МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра вычислительной техники

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О.Г. Локтионова
«_05_» _02_ 2021 г.

РАЗРАБОТКА БАЗЫ ДАННЫХ

методические указания к выполнению лабораторных работ по дисциплине «Архитектура систем обработки, анализа и интерпретации данных»

Курск 2021

УДК 004
Составитель: В.С. Панищев

Рецензент
Кандидат технических наук, доцент Ю.А. Халин

Разработка базы данных: методические указания к выполнению лабораторных работ по дисциплине «Архитектура систем обработки, анализа и интерпретации данных» / Юго-Зап. гос. ун-т; сост.: В.С. Панищев. – Курск, 2021. – 38 с.: Библиогр.: с. 30.

Методические указания предназначены для ознакомления студентов с принципами разработки базы данных и приобретения навыков в написании специализированных программ.

Предназначены для студентов направления подготовки 09.04.01 Информатика и вычислительная техника.

Текст печатается в авторской редакции

Подписано в печать _____. Формат 60x84 1/16.
Усл. печ. л. 2,21. Уч. – изд.л. 2,0. Тираж 50 экз. Заказ 376. Бесплатно.
Юго-Западный государственный университет
305040, Курск, ул. 50 лет Октября, 94.

Содержание

1 РАЗРАБОТКА БАЗЫ ДАННЫХ

1. Цель работы

Целью данной работы является изучение принципов создания баз данных, оформления документации и приобретение навыков в написании специализированных программ.

2. Задачи: разработка базы данных, получение практических навыков по работе со структурированными наборами данных.

3. Задание.

Написать программу, осуществляющую работу с базой данных (набор полей указан и выбирается в соответствии с индивидуальным вариантом задания). Хранение базы данных осуществлять в памяти в виде массива записей. Предусмотреть следующий набор действий: добавление данных, удаление данных, поиск данных, вывод содержимого базы на экран, загрузка данных из файла, сохранение данных в файл. Формат файла с данными разработать самостоятельно. Оформить программу в виде оконного приложения. Разработать интерфейс.

Предусмотреть возможность выгрузки данных в Excel.

Допускается подгружать изображения для полей БД.

Для хранения данных компонент StringGrid использовать нельзя (допускается использование только для вывода результирующих значений).

2 ВВЕДЕНИЕ

В последнее время информационные технологии стали неотъемлемой частью нашей жизни. Информационные системы, связанные с предоставлением и обработкой информации для всех уровней управления различными объектами, приобретают особую важность в общественной жизни. На данный момент невозможно представить пользователя, не применяющего компьютерных технологий.

Основными задачами данной работы являются:

- произвести исследование предметной области;
- на основании полученных знаний спроектировать структуру хранения информации;
- разработать программу для работы с базой данных (БД).

В методических указаниях рассмотрено проектирование БД в среде визуального программирования Lazarus (аналог Delphi). Это мощная система визуального объектно-ориентированного программирования. С ее помощью, даже начинающие программисты, могут создавать оконные интерфейсы, удовлетворяющие стандартам Windows, причем очень быстро. Спектр отраслей, в которых возможно применение средств визуального программирования, достаточно широк: инженерные, офисные, торговые и др. Также можно разрабатывать библиотеки .DLL компонентов, форм, функций.

Выбор среды проектирования и языка программирования осуществляется студентом самостоятельно (Lazarus, Visual Studio Express edition и т.п.).

Отчет по лабораторной работе должен содержать следующие пункты

Введение
1 Анализ задачи
1.1 Описание предметной области
1.2 Назначение программы
1.3 Организация входных и выходных данных
1.4 Выбор метода решения
1.5 Язык программирования
2 Разработка программного обеспечения
2.1 Проектирование интерфейса в среде визуального программирования
2.2 Описание структуры программы
2.3 Блок-схемы
3 Руководство пользователя
3.1 Системные требования
3.2 Инструкция пользователя
Выводы по работе

Варианты заданий:

1. База данных «Студенты». Набор полей: ФИО, группа, год поступления.
2. База данных «Детали». Набор полей: наименование детали, цвет покрытия, числовой код детали.
3. База данных «Грибы». Набор полей: название гриба, ядовитость, цвет.
4. База данных «Книги». Набор полей: автор, название произведения, количество страниц.
5. База данных «Библиотека». Набор полей: название книги, количество страниц, шифр.
6. База данных «Автомобили». Набор полей: производитель автомобиля, марка, цвет.
7. База данных «Процессоры». Набор полей: название процессора, размер КЭШа, поддержка технологии Hyper-Threading.

8. База данных «Модули памяти». Набор полей: brand name модуля, тип памяти (SDR, DDR, DDR2, RDRAM), объем.
9. База данных «Материнские платы». Набор полей: модель, тип сокета, год выпуска.
10. База данных «Мониторы». Набор полей: модель, размер диагонали, соответствие стандарту MPR II.
11. База данных «Клавиатуры». Набор полей: модель, количество клавиш, наличие мультимедийных функций.
12. База данных «Мыши». Набор полей: фирма-производитель, количество кнопок, наличие скролла.

Рассмотрим решение задачи на примере проектирования базы данных «Книги».

3 АНАЛИЗ ЗАДАЧИ

3.1 Описание предметной области

Предметной областью для данного проекта является формирование базы данных книг. Для учёта книг необходимо оформлять и вести довольно большое количество документации. В современном мире почти каждый человек использует на своей работе вычислительную технику, так же и при учете книг работа ведётся с помощью автоматизированных систем и технологий.

Основные данные, которые использовались в данной информационной системе, были данные о книгах:

- инвентарный номер;
- автор;
- название произведения;
- количество страниц.

Входная информация: данные о доступных книгах и авторах. Информация, полученная на входе, обрабатывается и преобразуется во входные данные.

3.2 Назначение программы

Программа предназначена для ведения базы данных по книгам (в домашней библиотеке, магазине и т.п.).

Программа позволяет выполнять:

- добавление новых книг,
- удаление книг,
- редактирование записей,
- сортировку по различным полям,

- поиск по составному критерию.

3.3 Организация входных и выходных данных

Входные данные:

количество записей,

массив записей, каждая из которых содержит данные по книге:

- инвентарный номер;
- автор;
- название произведения;
- количество страниц.

Выходные данные: результат обработки исходных массивов данных.

Для хранения информации о книге будет использован тип - запись.

Запись представляет собой совокупность ограниченного числа логически связанных компонент, принадлежащих к разным типам. Компоненты записи называются полями, каждое из которых определяется именем. Поле записи содержит имя поля, вслед за которым через двоеточие указывается тип этого поля. Поля записи могут относиться к любому типу, допустимому в языке Паскаль, за исключением файлового типа.

Описание записи в языке Object Pascal осуществляется с помощью служебного слова RECORD, вслед за которым описываются компоненты записи. Завершается описание записи служебным словом END.

Например, записная книжка содержит фамилии, инициалы и номера телефона, поэтому отдельную строку в записной книжке удобно представить в виде следующей записи:

```
type   Row=Record
                FIO: String[20];
                TEL: String[7]
               end;
     var    str: Row;
```

Описание записей возможно и без использования имени типа, например:

```
var  str: Record
                FIO: String[20];
                TEL: String[7]
               end;
```

Обращение к записи в целом допускается только в операторах присваивания, где слева и справа от знака присваивания используются имена записей одинакового типа. Во всех остальных случаях оперируют отдельными полями записей. Чтобы обратиться к отдельной компоненте записи, необходимо задать имя записи и через точку указать имя нужного поля, например:

str.FIO, str.TEL

Такое имя называется составным. Компонентой записи может быть также запись, в таком случае составное имя будет содержать не два, а большее количество имен.

Обращение к компонентам записей можно упростить, если воспользоваться оператором присоединения with.

Он позволяет заменить составные имена, характеризующие каждое поле, просто на имена полей, а имя записи определить в операторе присоединения:

with M do OP;

Здесь M – имя записи, OP – оператор, простой или составной. Оператор OP представляет собой область действия оператора присоединения, в пределах которой можно не использовать составные имена.

Иногда содержимое отдельной записи зависит от значения одного из ее полей. В языке Object Pascal допускается описание записи, состоящей из общей и вариантной частей. Вариантная часть задается с помощью конструкции

case P of,

где P – имя поля из общей части записи. Возможные значения, принимаемые этим полем, перечисляются так же, как и в операторе варианта. Однако вместо указания выполняемого действия, как это делается в операторе варианта, указываются поля варианта, заключенные в круглые скобки. Описание вариантной части завершается служебным словом end. Тип поля P можно указать в заголовке вариантной части, например:

case P: Integer of

Инициализация записей осуществляется с помощью типизированных констант:

```
type
  RecType= Record
            x,y: Word;
            ch: Char;
            dim: Array[1..3] of Byte
           end;
const
  Rec: RecType= ( x: 127; y: 255;
                  ch: 'A';
                  dim: (2, 4, 8) );
```

3.4 Выбор метода решения

Для решения поставленной задачи будем использовать:

– поиск;

– сортировку.

Для сортировки массива будем использовать алгоритм из семейства алгоритмов сортировки – методов обменов или транспозиций, предусматривающих систематический обмен местами между элементами пар, в которых нарушается упорядоченность, до тех пор, пока таких пар не останется. Имеются четыре основных методов сортировки, для которых обмен является основной операцией: обменную сортировку с выбором (метод пузырька), обменную сортировку со слиянием (параллельную сортировку Бэтчера), обменную сортировку с разделением (быструю сортировку Хоара) и поразрядную обменную сортировку. Мы будем использовать метод пузырька. Пожалуй, наиболее очевидный способ обменной сортировки – сравнить $К_1$ с $К_2$, меняя местами R_1 и R_2, если их ключи расположены не в нужном порядке, и затем проделать то же самое с R_2 и R_3, R_3 и R_4 и т. д. При выполнении этой последовательности операций записи с большими ключами будут продвигаться вправо; и действительно, все это закончится тем, что запись с наибольшим ключом займет положение R_N. При многократном выполнении данного процесса соответствующие записи попадут в позиции $R_{N–1}$, $R_{N–2}$ и т. д., так что, в конце концов, все записи будут упорядочены. Метод назван «методом пузырька» потому что большие элементы, подобно пузырькам, «всплывают» на соответствующую позицию в противоположность «методу погружения» (т. е. методу простых вставок), в котором элементы погружаются на соответствующий уровень. Метод пузырька известен и под более прозаическими именами, такими как «обменная сортировка с выбором» и «метод распространения»

Нетрудно видеть, что после каждого просмотра последовательности все записи, расположенные выше самой последней, которая участвовала в обмене, и сама эта запись должны занять свои окончательные позиции, так что их не нужно проверять при последующих просмотрах.

Алгоритм В (Метод пузырька). Записи R_1, ..., R_N перекомпоновываются в пределах того же пространства памяти; после завершения сортировки их ключи будут упорядочены так: $К_1 \le \ldots \le К_N$.

В1. [Начальная установка BOUND.] Присвоить BOUND $\leftarrow$ N. (BOUND – индекс самого верхнего элемента, о котором еще не известно, занял ли он свою окончательную позицию; таким образом, можно отметить, что в начале сортировки еще ничего не известно о порядке размещения записей.)

В2. [Цикл по j.] Присвоить t $\leftarrow$ 0. Выполнить шаг В3 при j = 1, 2, ..., BOUND – 1. Затем перейти к шагу В4. (Если BOUND = 1, то сразу перейти к В4.)

В3. [Сравнение/обмен R_j : R_{j+1}] Если $K_j > K_{j+1}$, то поменять местами $R_j \leftrightarrow R_{j+1}$ и установить t $\leftarrow$ j.

В4. [Были ли обмены?] Если t = 0, завершить выполнение процедуры. В

противном случае присвоить BOUND ← t и возвратиться к шагу B2.

Последовательный поиск

«Начать с начала и продолжать, пока не будет найден искомый ключ; затем остановиться.» Эта последовательная процедура представляет собой очевидный путь поиска и может служить отличной отправной точкой для рассмотрения множества алгоритмов поиска, поскольку они основаны на последовательной процедуре. Мы увидим, что за простотой последовательного поиска скрывается ряд очень интересных, несмотря на их простоту, идей.

Алгоритм S (Последовательный поиск (Sequential search)). Дана таблица записей R_1, R_2, ..., R_N с ключами K_1, K_2, ..., K_N соответственно. Алгоритм предназначен для поиска записи с заданным ключом K. Предполагается, что $N \geq 1$.

S1. [Инициализация.] Установить i ← 1.

S2. [Сравнение.] Если $K = K_i$, алгоритм заканчивается успешно.

S3. [Продвижение.] Увеличить i на 1.

S4. [Проверка окончания?] Если $i \leq N$, перейти к шагу S2. В противном случае алгоритм заканчивается неудачно.

Обратите внимание на то, что этот алгоритм может завершиться успешно (искомый ключ найден) и неудачно (искомый ключ отсутствует).

Данный алгоритм, несомненно, знаком всем программистам, но лишь некоторые из них знают, что этот способ – не самая лучшая реализация последовательного поиска! Небольшое изменение – и алгоритм выполняется существенно быстрее (если записей не слишком мало).

3.5 Язык программирования

В качестве средства разработки приложения была выбрана система программирования Delphi и язык Object Pascal.

Delphi позволяет:

– Создавать законченные приложения для Windows самой различной направленности, от чисто вычислительных и логических, до графических и мультимедиа.

– Быстро создавать (даже начинающим программистам) профессионально выглядящий оконный интерфейс для любых приложений, написанных на любом языке; интерфейс удовлетворяет всем требованиям Windows и автоматически настраивается на ту систему, которая установлена на компьютере пользователя, поскольку использует многие функции, процедуры, библиотеки Windows.

– Создавать мощные системы работы с локальными и удаленными базами данных любых типов; при этом имеются средства автономной отладки приложений с последующим выходом в сеть.

– Создавать многозвенные распределенные приложения, основанные на различных технологиях.

– Создавать приложения, которые управляют другими приложениями, в частности такими программами Microsoft Office, как Word, Excel и др.

– Создавать кросс-платформенные приложения, которые можно компилировать и эксплуатировать как в Windows, так и в системе Linux.

– Создавать приложения различных классов для работы в Интернет и в интранет.

– Создавать профессиональные программы установки для приложений Windows, учитывающие всю специфику и все требования Windows.

– и многое, многое другое, включая создания отчетов, справочных систем, библиотек DLL, компонентов ActiveX и т.п.

Delphi - прекрасная система визуального объектно-ориентированного проектирования, одинаково радующая и новичков в программировании, и профессионалов. Начинающим Delphi позволяет сразу, с небольшими затратами времени и сил создавать прикладные программы, которые внешне неотличимы от программ, созданных профессионалами. А для опытного программиста Delphi открывает неограниченные возможности для создания сколь угодно сложных программ любого типа, в том числе, распределённых приложений, работающих с любыми базами данных.

Delphi - это средство визуального программирования, в основе которого лежит объектно-ориентированный язык. В Delphi многие действия требуют гораздо меньше времени и выполняются более интуитивно.

Для быстрого создания приложений необходим иной взгляд на программирование вообще. Для этого основой Delphi стал объектно-ориентированный Pascal (который так и называется Object Pascal и сильно отличается от стандарта языка).

Теперь программист не пишет стандартный код, а оперирует с более абстрактными понятиями - классами, событиями, свойствами, компонентами.

Для ясности необходимо определить понятие компонента. Взгляните на ваш Windows или на Internet Explorer. Перед Вами кнопки, полосы прокрутки, выпадающие списки, меню и т.д. Все это - компоненты. А зачем писать что-то по сто раз, когда можно использовать библиотеку визуальных компонентов - VCL (Visual Component Library). Вам нужна кнопка. Зачем вспоминать длинные строчки кода - просто поместите на рабочую формочку нужный компонент.

Delphi - это продукт, уникальным образом сочетающий высокопроизводительный компилятор, объектно-ориентированные средства

визуального программирования и универсальный механизм доступа к базам данных.

Для многих программистов немаловажным фактором является способность системы работать с базами данных. Delphi поддерживает базы данных, причем с той же присущей системе визуальностью. С таким же изяществом позволяет она разрабатывать и клиентский уровень СУБД "клиент-сервер", что чрезвычайно актуально сейчас. Встроенные мастера позволяют быстро создать необходимые компоненты и процедуры, а программисту остается лишь дописать свой код.

Действительно, система настолько интуитивна и интеллектуальна, что её могут использовать все - от самого крутого программиста до простого пользователя Windows.

Таким образом, в данной работе именно система Delphi выбрана базовой для разработки пользовательских WINDOWS–приложений.

4 РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

4.1 Проектирование интерфейса

Запустим Delphi выбрав в меню Borland Delphi 7 команду Delphi 7 и в Начать новый проект можно несколькими способами:

1. Команда File → New → Application.
2. Команда File → New → Other.
3. Кнопка быстрого вызова  на панели инструментов.

При выборе этой команды открывается окно депозитария New Items, хранящее образцы компонентов, форм и проектов (рис. 1).

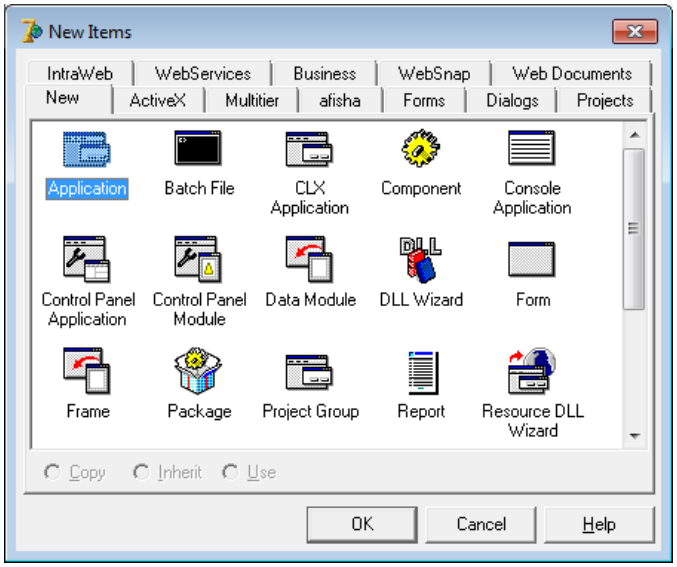

Рис. 1. Страница New окна депозитария New Items

После того, как выбрано создание нового проекта, на экране дисплея

появится главное окно среды программирования, вид которого представлен на рис. 2.

В верхней части главного окна размещены: меню, настраиваемая панель инструментов (слева) и палитра компонентов (справа). В средней части окна размещены панель Инспектора объектов (слева) и форма нового приложения (справа). Форма почти полностью закрывает окно редактора кода. На палитре компонентов, выполненной в виде многостраничного блокнота, размещены объекта (компоненты), используемые при визуальном проектировании программы.

Окно стартовой формы (Forml) представляет собой заготовку главного окна разрабатываемого приложения.

Окно Object Inspector – окно редактора свойств объектов предназначено для редактирования значений свойств объектов. В терминологии визуального проектирования объекты – это диалоговые окна и элементы управления (поля ввода и вывода, командные кнопки, переключатели и др.). Свойства объекта – это характеристики, определяющие вид, положение и поведение объекта. Например, свойства Width и Height задают размер (ширину и высоту) формы, свойства Top и Left – положение формы на экране, свойство Caption – текст заголовка.

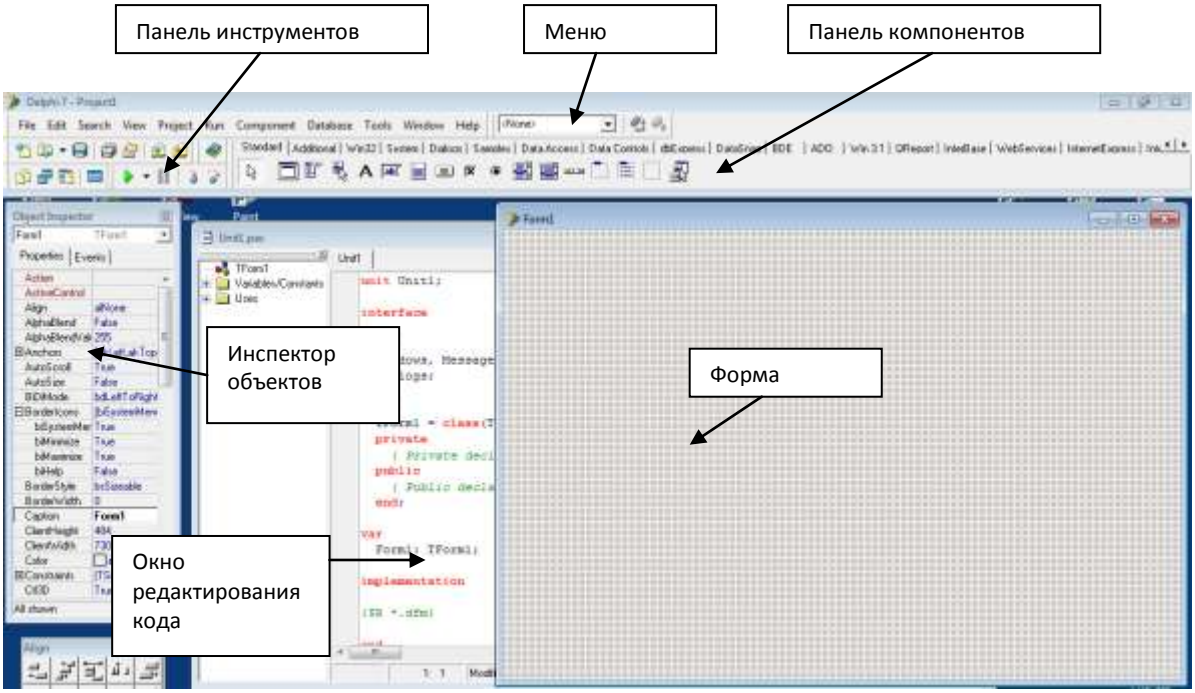

Рис. 2. Главное окно среды программирования на языке Borland Delphi 7.

В окне редактора кода, которое можно увидеть, отодвинув в сторону окно формы, следует набирать текст программы. В начале работы над новым проектом это окно редактора кода содержит сформированный Delphi шаблон программы.

Компонент можно выбрать щелчком мыши на нем в Форме или из списка Инспектора объектов. Каждый компонент имеет свои свойства и свои события.

Скрыть/Показать окно можно через команду Вид (View).

Созданное в Delphi приложение – это проект, состоящий из элементов:

- код проекта (.bpr),
- описания форм (.dfm),
- модули форм (.cpp),
- заголовочные модули (.p),
- описание ресурсов (.res).

Целесообразно для каждого нового проекта создавать свою папку, чтобы не путаться в многочисленных файлах, относящихся к конкретному проекту.

Приступаем к размещению на форме элементов управления. Это производится следующим образом: выбираем необходимый элемент управления, щелкаем по нему, переводим мышь на поле формы и нажав на клавишу мыши растянем пунктирный прямоугольник до нужного нам размера. Отпустив клавишу мыши увидим, что на форме появился выбранный элемент. Также элемент управления помещается на форму после двойного щелчка на нем.

В окне свойств мы увидим его свойства, которые при необходимости будем изменять. Имя элемента управления определяется свойством Name и задается средой программирования по умолчанию. При необходимости его можно изменить.

Размещаем на главной форме следующие компоненты:

CheckBox (страница Standard) - кнопка с независимой фиксацией (флажком) и служит в качестве двоичного переключателя режимов в программе (переключается одинарным щелчком мыши на компоненте). В программе будет использована для переключения режима автосохранения записей.

ImageList – набор рисунков (страница Win32). Представляет собой хранилище для нескольких рисунков одинакового размера. Используется для хранения пиктограмм кнопок инструментальной панели.

MainMenu – главное меню программы (страница Standard). Компонент способен создавать и обслуживать сложные иерархические меню. После установки компонента на форму необходимо создать его опции. Для этого следует дважды щелкнуть по компоненту левой кнопкой мыши, либо нажать на нем правую кнопку и выбрать продолжение Menu Designer в появившемся вспомогательном меню, либо, наконец, щелкнуть по кнопке в правой половине строки Items Инспектора объектов.

Создание опций. Перейдите в окно Инспектора объектов и введите текст опции в строке Caption, после чего нажмите Enter – опция готова, и можно переходить к следующей.

OpenDialog - открыть (страница Dialogs). Реализует стандартное диалоговое окно «Открыть файл». Используется для выбора файла из которого будут загружены данные.

SaveDialog - сохранить (страница Dialogs). Реализует стандартное диалоговое окно «Сохранить файл». Используется для выбора файла в котором будут сохранены данные.

StatusBar - панель статуса (страница Win32). Предназначена для размещения служебной информации: количества записей, номера текущей записи, имени текущего файла данных.

StringGrid - таблица строк (страница Additional). Этот компонент обладает мощными возможностями для представления текстовой информации в табличном виде. В нем будут отображаться записи базы данных.

ToolBar – инструментальная панель (страница Win32). Компонент служит контейнером для командных кнопок. Чтобы вставить его в инструментальную панель нужно щелкнуть правой кнопкой на компоненте TToolBar и выбрать NewButton или NewSeparator.

Макет формы приведен на рис. 3.

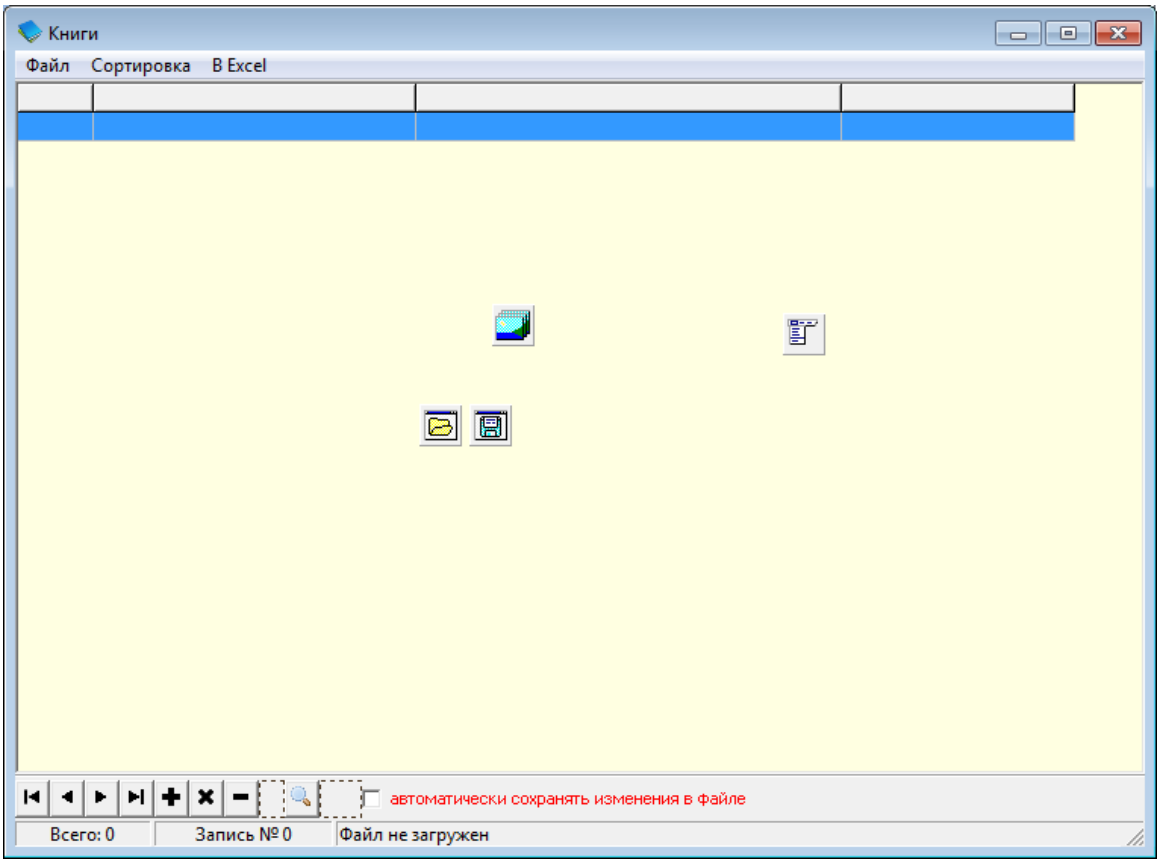

Рис. 3. Макет главной формы

Вторая форма предназначена для работы с записями. Чтобы добавить форму к проекту необходимо нажать кнопку New Form  на панели инструментов (или из меню File выбрать New → Form). На неё помещаем компоненты:

BitBtn - командная кнопка с надписью и пиктограммой (страница Additional). Размещаем две кнопки «ОК» и «Отмена».

Edit - строка ввода (страница Standard). Предназначена для ввода, отображения или редактирования полей одной записи.

Label - представляет собой статический текст и служит для отображения информации (страница Standard). В метки выводятся пояснения к полям ввода.

Макет формы приведен на рис. 4.

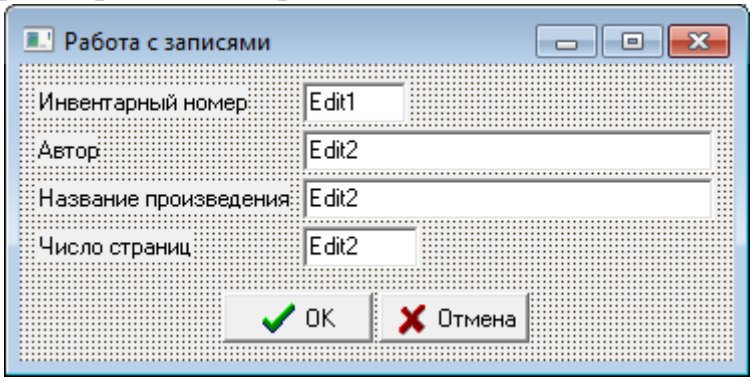

Рис. 4. Макет формы работы с записями

Третья форма предназначена для вывода результатов поиска. На ней размещен один компонент StringGrid - таблица строк. В ней будут отображаться найденные записи базы данных.

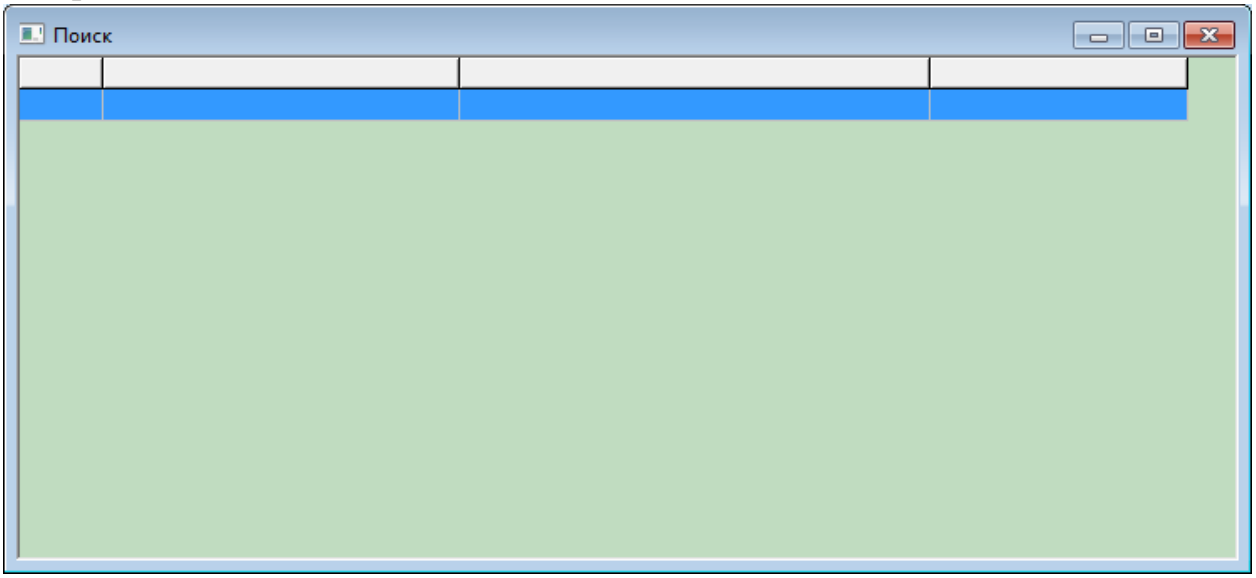

Рис. 5. Макет формы вывода результатов поиска

С помощью команды Save Project As меню пункта File сохраним проект. После выбора этой команды появится диалоговое окно для сохранения файла кода программы. По умолчанию файлу дается расширение .PAS, указанное в окне «тип файла». Вводим значения для всех форм и нажимаем OK. Далее появится следующее диалоговое окно для сохранения файла проекта (расширение DPR). Вводим имя проекта – books и нажимаем OK. Для последующего сохранения ранее созданного проекта достаточно выполнить команду Save all этого же меню.

4.2 Описание структуры программы

Состав файлов проекта приведен в таблице 1.

Таблица 1. Состав проекта

Наименование	**Обозначение**	**Примечание**
Главная форма Form1		
frm_main.dcu	Объектный файл для frm_main	Скомпилированный файл
frm_main.dfm	Файл формы Form1	Содержит список свойств всех компонентов.
frm_main.pas	Исходный код главной формы	Исходный код
Форма работы с записями Form2		
frm_recs.dcu	Объектный файл для frm_recs	Скомпилированный файл
frm_recs.dfm	Файл формы Form2	Содержит список свойств всех компонентов.
frm_recs.pas	Исходный код формы работы с записями	Исходный код
Форма вывода результатов запросов Form3		
frm_Find.dcu	Объектный файл для frm_Find	Скомпилированный файл
frm_Find.dfm	Файл формы Form3	Содержит список свойств всех компонентов.
frm_Find.pas	Исходный код формы вывода результатов запросов	Исходный код
Файлы проекта		
books.dpr	Файл проекта	Связывает все файлы из которых состоит приложение
books.dof	Файл параметров проекта	Содержит текущие установки проекта
books.res	Содержит ресурсы	Содержит иконку

Описание полей структуры данных TBook, описывающей книгу, приведено в таблице 2.

Таблица 2. Поля структуры данных TBook

Наименование	**Назначение**	**Тип**	**Размер**

Invn	инвентарный номер	integer	
avtor	автор	string	50
Nazv	название произведения	string	100
Stran	количество страниц	integer	

Глобальные переменные:

Books - массив записей, тип array of TBook.

Nrec - число записей, тип integer.

curFileName - имя текущего загруженного файла данных, тип string.

Процедуры и функции общего назначения.

ToGrid(row, temp, StringGrid) - помещение записи в таблицу.

Входные параметры:

row - номер строки в которую помещается запись, тип integer;

temp - запись, тип TBook;

StringGrid - указатель на таблицу, тип TStringGrid.

Функция FromGrid(row) - извлечение записи из таблицы.

Входные параметры:

row - номер строки из которой читается запись, тип integer.

Возвращает значение типа TBook с информацией по записи.

MasToGrid - вывод массива в таблицу.

Локальные переменные:

i - счетчик цикла, тип integer.

FileToGrid(fname, N) - чтение записей из файла и помещение их в таблицу и массив

Входные параметры:

fname - имя файла, тип string.

N - возвращает количество считанных записей, тип integer.

SaveData(fname) - сохранить записи в файле.

Входные параметры:

fname - имя файла, тип string.

Событийные процедуры.

FormCreate - инициализация.

mnuOtkrytClick - открыть файл и загрузить данные.

mnuVyhodClick - выход из программы.

firstButtonClick - перейти к первой записи таблицы.

prevButtonClick - перейти к предыдущей записи таблицы.

nextButtonClick - перейти к следующей записи таблицы.

lastButtonClick - перейти к последней записи таблицы.

StringGrid1SelectCell - вывод номера выбранной записи в строку статуса.

mnuSohranitClick - сохранить записи в файле.

mnuSohranitKakClick - сохранить файл под новым именем.
addButtonClick - добавить запись.
Локальные переменные:
temp - информация о книге, тип TBook.
i - счетчик цикла, тип integer.
maxin - максимальный инвентарный номер в базе, тип integer.
e**ditButtonClick** - редактировать запись.
Локальные переменные:
temp - информация о книге, тип TBook.
r - номер выбранной записи, тип integer.
deleteButtonClick - удалить запись из таблицы.
Локальные переменные:
i - счетчик, тип integer.
r - номер выбранной записи, тип integer.
CheckBox1Click - включение / выключение режима автосохранения.
mnuSortClick - сортировка записей.
Локальные переменные:
i, k - счетчики цикла, тип integer.
field - номер поля сортировки, тип integer.
crit - критерий сортировки, тип boolean.
temp - информация, тип TBook.
findButtonClick - поиск записей.
Локальные переменные:
n - количество найденных записей, тип integer.
i - счетчик цикла, тип integer.
crit1, crit2 - значения критериев по полям записи, тип boolean.
temp - информация, тип TBook.
mnuExportClick - экспорт базы данных в Excel.
Локальные переменные:
MsExcel - указатель на табличный процессор Excel, тип variant.
i - счетчик цикла, тип integer.

4.3 Блок-схемы

Примеры блок схем для ряда функций проекта приведены ниже.

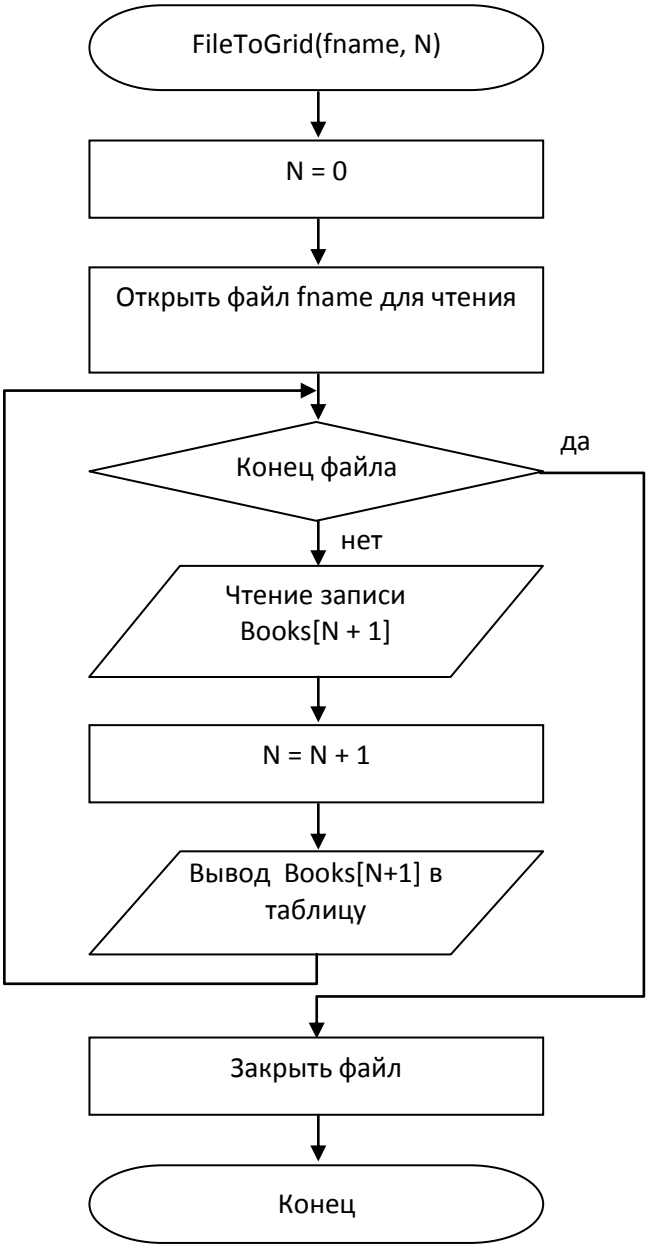

Рис. 6. Блок - схема алгоритма процедуры загрузки данных из файла

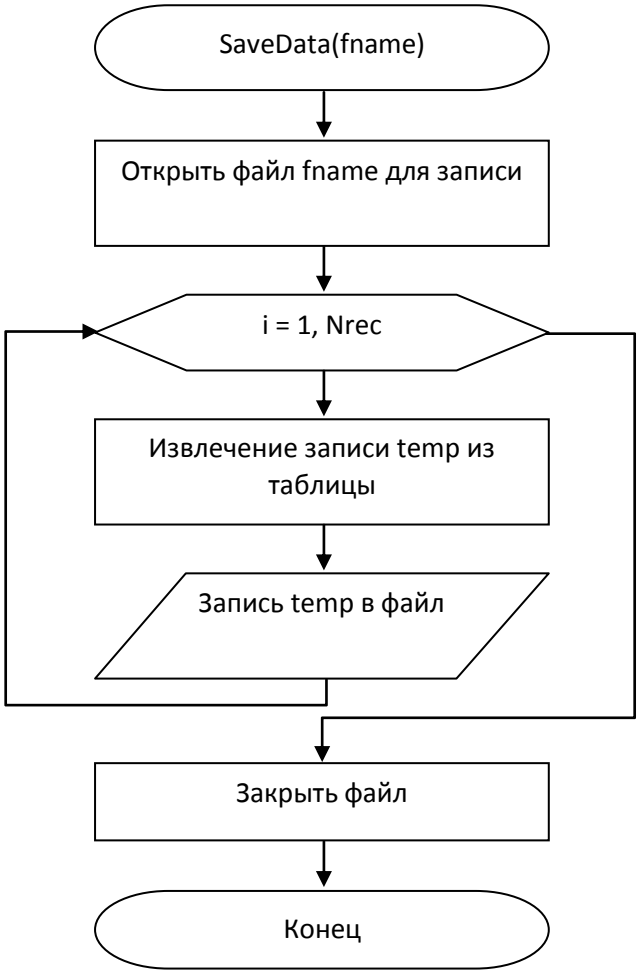

Рис. 7. Блок - схема алгоритма процедуры сохранения записей в файле

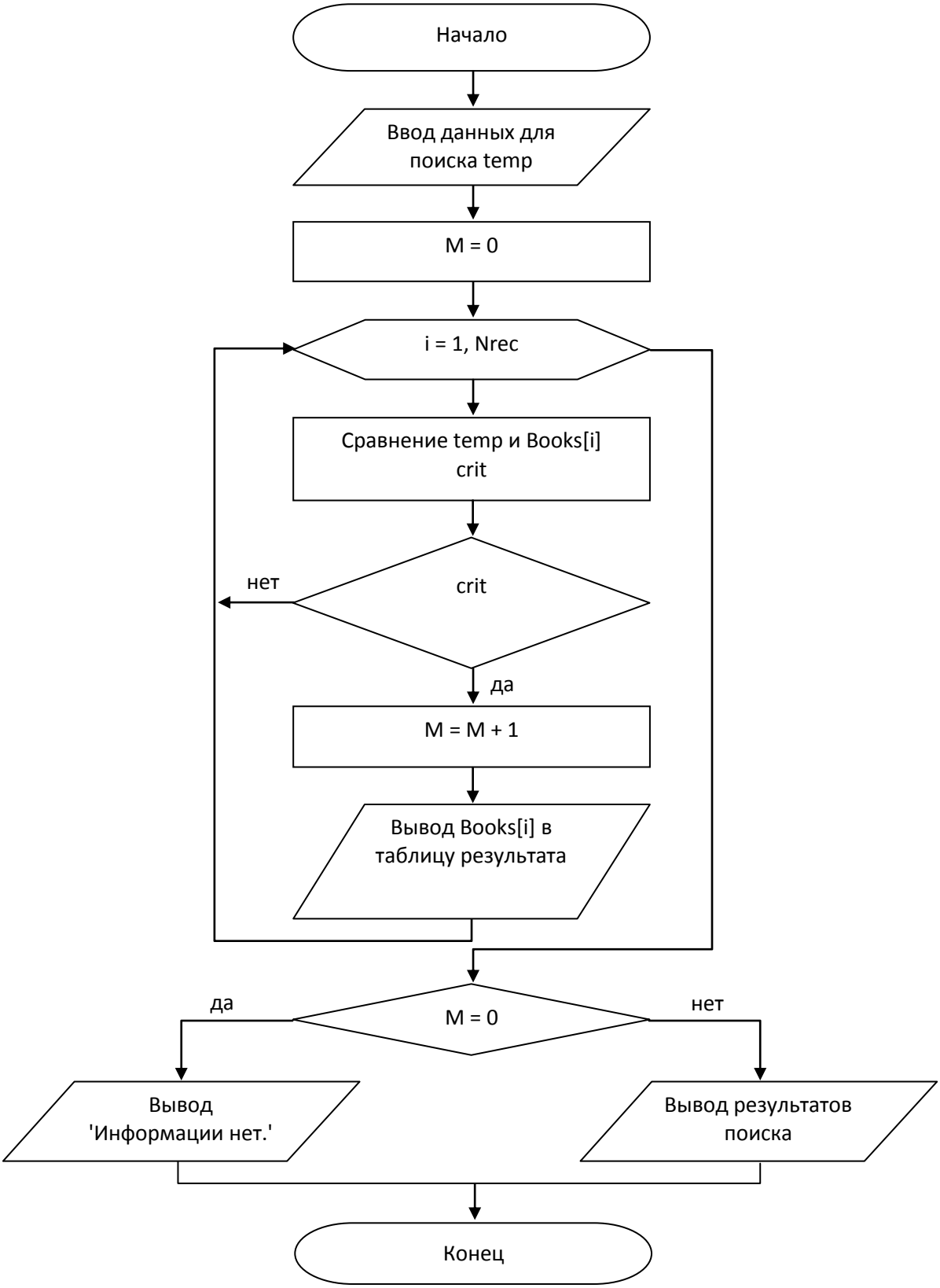

Рис. 8. Блок - схема алгоритма процедуры поиска

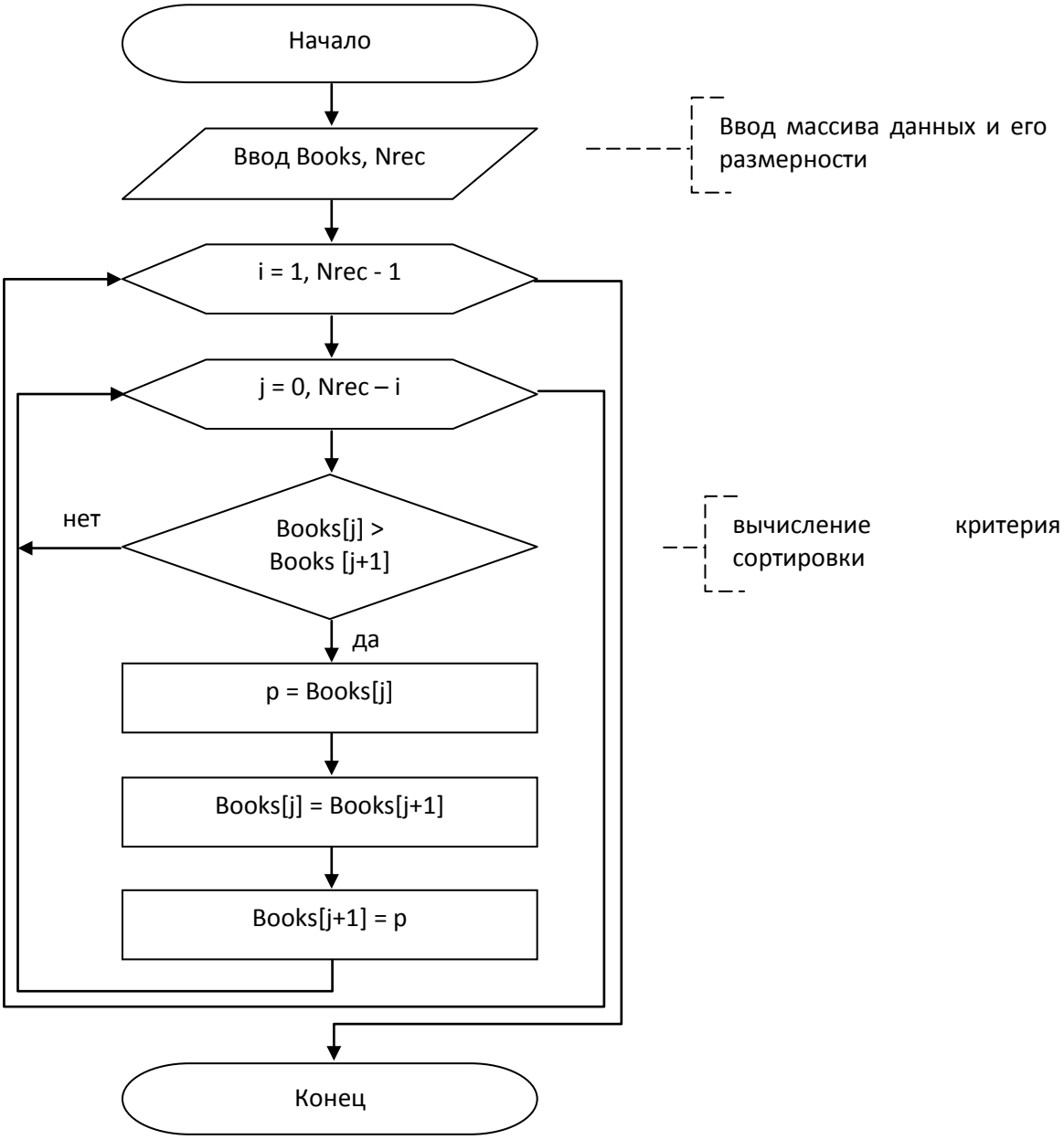

Рис. 9. Блок - схема алгоритма метода пузырька

5 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

5.1 Системные требования

Ниже приведены рекомендуемые требования для нормальной работы программы.

Компьютер на базе процессора Intel Pentium 1 ГГц или более мощный.

1 Гбайт оперативной памяти.

1 Мбайт свободного места на жестком диске.

Операционная система Windows 7, 8, 10.

5.2 Инструкция пользователя

Для установки программ на персональный компьютер пользователя,

необходимо скопировать загрузочный модуль books.exe с внешнего носителя на рабочий стол или в специально созданный каталог. В этот же каталог необходимо поместить (если он есть) файл базы данных baza.dat.

Для запуска программы необходимо с помощью окна проводника выбрать диск, на котором расположена программа – дважды щелкнуть папку с программой – установить указатель на файл books.exe и дважды щелкнуть его мышью. После запуска приложения на экран будет выведено окно консоли с меню (рис. 10).

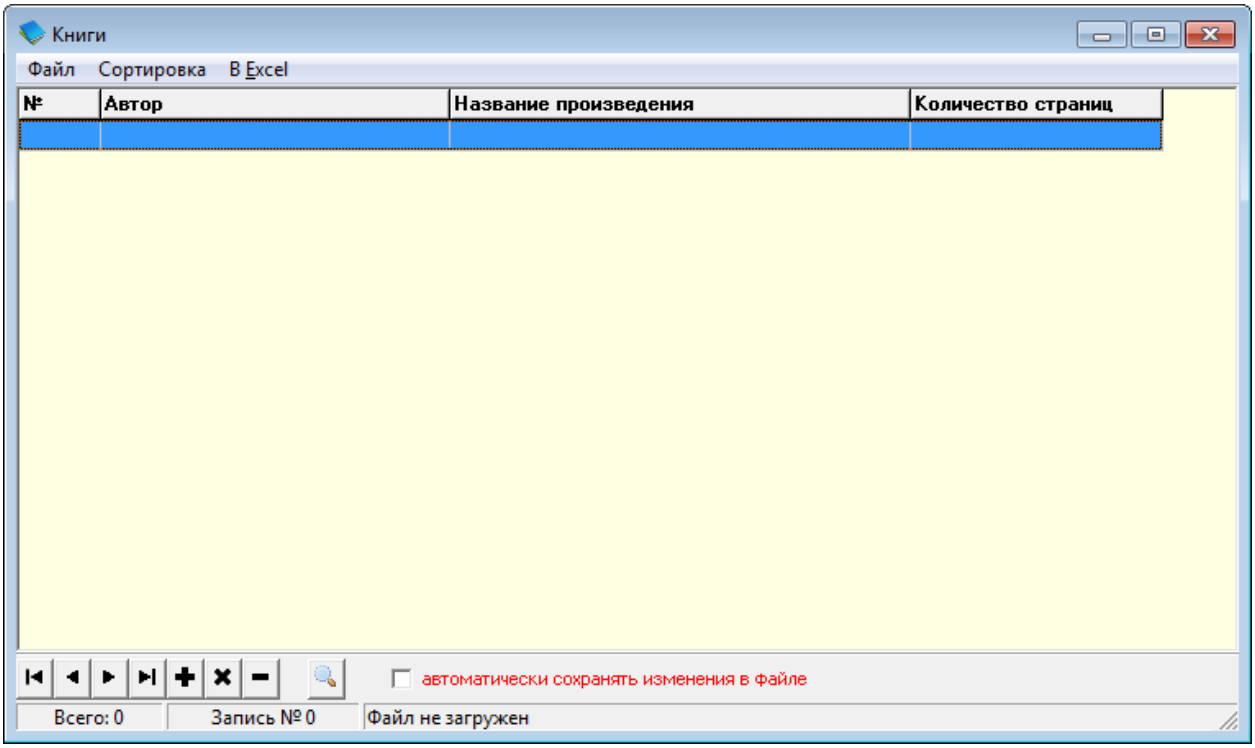

Рис. 10. Главное окно

В верхней части окна находится главное меню.

В меню «Файл» содержатся команды:

«Загрузить» – загрузка базы данных из файла;

«Сохранить» – сохранение базы данных в файл;

«Сохранить как» – сохранение базы данных в файл под новым именем;

«Выход» – завершение работы с программой.

В меню «Сортировка» содержатся команды, выполняющие сортировку по соответствующему полю:

- по номеру
- по автору
- по названию
- по количеству страниц.

Меню «В Excel» выполняет экспорт базы данных в табличный процессор Excel.

В нижней части окна находится инструментальная панель с кнопками

управления записями и информационная строка.

Назначение кнопок управления записями (слева направо)

- переход к первой записи;
- переход к предыдущей записи;
- переход к следующей записи;
- переход к последней записи;
- вставить новую запись;
- удалить текущую запись;
- редактировать запись;
- поиск.

Остальную часть окна занимает таблица базы данных.

Прежде чем приступать к непосредственной работе с данными кандидатов и вакансиями, необходимо заполнить базу данных (или загрузить ее с диска командой меню «Файл» - «Загрузить»).

Добавление записи выполняется после нажатия кнопки  . Данные в справочник вводятся в поля, расположенные в отдельном окне диалога (рис. 11) непосредственно с клавиатуры.

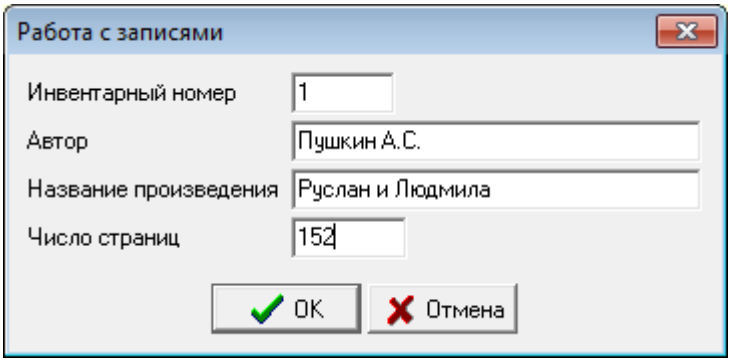

Рис. 11. Окно работы с записями

Значение в поле «Инвентарный номер» вставляется автоматически и изменению не подлежит.

Для подтверждения добавления необходимо нажать кнопку «ОК», отмена операции - кнопка «Отмена».

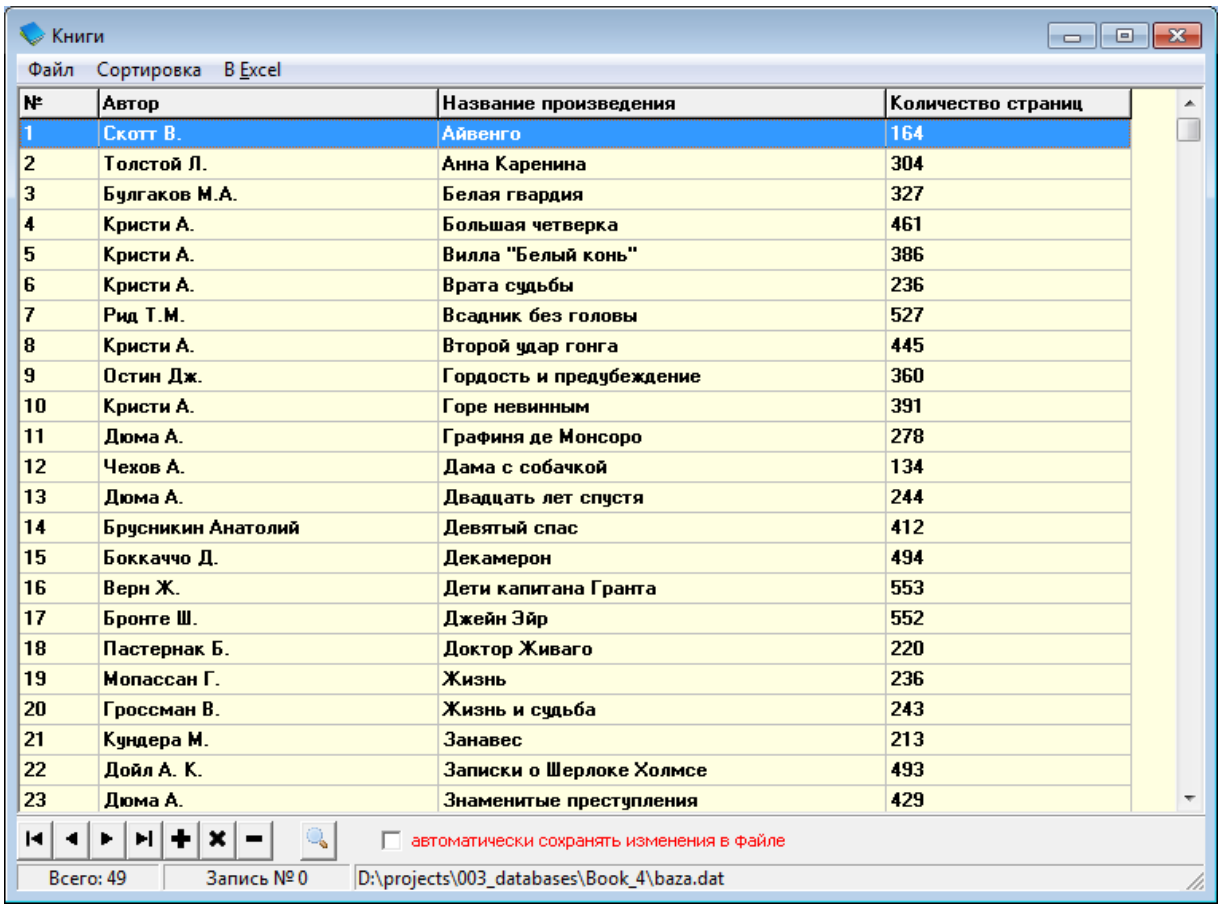

Рис. 12. Окно программы с введенными записями

После того, как в справочник введена информация, можно приступать к обработке.

Удаление записи. Предварительно удаляемая запись выделяется в таблице, после чего нажимается кнопка  . Программа выводит окно с подтверждением удаления (рис. 13). Для удаления необходимо нажать кнопку «Да», отмена операции - кнопка «Нет».

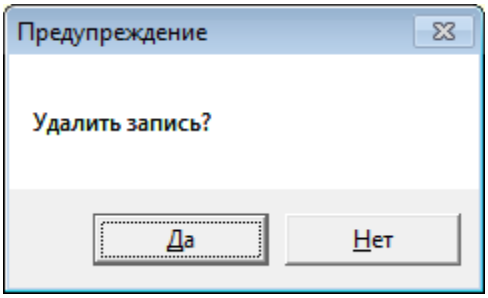

Рис. 13. Запрос на удаление записи

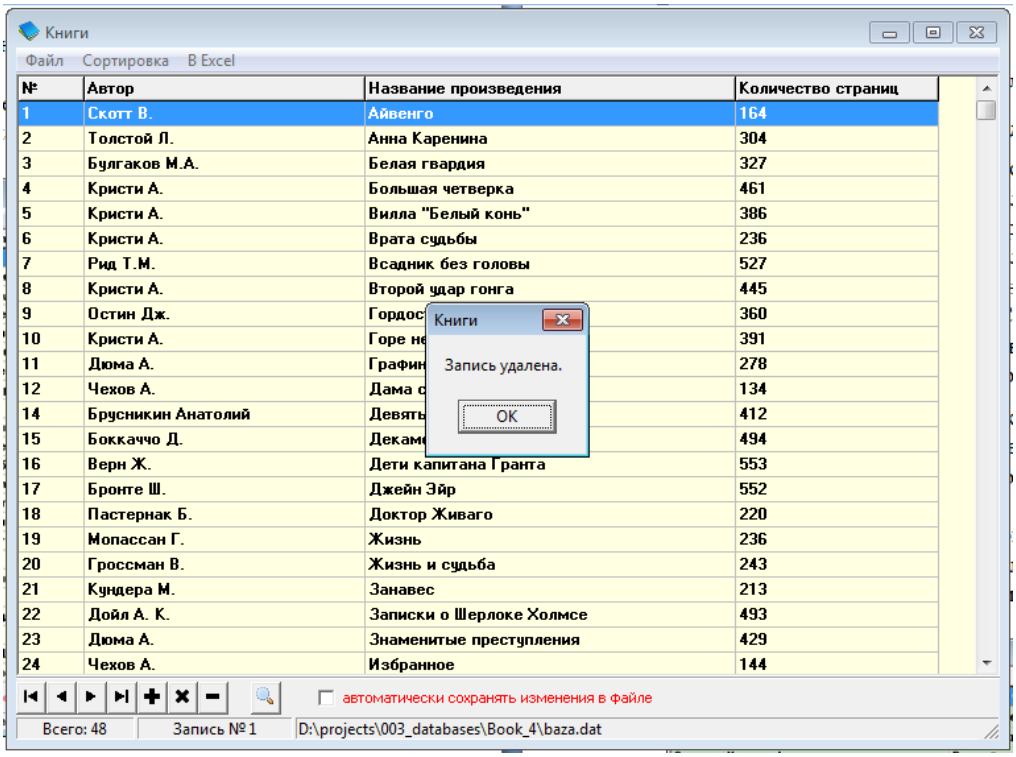

Рис. 14. Окно после удаления записи с №13

Редактирование записи. Предварительно изменяемая запись выделяется в таблице, после чего нажимается кнопка [−]. Программа выводит окно работы с записями (см. рис. 11). Для подтверждения изменений необходимо нажать кнопку «ОК», отмена операции - кнопка «Отмена».

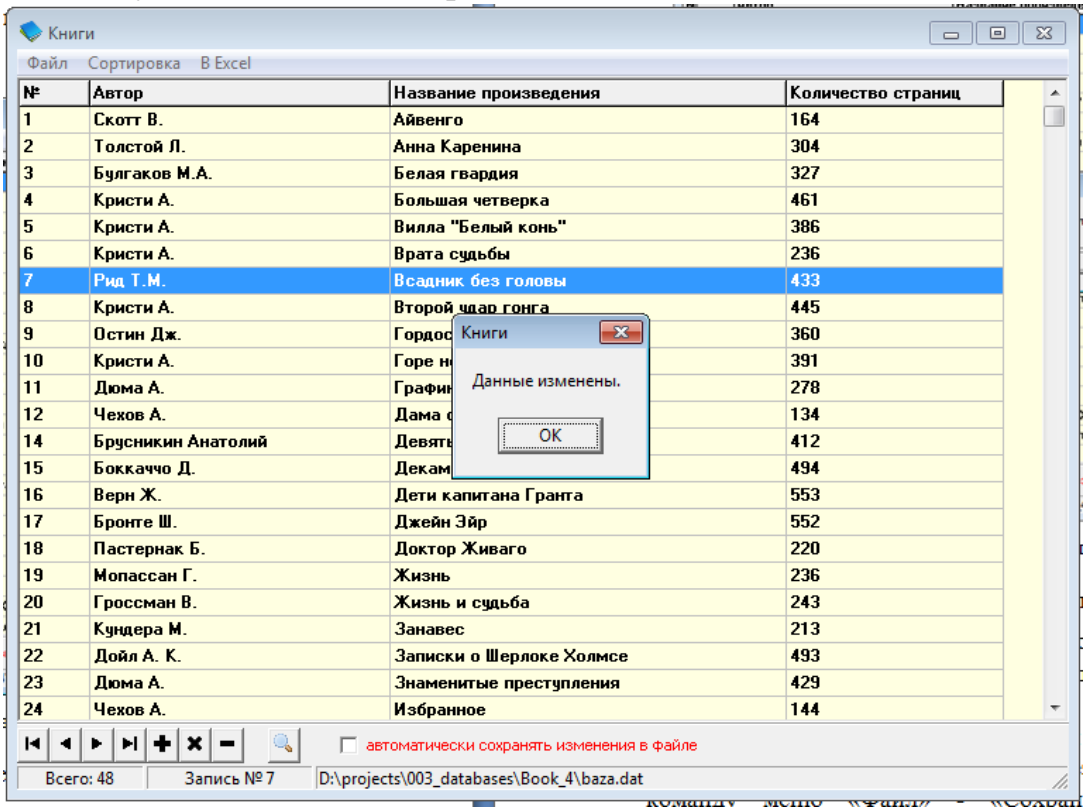

Рис. 15. Окно после изменения числа страниц в записи №7

Операции добавления, удаления и редактирования выполняются в оперативной памяти, для сохранения результатов необходимо использовать команду меню «Файл» - «Сохранить…» или установить флажок «автоматически сохранять изменения в файле». В этом случае после каждой операции будет выполняться сохранение результатов.

Для поиска необходимо нажать кнопку  и в окне диалога (см. рис. 11) задать значения для полей автор и название произведения. Критерий поиска может быть задан как полностью, так и его любая часть. Регистр символов значения не имеет.

После задания параметров нажимаем кнопку «ОК», результат выводится в таблицу в окне результата, если поиск удачен или выводится сообщение об отсутствии информации.

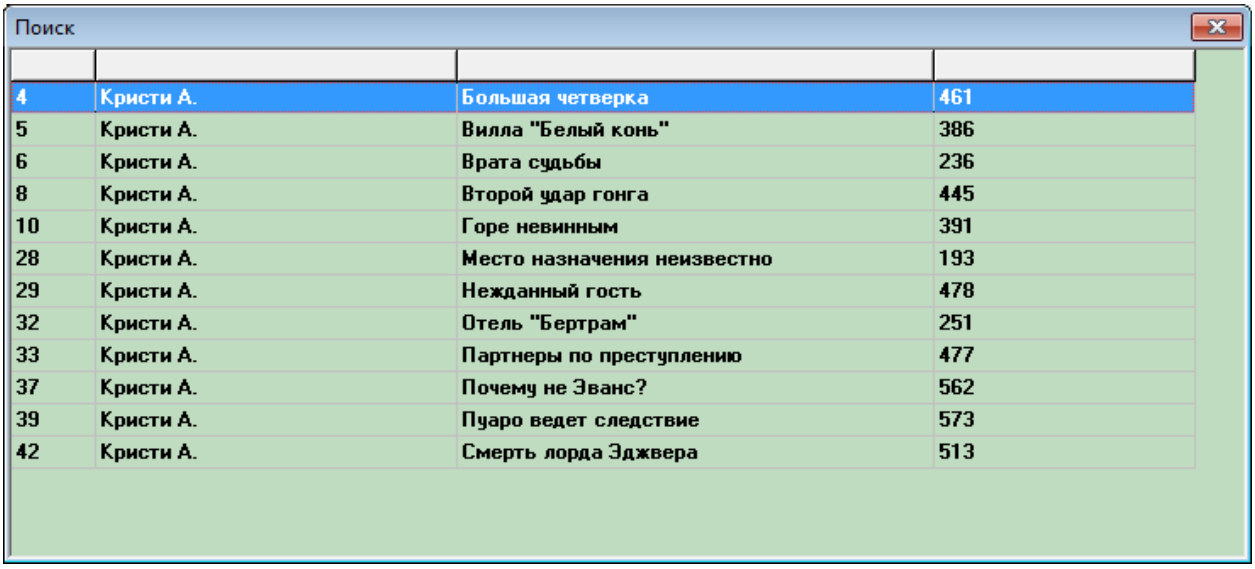

Рис. 16. Поиск для автора «Кристи»

Для сортировки, необходимо в меню «Сортировка» выбрать нужное поле.

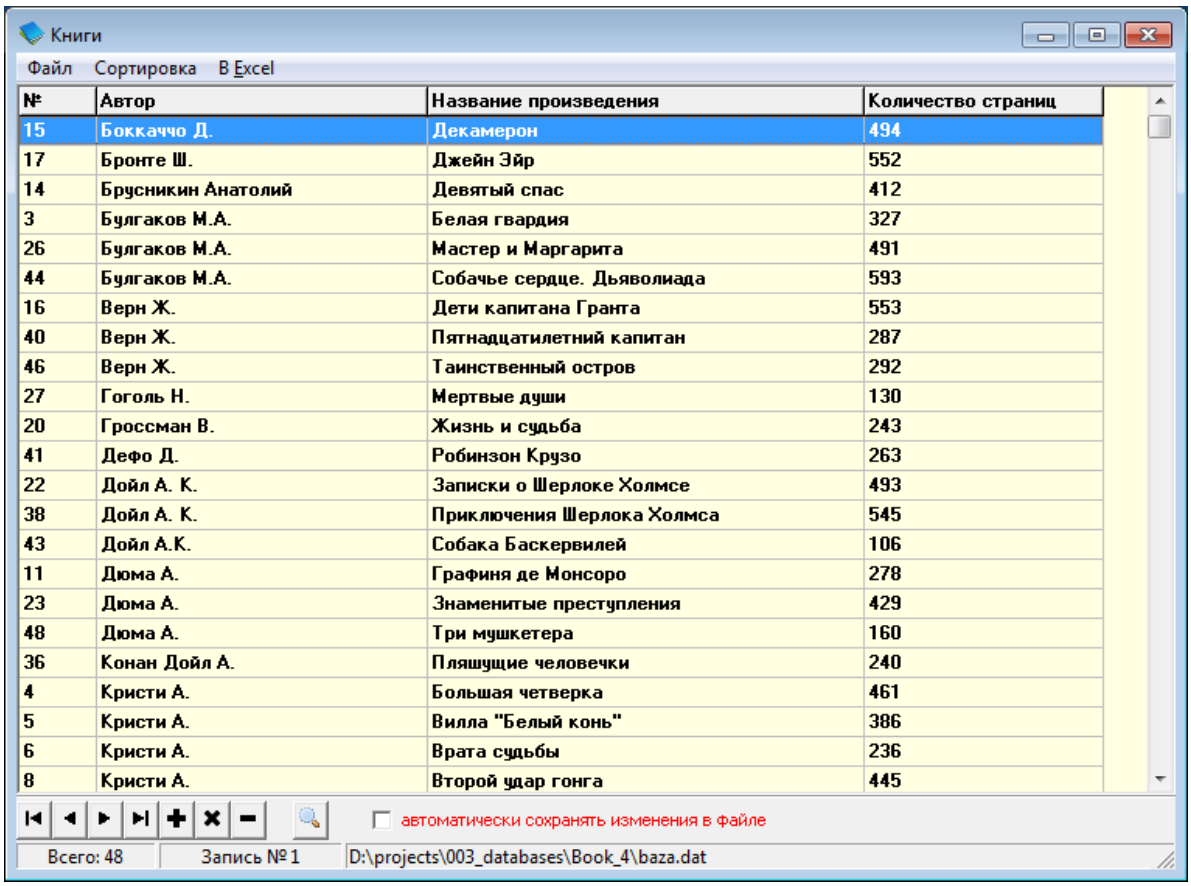

Рис. 17. Сортировка по автору

6 ВЫВОДЫ ПО РАБОТЕ

В результате выполнения работы разработана программа работы с базой данных «Книги» предназначенная для ведения учета каталога книг, поиска по каталогу и многое другое. Представляемый комплекс является полнофункциональной системой программного обеспечения личной (и не только) библиотеки и может использоваться всеми желающими для систематизации и учета книг в домашних условиях.

В ходе выполнения работы были рассмотрены основные возможности среды программирования при работе с базами данных. Изучены способы создания таблиц баз данных, с заданной структурой, способы создания простых и сложных запросов, осуществляя поиск в базах данных по различным критериям. Изучены способы создания пользовательского интерфейса с использованием форм. Разработана документация по использованию программы.

Данный программный продукт успешно прошел полное тестирование и отладку, что говорит о соответственном качестве программного продукта.

По завершению работы над этим программным средством были осуществлены все поставленные задачи.

7 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Как называются базы данных, доступ к которым невозможен по сети?
2. Базы данных, доступ к которым возможен через вычислительную сеть, называются?
3. Где централизованные базы данных хранят данные ?
4. Как называется программный код, организующий взаимодействие пользователя с базой данных ?
5. Какие основные функции ядра системы управления базами данных (СУБД)?
6. Компонент системы базы данных, который организует взаимодействие приложения БД с ядром СУБД, называется?
7. Какая модель данных используется для структурирования данных на первом этапе проектирования базы данных?
8. Какая модель данных используется для структурирования групп пользователей по правам доступа на третьем этапе проектирования базы данных?
9. Атрибутом (в реляционной модели данных) называется?
10. Первичный ключ (Primary key) в реляционной таблице – это?
11. Последовательность запросов к базе данных, выполняемых как единое, неделимое целое, называется?
12. Индекс – это?

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Гофман В.Э. Delphi 7. –Спб.: BHV–Петербург, 2013. –488 с.
2. Желонкин А. Основы программирования в интегрированной среде Delphi. Практикум. –М.: Бином. Лаборатория знаний, 2016. -240 с.
3. Кнут Д.Е. Искусство программирования. т. 3. –М.: Мир, 1984. –724 с.
4. Лишнер Рэй. Delphi. Справочник. –М.: БЕК, 2015. –715 с.
5. Пестриков В. М., Маслобоев А. Н. Delphi на примерах. –СПб.: БХВ–Петербург, 2015. –496 с.
6. Сухарев М. Delphi. Полное руководство. –М.: Наука и техника, 2017. – 1040 с.
7. Фаронов В.В. Учебный курс Delphi. -М.: Нолидж, 2010. -608 с.
8. Фаронов, В.В. Программирование в Delphi 7: учеб. пособие / В.В. Фаронов. –М.: Нолидж, 2016. –412 с.
9. Фленов М. Библия Delphi –СПб.: БХВ-Петербург, 2011. -686 с.
10. Эйдлина Г.М., Милорадов К.А. Delphi. Программирование в примерах и задачах. -М.: РИОР, Инфра-М, 2016. -116 с.

ПРИЛОЖЕНИЕ А. ТЕКСТ ПРОГРАММЫ

```
unit frm_main;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ComCtrls, StdCtrls, ToolWin, ImgList, Grids, Menus, ComObj;

type
  TBook = record//запись "Книги"
    invn: integer;//инвентарный номер
    avtor: string[50];//автор
    nazv: string[100];//название произведения
    stran: integer;//количество страниц
  end;
  TForm1 = class(TForm)
    StringGrid1: TStringGrid;
    ImageList1: TImageList;
    ToolBar2: TToolBar;
    firstButton: TToolButton;
    prevButton: TToolButton;
    nextButton: TToolButton;
    lastButton: TToolButton;
    addButton: TToolButton;
    deleteButton: TToolButton;
    editButton: TToolButton;
    ToolButton1: TToolButton;
    CheckBox1: TCheckBox;
    StatusBar1: TStatusBar;
    OpenDialog1: TOpenDialog;
    SaveDialog1: TSaveDialog;
    MainMenu1: TMainMenu;
    N1: TMenuItem;
    mnuOtkryt: TMenuItem;
    mnuSohranit: TMenuItem;
    mnuSohranitKak: TMenuItem;
    N4: TMenuItem;
    mnuVyhod: TMenuItem;
    N2: TMenuItem;
    mnuInvn: TMenuItem;
    mnuAvtor: TMenuItem;
    mnuNazv: TMenuItem;
    mnuStranic: TMenuItem;
    findButton: TToolButton;
    ToolButton3: TToolButton;
    mnuExport: TMenuItem;
    procedure FormCreate(Sender: TObject);
    function FromGrid(row: integer): TBook;
    procedure ToGrid(row: integer; temp: TBook; StringGrid: TStringGrid);
    procedure FileToGrid(fname: string; var N: integer);
    procedure MasToGrid;
    procedure SaveData(fname: string);
    procedure mnuOtkrytClick(Sender: TObject);
    procedure mnuVyhodClick(Sender: TObject);
    procedure firstButtonClick(Sender: TObject);
    procedure prevButtonClick(Sender: TObject);
    procedure nextButtonClick(Sender: TObject);
```

```
    procedure lastButtonClick(Sender: TObject);
    procedure StringGrid1SelectCell(Sender: TObject; ACol, ARow: Integer;
      var CanSelect: Boolean);
    procedure mnuSohranitClick(Sender: TObject);
    procedure mnuSohranitKakClick(Sender: TObject);
    procedure addButtonClick(Sender: TObject);
    procedure editButtonClick(Sender: TObject);
    procedure deleteButtonClick(Sender: TObject);
    procedure CheckBox1Click(Sender: TObject);
    procedure mnuSortClick(Sender: TObject);
    procedure findButtonClick(Sender: TObject);
    procedure mnuExportClick(Sender: TObject);
  private
    { Private declarations }
  public
    Books: array of TBook;//массив записей
    Nrec: integer;//число записей
    curFileName: string;//имя текущего загруженного файла данных
  end;

var
  Form1: TForm1;

implementation

uses frm_recs, frm_Find;

{$R *.dfm}
procedure TForm1.ToGrid(row: integer; temp: TBook; StringGrid: TStringGrid);
//помещение записи в таблицу
begin
  if (row < 1) then//номер строки недопустимый
    Exit;
  StringGrid.Cells[0, row] := IntToStr(temp.invn);//помещаем в ячейки
  StringGrid.Cells[1, row] := temp.avtor;
  StringGrid.Cells[2, row] := temp.nazv;
  StringGrid.Cells[3, row] := IntToStr(temp.stran);
end;
function TForm1.FromGrid(row: integer): TBook;
//извлечение записи из таблицы
begin
  if (row < 1) or (row > StringGrid1.RowCount - 1) then//номер строки недопустимый
    Exit;
  Result.invn := StrToInt(StringGrid1.Cells[0, row]);//читаем запись из ячеек таблицы
  Result.avtor := StringGrid1.Cells[1, row];
  Result.nazv := StringGrid1.Cells[2, row];
  Result.stran := StrToInt(StringGrid1.Cells[3, row]);
end;
procedure TForm1.MasToGrid;
//вывод массива в таблицу
var
  i: integer;//счетчик цикла
begin
  if Nrec = 0 then//установили число строк таблицы
    begin
      StringGrid1.RowCount := 2;
      for i := 0 to 3 do
        StringGrid1.Cells[i, 1] := '';
      Exit;
```

```
    end;
  StringGrid1.RowCount := Nrec + 1;
  for i := 1 to Nrec do
    ToGrid(i, Books[i], StringGrid1);//вывели в таблицу
end;
procedure TForm1.FileToGrid(fname: string; var N: integer);
//чтение записей из файла и помещение их в таблицу и массив
var
  f: file of TBook;//указатель на файл
  i: integer;//счетчик цикла
  temp: TBook;//запись из файла
begin
  N := 0;
  if not FileExists(fname) then//файл не существует
    begin
      ShowMessage('Файл ' + fname + ' не найден!');
      StringGrid1.RowCount := 2;
      for i := 0 to 3 do
        StringGrid1.Cells[i, 1] := '';
      Exit;
    end;
  AssignFile(f, fname); Reset(f);//открыли файл для чтения
  while not EOF(f) do//пока не конец файла
    begin
      Read(f, temp);//чтение записи
      N := N + 1;//увеличили счетчик записей
      SetLength(Books, N + 1);
      Books[N] := temp;
      //ToGrid(N, temp);//вывели в таблицу
    end;
  Closefile(f);//закрыть файл
  Nrec := N;
  MasToGrid;
end;
procedure TForm1.SaveData(fname: string);
//сохранить записи в файле
var
  f: file of TBook;//указатель на файл
  i: integer;//счетчик цикла
begin
  if Nrec = 0 then Exit;
  AssignFile(f, fname); Rewrite(f);
  for i := 1 to Nrec do
    Write(f, Books[i]);
  CloseFile(f);
end;
procedure TForm1.FormCreate(Sender: TObject);
//инициализация
begin
  StringGrid1.Cells[0, 0] := '№';
  StringGrid1.Cells[1, 0] := 'Автор';
  StringGrid1.Cells[2, 0] := 'Название произведения';
  StringGrid1.Cells[3, 0] := 'Количество страниц';
  curFileName := '';
  Nrec := 0;
end;

procedure TForm1.mnuOtkrytClick(Sender: TObject);
//открыть файл и загрузить данные
```

```
begin
  if not OpenDialog1.Execute then
    Exit;
  FileToGrid(OpenDialog1.FileName, Nrec);
  curFileName := OpenDialog1.FileName;
  StatusBar1.Panels[2].Text := curFileName;
  StatusBar1.Panels[0].Text := 'Всего: ' + IntToStr(Nrec);
end;

procedure TForm1.mnuVyhodClick(Sender: TObject);
//выход из программы
begin
  Close;
end;

procedure TForm1.firstButtonClick(Sender: TObject);
//перейти к первой записи таблицы
begin
  if Nrec = 0 then Exit;
  StringGrid1.Row := 1;
end;

procedure TForm1.prevButtonClick(Sender: TObject);
//перейти к предыдущей записи таблицы
begin
  if Nrec = 0 then Exit;
  if StringGrid1.Row <> 1 then
    StringGrid1.Row := StringGrid1.Row - 1;
end;

procedure TForm1.nextButtonClick(Sender: TObject);
//перейти к следующей записи таблицы
begin
  if Nrec = 0 then Exit;
  if StringGrid1.Row <> Nrec then
    StringGrid1.Row := StringGrid1.Row + 1;
end;

procedure TForm1.lastButtonClick(Sender: TObject);
//перейти к последней записи таблицы
begin
  if Nrec = 0 then Exit;
  StringGrid1.Row := Nrec;
end;

procedure TForm1.StringGrid1SelectCell(Sender: TObject; ACol,
  ARow: Integer; var CanSelect: Boolean);
//вывод номера выбранной записи в строку статуса
begin
  StatusBar1.Panels[1].Text := 'Запись № ' + IntToStr(ARow);
end;

procedure TForm1.mnuSohranitClick(Sender: TObject);
//сохранить записи в файле
begin
  if Nrec = 0 then Exit;
  if curFileName <> '' then
    begin
      SaveData(curFileName);
```

```
    ShowMessage('Информация сохранена');
   end
  else
   mnuSohranitKakClick(nil);
  StatusBar1.Panels[2].Text := curFileName;
end;

procedure TForm1.mnuSohranitKakClick(Sender: TObject);
//сохранить файл под новым именем
begin
  if Nrec = 0 then Exit;
  if not SaveDialog1.Execute then
   Exit;
  SaveData(SaveDialog1.FileName);
  curFileName := SaveDialog1.FileName;
  ShowMessage('Информация сохранена');
  StatusBar1.Panels[2].Text := curFileName;
end;

procedure TForm1.addButtonClick(Sender: TObject);
//добавить запись
var
  temp: TBook;//информация о книге
  i,//счетчик цикла
  maxin: integer;//максимальный номер в базе
begin
  maxin := 0;
  for i := 1 to Nrec do
   if Books[i].invn > maxin then
    maxin := Books[i].invn;
  Form2.Edit1.Text := IntToStr(maxin + 1);
  Form2.Edit2.Text := '';
  Form2.Edit3.Text := '';
  Form2.Edit4.Text := '0';
  if Form2.ShowModal <> mrOK then
   Exit;
  if (Form2.Edit1.Text = '') or (Form2.Edit2.Text = '') or
    (Form2.Edit3.Text = '') or (Form2.Edit4.Text = '') then
     begin
      ShowMessage('Необходимо заполнить все поля!');
      Exit;
     end;
  temp.invn := StrToInt(Form2.Edit1.Text);
  temp.avtor := Form2.Edit2.Text;
  temp.nazv := Form2.Edit3.Text;
  temp.stran := StrToInt(Form2.Edit4.Text);
  Inc(Nrec);
  SetLength(Books, Nrec + 1);
  Books[Nrec] := temp;
  StatusBar1.Panels[0].Text := 'Всего: ' + IntToStr(Nrec);
  MasToGrid;
  StringGrid1.Row := Nrec;
  ShowMessage('Запись добавлена.');
  if CheckBox1.Checked then //автосохранение
   mnuSohranitClick(nil);
end;

procedure TForm1.editButtonClick(Sender: TObject);
//редактировать запись
```

```
var
  temp: TBook;//информация о книге
  r: integer;//номер выбранной записи
begin
  r := StringGrid1.Row;
  if (r < 1) or (r > Nrec) then
    Exit;
  temp := Books[r];
  Form2.Edit1.Text := IntToStr(temp.invn);
  Form2.Edit2.Text := temp.avtor;
  Form2.Edit3.Text := temp.nazv;
  Form2.Edit4.Text := IntToStr(temp.stran);
  if Form2.ShowModal <> mrOK then
    Exit;
  if (Form2.Edit1.Text = '') or (Form2.Edit2.Text = '') or
     (Form2.Edit3.Text = '') or (Form2.Edit4.Text = '') then
       begin
         ShowMessage('Необходимо заполнить все поля!');
         Exit;
       end;
  temp.invn := StrToInt(Form2.Edit1.Text);
  temp.avtor := Form2.Edit2.Text;
  temp.nazv := Form2.Edit3.Text;
  temp.stran := StrToInt(Form2.Edit4.Text);
  Books[r] := temp;
  MasToGrid;
  StringGrid1.Row := r;
  ShowMessage('Данные изменены.');
  if CheckBox1.Checked then //автосохранение
    mnuSohranitClick(nil);
end;

procedure TForm1.deleteButtonClick(Sender: TObject);
//удалить запись из таблицы
var
  i,//счетчик
  r: integer;//номер выбранной записи
begin
  r := StringGrid1.Row; //номер выбранной записи
  if r < 1 then Exit;;//если запись не выбрана, выход
  if Nrec = 0 then Exit;//записей нет, выход
  if Application.MessageBox('Удалить запись?', 'Предупреждение', MB_YESNO) <> idYes then
    Exit;
  for i := r to Nrec - 1 do//удаляем запись
    Books[i] := Books[i + 1];
  Dec(Nrec);//уменьшаем число записей
  MasToGrid;
  StatusBar1.Panels[0].Text := 'Всего: ' + IntToStr(Nrec);
  StringGrid1.Row := 1;
  ShowMessage('Запись удалена.');
  if CheckBox1.Checked then //автосохранение
    mnuSohranitClick(nil);
end;

procedure TForm1.CheckBox1Click(Sender: TObject);
//включение / выключение режима автосохранения
begin
  if CheckBox1.Checked then
    CheckBox1.Font.Color := clGreen
```

```
  else
    CheckBox1.Font.Color := clRed;
end;

procedure TForm1.mnuSortClick(Sender: TObject);
//сортировка записей
var
  i, k,//счетчик цикла
  field: integer;//номер поля сортировки
  crit: boolean;//критерий сортировки
  temp: TBook;//информация
begin
  field := (Sender as TMenuItem).Tag;
  for k := Nrec downto 2 do //сортировка методом пузырька
    for i := 1 to k - 1 do
      begin
        case field of
          2: crit := Books[i].avtor > Books[i + 1].avtor;
          3: crit := Books[i].nazv > Books[i + 1].nazv;
          4: crit := Books[i].stran > Books[i + 1].stran;
          else
            crit := Books[i].invn > Books[i + 1].invn;
        end;
        if crit then
          begin
            temp := Books[i];
            Books[i] := Books[i + 1];
            Books[i + 1] := temp;
          end;
      end;
  MasToGrid;
  StringGrid1.Row := 1;
end;
procedure TForm1.findButtonClick(Sender: TObject);
//поиск записей
var
  n: integer;//количество найденных записей
  i: integer;//счетчик цикла
  crit1, crit2: boolean;
  temp: TBook;//информация
begin
  if Nrec = 0 then Exit;
  Form2.Edit1.Text := '0';
  Form2.Edit2.Text := '';
  Form2.Edit3.Text := '';
  Form2.Edit4.Text := '0';
  if Form2.ShowModal <> mrOK then
    Exit;
  n := 0;
  temp.invn := 0;
  temp.avtor := Form2.Edit2.Text;
  temp.nazv := Form2.Edit3.Text;
  temp.stran := StrToInt(Form2.Edit4.Text);
  for i := 1 to Nrec do
    begin
      if temp.avtor = '' then
        crit1 := true
      else
        crit1 := Pos(AnsiUpperCase(temp.avtor), AnsiUpperCase(Books[i].avtor)) <> 0;
```

```
    if temp.nazv = '' then
      crit2 := true
    else
      crit2 := Pos(AnsiUpperCase(temp.nazv), AnsiUpperCase(Books[i].nazv)) <> 0;
    if crit1 and crit2 then
      begin
        n := n + 1;
        Form3.StringGrid1.RowCount := n + 1;
        ToGrid(n, Books[i], Form3.StringGrid1);
      end;
  end;
 if n = 0 then
   ShowMessage('Книги не найдены.')
 else
   Form3.ShowModal;
end;

procedure TForm1.mnuExportClick(Sender: TObject);
//экспорт базы данных в Excel
var
 MsExcel: Variant;//указатель на Excel
 i: integer;//счетчик цикла
begin
 try    // Если Excel уже запущен
   MsExcel := GetActiveOleObject('Excel.Application');
  // Взять ссылку на запущенный OLE объект
 except
   try  //Excel не запущен, запустить
     MsExcel := CreateOleObject('Excel.Application');
     // Создать ссылку на зарегистрированный OLE объект
     MsExcel.Visible := True;
   except
      ShowMessage('Не могу запустить Microsoft Excel!');
      Exit;
   end;
 end;
 MsExcel.Workbooks.Add;
 MsExcel.Worksheets[1].Columns[1].ColumnWidth := 6;
 MsExcel.Worksheets[1].Columns[2].ColumnWidth := 35;
 MsExcel.Worksheets[1].Columns[3].ColumnWidth := 80;
 MsExcel.Worksheets[1].Columns[4].ColumnWidth := 14;

 MsExcel.Worksheets[1].Cells[1, 1].Value := '№';
 MsExcel.Worksheets[1].Cells[1, 2].Value := 'Автор';
 MsExcel.Worksheets[1].Cells[1, 3].Value := 'Название произведения';
 MsExcel.Worksheets[1].Cells[1, 4].Value := 'Число страниц';
 for i := 1 to Nrec do
   begin
     MsExcel.Worksheets[1].Cells[i + 1, 1].Value := Books[i].invn;
     MsExcel.Worksheets[1].Cells[i + 1, 2].Value := Books[i].avtor;
     MsExcel.Worksheets[1].Cells[i + 1, 3].Value := Books[i].nazv;
     MsExcel.Worksheets[1].Cells[i + 1, 4].Value := Books[i].stran;
   end;
end;

end.
```