

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Забайкальский государственный университет»
(ФГБОУ ВО «ЗабГУ»)

Факультет: Энергетический факультет

Кафедра: Информатики и вычислительной техники

Курсовая работа

по Программированию

На тему: База данных расходуемых материалов на стройке

Выполнил ст. гр. ИВТ-19-2

Михалева Е.Д.

Проверил Доцент кафедры ИВТ и ПМ

Соловьёв В.А.

Чита

2020

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«Забайкальский государственный университет»
(ФГБОУ ВО «ЗабГУ»)

Факультет: Энергетический факультет

Кафедра: Информатики и вычислительной техники

ЗАДАНИЕ
для курсовой работы

По дисциплине: Программирование

Студенту Михалевой Е.Д. специальности

Информатика и Вычислительная техника

1 Тема курсовой работы: создание базы данных нарушений ПДД
пешеходами

2 Срок подачи студентом законченной работы: 1.06.2020

3 Исходные данные к работе: база данных, содержащая перечень
расходуемых вариантов на стройке. В перечень входят: наименование
материала, количество, Ф.И.О. получившего, Ф.И.О. отпустившего. В
конце дня подводится итог.

4 Перечень подлежащих разработке в курсовой работе вопросов:

Программа должна предоставлять возможность просматривать, добавлять,
удалять, копировать, хранить данные. В программе использовать модули,
функции, процедуры, записи, списки и файлы.

. Дата выдачи задания: 13.02.2020

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«Забайкальский государственный университет»

(ФГБОУ ВО «ЗабГУ»)

Факультет: Энергетический факультет

Кафедра: Информатики и вычислительной техники

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе (проекту)

по Программированию

на тему: База данных продажи товаров в магазине

Целью курсовой работы является разработка приложения, с минимальным интерфейсом, которое будет работать с базой данных для продажи товаров в магазинах. Также целью будет являться: Применение типов данных, изученных во время обучения, Использование процедур и функций, модулей и файлов, а также создание минимального интерфейса

Выполнил студент группы ИВТ-19-2 Михалева Е.Д.

Руководитель работы: Доцент кафедры ИВТ и ПМ Соловьёв В.А.

Календарный план

№	Наименование раздела курсовой работы	Неделя			
		1	2	3	4
1	Введение	+			
2	Глава 1		+		
3	Глава 2			+	
4	Глава 3			+	
5	Заключение				+
6	Литература, приложение				+

Оглавление

Введение	2
Глава 1. Типы данных и операции, реализуемые в курсовой работе [3]	4
1.1. Типы данных, используемые в курсовой работе.....	5
1.2. Операции, реализуемые в курсовой работе.....	6
Глава 2. Интерфейс приложения	11
2.1. Компоненты интерфейса, используемых в курсовой работе. Их описание в программе.....	11
2.2. Реализация обработчиков событий. [1],[3]	16
Глава 3. Тестирование созданного приложения, проверка полученных результатов	24
Заключение	28
Литература.....	29
Приложение	30

Введение

Актуальность данной темы заключается в том, что сложно уследить за расходом материала на стройках или других подобных стройке работ, потому что рабочих много и всех их запомнить сложно. Потому на помощь придут современные технологии и база данных, которая облегчит эту работу для тех, кто отвечает за материалы, необходимые на стройке.

База данных - представленная в объективной форме совокупность самостоятельных материалов, систематизированных таким образом, чтобы материалы могли быть найдены и обработаны с помощью Электронно-вычислительной машины (ЭВМ). [2]

Для студентов это является хорошей практикой для нашей работы в будущем. Уже при работе над данной темой мы научимся создавать свою базу данных, а также проработать всё, что касается типов данных, динамических линейных списков, а также работе с Delphi и созданию примитивного интерфейса, понятного каждому человеку, который будет работать с этой программой.

В данном приложении должны быть введены такие функции:

- Хранение информации.
- Создание базы данных.
- Внесение изменений в базу данных.
- Удаление информации из базы данных.
- Сортировка данных.
- Подведение итога дня.

Объектом исследования данной курсовой работы являются стройка, расчёт затраченных материалов на стройке и применение на практике знаний, полученных на лекциях и домашнем обучении, по теме базы данных, типы данных, списки.

Предметом исследования является двунаправленный линейный список, база данных и динамические типы данных в целом

Для выполнения этой курсовой работы были использованы такие источники информации:

Электронные ресурсы:

http://www.pascal.helpov.net/index/dynamic_lists_pascal

https://ru.wikipedia.org/wiki/База_данных

Книги:

А.Я. Архангельский. Язык Pascal и основа программирования в Delphi. Учебное пособие – М.: ООО«Бином-Пресс», 2004г.-496с.

Глава 1. Типы данных и операции, реализуемые в курсовой работе [3]

1.1. Типы данных, используемые в курсовой работе

В этой работе подразумевается использование списка записей, которые можно хранить в файле на ЭВМ. Поэтому самым первым мы создадим тип записи, содержащий в себе основные элементы базы данных, для выполнения задачи курсовой работы: Название материала, кол-во материала, ФИО принявшего материал, ФИО отпустившего материал

Чтобы список существовал, нужно будет создать ещё один тип данных – запись, содержащая в себе адрес следующего узла списка, адрес предыдущего узла списка и данные, хранящиеся в каждом узле списка.

Следующим типом данных будет тип указатель на запись, создающую список.

И последним типом данных, который мы опишем в секции type, будет тип файла, способный хранить в себе базу данных.

Секция type:

Type

```
Building = record //Тип записи для нашей задачи, который  
будет храниться в базе данных
```

```
    Material:string[30]; //название материала
```

```
    KolMat:integer; //Количество данного материала
```

```
    FIOPrin:string[30]; //ФИО принявшего материал
```

```
    FIOOtp:string[30]; //ФИО отпустившего материал
```

```
end;
```


`PUzel = ^Z1; // Указатель на тип данных, создающий список`

`Z1 = record // Запись, реализующая список`

`x: Building; // Информация, хранящаяся в узлах списка`

`next: puzel; // Указатель на следующий узел`

`pred: puzel; // Указатель на предыдущий узел`

`end;`

`Fzap = file of Building; // Типизированный файл, для хранения
базы данных`

1.2. Операции, реализуемые в курсовой работе

В задании для этой курсовой работы требуется создать несколько операций, чтобы работать с базой данных, которую мы создадим. Это операции:

- Сохранение списка в файл
- Открывание списка из файла
- Построение списка
- Дополнение списка
- Сортировка списка
- Просмотр списка
- Удаление списка
- Подведение итога дня на стройке

Список операций для данной курсовой работы:

- *Сохранение списка в файл* необходимо для того, чтобы сохранить базу данных на носителе данных, чтобы в дальнейшем его можно было открыть и использовать. Для такого мы будем использовать типизированный файл. Для сохранения результатов будем использовать текстовый файл, так как его можно без проблем прочитать в любом блокноте.
- *Загрузка списка из файла*, нужна как раз таки, чтобы открыть файл, сохранённый ранее, и взять оттуда список для дальнейшего использования.
- *Построение списка* нужно для того, чтобы построить список и начать делать свою базу данных.
- *Дополнение списка* будет использоваться, чтобы ввести новые данные в уже существующий список.
- *Просмотр списка* нужен, чтобы просмотреть содержимое списка.
- *Сортировка* нужна, чтобы сделать список упорядоченным и облегчить поиск по базе данных необходимых элементов. Сортировка будет проводиться по 4 признакам: Наименование материала, кол-во товара(По возрастанию), ФИО принявшего, ФИО отпустившего.
- *Удаление списка* будет удалять базу данных, если она неактуальна.
- *Подведение итогов дня на стройке* – это заключительная операция в данной работе, необходима для того, чтобы подсчитать общее кол-во отпущенного материала за день.

Операции в созданном модуле:

procedure BuildSpisok(var f: PUzel); //Процедура, для построения списка

procedure AddFirst(var f: PUzel; a: PUzel);//Вставить узел а
первым в список

procedure AddAfter(var old:PUzel; a: PUzel);//Вставить узел а
после old

procedure WriteSpTip(var f: PUzel; var ftip:Fzap);//Записать
данные списка в типизированный файл

procedure WriteSpText(var f: PUzel; var ftxt:Text); //Записать в
текстовый файл

procedure BuildSpisokFromTip(var f:puzel;var ftip:Fzap);
//Построить список из данных, взятых из тип. файла

procedure DelFirstElement(var f,a: PUzel);//Выделить первый
элемент списка

procedure DelElement(var old,a: PUzel);//Выделить элемент из
списка, следующий за old

procedure ItogDay(var f:puzel;var ftxt:text);//Процедура для
выполнения задачи

procedure DobVSp(var f:puzel);//Добавляет в список новые
элементы, не удаляя старые

procedure DelSpisok(var f: PUzel);//Удалить список

procedure PoslElem(f:puzel;var a:puzel);//Выбирает последний
элемент в списке

procedure Sort(f:puzel;var b:puzel);//Сортирует данный список

Примеры использования некоторых операций:

Итог дня:

Допустим, у нас имеется список состоящий из:

Кирпич отпущенный строителем по ФИО: Иванов И.И.

в количестве: 50 шт. был принят строителем по ФИО: Сидоров С.С.

Шифер отпущенный строителем по ФИО: Иванов И.И.

в количестве: 25 шт. был принят строителем по ФИО: Петров К.Н.

Кирпич отпущенный строителем по ФИО: Иванов И.И.

в количестве: 30 шт. был принят строителем по ФИО: Николаев И.С.

Тогда, результатом данной операции будет:

Итого за день были отпущены такие материалы:

Кирпич: 80 шт. Шифер: 25 шт.

Сортировка:

Имеется база данных, состоящая из нескольких элементов:

	ФИО отпустившего	Название материала	Кол-во материала	ФИО принявшего
1	Сидоров С.С.	Кирпич	30шт.	Иванов И.И.
2	Сидоров С.С.	Кирпич	60шт.	Говорайло
3	Сидоров С.С.	Шифер	25шт.	Николаев Н.Н.

Тогда при использовании сортировки, к примеру, по ФИО принявшего, мы получим вот такой упорядоченный список:

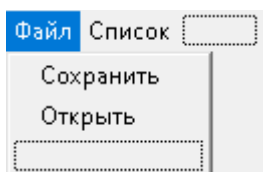
	ФИО отпустившего	Название материала	Кол-во материала	ФИО принявшего
1	Сидоров С.С.	Кирпич	30шт.	Говорайло
2	Сидоров С.С.	Кирпич	60шт.	Иванов И.И.
3	Сидоров С.С.	Шифер	25шт.	Николаев Н.Н.

Глава 2. Интерфейс приложения

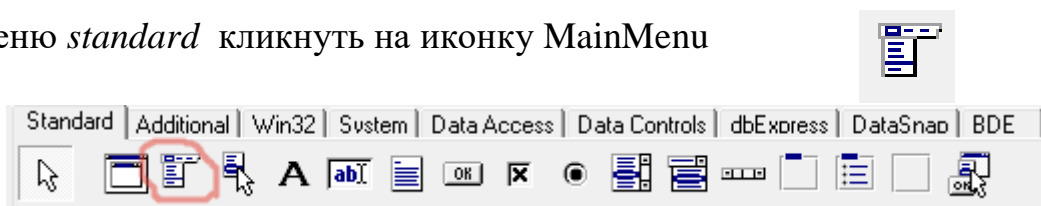
В данной работе не требуется создавать какой-то яркий и привлекающий дизайн, интерфейс должен быть простым и лёгким в понимании, чтобы любой мог разобраться в нём. Также он должен быть компактным и не иметь множество различных кнопок и т.д. В этом нам помогут компоненты интерфейса Delphi: TMainMenu, TStringGreed, SaveDialog, OpenDialog.

2.1. Компоненты интерфейса, используемых в курсовой работе. Их описание в программе.

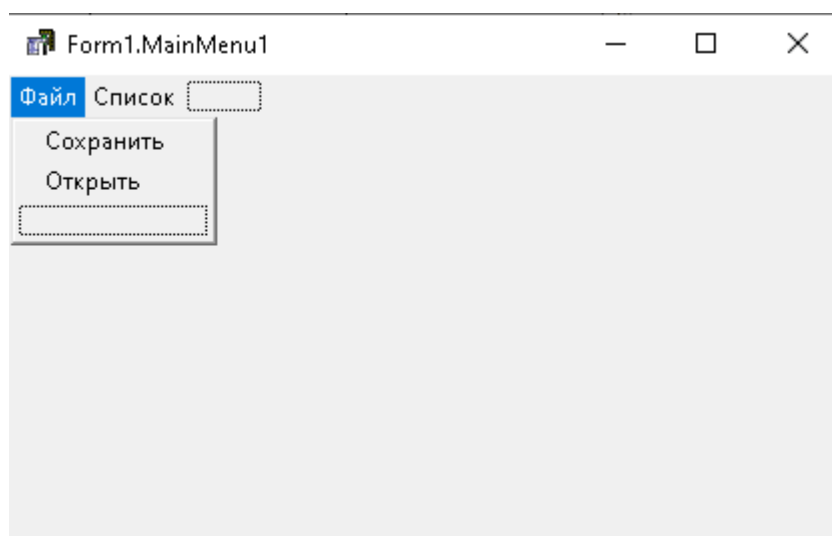
Для того, чтобы не загромождать всю форму кнопками, надписями, картинками и т.д., мы будем использовать удобный компонент интерфейса Delphi – MainMenu. Он представляет из себя небольшую полоску наверху формы программы, которая содержит в себе разные меню. Меню в свою очередь состоят из подменю, таким образом, представляя структурированную область для создания множества обработчиков событий, не занимающих много места.



Чтобы его добавить на форму, нужно в меню компонентов в подменю *standard* кликнуть на иконку MainMenu



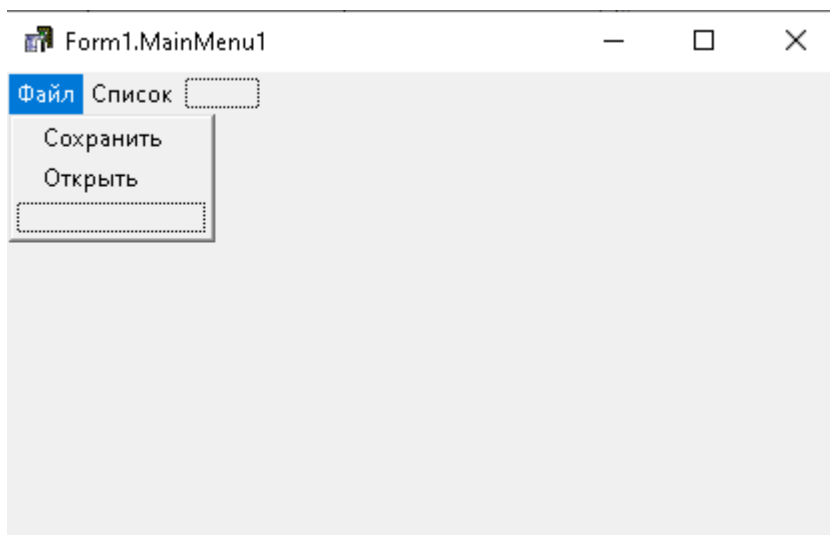
И разместить на форме в любом месте. После нужно сделать 2-ой клик по значку MainMenu на форме, и тогда откроется редактор меню, в котором мы можем добавлять подменю:



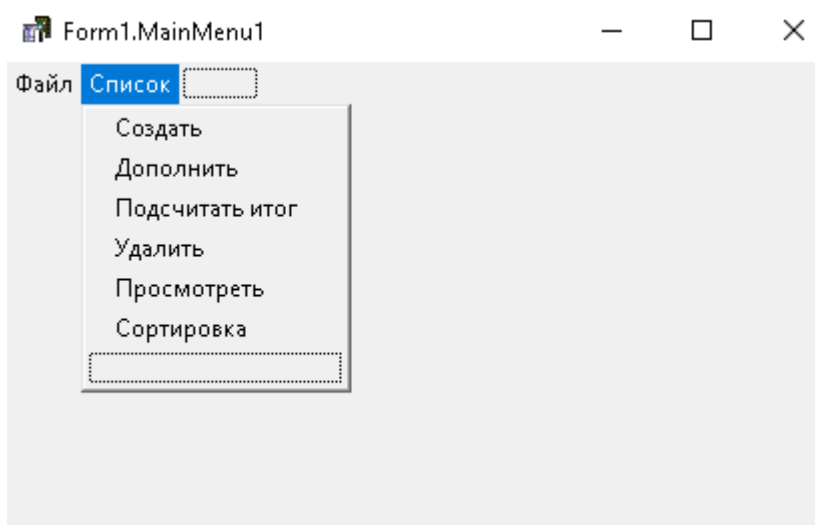
После, добавленные подменю появятся на основной форме и дереве компонентов программы, где мы можем при двойном клике на них, либо через редактор обработчиков событий, создавать обработчик событий,.

В нашей программе будет 2 меню: *Файл* и *Список* – они будут содержать в себе основные функции, нужные для работы с базой данных в данной программе.

В меню *Файл* будут находиться подменю: *Сохранить* и *Открыть* – они будут сохранять и открывать список при помощи файлов.



В меню *Список* будут находится такие подменю как: *Создать*, *Дополнить*, *Подсчитать итог*, *Удалить*, *Просмотреть*, *Сортировка*. Они будут работать со списком, являющимся нашей базой данных, то есть создать, дополнить, удалить, просмотреть или сортировать список, а также подводить итог дня.

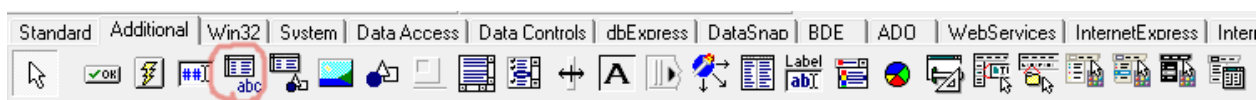


Второй элемент, который будет помогать создавать минимальный, но функциональный интерфейс – таблица *StringGreed*. Это обыкновенная таблица, в которую мы будем выводить базу данных. Заголовками в таблице будут: ФИО отпущившего, . В самом конце будет подводиться итог.

Таблица, как она выглядит в программе:

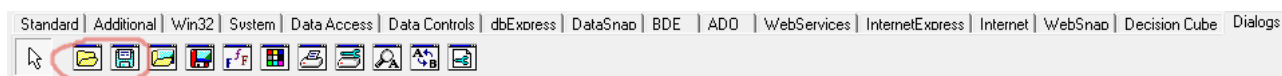
	ФИО отпустившего	Название материала	Кол-во материала	ФИО принявшего
1	Сидоров С.С.	Кирпич	30шт.	Говорайло
2	Сидоров С.С.	Кирпич	60шт.	Иванов И.И.
3	Сидоров С.С.	Шифер	25шт.	Николаев Н.Н.

Чтобы добавить таблицу на форму, необходимо в меню компоненты в подменю Additional найти значок StringGreed

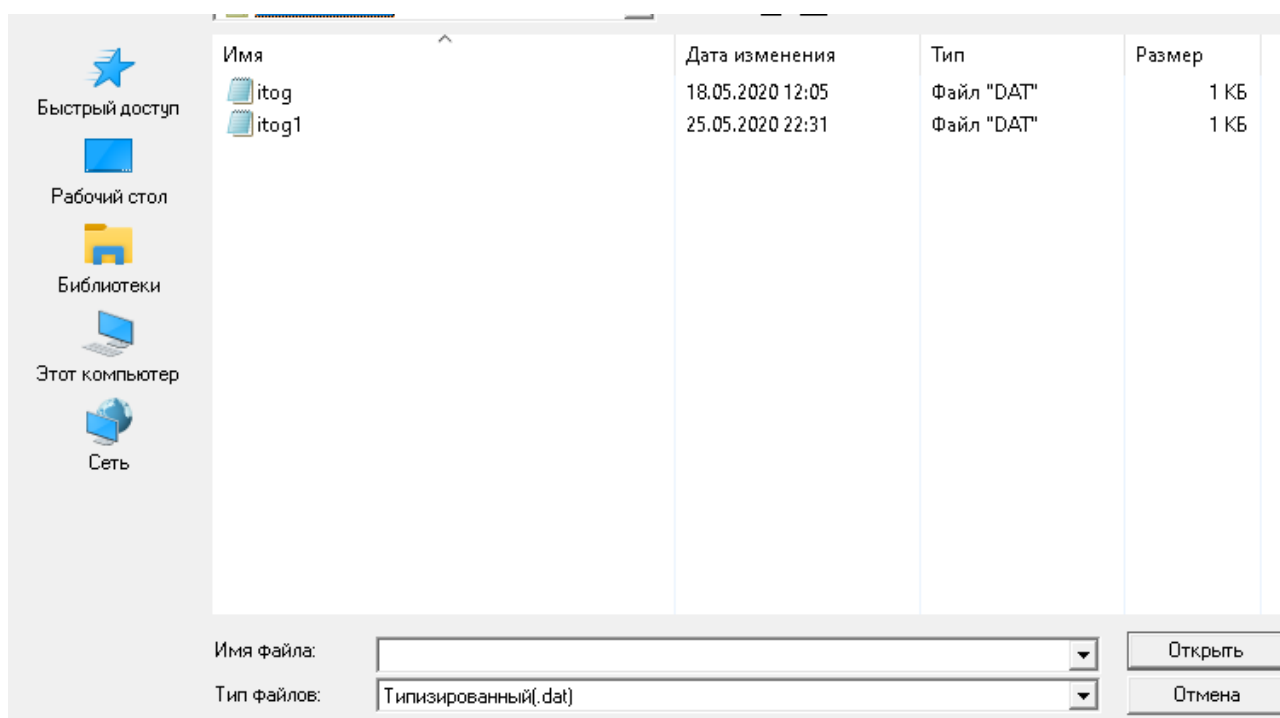
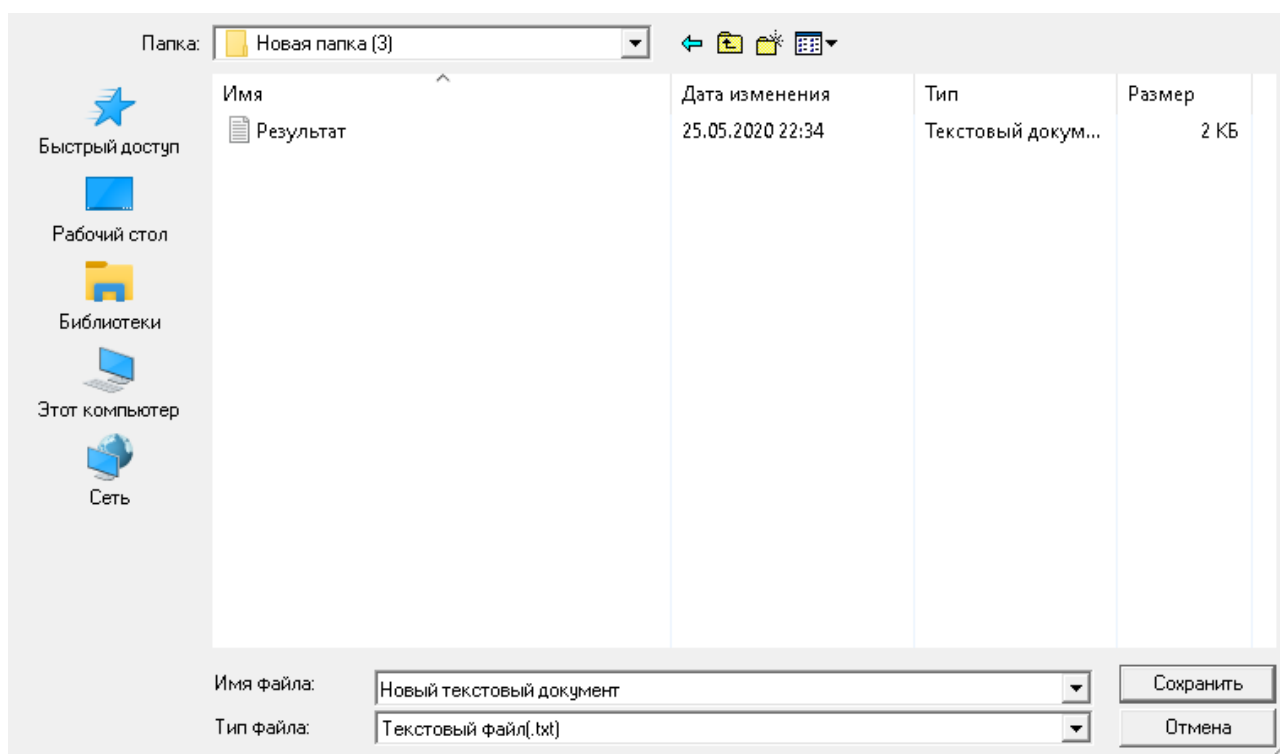


И добавить в любое место на форме. После с таблицей можно начать работать, у неё есть параметр Cells[Номер столбца, Номер строки], который позволяет заполнять таблицу, нумерация столбцов и строк начинается от нуля. Чтобы заполнить таблицу в программе, необходимо заполнить список, либо загрузить его из файла, и нажать на *Список - Просмотреть*. Тогда таблица выведет все данные из данного списка.

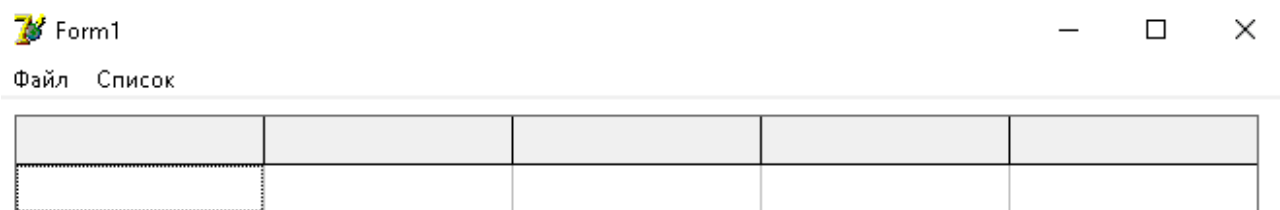
Следующие компоненты это SaveDialog и OpenFileDialog. Они находятся в меню инструментов в подменю Dialogs



При клике на них, на первый план выходит форма и на неё и выставляются данные компоненты. При обращении к этому компоненту вызывается стандартное диалоговое окно открытия или сохранения файла



При запуске приложение выглядит очень просто – Таблица на белом фоне, пока ещё не заполненная данными, и меню Файл и Список выше



Интерфейс программы не содержит лишних кнопок, в которых можно запутаться, не содержит лишних надписей, а также выполняет все необходимые функции

2.2. Реализация обработчиков событий. [1],[3]

В данной главе будут представлены коды для обработчиков событий, описанных выше.

Файл – Сохранить:

```
procedure TForm1.SaveToTxt1Click(Sender: TObject);
//Сохраняет список в текстовый и типизированный файл

var

    s,s1:string;//Имена файлов, в которые будут произведены
сохранения

begin

    if (savedialogTxt.execute) then begin
```

```

        s:=SaveDialogTxt.FileName; //Присваивание переменной s
название текстового файла

        AssignFile(ftxt,s);

        Append(ftxt);

        WriteSpText(Sp,ftxt); //Сохранение списка в текстовый
файл

        closefile(ftxt);

    end

    else exit;

    if savedialogTip.execute then begin

        s1:=SaveDialogTip.FileName; //Присваивание переменной s1
название типизированного файла

        AssignFile(ftip,s1);

        rewrite(ftip);

        WriteSpTip(Sp,ftip); //Сохранение списка в типизированный
файл

        closefile(ftip);

    end

    else exit;

end;

```

Файл – Открыть:

```
procedure TForm1.Open1Click(Sender: TObject); //Открывает  
типизированный файл и загружает оттуда список, сохранённый  
ранее
```

```
var  
  
s:string; //Имя файла, который будет открыт  
  
begin  
  
if not OpenFileDialog1.Execute then exit;  
  
s:=OpenDialog1.FileName; // Присваивание переменной s  
название текстового файла  
  
Assignfile(ftip,s);  
  
reset(ftip);  
  
BuildSpisokFromTip(Sp,ftip); //Построение списка из файла  
  
closefile(ftip);  
  
end;
```

Список – Построить:

```
procedure TForm1.Build1Click(Sender: TObject); //Создаёт  
новый список  
  
begin  
  
BuildSpisok(Sp); //Процедура построения списка
```

end;

Список – Дополнить:

```
procedure TForm1.DobavitVSpisok1Click(Sender: TObject);  
//Добавляет в список новые элементы  
  
begin  
  
    DobVSp(Sp); //Процедура добавления новых элементов в  
список  
  
end;
```

Список – Итог дня:

```
procedure TForm1.Itog1Click(Sender: TObject); //Рассчитывает  
итог дня и записывает его в текстовый файл  
  
var  
  
    s:string; // Переменная для названия файла  
  
begin  
  
    if not OpenFileDialog2.Execute then exit;  
  
    s:=OpenDialog2.FileName; // Присваивание переменной s  
название файла  
  
    assignfile(ftxt,s);  
  
    append(ftxt);
```

```
    ItogDay(Sp,ftxt); //Процедура подведения итога дня  
  
    closeFile(ftxt);  
  
end;
```

Список – Удалить:

```
    procedure TForm1.Delete1Click(Sender: TObject); // Удаляет  
список  
  
    begin  
  
        DelSpisok(Sp); //Процедура удаления списка  
  
    end;
```

Список – Просмотреть:

```
    procedure TForm1.Look1Click(Sender: TObject);  
//Выводит список в таблицу, давая возможность просмотреть  
список  
  
    var  
  
        a:puzel; // Узел необходимый для данной процедуры  
  
        k:integer; // k – счётчик для кол-ва элементов в списке  
  
    begin  
  
        a:=sp;
```

```

    k:=1;

    tabl.cells[1,0]:='ФИО отпустившего';//Заполняем
фиксированные строки таблицы в самом верху

    tabl.cells[2,0]:='Название материала';

    tabl.cells[3,0]:='Кол-во материала';

    tabl.cells[4,0]:='ФИО принявшего';

    While not(a=nil) do begin //Цикл для заполнения
остальной таблицы

        Tabl.cells[0,k]:=IntToStr(k);

        Tabl.cells[1,k]:=a^.x.FIOOtp;

        Tabl.cells[2,k]:=a^.x.Material;

        Tabl.cells[3,k]:=IntToStr(a^.x.KolMat)+'шт.';

        Tabl.cells[4,k]:=a^.x.FIOPrin;

        inc(k);

        Tabl.RowCount:=k+1;

        Tabl.Height:=Tabl.RowCount*26;

        Form1.Height:=Tabl.height+(3*26);

        a:=a^.next;

    end;

end;

```

Список – Сортировка:


```

    procedure TForm1.Poisk1Click(Sender: TObject); //Сортирует
    список

    var

        a:puzel;

        k:integer;

    begin

        SpP:=nil;

        a:=Sp;

        sort(a,SpP);

        k:=1;

        While not(a=nil) do begin

            Tabl.cells[0,k]:=IntToStr(k);

            Tabl.cells[1,k]:=a^.x.FIOOtp;

            Tabl.cells[2,k]:=a^.x.Material;

            Tabl.cells[3,k]:=IntToStr(a^.x.KolMat)+'шт.';

            Tabl.cells[4,k]:=a^.x.FIOPrin;

            Tabl.RowCount:=k+1;

            Tabl.Height:=Tabl.RowCount*26;

            Form1.Height:=Tabl.height+(3*26);

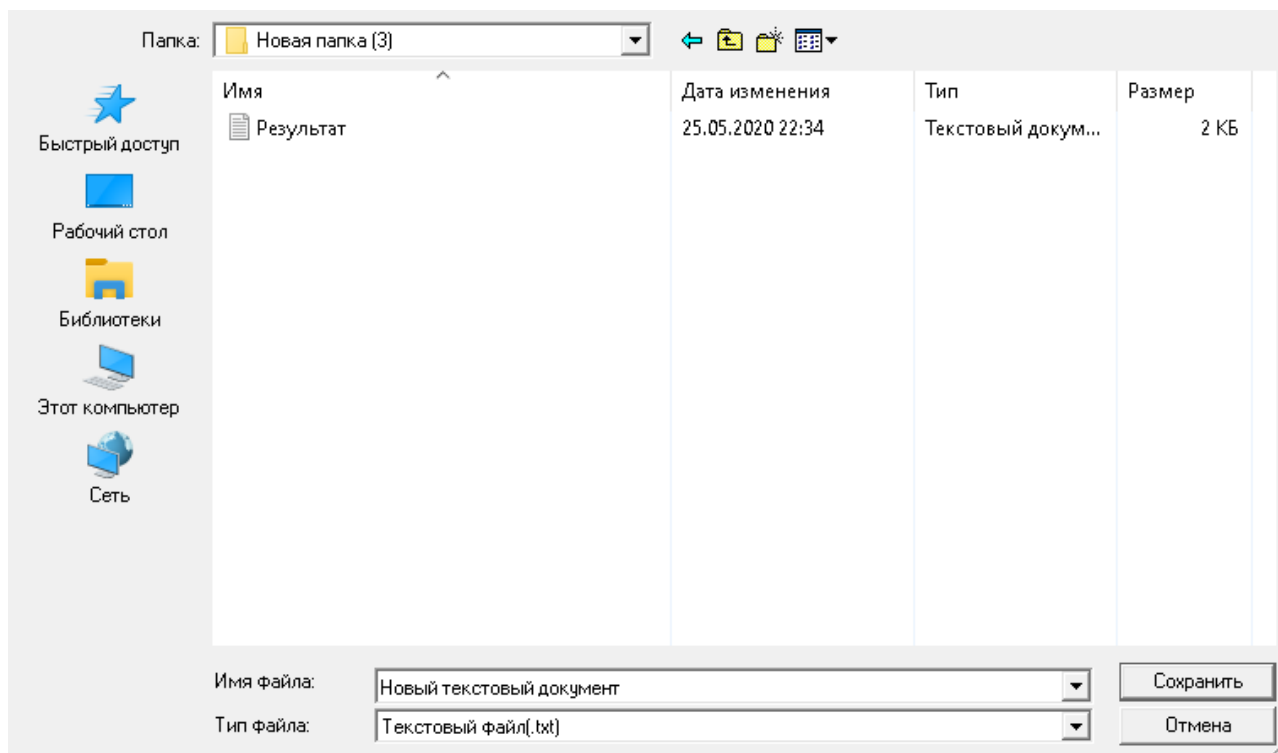
            inc(k);

```

```
        a:=a^.next;  
    end;  
end;
```

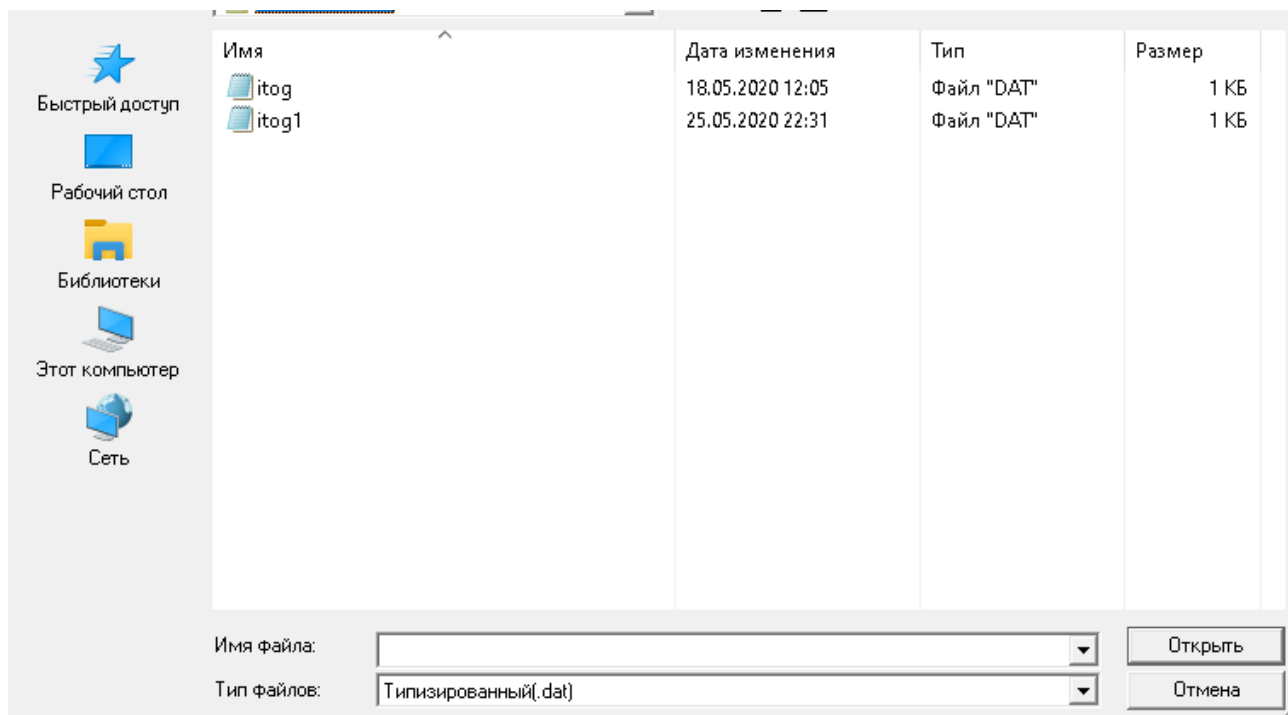
Глава 3. Тестирование созданного приложения, проверка полученных результатов

При клике на *Файл – Сохранить* будет появляться окно,



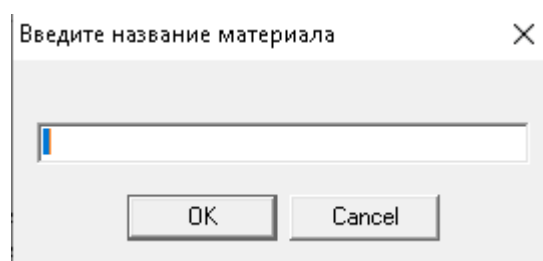
В котором мы можем выбрать или создать файл, в который мы хотим сохранить наш список

При клике на *Файл – Открыть* появится похожее окно, но с одним лишь отличием, тут можно будет только выбрать файл, который хотим открыть.

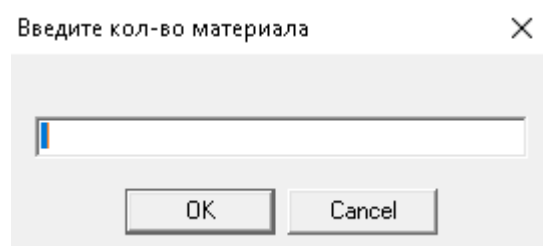


При нажатии на *Список – Построить*, *Список - Дополнить* программа будет поочерёдно спрашивать вас о том, что вы хотите ввести, но при нажатии на *Дополнить* список не будет заполняться сначала, он продолжит тот список, который был уже создан:

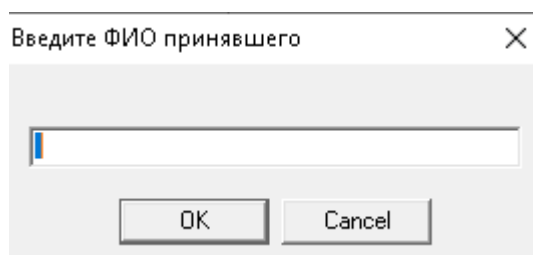
Название материала:



Кол-во материала:



ФИО принявшего (Желательно вида: Иванов И.И.):

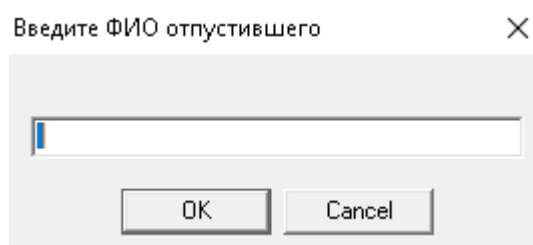


Введите ФИО принявшего

Text input field with a blue cursor.

OK Cancel

ФИО отпустившего (Желательно вида: Иванов И.И.):

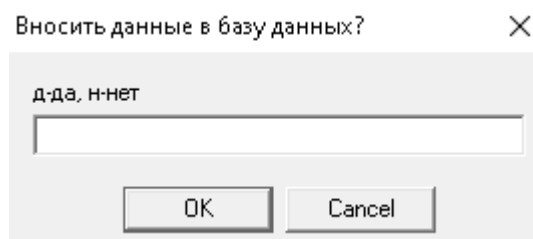


Введите ФИО отпустившего

Text input field with a blue cursor.

OK Cancel

После заполнения данных, программа спросит о том, вносить ли данные в базу данных, где, если вы неправильно заполнили данные, можно отменить их внос в базу данных и заполнить заново.



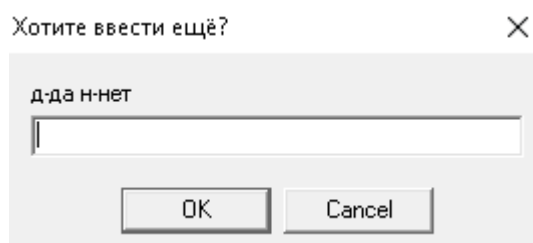
Вносить данные в базу данных?

д-да, н-нет

Text input field.

OK Cancel

После вноса данных в базу данных, программа спросит вас о дальнейшем вводе. Если вы согласитесь, программа вернёт вас в начало ввода, и вы начнёте заполнять новые данные.



Хотите ввести ещё?

д-да н-нет

Text input field.


OK Cancel

При нажатии на *Список - Итог дня* программа подсчитывает общую сумму, и записывает результат в текстовый файл в виде

Кирпич отпущенный строителем по ФИО: Сидоров С.С. В количестве: 30шт был принят строителем по ФИО: Иванов И.И.
 Кирпич отпущенный строителем по ФИО: Сидоров С.С. В количестве: 60шт был принят строителем по ФИО: Говорайло
 Шифер отпущенный строителем по ФИО: Сидоров С.С. В количестве: 25шт был принят строителем по ФИО: Николаев Н.Н.

При нажатии на *Список – Удалить* программа удаляет данный список.

При нажатии на *Список – Просмотреть* программа выводит список в таблицу.



	ФИО отпустившего	Название материала	Кол-во материала	ФИО принявшего
1	Сидоров С.С.	Кирпич	30шт.	Иванов И.И.
2	Сидоров С.С.	Кирпич	60шт.	Говорайло
3	Сидоров С.С.	Шифер	25шт.	Николаев Н.Н.

При нажатии на *Список – Сортировка* программа сортирует данные по возрастанию и алфавиту:

Имеется база данных, состоящая из нескольких элементов:

	ФИО отпустившего	Название материала	Кол-во материала	ФИО принявшего
1	Сидоров С.С.	Кирпич	30шт.	Иванов И.И.
2	Сидоров С.С.	Кирпич	60шт.	Говорайло
3	Сидоров С.С.	Шифер	25шт.	Николаев Н.Н.

Тогда при использовании сортировки, к примеру, по ФИО принявшего, мы получим вот такой упорядоченный список:

	ФИО отпустившего	Название материала	Кол-во материала	ФИО принявшего
1	Сидоров С.С.	Кирпич	30шт.	Говорайло
2	Сидоров С.С.	Кирпич	60шт.	Иванов И.И.
3	Сидоров С.С.	Шифер	25шт.	Николаев Н.Н.

Заключение

При написании данной курсовой работы было использовано четыре источника. Так как выполнить эту работу без изучения теории было бы невозможно, была изучена теория по работе с компонентами интерфейса Delphi: StringGreed, MainMenu – а также компоненты Delphi: SaveDialog и OpenDialog. Также была изучена теория для работы со списками и типами данных. Для работы приложения были созданы такие операции как:

- Сохранение списка
- Загрузка списка из типизированного файла
- Создание списка
- Дополнение списка
- Удаление списка
- Просмотр списка
- Сортировка списка
- Подведение итога дня

Операции работают отлично, ошибки возникают при вводе некорректного значения в поля ввода при создании списка, чтобы исправить данную проблему необходимо перезапустить программу. Программа была протестирована несколько раз, и можно утверждать исправность данной программы, если в будущем понадобится - программа будет доделана и ошибок не будет.

Руководство пользователя написано подробно с использованием скриншотов. Любой пользователь сможет понять как работать с программой по нему.

Литература

1. Pascal-helpov Программирование. Динамические списки Паскаль [Электронный ресурс] / Режим доступа:
http://www.pascal.helpov.net/index/dynamic_lists_pascal_programming
2. Wikipedia База данных [Электронный ресурс] / Режим доступа:
https://ru.wikipedia.org/wiki/База_данных
3. А.Я. Архангельский. Язык Pascal и основа программирования в Delphi. Учебное пособие – М.: ООО«Бином-Пресс», 2004г.-496с.
4. Оформление курсовой работы [Электронный ресурс] / Общие требования к построению и оформлению текстовой документации ЗабГУ. – Режим доступа:
http://zabgu.ru/files/html_document/pdf_files/fixed/Normativny'e_dokumenty'/MI__01-02-2018_Obshhie_trebovaniya_k_postroeniyu_i_oformleniyu_uchebnoj_tekstovoj_dokumentacii.pdf

Приложение

Модуль формы:

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics,
Controls, Forms, Dialogs, Menus, UnitmodulK, Grids;

type

TForm1 = class(TForm)

MainMenu1: TMainMenu;

File1: TMenuItem;

SaveToTxt1: TMenuItem;

Open1: TMenuItem;

Spisok1: TMenuItem;

Build1: TMenuItem;

Itog1: TMenuItem;

Delete1: TMenuItem;

Look1: TMenuItem;

SaveDialogTxt: TSaveDialog;

SaveDialogTip: TSaveDialog;

OpenDialog1: TOpenDialog;

```

DobavitVSpisok1: TMenuItem;

OpenDialog2: TOpenDialog;

Tab1: TStringGrid;

Poisk1: TMenuItem;

procedure SaveToTxt1Click(Sender: TObject);//Сохранение в
текстовый файл

procedure Open1Click(Sender: TObject);//Открытие из
типизированного файла

procedure Build1Click(Sender: TObject);//Построение списка

procedure Itog1Click(Sender: TObject);//Подведение итога

procedure Delete1Click(Sender: TObject);//Удаление списка

procedure DobavitVSpisok1Click(Sender:
TObject);//Добавление в список новых элементов

procedure Look1Click(Sender: TObject);//Просмотр списка

procedure Poisk1Click(Sender: TObject);//Сортировка списка

private
    { Private declarations }
public
    { Public declarations }
end;

var

    Form1: TForm1;
    ftxt:text;
    ftip:fzap;

```

Sp,SpP:puzel;

Созданный модуль:

uses

SysUtils, Dialogs;

Type

Building = record //Тип записи для нашей задачи,
который будет храниться в базе данных

Material:string[30]; //название материала

KolMat:integer; //Количество данного материала

FIOPrin:string[30]; //ФИО принявшего материал

FIOOtp:string[30]; //ФИО отпустившего материал

end;

PUzel = ^Zl; //Указатель на тип данных, создающий
список

Zl = record //Запись, реализующая список

x:Building; //Информация, хранящаяся в узлах
списка

next:puzel; //Указатель на следующий узел

pred:puzel; //Указатель на предыдущий узел

end;

Fzap = file of Building; //Типизированный файл, для хранения базы данных

procedure BuildSpisok(var f: PUzel); //Процедура, для построения списка

procedure AddFirst(var f: PUzel; a: PUzel); //Вставить узел а первым в список

procedure AddAfter(var old:PUzel; a: PUzel); //Вставить узел а после old

procedure WriteSpTip(var f: PUzel; var ftip:Fzap); //Записать данные списка в типизированный файл

procedure WriteSpText(var f: PUzel; var ftxt:Text); //Записать в текстовый файл

procedure BuildSpisokFromTip(var f:puzel;var ftip:Fzap);
//Построить список из даныхх, взятых из тип. файла

procedure DelFirstElement(var f,a: PUzel); //Выделить первый элемент списка

procedure DelElement(var old,a: PUzel); //Выделить элемент из списка, следующий за old

procedure ItogDay(var f:puzel;var ftxt:text); //Процедура для выполнения задачи

procedure DobVSp(var f:puzel); //Добавляет в список новые элементы, не удаляя старые

procedure DelSpisok(var f: PUzel); //Удалить список

```
procedure PoslElem(f:puzel;var a:puzel);//Выделяет  
последний элемент списка
```

```
procedure Sort(f:puzel;var b:puzel);//Сортирует список
```