X областной конкурс работ исследовательского характера (конференция) учащихся по учебным предметам «Хрустальная Альфа»

Секция «Информатика»

АВТОМАТИЗИРОВАННЫЙ ПОДБОР УЧАСТНИКОВ И УЧЁТ РЕЗУЛЬТАТОВ ФИЗИЧЕСКОЙ ПОДГОТОВЛЕННОСТИ УЧАЩИХСЯ ПО ЛЕГКОЙ АТЛЕТИКЕ

Автор работы:
Дервис Максим Владимирович,
8 «Б» класс,
Государственное учреждение образования «Средняя школа № 3 г. Ошмяны»,
Тел. +375291873435

Руководитель работы:
Толочко Екатерина Ивановна,
учитель информатики,
Государственное учреждение образования «Средняя школа № 3 г. Ошмяны»

2018

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Один из основных и наиболее массовых видов спорта является легкая атлетика. Систематические занятия легкоатлетическими упражнениями развивают силу, быстроту, выносливость, гибкость и ловкость, т.е. качества, необходимые каждому человеку в повседневной жизни. Повседневные занятия физической культурой положительно влияет на сердечно-сосудистую, мышечную и нервную систему, оберегая хрупкий подростковый организм от стрессов и депрессий, восстанавливая сердечный ритм и поддерживая мышцы в тонусе. Некоторые ученики наивно полагают, что лёгкая атлетика оказывает наименьшую нагрузку на мышцы и организм в целом. Но, только начав заниматься этим спортом, очень многие признают, что они глубоко ошибались.

Первостепенной задачей для меня являлось изучить влияние лёгкой атлетики на физическую подготовку учащихся. Но так как я увлекаюсь языками программирования, я решил разработать программу, которая сможет хранить, отслеживать результаты физической подготовленности учащихся по легкой атлетике.

Цель: разработать программу для хранения и использования результатов физической подготовленности учащихся по лёгкой атлетике.

Разработка программы началась на языке высокого уровня Delphi.

Delphi – среда разработки программ на Object Pascal. Слово Delphi – это название города в древней Греции, в котором пророчествовали оракулы. Такое название было выбрано разработчиками Delphi для того, чтобы подчеркнуть способность программ, создаваемых в Delphi, взаимодействовать с базами данных Oracle. Система Delphi создана в полном согласии с концепцией RAD (Rapid Application Development – быстрая разработка приложений), ее использование значительно повышает скорость разработки приложений для Windows.

Базы данных считаются основным достоинством Delphi. Практически любую задачу в этой предметной области можно реализовать средствами этого языка, причём за довольно короткий промежуток времени. Главное здесь то, что реализация приложения очень удобна и проста в понимании.

Базы данных обеспечивают быстрый доступ к информации и дают полное представление её. Поэтому тема данного проекта “Автоматизированный подбор участников и учёт результатов соревнований по легкой атлетике” является очень актуальной, а разработка её в объектно-ориентированной среде Delphi после создания базы данных в MS Access предоставляет пользователю удобный интерфейс доступа к данным и использования их.

Мое исследование пропагандирует здоровый образ жизни, занятия спортом и физической культурой. Все мы осознаем их важность в связи с настоящим положением спорта в мире. Мы стремительно теряем здоровых и активных молодых людей, для которых спорт – не просто занятия, а получение удовольствия от них.

Для себя я выбрал следующие методы исследования: анкетирование, анализ результатов, наблюдение и разработка базы данных.

1 ОБЬЕКТНО-ОРИЕНТИРОВАННЫЙ АНАЛИЗ И ПРОЕКТИРОВАНИЕ СИСТЕМЫ

1.1 Описание предметной области

В данной работе необходимо реализовать в виде базы данных учёт результатов соревнований по легкой атлетике. Приложение должно обеспечивать:

1. Корректировку базы данных (добавление новых спортсменов, удаление выбывших спортсменов, редактирование любого поля).
2. Поиск по критериям: по возрасту и по фамилии спортсмена.
3. Подбор всех возможных противников по возрасту для конкретного спортсмена.

1.2 Инструменты разработки

В данном проекте приложение реализуется на языке Delphi.

Наряду с традиционными инструментами доступа к данным Borland Database Engine и ODBC в приложениях Delphi можно применять технологию Microsoft ActiveX Data Objects (ADO), которая основана на возможностях COM, а именно интерфейсов OLE DB.

Технология ADO завоевала популярность у разработчиков, благодаря универсальности - базовый набор интерфейсов OLE DB имеется в каждой современной операционной системе Microsoft. Поэтому для обеспечения доступа приложения к данным достаточно лишь правильно указать провайдер соединения ADO и затем переносить программу на любой компьютер, где имеется требуемая база данных и, конечно, установленная ADO.

Технология Microsoft ActiveX Data Objects обеспечивает универсальный доступ к источникам данных из приложений БД. Такую возможность предоставляют функции набора интерфейсов, созданные на основе общей модели объектов COM и описанные в спецификации OLE DB.

Технология ADO и интерфейсы OLE DB обеспечивают для приложений единый способ доступа к источникам данных различных типов. Например, приложение, использующее ADO, может применять одинаково сложные операции и к данным, хранящимся на корпоративном сервере SQL, и к электронным таблицам, и локальным СУБД. Запрос SQL, направленный любому источнику данных через ADO, будет выполнен.

Технология ADO в целом включает в себя не только сами объекты OLE DB, но и механизмы, обеспечивающие взаимодействие объектов с данными и приложениями. На этом уровне важнейшую роль играют провайдеры ADO, координирующие работу приложений с хранилищами данных различных типов.

Механизм доступа к данным через ADO и многочисленные объекты и интерфейсы реализованы в VCL Delphi в виде набора компонентов, расположенных на странице ADO [1].

В данной работе использовались ADOConnection, ADOTable, ADOQuery.

2 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

2.1 Взаимосвязь модулей

Используемые в программе модули описываются в двух местах в разделе описания главной формы. Перечни модулей отделены зарезервированным словом Uses. То есть:

Uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs, StdCtrls, Buttons, ExtCtrls;
и
Uses
DataModuleUnit, EditRecordUnit, FindUnit, PodborSopernikaUnit, ObAwtoreUnit, SorewnowaniyaUnit;

В первом случае список представляет собой перечень системных модулей, содержащих в себе принципы работы компонентов, основные библиотеки и системные интерфейсы для работы приложения.

Опишем их:

Windows – модуль отвечающий за связь программы со средой ОС Windows;

Messages – вывод различных сообщений на экран;

SysUtils – работа с системными данными и утилитами. Поддержка драйверов;

Graphics – работа с графикой;

Dialogs – включение в программу диалогов и принципа их работы на основе ОС Windows;

Menus – включение в программу меню. То есть программное меню будет таким же как стандартное меню ОС [3].

По записанным в этом списке модулям, программа определяет те компоненты, с которыми ей придется работать и организовывать с ними работу.

Второй перечень модулей предназначен для сообщения программе дополнительных форм, на которые возможно перейти (то есть на них имеются ссылки в данной программе) с этой формы. Если данный перечень отсутствует, это говорит о том, что вся программа выполнена на одной форме или другие формы просто нигде не оговорены (на событиях нажатий на кнопки или пункты меню).

В программу входят 6 юнитов, а следовательно и 6 форм. Опишем их:

SorewnowaniyaUnit –соответствует главной форме программы MainForm (см. рисунок А.1);

DataModuleUnit – соответствует форме DataModule1, на которой хранятся компоненты работы с базой данных db1 в Access (см. рисунок А.2);

FindUnit – соответствует форме FindForm ввода параметра для поиска (см. рисунок А.3);

PodborSopernikaUnit – соответствует форме SopernikForm ввода веса спортсмена для подбора ему противника (см. рисунок А.4);

EditRecordUnit – соответствует форме EditRecordForm - добавления/редактирования данных для главной формы (см. рисунок А.5);

ObAwtoreUnit – соответствует форме ObAwtoreForm - информации об авторе программы (см. рисунок А.6).

Таким образом каждый модуль соответствует какому-либо этапу выполнения работы данной программы.

2.2 Процедуры и их действие

Как уже было сказано выше, вся программа состоит из последовательно записанных процедур на каждое заранее продуманное действие, которое пользователь может предпринять в отношении какого либо компонента. Таким образом, наиболее часто встречаемая процедура в данной программе – это процедура, обрабатывающая нажатие на кнопку. Ее заголовок выглядит следующим образом:

```
Procedure TForm1.Button1Click (Sender: TObject);   //объявление процедуры.
Var                 // раздел описания переменных.
begin
…                  //  раздел операторов.
end;
```

При нажатии на большинство кнопок, используемых в программе, будет возникать событие OnClick и соответствующая данной кнопке процедура Button.Click будет выполнять операторы, записанные между операторными скобками begin и end.

Опишем другие процедуры, выполняемые в результате разных событий.

Немаловажной является процедура подбора противника по весу для конкретного спортсмена. Данная процедура создаёт условия для более расширенного поиска противника, что является более удобным для пользователя. При этом в процедуре происходит анализ, какие критерии выбраны/введены, после чего записывается уже сам SQL запрос в компонент ADOQuery и активируется. Код процедуры приведён ниже:

```
/*процедура подбора противника по весу для конкретного спортсмена */
procedure TSopernikForm.Button1Click(Sender: TObject);
var k:integer;
begin
if Edit1.Text<>''  then
begin
k:=StrToInt(Edit1.Text);
```

```
/*выбор весовой категории спортсмена */
if k<60 then
begin
DataModule1.ADOQuery1.Active:=False; //сброс запроса
DataModule1.ADOQuery1.SQL.Clear; //очистка предыдущего запроса
/*добавление SQL запроса */
DataModule1.ADOQuery1.SQL.Add('Select * ');
DataModule1.ADOQuery1.SQL.Add('From Спортсмены');
DataModule1.ADOQuery1.SQL.Add('Where Wes < :To');
/*присваивание параметру компонента ADOQuery категории поиска*/
DataModule1.ADOQuery1.Parameters.ParamByName('To').Value:=60;
DataModule1.ADOQuery1.Active:=true; //активация запроса
end
else
if (k>=60) and (k<70) then
begin
DataModule1.ADOQuery1.Active:=False;
DataModule1.ADOQuery1.SQL.Clear;
DataModule1.ADOQuery1.SQL.Add('Select * ');
DataModule1.ADOQuery1.SQL.Add('From Спортсмены');
DataModule1.ADOQuery1.SQL.Add('Where Wes >= :From and Wes < :To');
DataModule1.ADOQuery1.Parameters.ParamByName('From').Value:=60;
DataModule1.ADOQuery1.Parameters.ParamByName('To').Value:=70;
DataModule1.ADOQuery1.Active:=true;
end
else
if (k>=70) and (k<80) then
begin
DataModule1.ADOQuery1.Active:=False;
DataModule1.ADOQuery1.SQL.Clear;
DataModule1.ADOQuery1.SQL.Add('Select * ');
DataModule1.ADOQuery1.SQL.Add('From Спортсмены');
DataModule1.ADOQuery1.SQL.Add('Where Wes >= :From and Wes < :To');
DataModule1.ADOQuery1.Parameters.ParamByName('From').Value:=70;
DataModule1.ADOQuery1.Parameters.ParamByName('To').Value:=80;
DataModule1.ADOQuery1.Active:=true;
end
else
if (k>=80) and (k<90) then  begin
DataModule1.ADOQuery1.Active:=False;
DataModule1.ADOQuery1.SQL.Clear;
DataModule1.ADOQuery1.SQL.Add('Select * ');
DataModule1.ADOQuery1.SQL.Add('From Спортсмены');
```

```
DataModule1.ADOQuery1.SQL.Add('Where Wes >= :From AND Wes < :To');
DataModule1.ADOQuery1.Parameters.ParamByName('From').Value:=80;
DataModule1.ADOQuery1.Parameters.ParamByName('To').Value:=90;
DataModule1.ADOQuery1.Active:=true;
end
else
if (k>=90) and (k<100) then  begin
DataModule1.ADOQuery1.Active:=False;
DataModule1.ADOQuery1.SQL.Clear;
DataModule1.ADOQuery1.SQL.Add('Select * ');
DataModule1.ADOQuery1.SQL.Add('From Спортсмены');
DataModule1.ADOQuery1.SQL.Add('Where Wes >= :From and Wes < :To');
DataModule1.ADOQuery1.Parameters.ParamByName('From').Value:=90;
DataModule1.ADOQuery1.Parameters.ParamByName('To').Value:=100;
DataModule1.ADOQuery1.Active:=true;
end
else
if k>=100 then  begin
DataModule1.ADOQuery1.Active:=False;
DataModule1.ADOQuery1.SQL.Clear;
DataModule1.ADOQuery1.SQL.Add('Select * ');
DataModule1.ADOQuery1.SQL.Add('From Спортсмены');
DataModule1.ADOQuery1.SQL.Add('Where Wes >= :From ');
DataModule1.ADOQuery1.Parameters.ParamByName('From').Value:=100;
DataModule1.ADOQuery1.Active:=true;
end;
if DataModule1.ADOQuery1.RecordCount<>0  then
begin
Edit1.Clear;
Edit1.SetFocus;
Close;
MainForm.Label1.Visible:=false;
MainForm.DBGrid1.Visible:=false;
MainForm.N1.Visible:=false;
MainForm.N4.Visible:=false;
MainForm.N9.Visible:=false;
MainForm.N15.Visible:=false;
MainForm.Button1.Visible:=false;
MainForm.Button2.Visible:=true;
MainForm.DBGrid2.Visible:=true;
MainForm.Label2.Visible:=true;
end
else  begin
```

```
ShowMessage('По вашему запросу ничего не найдено!');
Close;  end
end
else  begin
/*сообщение, что не введён вес*/
ShowMessage('Введите ваш вес!');
Edit1.SetFocus;  end
end;
```

2.3 Классы

Класс — определенный пользователем тип данных, который обладает внутренними данными и методами в форме процедур или функций и обычно описывает родовые признаки и способы поведения ряда похожих объектов, он включает в себя данные различных типов и операции над ними.

Все классы в Delphi являются прямыми или косвенными наследниками TObject. Прямое наследование используется только при объявлении простых классов, объекты которых не являются компонентами, не могут присваиваться друг другу и не участвуют в операциях обмена с потомками. Подавляющее большинство классов являются косвенными наследниками TObject и производятся от промежуточных классов. Если при объявлении нового типа объектов не указывается класс предок, то Delphi считает TObject предком нового класса.

Класс TComponent является предком всех компонентов VCL. Используется в качестве основы для создания невидимых компонентов и реализует основные механизмы, которые обеспечивают функционирование любого компонента.

Обычно разработчик не задумывается о том, как объект будет создан, и что необходимо сделать для его корректного уничтожения. Компоненты VCL создаются и освобождают занимаемые ресурсы автоматически. Иногда разработчику приходится создавать и удалять объекты самостоятельно. Класс TComponent служит основой для всех компонентов Delphi. Этот класс задает базовое поведение всех компонентов - их основные свойства и методы. К ним относятся:

- возможность отображения компонента в палитре компонентов и управления им в дизайнере форм;
- возможность выступать контейнером для других компонентов;
- возможность выступать в качестве оболочки вокруг компонентов ActiveX и других объектов, реализующих интерфейсы.

Класс TComponent является базовым для создания невизуальных компонентов, которые могут располагаться в палитре компонентов и использоваться в дизайнере форм.

Класс TPersistent происходит непосредственно от класса TObject. Он обеспечивает своих потомков возможностью взаимодействовать с другими объектами и процессами на уровне данных. Его методы позволяют передавать

данные в потоки, а также обеспечивают взаимодействие объекта с инспектором объектов.

Для создания визуальных компонентов базовым является TControl, а для создания компонентов, имеющих окна, - класс TWinControl. Визуальные компоненты являются элементами управления. Элементы управления - это варианты стандартных элементов управления Windows. Примером элемента управления является "кнопка". Пользователь может произвести некоторое действие, выполнив щелчок на этой кнопке. Компонент "Меню" видим и доступен для редактирования только разработчику приложения. Для него этот компонент является инструментом, используемым для создания меню. Во время работы программы пользователь видит только результат работы "Меню", а не сам элемент.

При разработке приложений в Delphi программист имеет возможность использовать несколько механизмов, обеспечивающих обработку исключительных ситуаций. Это и специальные операторы языка Object Pascal и классы, предназначенные для программирования реакции на ошибки.

Исключительная ситуация - это нештатное событие, которое может повлиять на дальнейшее выполнение программы. Современный компилятор выдает код, который перехватывает любое такое событие, предотвращает нежелательные последствия, сохраняет необходимые данные о состоянии программы. С точки зрения Object Pascal исключительная ситуация - это объект.

Для обработки исключительных ситуаций предназначен класс Exception.

Тип Exception порождает многочисленные дочерние типы, соответствующие часто встречающимся случаям ошибок ввода/вывода, распределения памяти и т. п. Exception и его потомки представляют собой исключение из правила, предписывающего все объектные типы именовать с буквы "T". Имена потомков типа Exception начинаются с "E".

При создании программы были использованы следующие компоненты:

TLabel - элемент этого типа обычно используется, когда необходимо отобразить текст, который не может быть отредактирован пользователем, например заголовки компонентов, которые не имеют собственного свойства Caption;

TEdit - строка редактирования - прямоугольное окно, в котором возможен ввод и редактирование текста. С помощью компонента TEdit можно отображать и нередактируемый текст;

TImage – контейнер для загрузки изображений;

TButton – выполняет функции кнопки;

TADOQuery – обеспечивает доступ к базе данных с помощью языка SQL;

TADOTable – возвращает набор данных одной или нескольких таблиц базы данных;

TADOConnection – используется для соединения с хранилищами данных ADO:

DataSource – используется для связи компонентов;

DBGrid – таблица, через которую можно отображать таблицы базы данных [4].

3 ПРАКТИЧЕСКОЕ ПРИМЕНЕНИЕ

3.1 Области применения

Данная программа может найти применение в спортивных клубах для учёта результатов спортивных достижений спортсменов, в школе (учителями физической культуры и здоровья), а также на соревнованиях.

Программа существенно сэкономит время пользователей на добавление и удаление данных. Удобный поиск позволит легко и быстро найти необходимых спортсменов по корректно введённым данным.

3.2 Система справочной информации

Система справочной информации для проекта написана с помощью текстового редактора Блокнот, позволяющего создавать HTML-страницы (с расширением .htm).

Документ HT ML - это обычный текстовый файл, окончательный вид документа зависит от последовательностей символов, вставленных в текст. Эти последовательности называются HTML-тэгами. Преобразование справочной системы в файл справки с расширением .chm было осуществлено при помощи программы HTML Help Workshop.

Система справки выполнена в форме руководства пользователю со скриншотами программы и помогает быстро и легко найти интересующие сведения о работе данной программы.

Вызов справки в приложении осуществляется при помощи соответствующего раздела в меню.

ЗАКЛЮЧЕНИЕ

В результате выполнения поставленной задачи создан программный продукт, который содержит информацию по результатам физической подготовленности учащихся по легкой атлетике.

Программа проста и легка в использовании, не требует много ресурсов.

В разработанной программе размещена вся необходимая документация для учителей физической культуры (Нормативы уровня физической подготовленности, ИНСТРУКЦИЯ о порядке и условиях выдачи нагрудного значка лицам, выполнившим нормативы, предусмотренные Государственным физкультурно-оздоровительным комплексом Республики Беларусь, Программа физкультурно-спортивных многоборий).

Программа имеет функцию вывода на экран окна помощи для пользователей.

Для написания программы была использована интегрированная среда разработки приложений Delphi 7. Так же при написании справочной системы использовался язык HTML и для работы с базами данных язык SQL.

В процессе выполнения данной работы были получены теоретические и практические навыки написания Windows-приложений на языке Object Pascal, а также подробно изучены принципы работы с базами данных.

Важно отметить, что программа может содержать большое количество записей, что позволит учителям физической культуры хранить информацию о спортсменах легкоатлетах на протяжении нескольких лет.

Данная программа может найти применение в спортивных клубах для учёта спортсменов, в школе (учителями физической культуры и здоровья), а также на соревнованиях.

ЛИТЕРАТУРА

1 Митчелл К. Керман – Программирование и отладка в среде Delphi – учебный курс, Москва – Санкт Петербург – Киев, 2002.

2 Дарахвелидзе П.Г., Марков Е.П. Delphi – среда визуального программирования: - СПб.: BHV – Санкт-Петербург,1996.

3 С.Бобровский – Delphi – учебный курс, СПб: Издательство “Питер”,2000.

ПРИЛОЖЕНИЕ А

Программные формы

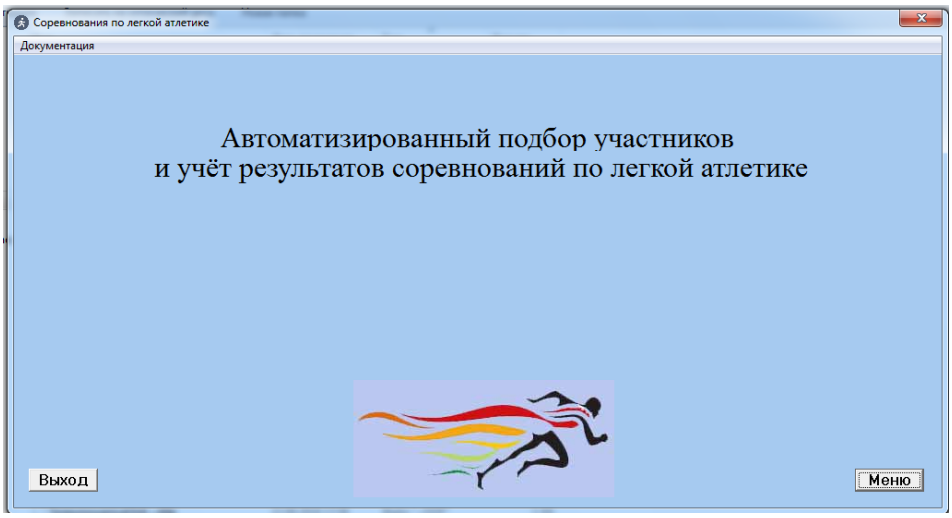

Рисунок 1 – Окно главной формы программы

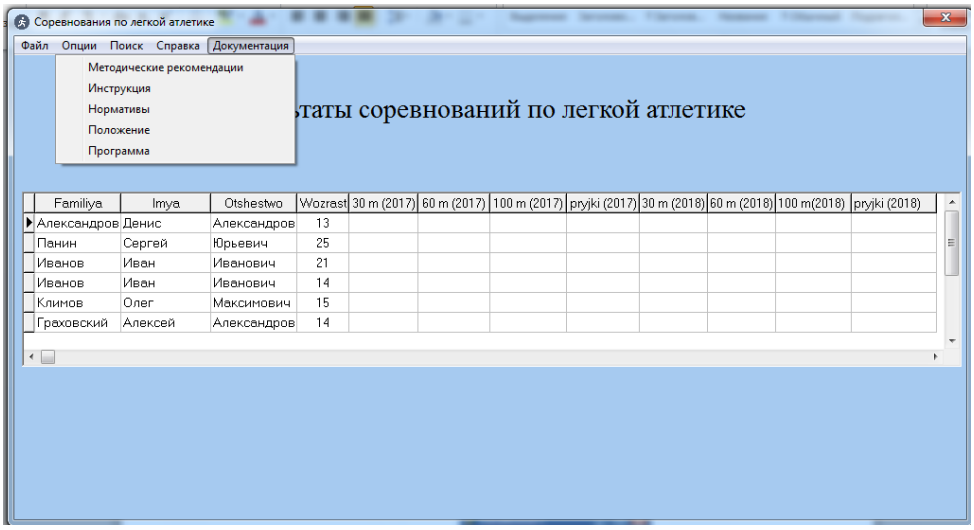

Рисунок 2 – Окно формы главного меню программы

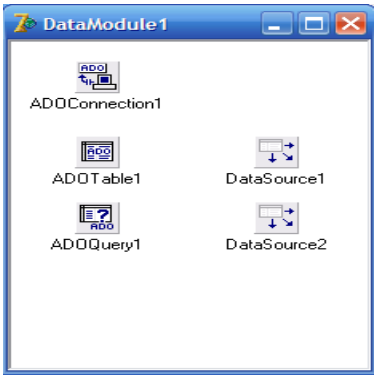

Рисунок 3 – Окно формы компонентов доступа к базе данных

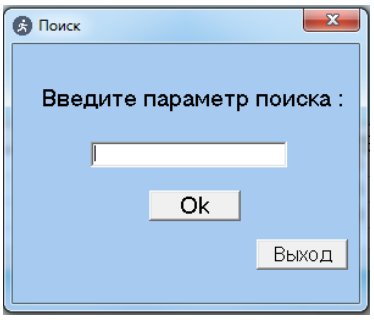

Рисунок 4 – Окно формы для ввода параметра поиска

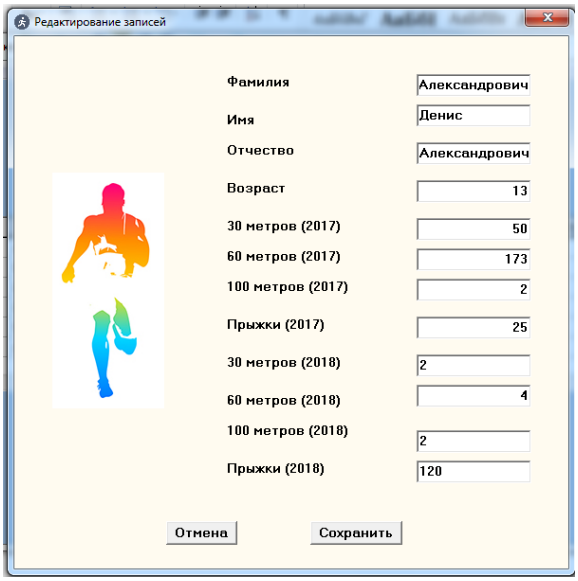

Рисунок 5 - Окно формы для редактирования записей

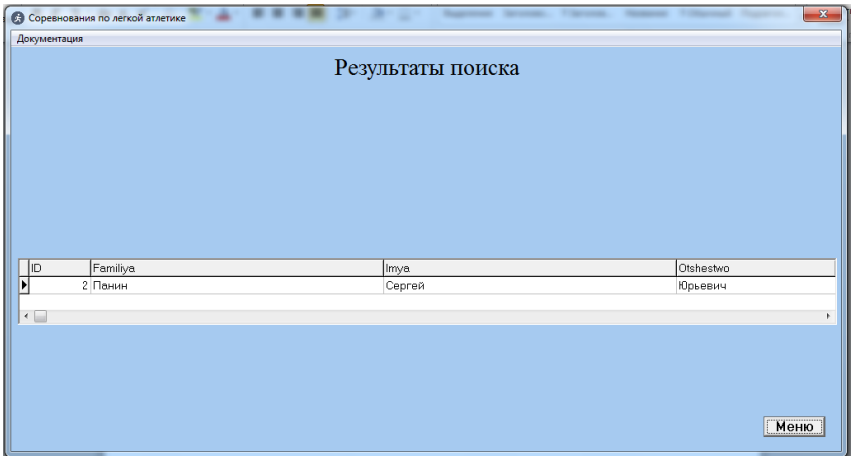

Рисунок 6 – Окно отображения результатов поиска