Ball")->AsString= Edit1->Text;
RiskSmertForm->RiskSmertTable->FieldByName("Proc")->AsString= Edit2->Text;
RiskSmertForm->RiskSmertTable->Post();
RiskSmertForm->Close();
}
//---------------------------------------------------------------------------
```

RiskSmert.h – заголовочный файл.

```
//---------------------------------------------------------------------------

#ifndef RiskSmertH
#define RiskSmertH
//---------------------------------------------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <DB.hpp>
#include <DBCtrls.hpp>
#include <DBGrids.hpp>
#include <DBTables.hpp>
#include <ExtCtrls.hpp>
#include <Grids.hpp>
//---------------------------------------------------------------------------
class TRiskSmertForm : public TForm
{
__published:      // IDE-managed Components
      TTable *RiskSmertTable;
      TDBGrid *DBGrid1;
      TDataSource *RiskSmertDataSource;
      TLabel *Label1;
      TLabel *Label2;
      TEdit *Edit1;
      TLabel *Label3;
      TEdit *Edit2;
      TButton *Button1;
      TLabel *Label32;
      TButton *Button2;
      TButton *Button3;
      TLabel *Label19;
      TLabel *Label20;
      TLabel *Label21;
      TLabel *Label22;
      void __fastcall Button1Click(TObject *Sender);
      void __fastcall Button2Click(TObject *Sender);
      void __fastcall Button3Click(TObject *Sender);
private:    // User declarations
public:           // User declarations
      __fastcall TRiskSmertForm(TComponent* Owner);
};
//---------------------------------------------------------------------------
extern PACKAGE TRiskSmertForm *RiskSmertForm;
//---------------------------------------------------------------------------
#endif
```