Федеральное агентство по образованию

Государственное образовательное учреждение высшего профессионального образования

‹‹Курский государственный технический университет››

**Кафедра\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**КУРСОВАЯ РАБОТА (ПРОЕКТ)**

по дисциплине ‹‹\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_››

(наименование учебной дисциплины)

на тему ‹‹\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Специальность (направление подготовки)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(код, наименование)

Авторы работы (проекта)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(инициалы, фамилия) (подпись, дата)

Группа\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель работы (проекта)\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(инициалы, фамилия) (подпись, дата)

Работа (проект) защищена \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

Оценка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Председатель комиссии\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(инициалы, фамилия) (подпись, дата)

Члены комиссии \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(инициалы, фамилия) (подпись, дата)

Курск, 20\_\_\_г.

Федеральное агентство по образованию

Государственное образовательное учреждение высшего профессионального образования

‹‹Курский государственный технический университет››

**Кафедра\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ (ПРОЕКТ)

Студент (слушатель)\_\_\_\_\_\_\_\_\_\_\_\_\_\_ шифр \_\_\_\_\_\_\_ группа \_\_\_\_\_\_\_\_\_

(фамилия, инициалы)

1. Тема\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Срок предоставления работы (проекта) к защите ‹‹\_\_\_››\_\_\_\_\_20\_\_\_г.

3. Исходные данные (*для проектирования, для научного исследования*):

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. Содержание пояснительно записки курсовой работы (проекта):

4.1.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.2.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.3.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.4.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.5.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.6.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.7.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Перечень графического материала:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель работы (проекта)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата) (инициалы, фамилия)

Задание принял к исполнению\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата)

Реферат

Представляемый документ содержит:

35 страниц текста, четыре рисунка, три используемых источника.

Перечень ключевых слов: база данных, учет товаров, программы для магазинов.

Объектом исследования является база данных компьютерного магазина.

Цель работы состоит в создании программного комплекса, обеспечивающего использование структурированной базы данных для компьютерного магазина. Методы, используемые при разработке – язык программирования высокого уровня C++. Созданный программный продукт обеспечивает выполнение всех требований технического задания.

Содержание

Введение

1 Техническое задание

1.1 Основание для разработки

1.2 Назначение разработки

1.3 Требований пользователя к программному изделию

1.4 Требования к программе или программному изделию

1.4.1 Входные данные

1.4.2 Процессы обработки

1.4.3 Выходные данные

1.4.4 Результирующие компоненты изделия

1.4.5 Носители информации

1.4.6 Требования к составу и параметрам технических средств

1.5 Требования к программной документации

1.6 Стадии и этапы разработки

2 Рабочий проект

2.1 Описание программы

2.1.1 Тип входных данных

2.1.2 Выходные данные

2.2.1 Форма

2.3.1 Компоненты на форме

2.4.1 Функциональные тесты

Заключение

Список использованных источников

Приложение А

Введение

Цель данной курсовой работы разработать программный продукт, предназначенный для учёта, реализации товаров в компьютерного магазине.

Область применения программного продукта. Программный продукт может быть использован для широкого потребления среди пользователей.

До внедрения первых ЭВМ в торговую отрасль вести учёт за товарами, которые находятся в наличии и которые только поступают в продажу было крайне трудно. Когда на производстве появились первые ЭВМ, то одновременно появились и программы систематизирующие и ускоряющие работу и учёт магазинов. Эти программы представляют собой базы данных в которых хранятся все данные о товарах. Современное программное обеспечение позволяет пользователям хранить огромные объёмы информации о продукции, вводить в базу данных значения введённые с клавиатуры, удалять любую строку по желанию пользователя, просматривать все записи, хранящиеся в базе данных, выводить количество всех записей на экран, легко и быстро находить нужный товар по необходимым параметрам, без сложной работы с документами производить быструю надбавку в процентном соотношении на нужный товар. Подобный программный продукт широко используется в большинстве современных сетей крупных магазинов и торговых центров.

Практическое значение. Программный продукт позволяет сократить время и автоматизировать учёт, реализацию товаров.

Актуальность разработки программного продукта состоит значительном упрощении и автоматизации учёта, приёма, продажи товаров в каком-либо техническом учреждении.

Программный продукт должен быть разработан на языке программирования высокого уровня C++, использовать принципы объектно-ориентированного программирования и структурный подход к решению поставленных задач.

Результатами выполнения дипломной работ должен стать готовый программный продукт, обеспечивающий всем требованиям технического задания.

# 1 Техническое задание

## Основание для разработки

Задача дана на основании курсовой работы. База данных для продуктового магазина.

## 1.2 Назначение разработки

Программный продукт разрабатывается с целью создания упорядоченной структуры данных, что позволяет существенно экономить используемое для работы время.

## 1.3 Требования пользователя к программному изделию

Необходимо реализовать программный продукт, позволяющий использовать структурированную базу данных.

## 

## 1.4 Требования к программе или программному изделию

### 1.4.1 Входные данные

Входными данными для программного продукта являются данные, вводимые пользователем с клавиатуры, такие как:

- Наименование товара;

- Код товара;

- Дата поступления;

- Цена товара;

- Количество;

- Имеющийся в наличии файл базы данных «Salon.txt».

Ячейки с соответствующими значениями будут использоваться для создания таблицы базы данных. Все вводимые значения являются переменными строкового типа.

### 1.4.2 Процессы обработки

С помощью кнопки «Добавить» значения, введенные пользователем во входных данных записываются в таблицу в соответствующие ячейки. После этого происходит очистка полей для ввода значений.

С помощью кнопки «Сохранить» таблицу со всеми добавленными в нее значениями сохранить в файл «Salon.txt», создающийся в каталоге с программой, или добавить значения в уже имеющийся файл. Полученный файл можно будет впоследствии использовать, не прибегая к работе программы.

Кнопка «Удалить» предназначена для удаления любой по выбору пользователя строки из уже добавленных в базу данных. В ходе этой операции последняя строка таблицы записывается на место удаленной, а количество строк таблицы соответственно уменьшается.

Кнопка «Количество» выводит на экран общее количество строк в таблице.

С помощью кнопки «Поиск» пользователь может определить, есть или в базе данных продукт, который удовлетворяет каким-либо критериям, интересующими пользователя. В ходе этого процесса все совпадающие значения будут выведены в поисковую таблицу. В случае не нахождения подобных значений в базе данных будет выведена пустая таблица.

С помощью кнопки «Накрутка» и значения, введенного пользователем в соответствующее поле можно регулировать выходную цену продукта. Цена будет изменяться в процентном отношении с ранее введенной. Счет идет до двух чисел после запятой.

### 

### 1.4.3 Выходные данные

Выходными данными для программного продукта являются значения, выводимые программой пользователю на экран или полученные в процессе обработки данных, такие как:

- файл базы данных «Salon.txt»;

- таблица, созданная пользователем в результате работы программы;

- таблица поиска с выведенными на экран табличными значениями и значениями, введенными пользователем;

- количество всех записей в базе данных.

1.4.4 Результирующие компоненты изделия

Заказчику предоставляются исполняемые файлы и отчёт о проделанной работе.

### 1.4.5 Носители информации

Продукт должен размещаться в виде файлов на CD дисках. В процессе работы могут использоваться Flash, CD-R/-RW, DVD-R/RW или другие носители информации.

### 1.4.6 Требования к составу и параметрам технических средств

Программный комплекс должен корректно работать на компьютере со следующими техническими характеристиками:

* процессор Pentium III 1000 МГц;
* оперативная память объемом 256 Мб;
* жесткий диск объемом 30 Гб;
* видеоадаптер SVGA;
* клавиатура;
* манипулятор типа “мышь”;

устройство удаленного доступа к сети: модем или сетевая карта.

## 1.5 Требования к программной документации

Программная документация должна включать следующие документы:

* техническое задание;
* рабочий проект;
* диск с приложением.

В приложении к документу "Рабочий проект" должен быть приведен листинг исходных текстов программного изделия.

## 1.6 Стадии и этапы разработки

Выполнение разработки должно осуществляться в три этапа:

* техническое задание;
* рабочий проект.

На стадии "Техническое задание" проводится:

* постановка задачи;
* разработка требований к программному изделию;
* изучение литературы по задаче.

На стадии "Рабочий проект" проводится:

* разработка схем алгоритмов;
* физическое проектирование программного изделия;
* тестирование и отладка программного изделия;
* оформление работы.

# 2 Рабочий проект

## 2.1 Описание программы:

Данная программа состоит из двух частей:

- Внешний вид форм;

- Компоненты на форме.

2.1.1 Тип входных данных

Входные данные, введенные пользователем с клавиатуры:

- Наименование товара (Edit1, тип AnsiString);

- Код товара (Edit3, тип AnsiString);

- Дата поступления (Edit2, тип AnsiString);

- Цена товара (Edit4, тип AnsiString);

- Количество(Edit5, тип AnsiString);

- Надбавка(Edit6, тип float).

Имеющийся файл базы данных «Salon.txt».

2.1.2 Выходные данные

Выходными данными для программного продукта являются значения, выводимые программой пользователю на экран или полученные в процессе обработки данных, такие как:

- файл базы данных «Salon.txt»;

- таблица, созданная пользователем в результате работы программы (DataGrid1);

- таблица поиска с выведенными на экран табличными значениями и значениями, введенными пользователем (SearchGrid1);

- количество всех записей в базе данных (Label6, тип Int);

- сообщение об удалении всех записей из базы данных

(ShowMessage( "Таблица пуста" ).

2.2.1 Форма

2.2.1 Стандартная форма

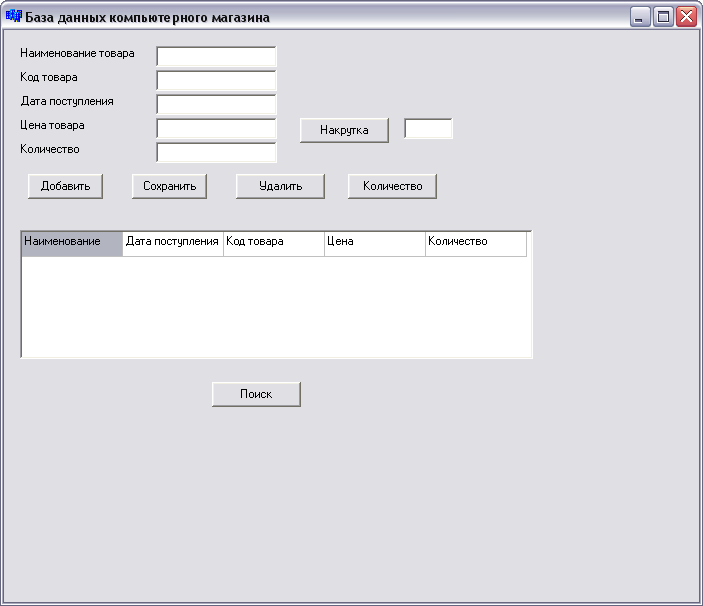


Рисунок 1 -- Стандартная форма программы

2.2.1.2 Форма с выведенным на экран количеством записей в базе данных и примером введенной таблицы

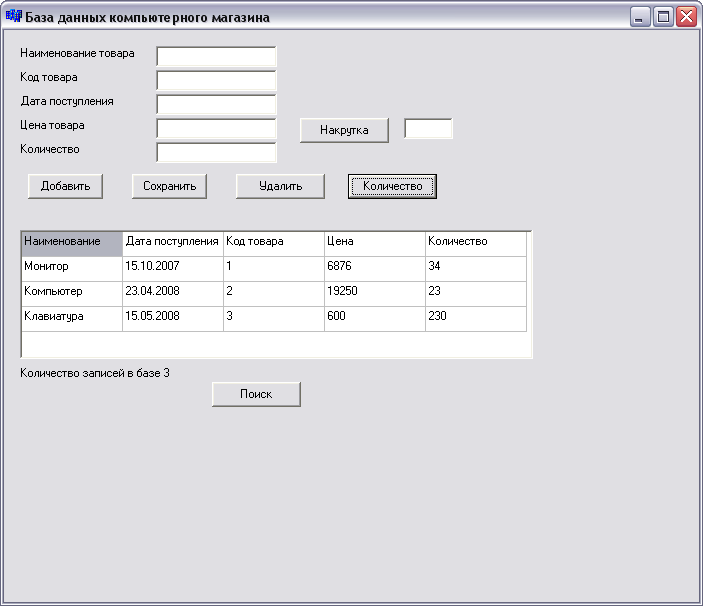


Рисунок 2 – Форма с выведенным на экран количеством записей в базе данных.

2.2.1.3 Форма с выведенной на экран таблицей поиска и совпадением критерия поиска пользователя и уже имеющейся в базе данных записи.

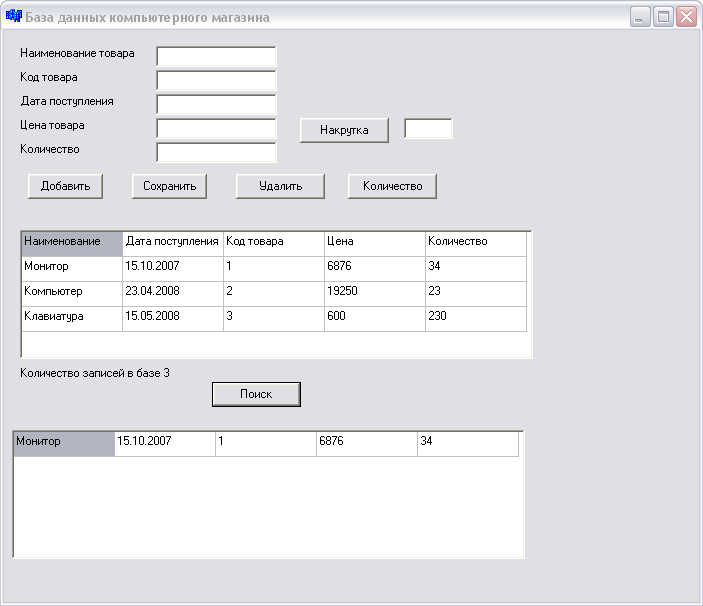


Рисунок 3 -- Форма с выведенной на экран таблицей поиска.

2.2.1.4. Форма с выводом сообщения об удалении всех строк в базе данных

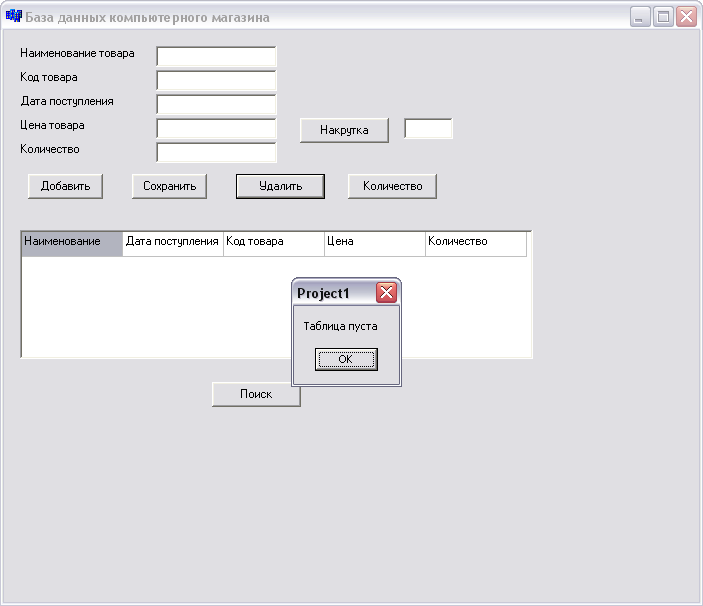


Рисунок 4 -- Форма с выводом сообщения об удалении всех строк в базе данных

2.3.1 Компоненты на форме.

2.3.1.1 Add

Add – эта кнопка предназначена для добавления в базу данных введенных с клавиатуры значений, таких как: наименование товара, код товара, дата поступления, дата окончания срока годности, цена товара, количество. Кнопка использует две функции.

2.3.1.1.1 add()

Функция add() используется для передачи значений полей для ввода в таблицу DataGrid1.

Входными данными для функции являются значения полей для ввода данных (Edit1, Edit3, Edit2, Edit4, Edit5), выходными данными являются значения соответствующих ячеек таблицы DataGrid1.

2.3.1.1.2 Clear()

Функция Clear() предназначена для очищения значений после ввода их с клавиатуры в базу данных. Не использует входных и выходных данных.

2.3.1.2 Save

Save – эта кнопка предназначена для сохранения готовой базы данных в файл с именем Salon.txt в каталоге с программой. Кнопка вызывает одну функцию.

2.3.1.2.1 form\_save()

Функция form\_save() предназначена для открытия или создания файла с именем Salon.txt, записи в него всех значений базы данных и закрытии файла.

Входными данными для функции являются значения соответствующих ячеек таблицы DataGrid1, выходным данным является файл с записанными в него этими значениями.

2.3.1.3 Delete

Delete – эта кнопка предназначена для удаления выбранной пользователем строки из базы данных. Кнопка вызывает одну функцию.

2.3.1.3.1 delet()

Функция delet() предназначена для вывода сообщения «Таблица пуста» в случае, если в базе данных не окажется записей, и для удаления выбранной пользователем строки в базе данных путем записи в эту строку последней строки в базе данных и уменьшении базы данных по размеру.

Входными данными для функции являются свойство RowCount таблицы DataGrid1 и значение указателя на строку rowtab, выходным данным является измененная таблица DataGrid1.

2.3.1.4 Number

Number – эта кнопка предназначена для вывода на экран количества всех записей в базе данных. Использует одну функцию.

2.3.1.4.1 all()

Функция All() предназначена для подсчета записей в базе данных и вывода на форму сообщения «Количество записей в базе ».

Входное данное для функции – переменная row (количество строк таблицы DataGrid1), выходным данным является метка на форме Label6, выводящая целые значения.

2.3.1.5 Search

Search – эта кнопка предназначена для сравнения значений, таких как: наименование товара, код товара, дата поступления, дата окончания срока годности, цена товара, количество, введенных с клавиатуры, с уже имеющимися соответствующими значениями в базе данных и вывода их на экран на отдельной таблице. Использует две функции.

2.3.1.5.1 search()

Функция search() предназначена для поиска совпадений в исходной базе данных и значениями, введенными пользователем с клавиатуры по любым критериям поиска и выводом возможных совпадений на экран.

Входными данными для функции являются значения полей для ввода данных (Edit1, Edit3, Edit2, Edit4, Edit5), переменная row (количество строк таблицы DataGrid1), значения соответствующих ячеек таблицы DataGrid1, выходным данным является таблица поиска SearchGrid1.

2.3.1.5.2 Clear()

Функция была описана в 2.3.1.1.2

2.3.1.6 plus

Plus – эта кнопка служит для расчета дополнительной стоимости продукта и вывода его в базу в соответствующей ячейке. Использует одну функцию.

2.3.1.6.1 plus()

Функция plus() предназначена для расчета и прибавления к стоимости продукта определенного значения, определяемого пользователем.

Входными данными для функции являются: переменная row (количество строк таблицы DataGrid1) и вещественная переменная к, вводимая пользователем в Edit6, выходными данными являются значения колонки "Цена" таблицы DataGrid1.

2.3.1.7 sozdanie\_form()

Функция sozdanie\_form() предназначена для записи заголовочных колонок в таблицу, создания либо открытия уже имеющегося файла.

Входными данными являются файл базы данных " Salon.txt", переменная row (количество строк таблицы DataGrid1), выходными данными являются значения ячеек таблицы DataGrid1.

2.4.1 Функциональные тесты

2.4.1.1 При вводе корректных данных в поля для ввода и нажатии на кнопку «Добавить» происходит запись этих данных в соответствующие ячейки в таблицу данных.

2.4.1.2 Если в уже имеющейся таблице выбрать строку, кроме строки с названиями колонок и нажать на кнопку «Удалить», то произойдет удаление выбранной строки, а на ее место в таблицу будет записана строка, которая была в таблице последней. При удалении всех строк в таблице будет выдаваться сообщение "Таблица пуста".

2.4.1.3 Если в процессе работы требуется вывести на экран количество всех записей в базе данных, то достаточно нажать на кнопку «Количество». Если таблица не пустая, то будет выведено количество строк таблицы, если же она пустая, то будет выведен 0.

2.4.1.4 В процессе работы может возникнуть потребность в поиске записей в базе данных по различным критериям. Для решения этой проблемы надо ввести соответствующие значения в поля для ввода (поиск может проходить как по одному критерию, так и по нескольким) и нажать кнопку «Поиск». После нажатия кнопки открывается поисковая таблица, в которую будут выведены все найденные совпадения по критериям поиска между пользовательскими значениями и значениями из базы данных. Если же совпадений найдено не будет, то таблица поиска останется пустой.

2.4.1.5 При введении в поле Edit6 вещественного значения (процентная накрутка к первоначальной стоимости товара) и нажатии на кнопку «Накрутка» происходит замена значений колонки «Цена» в главной таблице новыми значениями. Новые значения также являются денежными, т.е. после запятой в числе только две цифры.

2.4.1.6 Для сохранения изменений в базе данных надо нажать кнопку «Сохранить». При этом происходит создание или перезапись файла "Salon.txt" в каталоге с программой. Файл может быть использован впоследствии отдельно от основной программы и открывается любым текстовым редактором.

Заключение

Разработанный программный продукт обеспечивает выполнение всех требований, предъявленных к нему в техническом задании.

Программный продукт рекомендован к использованию для широкого круга пользователей. Использование программного продукта позволяет существенно сократить используемое место на жестком диске и время, проводимое за компьютером, что очень актуально в наши дни.

Список использованных источников

1. Архангельский, А.Я., Тагин, М.А. Программирование в C++Builder 6 и 2006 [Текст] / М..: ООО “Бином-Пресс”, 2002. – 1184 с.
2. Архангельский, А.Я. Компоненты общего назначения библиотеки Delphi 5.0 [Текст] / А.Я. Архангельский – М.: Бином, 2001. – 416 с.
3. Архангельский, А.Я. Программирование в Delphi 6 [Текст] / А.Я. Архангельский – С.-П.: Бином, 2002. – 1120 с.

Приложение А

Текст программы

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

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

#include "stdio.h"

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

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

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

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

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

int row = 1;

int rowtab = -1;

class Ta

{

public:

AnsiString Naimenovanie, Date1, Kod, Price, Quantity;

FILE \*f;

void input(AnsiString, AnsiString, AnsiString, AnsiString, AnsiString);

void sozdanie\_form();

void add();

void clear();

void delet();

void close();

void form\_save();

void all();

void search();

void plus();

};

void Ta::input(AnsiString Naimenovanie, AnsiString Date, AnsiString Kod, AnsiString Price, AnsiString Quantity)

{

this->Naimenovanie = Naimenovanie ;

this->Date1 = Date1 ;

this->Kod = Kod;

this->Price = Price;

this->Quantity = Quantity;

}

void Ta::sozdanie\_form()

{

Form1->DataGrid1->RowCount = 1;

Form1->DataGrid1->Cells[0][0] = "Íàèìåíîâàíèå";

Form1->DataGrid1->Cells[1][0] = "Äàòà ïîñòóïëåíèÿ";

Form1->DataGrid1->Cells[2][0] = "Êîä òîâàðà";

Form1->DataGrid1->Cells[3][0] = "Öåíà";

Form1->DataGrid1->Cells[4][0] = "Êîëè÷åñòâî";

char a1[50];

char b1[50];

char c1[50];

char d1[50];

char e1[50];

if ((this->f = fopen("salon.txt","rb")) != NULL)

{

f = fopen("salon.txt","rb");

while(!(feof(this->f)))

{

fread(a1,sizeof(char),50,this->f);

fread(b1,sizeof(char),50,this->f);

fread(c1,sizeof(char),50,this->f);

fread(d1,sizeof(char),50,this->f);

fread(e1,sizeof(char),50,this->f);

this->Naimenovanie = a1;

this->Date1 = b1;

this->Kod = c1;

this->Price = d1;

this->Quantity = e1;

if (!(feof(this->f)))

{

row++;

Form1->DataGrid1->RowCount = row;

Form1->DataGrid1->Cells[0][row - 1] = this->Naimenovanie;

Form1->DataGrid1->Cells[1][row - 1] = this->Date1;

Form1->DataGrid1->Cells[2][row - 1] = this->Kod;

Form1->DataGrid1->Cells[3][row - 1] = this->Price;

Form1->DataGrid1->Cells[4][row - 1] = this->Quantity;

}

}

}

else {this->f = fopen("salon.txt","wb");};

fclose(this->f);

}

void Ta::add()

{

Form1->DataGrid1->Cells[0][Form1->DataGrid1->RowCount] = Form1->Edit1->Text;

Form1->DataGrid1->Cells[1][Form1->DataGrid1->RowCount] = Form1->Edit2->Text;

Form1->DataGrid1->Cells[2][Form1->DataGrid1->RowCount] = Form1->Edit3->Text;

Form1->DataGrid1->Cells[3][Form1->DataGrid1->RowCount] = Form1->Edit4->Text;

Form1->DataGrid1->Cells[4][Form1->DataGrid1->RowCount] = Form1->Edit5->Text;

row++;

Form1->DataGrid1->RowCount = row;

Form1->Edit1->Clear();

Form1->Edit2->Clear();

Form1->Edit3->Clear();

Form1->Edit4->Clear();

Form1->Edit5->Clear();

}

void Ta::form\_save()

{

this->f = fopen("salon.txt", "w+b");

char one[50];

char two[50];

char three[50];

char four[50];

char five[50];

for(int i = 1; i < row; i++)

{

AnsiString a1 = Form1->DataGrid1->Cells[0][i];

AnsiString a2 = Form1->DataGrid1->Cells[1][i];

AnsiString a3 = Form1->DataGrid1->Cells[2][i];

AnsiString a4 = Form1->DataGrid1->Cells[3][i];

AnsiString a5 = Form1->DataGrid1->Cells[4][i];

strncpy(one,a1.c\_str(),50);

strncpy(two,a2.c\_str(),50);

strncpy(three,a3.c\_str(),50);

strncpy(four,a4.c\_str(),50);

strncpy(five,a5.c\_str(),50);

fwrite(one,sizeof(char),50,this->f);

fwrite(two,sizeof(char),50,this->f);

fwrite(three,sizeof(char),50,this->f);

fwrite(four,sizeof(char),50,this->f);

fwrite(five,sizeof(char),50,this->f);

}

fclose(this->f);

}

void Ta::delet()

{

int k = 0;

for(int i = 1; i < row; i++)

k++;

if (Form1->DataGrid1->RowCount == 1)ShowMessage( "Òàáëèöà ïóñòà" );

else

{

Form1->DataGrid1->Cells[0][rowtab] = Form1->DataGrid1->Cells[0][k];

Form1->DataGrid1->Cells[1][rowtab] = Form1->DataGrid1->Cells[1][k];

Form1->DataGrid1->Cells[2][rowtab] = Form1->DataGrid1->Cells[2][k];

Form1->DataGrid1->Cells[3][rowtab] = Form1->DataGrid1->Cells[3][k];

Form1->DataGrid1->Cells[4][rowtab] = Form1->DataGrid1->Cells[4][k];

Form1->DataGrid1->Cells[5][rowtab] = Form1->DataGrid1->Cells[5][k];

Form1->DataGrid1->RowCount--;

}

}

void Ta::all()

{

int k = 0;

for(int i = 1; i < row; i++)

k++;

Form1->Label6->Caption = "Êîëè÷åñòâî çàïèñåé â áàçå " + IntToStr(k);

}

void Ta::search()

{

int k = 0;

for (int i = 1; i < row; i++)

{

if ((Form1->DataGrid1->Cells[0][i] == Form1->Edit1->Text)||(Form1->DataGrid1->Cells[1][i] == Form1->Edit2->Text)||(Form1->DataGrid1->Cells[2][i] == Form1->Edit3->Text)||(Form1->DataGrid1->Cells[3][i] == Form1->Edit4->Text)||(Form1->DataGrid1->Cells[4][i] == Form1->Edit5->Text))

{

Form1->SearchGrid1->Cells[0][k] = Form1->DataGrid1->Cells[0][i];

Form1->SearchGrid1->Cells[1][k] = Form1->DataGrid1->Cells[1][i];

Form1->SearchGrid1->Cells[2][k] = Form1->DataGrid1->Cells[2][i];

Form1->SearchGrid1->Cells[3][k] = Form1->DataGrid1->Cells[3][i];

Form1->SearchGrid1->Cells[4][k] = Form1->DataGrid1->Cells[4][i];

Form1->SearchGrid1->Cells[5][k] = Form1->DataGrid1->Cells[5][i];

k++;

Form1->SearchGrid1->RowCount = k;

}

Form1->SearchGrid1->Visible = True;

}

Form1->Edit1->Clear();

Form1->Edit2->Clear();

Form1->Edit3->Clear();

Form1->Edit4->Clear();

Form1->Edit5->Clear();

}

void Ta::plus()

{

float k = StrToFloat(Form1->Edit6->Text);

k = k/100;

for (int i = 1; i < row; i++)

Form1->DataGrid1->Cells[3][i] = ((float)((int)((Form1->DataGrid1->Cells[3][i]\*(1 + k))\*100))/100);

}

void \_\_fastcall TForm1::AddClick(TObject \*Sender)

{

Ta a;

a.add();

}

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

void \_\_fastcall TForm1::SaveClick(TObject \*Sender)

{

Ta a;

a.form\_save();

}

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

void \_\_fastcall TForm1::FormCreate(TObject \*Sender)

{

Ta a;

a.sozdanie\_form();

}

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

void \_\_fastcall TForm1::DeleteClick(TObject \*Sender)

{

Ta a;

a.delet();

}

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

void \_\_fastcall TForm1::SearchClick(TObject \*Sender)

{

Ta a;

a.search();

}

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

void \_\_fastcall TForm1::NumberClick(TObject \*Sender)

{

Ta a;

a.all();

}

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

void \_\_fastcall TForm1::plusClick(TObject \*Sender)

{

Ta a;

a.plus();

}

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

void \_\_fastcall TForm1::DataGrid1SelectCell(TObject \*Sender, int ACol,

int ARow, bool &CanSelect)

{

rowtab = ARow;

}

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