МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

кафедра Компьютерных технологий

«Допущен к защите»
Заведующий кафедрой д. ф.-м. н.
профессор Куралбаев З. К.
(Ф.И.О., ученая степень, звание)
_____ «____» _____ **20__ г.**
(подпись)

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Разработка автоматизированного рабочего места специалиста отдела продаж

Специальность 050704 – „Вычислительная техника и программное обеспечение"

Выполнил (а) Айтжанов М.Б. ВТ-10-5
(Фамилия и инициалы) группа

Научный руководитель Кожамбердиева М.И. к.п.н., доцент
(Фамилия и инициалы, ученая степень, звание)

Консультанты:

по экономической части:
Ержанова З.Д., ст. преподаватель
(Фамилия и инициалы, ученая степень, звание)
Ержанова «23» 05 2014 г.
(подпись)

по безопасности жизнедеятельности:
Ахметова М.Е., д.т.н., профессор
(Фамилия и инициалы, ученая степень, звание)
_____ «29» 05 2014 г.
(подпись)

по применению вычислительной техники:
Кожамбердиева М.И. к.п.н., доцент
(Фамилия и инициалы, ученая степень, звание)
_____ «____» _____ 20__ г.
(подпись)

_____
(Фамилия и инициалы, ученая степень, звание)
_____ «____» _____ 20__ г.
(подпись)

Нормоконтролер: Тулпов Д.М.
(Фамилия и инициалы, ученая степень, звание)
_____ «31» мая 2014 г.
(подпись)

Рецензент: _____
(Фамилия и инициалы, ученая степень, звание)
_____ «____» _____ 20__ г.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

Факультет Информационных технологий
Специальность 050704- Вычислит. техника и программн. обеспеч.
Кафедра компьютерные технологии

ЗАДАНИЕ
на выполнение дипломного проекта

Студент Айтжанов Музафар Болатбекович
(фамилия, имя, отчество)

Тема проекта Разработка автоматизированного рабочего места специалиста отдела продаж

утверждена приказом ректора № ___ от «___» сентября 20___ г.
Срок сдачи законченной работы «___» _____ 20___ г.
Исходные данные к проекту требуемые параметры результатов проектирования (исследования) и исходные данные объекта
проанализировать деятельность предприятия „Компьютерный сервис SUPPORT", выбрать средства разработки, спроектировать структуру разрабатываемой системы, разработать пользовательский интерфейс.

Перечень подлежащих разработке дипломного проекта вопросов или краткое содержание дипломного проекта:
1) Исследование предметной области
2) Выбор среды реализации
3) Реализация проекта
4) Экономический анализ

Перечень графического материала (с точным указанием обязательных чертежей)

Проверка пользователя, Основное Меню, Продажа компьютеров, Список комплектующих, Ремонтируемые изделия

Рекомендуемая основная литература

Епанешников А.М., Епанешников В.А. Программирование в среде Turbo Pascal 7.0 М.: Диалог МИФИ, 1998.

Леонтьев В. Delphi 5. -М.: Олма-пресс, 1998.

Моисеев А. Object Pascal, СПб, 2000

Немнюгин С.А. Программирование, Питер, 1999.

Консультанты по проекту с указанием относящихся к ним разделов

Раздел	Консультант	Сроки	Подпись
Экономика	Ермеева	15.04-23.05.14	Ермеева
БЖД	Прихотько И.Г.	11.04-29.05.14	
Нормоконтроль	Турлов Д.М.	31.05.14г	
Каштанбердиева	Каштанбердиева Н.А.	24.05.14.	

Г Р А Ф И К
подготовки дипломного проекта

№ п/п	Наименование разделов, перечень разрабатываемых вопросов	Сроки представления руководителю	Примечание
1.	Исследование предметной области	29.03. 2014г.	
2.	Выбор среды реализации	10.04. 2014г.	
3.	Реализация проекта	18.04 2014г.	
4.	Экономический анализ	15.05. 2014г.	

Дата выдачи задания «____»_____20___ г.

Заведующий кафедрой _____ _____
(подпись) (Фамилия и инициалы)

Руководитель _____ Кожамбердиева М.И.
(подпись) (Фамилия и инициалы)

Задание принял к исполнению

А

ннотация

Циркулирующие потоки информации, которые нас окружают в мире, огромны. Во времени они имеют тенденцию к увеличению. Поэтому в любой организации, как большой, так и маленькой, возникает проблема такой организации управления данными, которая обеспечила бы наиболее эффективную работу.

В данном дипломном проекте разработано программное обеспечение для автоматизации рабочего места специалиста отдела продаж. Данное программное обеспечение, разработанное в среде Delphi7, позволяет достигнуть уменьшение времени необходимого для обработки информации и ускорить процесс обслуживания клиентов.

Также в проекте рассмотрен вопрос безопасности труда и рассчитана экономическая эффективность от внедрения программы.

Аңдатпа

Айналамызда таратылып жатқан ақпарат тасқындары тым алып . Уақыт өте олар көбейе береді. Сол себептен, кезкелген уйымда, үлкен немесе кіші болсын, тиімді жұмыстпен қамсыздандыратын деректерлермен басқару мәселесі туады.

Бұл дипломдық жобада сауда бөлімінің маманының жұмыс орнын автоматтандыратын бағдарламалық қамсыздандыру әзірленген. Delphi7 ортасында әзірлеген бұл бағдарламалық қамсыздандыру, ақпараттың өңдеу уақытын азайтуға және клиенттерге қызмет ету үдерісін жылдамдатуға ерік береді.

Сонымен қатар жобада еңбек қауіпсіздігі сұрағы қарастырылған және бағдарламаның енгізуінің экономикалық тиімділігі өлшелген.

Annotation

Recirculating flows of information that surround us in the world are enormous. In time they tend to increase. Therefore, in any organization as large and small, there is a problem of such an organization of data management that would ensure the most efficient operation.

In this thesis project developed software for workplace automation specialist sales team. This software developed in the medium Delphi7, can achieve reduction of the time required to process information and to accelerate the process of customer service.

Also in the project considered safety and economic efficiency is calculated from the implementation of the program.

Содержание

Введение

Современные масштабы и темпы внедрения средств автоматизации управления в народном хозяйстве с особой остротой ставят задачу проведения комплексных исследований, связанных со всесторонним изучением и обобщением возникающих при этом проблем как практического, так и теоретического характера.

В последние годы возникает концепция распределённых систем управления народным хозяйством, где предусматривается локальная обработка информации. Для реализации идеи распределённого управления необходимо создание для каждого уровня управления и каждой предметной области автоматизированных рабочих мест (АРМ) на базе профессиональных персональных ЭВМ.

На современном этапе автоматизации управления производством наиболее перспективным является автоматизация планово-управленческих функций на базе персональных ЭВМ, установленных непосредственно на рабочих местах специалистов. Совокупность АРМ и модулей сервисной поддержки составляет информационная система предприятия. Это позволит использовать систему людям, не имеющим специальных знаний в области программирования, и одновременно позволит дополнять систему по мере надобности.

Для каждого объекта управления нужно предусмотреть автоматизированные рабочие места, соответствующие их функциональному назначению. Однако принципы создания АРМ должны быть общими: системность, гибкость, устойчивость, эффективность.

Согласно принципу системности АРМ следует рассматривать как системы, структура которых определяется функциональным назначением.

Принцип гибкости означает приспособляемость системы к возможным перестройкам благодаря модульности построения всех подсистем и стандартизации их элементов.

Принцип устойчивости заключается в том, что система АРМ должна выполнять основные функции независимо от воздействия на неё внутренних и внешних возможных факторов. Это значит, что неполадки в отдельных её частях должны быть легко устранимы, а работоспособность системы быстро восстановима.

Эффективность АРМ следует рассматривать как интегральный показатель уровня реализации приведённых выше принципов, отнесённого к затратам по созданию и эксплуатации системы.

Функционирование АРМ может дать численный эффект только при условии правильного распределение функций и нагрузки между человеком и машинными средствами обработки информации, ядром которых является ЭВМ. Лишь тогда АРМ станет средством повышения не только производительности

труда и эффективности управления, но и социальной комфортности специалистов.

Практически любая деятельность современного делового человека связана с обработкой информации. И тот, кто может качественно обработать существенный объем информации за наименьшее время, имеет наибольшие гарантии на успех. Вот почему проблема создания различных вспомогательных средств и методов оперирования со всевозможной информацией становится все насущнее.

Основное преимущество автоматизации - сокращение избыточности хранимых данных, а следовательно, экономия объема используемой памяти, уменьшение затрат на многократные операции обновления избыточных копий, увеличение степени достоверности информации и увеличение скорости обработки информации, сокращение количества внутренних промежуточных документов, журналов, папок и т.д. Значительно сокращает время автоматического поиска информации, который производится из специальных экранных форм, в которых указываются параметры поиска объекта.

Основной задачей данной проектируемой системы является улучшение учета и оперативное регулирование складских операций, подготовки стандартных документов для внутренней среды (отчетов, накладных, счетов-фактур).

Целью дипломной работы является разработка информационной системы для предприятия "Компьютерный сервис SUPPORT".

Для достижения цели необходимо решить следующие задачи:

1) проанализировать деятельность предприятия "Компьютерный сервис SUPPORT";
2) изучить функциональные обязанности диспетчера и сущность процесса учета и контроля заявок;
3) выбрать средства разработки;
4) спроектировать структуру разрабатываемой системы;
5) разработать пользовательский интерфейс;
6) реализовать и протестировать на данных конкретного примера;
7) разработать соответствующую документацию к информационной системе;
8) рассчитать экономический эффект от внедрения.

1 Исследование предметной области

1.1 Характеристика деятельности предприятия

Предприятие "Компьютерный сервис SUPPORT" - компания, образованная в 2010 году, утвердившаяся на региональном рынке как поставщик компьютерной и копировальной техники, выполняющий так же ремонт и сборку компьютеров и установку программного обеспечение. Специалисты имеют большой опыт работы с компьютерной техникой различных производителей. Предприятие "Компьютерный сервис SUPPORT" на протяжении 4 лет работы на рынке завоевало репутацию всегда выполняющего свои гарантийные обязательства перед клиентами.

Высокое качество и надежность компьютеров компании "SUPPORT" гарантированы благодаря многоступенчатой системе проверки качества на каждом этапе производственного процесса. В сборочном цеху компании работают квалифицированные инженеры и технические специалисты. Сборка производится из высококачественных, прошедших тестирование, комплектующих.

Компания предлагает широкий выбор высококачественной продукции известных брэндов, таких как Acer, Asus, Benq, Creative, D-Link, Elitegroup, Inno Vision, Hewlett-Packard, LG, Liteon, U.S. Robotics, Samsung, Segate, Sony, Western Digital, Zyxel и многих других.

Основные направления деятельности:

1) поставка компьютерной и копировальной техники;
2) модернизация и ремонт компьютерной техники;
3) поставка программного обеспечения Microsoft, лаборатории Касперского;
4) установка программного обеспечения Microsoft, лаборатории Касперского;
5) поставка расходных материалов для компьютерной и копировальной техники;
6) ремонт, гарантийное и сервисное обслуживание компьютерной и копировальной техники;
7) заправка всех видов картриджей;
8) поставка ноутбуков и ПК;
9) поставка цифровых фотоаппаратов и аксессуаров к ним;
10) поставка DVD-плееров, DVD-театров и акустических систем BBK;
11) поставка активного сетевого оборудования;
12) подключение к сети Интернет;
13) создания почтового электронного ящика.

Компания работает во всех областях современных информационных технологий и ее службы готовы решать широкий спектр задач:

1) консультация, проектирование;
2) внедрение комплексных систем автоматизации предприятия;
3) максимальное удовлетворение запросов клиентов;
4) достижение высокого уровня обслуживания;
5) укрепление престижа компании на освоенном рынке;
6) поставка персональных компьютеров и ноутбуков;
7) поставка любого периферийного и офисного оборудования.

Организационная структура предприятия представлена на рисунке 1.1.

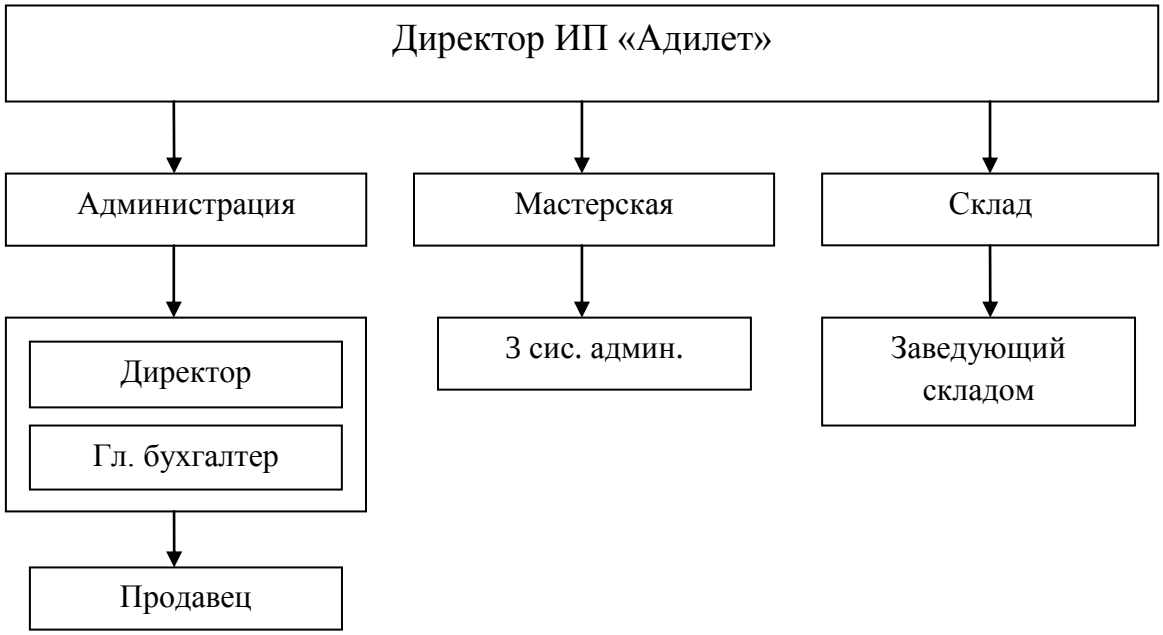

Рисунок 1.1 - Организационная структура предприятия

Клиент приходит в головной офис для заказа того или иного устройства или ПК на свой выбор и по цене, которая его устраивает. Делает заказ. С головного офиса отправляют накладная в мастерскую, где обозначены какие устройства или комплектующие надо заказчику. В мастерской отправляют накладную для получения того или иного устройства или комплектующих на складе. При получении документа на складе вносят в базу данные для учета, записывают серийные номера и на какое время дается гарантия для обслуживания тех или иных устройств, которая заносится в гарантийный талон. Затем этот документ с устройством или комплектующими отправляют обратно в мастерскую.

Если надо собрать ПК, то его собирают в мастерской и устанавливают ПО. После этого отправляют устройства или готовый ПК в офис. В головном офисе заносят номер гарантийного талона для бухгалтерского учета и ставят печать на документе. Клиент рассчитывается в кассе и забирает ПК или другое

устройство с гарантией. В течении срока, который указан в гарантийном талоне, клиент может обслуживаться при предъявлении данного документа.

На каждом предприятии возникают различные проблемы с оборудованием или с обслуживанием клиентов. Для этого выявляют проблему и решают разными методами.

Менеджер по продажам подчиняется директору фирмы. Менеджер по продажам продает товар и все проданное записывается в программу 1С предприятия.

В ИП "Адилет" имеются ряд различных проблем:

1) все данные хранятся в программе 1С Предприятие 7.7, которая слишком громоздкая в силу своей многофункциональности и клиент не имеет доступа к этой программе;

2) список комплектующих и периферийных устройств и др. находится в прайс-листе. Менеджеру по продажам приходится очень тяжело: обслуживать несколько клиентов сразу и объяснять им об комплектующих их характеристику, т.к. в прайс-листе не содержится никакого описания о том или ином товаре;

3) оставленные на ремонт компьютеры регистрируются вручную: выписывается квитанция, в которой указывается проблема "поломка", номер квитанции, телефон и имя клиента, делается две копии квитанции: одна отдается клиенту, другая приклеивается к оставленному ПК или другому оборудованию. Также информация о клиенте и о заказе записывается в тетрадь.

1.2 Формулировка задачи в общем виде

Постановка задачи - это документ, в котором сформулированы основные цели, разработки требования программному продукту определены сроки и этапы разработки. В разработки технического задания участвуют как представители заказчика, так и представители исполнителя. Основные факторы, определяющие характеристики разрабатываемого программного обеспечения:

1) Назначение разрабатываемой системы.
2) Периодичность решения задач.
3) Входные и выходные документы.
4) Сама задача.
5) Ограничения.

В Республике Казахстан в основном все компьютерные фирмы подобного профиля для регистрации комплектующих пользуются специально разработанными программами, где удобно вести заполнения, поиск, подсчеты, заполнение накладной и т.д. Также пользуются специально предназначенной программой 1С.

Для решения выше описанных проблем необходимо разработать программу "Разработка автоматизированного рабочего места специалиста отдела продаж", которая будет выполнять следующие функции:

1) выводить на экран цену и количество товара на складе;
2) осуществлять поиск определенного устройства или комплектующих через программу;
3) вводить в базу определенный товар его сумму и количество;
4) рассчитывать сумму заказа;
5) выдавать сообщение о дате, когда закончится определенный товар на складе;
6) сохранять накладную на сборку компьютера;
7) записывать список услуг, сделанных клиенту и их стоимость;
8) печатать накладную на сборку компьютера;
9) осуществлять печать квитанции на приобретенный товар;
10) печатать гарантийный талон на приобретенный товар;
11) выдавать квитанцию на компьютер, оставленный клиентом для ремонта;
12) сохранять адрес клиента, при покупке компьютера, с целью доставки.

1.3 Формы документов, используемых для ввода исходных данных

В программе используются входные документы для регистрации в базу данных данные о продукции, которая поступает на склад ИП "Адилет":

1) Приходная накладная (в ней указывается наименование организации, реализующей товар, наименование склада, номер документа, дата составления, основание на котором приобретен товар, наименование приобретаемого товара, срок гарантии количество и общая сумма).
2) Удостоверение личности.

1.4 Формы выходных документов, которые формируются при работе программы

1) Накладная на сборку компьютера.
2) Гарантийный талон.
3) Расходная накладная.
4) Квитанция на ремонт.

2 Выбор среды реализации

2.1 Обзор программных средств

2.1.1 C++ Builder

C++ Builder - программный продукт, инструмент быстрой разработки приложений (RAD), интегрированная среда программирования (IDE), система, используемая программистами для разработки программного обеспечения на языке C++.

C++ Builder представляет собой SDI-приложение, главное окно которого содержит настраиваемую инструментальную панель (слева) и палитру компонентов (справа). Помимо этого, по умолчанию при запуске C++ Builder появляются окно инспектора объектов и форма нового приложения (Рисунок 2.1). Под окном формы приложения находится окно редактора кода.

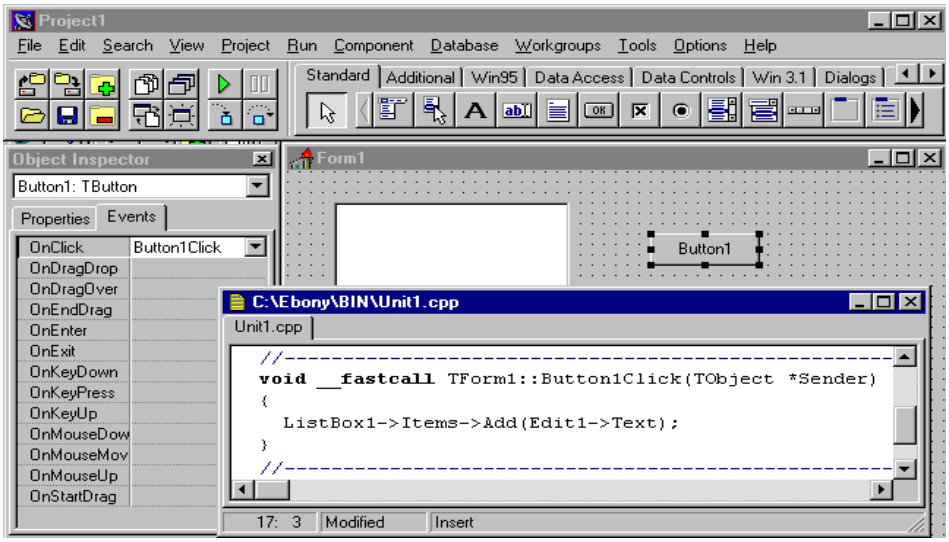

Рисунок 2.1 - Среда разработки C++ Builder

Формы являются основой приложений C++ Builder. Создание пользовательского интерфейса приложения заключается в добавлении в окно формы элементов объектов C++ Builder, называемых компонентами. Компоненты C++ Builder располагаются на палитре компонентов, выполненной в виде многостраничного блокнота. Важная особенность C++ Builder состоит в том, что он позволяет создавать собственные компоненты и настраивать палитру компонентов, а также создавать различные версии палитры компонентов для разных проектов.

Компоненты C++ Builder

Компоненты разделяются на видимые (визуальные) и невидимые (невизуальные). Визуальные компоненты появляются во время выполнения точно так же, как и во время проектирования (Рисунок 2.2). Примерами являются кнопки и редактируемые поля. Невизуальные компоненты появляются во время проектирования как пиктограммы на форме. Они никогда не видны во время выполнения, но обладают определенной функциональностью (например, обеспечивают доступ к данным, вызывают стандартные диалоги Windows 95 и др.).

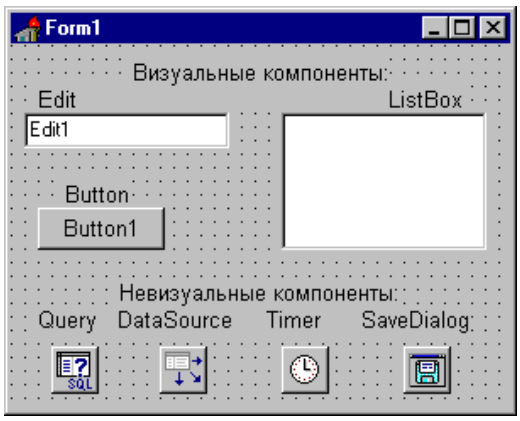

Рисунок 2.2 - Пример использования видимых и невидимых компонентов

Для добавления компонента в форму можно выбрать мышью нужный компонент в палитре и щелкнуть левой клавишей мыши в нужном месте проектируемой формы. Компонент появится на форме, и далее его можно перемещать, менять размеры и другие характеристики.

Каждый компонент C++Builder имеет три разновидности характеристик: свойства, события и методы.

Если выбрать компонент из палитры и добавить его к форме, инспектор объектов автоматически покажет свойства и события, которые могут быть использованы с этим компонентом. В верхней части инспектора объектов имеется выпадающий список, позволяющий выбирать нужный объект из имеющихся на форме.

2.1.2 Visual studio

Microsoft Visual Studio - линейка продуктов компании Майкрософт, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств.

Основным языком Microsoft Visual Studio является Visual Basic .NET - это объектно-ориентированный язык программирования, который можно рассматривать как очередной виток эволюции Visual Basic, реализованный на платформе Microsoft .NET.

Microsoft Visual Studio поддерживает работу с приложениями Microsoft Office, интеграцию в них кода программ.

2.1.3 Delphi 7

Borland Delphi - интегрированная среда разработки ПО для Microsoft Windows на языке Delphi (ранее носившем название Object Pascal).

Delphi позволяет разрабатывать приложения быстрым процессом лишь за счет средств визуализации, ибо визуальное программирование как бы добавляет новое измерение при создании приложений, давая возможность изображать эти объекты на экране монитора до выполнения самой программы. Без визуального программирования процесс отображения требует написания фрагмента кода, создающего и настраивающего объект "по месту". Увидеть закодированные объекты было возможно только в ходе исполнения программы. При таком подходе достижение того, чтобы объекты выглядели и вели себя заданным образом, становится утомительным процессом, который требует неоднократных исправлений программного кода с последующей прогонкой программы и наблюдения за тем, что в итоге получилось.

Благодаря средствам визуальной разработки можно работать с объектами, держа их перед глазами и получая результаты практически сразу. Способность видеть объекты такими, какими они появляются в ходе исполнения программы, снимает необходимость проведения множества операций вручную, что характерно для работы в среде, не обладающей визуальными средствами, вне зависимости от того, является она объектно-ориентированной или нет. После того, как объект помещен в форму среды визуального программирования, все его атрибуты сразу отображаются в виде кода, который соответствует объекту как единице, исполняемой в ходе работы программы.

Размещение объектов в Delphi связано с более тесными отношениями между объектами и реальным программным кодом. Объекты помещаются в вашу форму, при этом код, отвечающий объектам, автоматически записывается в исходный файл. Этот код компилируется, обеспечивая существенно более высокую производительность, чем визуальная среда, которая интерпретирует информацию лишь в ходе исполнения программы.

Borland Delphi позволяет добавлять к окнам поля ввода, меню, командные кнопки, переключатели, флажки, списки, линейки прокрутки, а также диалоговые окна для выбора файла или каталога. Программист может использовать сетку для обработки табличных данных, организовать взаимодействие с другими приложениями Windows и доступ к базам данных. Borland Delphi такие компоненты обычно называют элементами управления.

Замечательным достоинством системы является и то, что размещение компонентов на экране, а также задание начальных значений их свойств (размеры, цвет, вид и др.) Delphi позволяет осуществлять на этапе конструирования формы без написания какой-либо программы.

Для этой цели предусмотрено специальное окно, называемое Инспектором объектов, в котором перечислены все доступные в режиме проектирования свойства выделенного компонента и их текущие значения.

Изменение свойства какого-либо объекта незамедлительно отразится на внешнем виде и коде программы. Это позволяет, уже до запуска программы видеть как будет выглядеть проектируемая форма. Такой способ работы с объектами, имеющими графическое представление, принято называть объектно-ориентированным программированием.

Преимущества Delphi по сравнению с аналогичными программными продуктами:

1) быстрота разработки приложения;
2) высокая производительность разработанного приложения;
3) низкие требования разработанного приложения к ресурсам компьютера;
4) наращиваемость за счет встраивания новых компонент и инструментов в среду Delphi;
5) возможность разработки новых компонент и инструментов собственными средствами Delphi (существующие компоненты и инструменты доступны в исходных кодах);
6) удачная проработка иерархии объектов.

Требования к интерфейсу Windows-приложения

Под графическим интерфейсом пользователя (Graphical User Interface - GUI) подразумевается тип экранного представления, при котором пользователь может выбирать команды, запускать задачи и просматривать списки файлов, указывая на пиктограммы или пункты в списках меню, показанных на экране. Действия могут, как правило, выполняться с помощью мыши, либо нажатием клавиш на клавиатуре.

Delphi предоставляет разработчику приложения с широкими возможностями быстрого и качественного проектирования графического интерфейса пользователя - различных окон, кнопок, меню и т.д. Есть определенные принципы построения графического интерфейса пользователя, и пренебрегающий ими обречен на то, что его приложение будет выглядеть чужеродным объектом в среде Windows.

Для пользователя одним из принципиальных преимуществ работы с Windows является то, что большинство имеющихся приложений выглядят и ведут себя сходным образом.

Чаще всего сколько-нибудь сложное приложение не может ограничиться одним окном. Поэтому нужно решить вопрос управления окнами. Есть две различные модели приложений: с интерфейсом одного документа (SDI) и с интерфейсом множества документов (MDI).

В большинстве случаев следует отдавать предпочтение интерфейсу SDI. Этот интерфейс не обязательно предполагает наличие действительно только

одного окна, как в приложениях Windows, типа "Калькулятор". Такое приложение, как "Проводник" Windows, также является SDI приложением, но в нужные моменты оно создает вторичные окна для поиска файлов или папок, задания параметров, просмотра свойств файлов и других целей.

Основным элементом любого приложения является форма - контейнер, в котором размещаются другие визуальные и не визуальные компоненты. С точки зрения пользователя форма - это окно, в котором он работает с приложением.

К внешнему виду окон в Windows предъявляются определенные требования. Delphi автоматически обеспечивает стандартный для Windows вид окон приложения. Надо продумать и указать, какие кнопки в полосе системного меню должны быть доступны в том или ином окне, должно ли окно допускать изменение пользователем его размеров, каким должен быть заголовок окна. Все эти характеристики окон обеспечиваются установкой и управлением свойствами формы.

Цвет является мощным средством воздействия на человека. Неудачное цветовое решение может приводить к быстрому утомлению пользователя, работающего с вашим приложением, к рассеиванию его внимания, к частым ошибкам. Слишком яркий или неподходящий цвет может отвлекать внимание пользователя или вводить его в заблуждение, создавать трудности в работе. А удачно подобранная гамма цветов, осмысленные цветовые акценты снижают утомляемость, сосредоточивают внимание пользователя на выполняемых в данный момент операциях, повышают эффективность работы. Цвет может также связываться с различными состояниями объектов.

Цвет не должен использоваться в качестве основного средства передачи информации. Можно использовать различные панели, формы, штриховку и другие методики выделения областей экрана. Microsoft даже рекомендует разрабатывать приложение сначала в черно-белом варианте, а уже потом добавлять к нему цвет.

Использование шрифтов по умолчанию: System или MS Sans Serif, чаще всего позволяет избежать неприятностей. Если использовать для надписей русские тексты, то при запуске приложения на компьютере с нерусифицированным Windows иногда возможны неприятности. Для подобных случаев все-таки полезно приложить файлы использованных шрифтов к программе.

Другой выход из положения - ввести в приложение команду выбора шрифта пользователем. Это позволит ему выбрать подходящий шрифт из имеющихся в его системе. Проведенную пользователем установку можно запоминать в файле .INI, в реестре или в файле конфигурации и читать автоматически информацию из этого файла при каждом запуске приложения.

Практически любое приложение должно иметь меню, поскольку именно меню дает наиболее удобный доступ к функциям программы. Существует несколько различных типов меню: главное меню с выпадающими списками

разделов, каскадные меню, в которых разделу первичного меню ставится в соответствие список подразделов, и всплывающие или контекстные меню, появляющиеся, если пользователь щелкает правой кнопкой мыши на каком-то компоненте.

Основное требование к меню - их стандартизация. Это требование относится ко многим аспектам меню: месту размещения заголовков меню и их разделов, форме самих заголовков, клавишам быстрого доступа, организации каскадных меню. Цель стандартизации - облегчить пользователю работу с приложением. Надо, чтобы пользователю не приходилось думать, в каком меню и как ему надо открыть или сохранить файл, как ему получить справку, как работать с буфером обмена Clipboard и т.д. Для осуществления всех этих операций у пользователя, поработавшего хотя бы с несколькими приложениями Windows, вырабатывается стойкий автоматизм действий и недопустимо этот автоматизм ломать.

Рассмотрение требований с размещения заголовков меню. Состав меню зависит от конкретного приложения. Но размещение общепринятых разделов должно быть стандартизированным. Все пользователи уже привыкли, что меню Файл размещается слева в полосе главного меню, раздел справки - справа, перед ним в приложениях MDI размещается меню Окно и т.д. Главное меню должно также снабжаться инструментальной панелью, быстрые кнопки, которые дублируют наиболее часто используемые команды меню. На этих кнопках надо использовать, по возможности, привычные картинки.

По возможности стандартным должно быть и расположение разделов в выпадающих меню.

Группы функционально связанных разделов отделяются в выпадающих меню разделителями.

Названия разделов меню должны быть привычными пользователю. Названия должны быть краткими и понятными. Названия разделов должны начинаться с заглавной буквы.

Названия разделов меню, связанных с вызовом диалоговых окон, должны заканчиваться многоточием, показывающим пользователю, что при выборе этого раздела ему предстоит установить в диалоге еще какие-то параметры.

Разделы, к которым относятся каскадные меню должны заканчиваться стрелкой, указывающей на наличие дочернего меню данного раздела.

В каждом названии раздела должен быть выделен подчеркиванием символ, соответствующий клавише быстрого доступа к разделу (клавиша Alt плюс подчеркнутый символ). Хотя вряд ли такими клавишами часто пользуются, но традиция указания таких клавиш незыблема.

Каждое окно, введенное в приложение, должно быть тщательно продумано и скомпоновано. Удачная компоновка может стимулировать эффективную работу пользователя, а неудачная - рассеивать внимание, отвлекать, заставлять тратить лишнее время на поиск нужной кнопки или индикатора.

Управляющие элементы и функционально связанные с ними компоненты экрана должны быть зрительно объединены в группы, заголовки которых коротко и четко поясняют их назначение.

Каждое окно должно иметь некоторую центральную тему, которой подчиняется его композиция. Пользователь должен понимать, для чего предназначено данное окно и что в нем наиболее важно. При этом недопустимо перегружать окно большим числом органов управления, ввода и отображения информации. В окне должно отображаться главное, а все детали и дополнительную информацию можно отнести на вспомогательные окна.

Помогают также разгрузить окно многостраничные компоненты с закладками. Они дают возможность пользователю легко переключаться между разными по тематике страницами, на каждой из которых имеется необходимый минимум информации.

При проектировании приложения важно правильно определить последовательность табуляции оконных компонентов. Под этим понимается последовательность, в которой переключается фокус с компонента на компонент, когда пользователь нажимает клавишу табуляции Tab. Это важно, поскольку в ряде случаев пользователю удобнее работать не с мышью, а с клавиатурой. Пусть, например, вводя данные о каком-то объекте, пользователь должен в отдельных окнах редактирования указать наименование, дату изготовления, страну изготовителя и так далее. Конечно, набрав наименование, ему удобнее нажать клавишу Tab и набирать дату изготовления, а потом опять, нажав Tab, набирать следующий реквизит, чем каждый раз отрываться от клавиатуры, хватать мышь и переключаться в новое окно редактирования.

Приложение должно предельно облегчать работу пользователя, снабжая его системой подсказок, помогающих сориентироваться в приложении. Эта система включает в себя:

1) ярлычки, которые всплывают, когда пользователь задержит курсор мыши над каким-то элементом окна приложения. В частности, такими ярлычками обязательно должны снабжаться быстрые кнопки инструментальных панелей, поскольку нанесенные на них пиктограммы часто не настолько выразительны, чтобы пользователь без дополнительной подсказки мог понять их назначение;

2) более развернутые подсказки в панели состояния или в другом отведенном под это месте экрана, которые появляются при перемещении курсора мыши в ту или иную область окна приложения;

3) встроенную систему контекстно-зависимой оперативной справки, вызываемую по клавише F1;

4) раздел меню Справка, позволяющий пользователю открыть стандартный файл справки Windows.hlp, содержащий в виде гипертекста развернутую информацию по интересующим пользователя вопросам.

При работе программы могут возникать различного рода ошибки: переполнение, деление на нуль, попытка открыть несуществующий файл и т.п.

При возникновении таких исключительных ситуаций, программа генерирует так называемое исключение, а выполнение дальнейших вычислений в данном блоке прекращается. Исключение - это объект специального вида, характеризующий возникшую в программе исключительную ситуацию. Он может также содержать в виде параметров некоторую уточняющую информацию. Особенностью исключений является то, что это сугубо временные объекты. Как только они обработаны каким-то обработчиком, они разрушаются.

Программист должен принять все мыслимые меры, чтобы ни при каких ошибках пользователя и ни при каких сочетаниях данных приложение не заканчивалось бы аварийно. Но если все-таки аварийное завершение происходит, необходима полная зачистка "мусора" - удаление временных файлов, освобождение памяти, разрыв связей с базами данных и т.д.

Выбор языка программирования обусловлен следующими причинами:

1) устраняется необходимость в повторном вводе данных;
2) обеспечивается согласованность проекта и его реализации;
3) увеличивается производительность разработки и переносимость программ.

Интерфейс Borland Delphi несложен. Ниже приведено краткое описание стандартных элементов Borland Delphi.

При открытии программы, открываются четыре окна Delphi. Вверху, на рисунке 2.3, во всю ширину экрана окно управления проектом и средой разработки - главное окно Delphi.

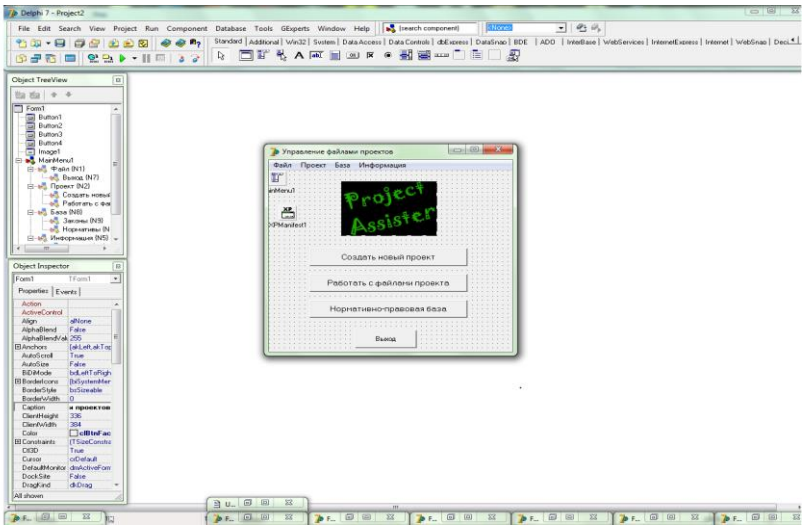

Рисунок 2.3 - Интерфейс Borland Delphi

При его сворачивании сворачиваются и все остальные. Слева - Инспектор объектов. В нём задаются свойства составляющих нашу программу компонентов. И наконец, в центре одно над другим два окна Delphi, окно формы будущей программы и окно программной начинки. Прямо перед нами -

окно, которое в Delphi называется Форма. Именно Форма является визуальным прообразом нашей будущей программы.

В пустой форме смысла нет. Наполнить её содержанием помогут компоненты Delphi. Они располагаются на соответствующих вкладках палитры компонентов на главном окне. Все основные компоненты Delphi находятся на первых четырёх вкладках, отмеченных на рисунке 2.4. Эти вкладки: Standard, Additional, Win32, System.

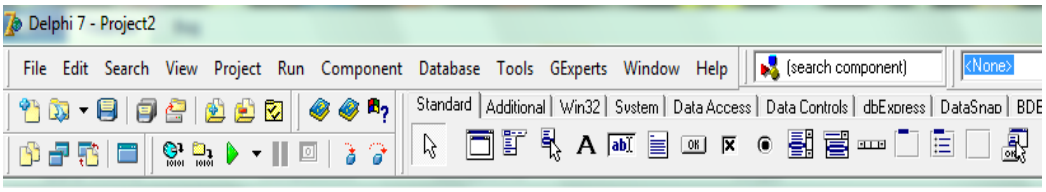

Рисунок 2.4 - Вкладки

Их названия всплывают в виде подсказок при наведении мышки на пиктограммы. Чтобы перенести компонент на форму, нужно щёлкнуть его мышкой (при этом выбранный компонент выделяется), и затем щёлкнуть в том месте формы, где его предполагается разместить. В дальнейшем компоненты можно свободно перетаскивать по форме мышкой, "конструируя" нужный интерфейс. Если, щёлкнув по компоненту, вы захотите отказаться от его переноса на форму, щёлкните по стрелке, расположенной слева на вкладке Standard. Выделение компонента снимется.

При "конструировании" формы в Инспекторе объектов можно придать свойствам компонентов любые значения. Случайно можно задать такие размеры или координаты, что компонент "исчезнет" - сделается невидимым или спрячется за другими компонентами. Его легко найти с помощью того же Инспектора Объектов - все компоненты, находящиеся на Форме, перечислены в его выпадающем списке. Достаточно выбрать нужный компонент, и он или его контуры (если он заслонён другими компонентами) появятся на Форме, а в Инспекторе объектов - его свойства.

Операционная система Windows - многозадачная, то есть несколько программ в ней могут функционировать одновременно. Когда, например, пользователь щёлкает по кнопке в окне программы, система Windows определяет, что произошло событие именно в этой программе, и посылает ей сообщение об этом. Программа должна соответствующим образом отреагировать на него. Для этого программист, должен написать код-обработчик этого события. Таким образом, структура программы для Windows представляет собой набор подпрограмм, каждая из которых ответственна за обработку конкретного события и вызывается только при его возникновении. Удобство Delphi состоит в том, что программисты избавлены от необходимости получать сообщения от Windows сами, Delphi это делает сама. Каждый компонент имеет впечатляющий набор событий, на которые он может

реагировать. Программист сам определяет, какие события в программе требуется обрабатывать. На рисунках 2.5 и 2.6 изображены окно обработчика событий и программный код.

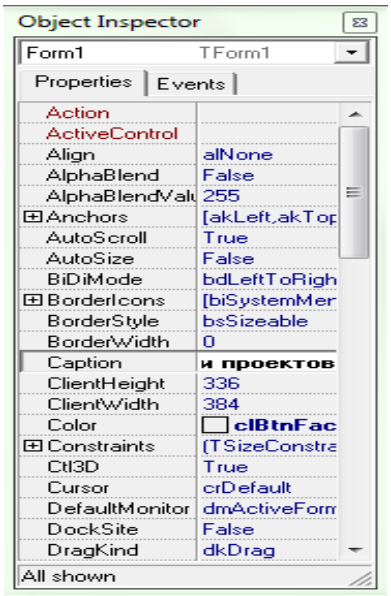

Рисунок 2.5 - Окно обработчика событий

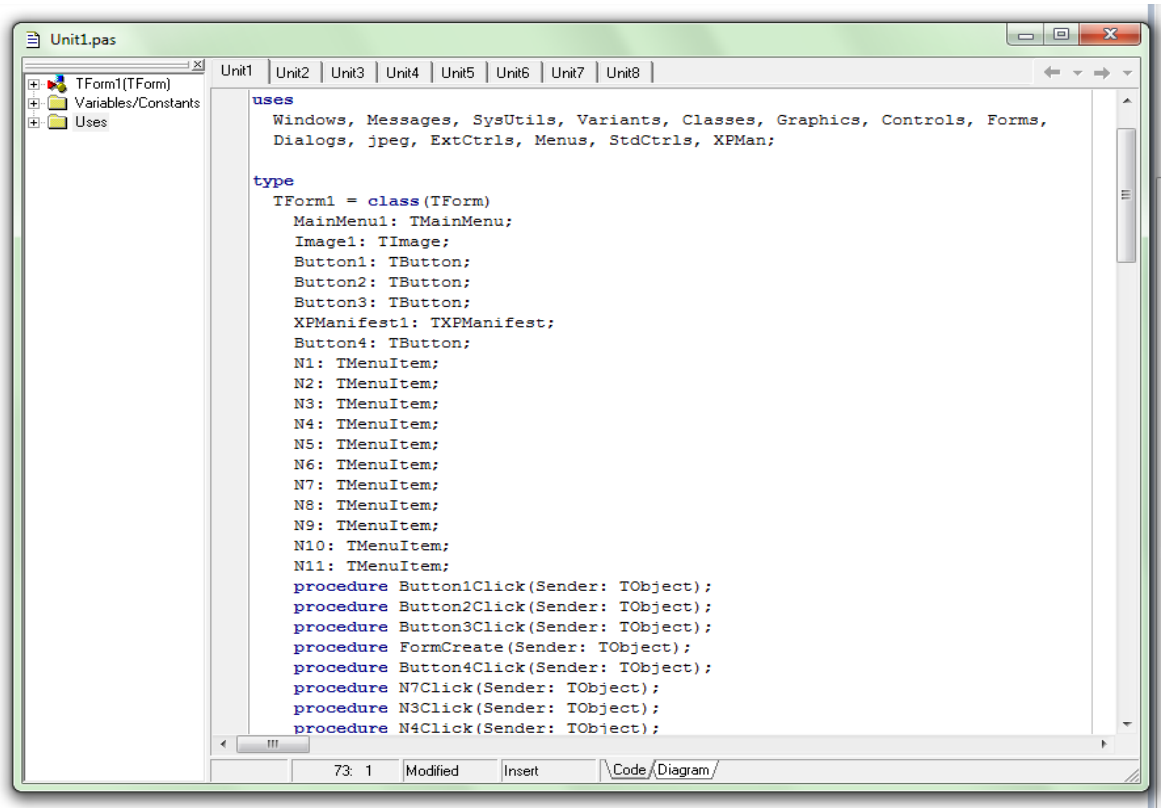

Рисунок 2.6 - Программный код

2.1.4 Microsoft Office Access

Системы управления базами данных (СУБД) - это программные средства, с помощью которых можно создавать базы данных, наполнять их и работать с ними.

В мире существует немало различных систем управления базами данных. Многие из них на самом деле являются не законченными продуктами, а специализированными языками программирования, с помощью которых каждый, освоивший данный язык, может сам создавать такие структуры, какие ему удобны, и вводить в них необходимые элементы управления. К подобным языкам относятся Clipper, Paradox, FoxPro и другие.

Положение изменилось с появлением в составе пакета Microsoft Office системы управления базами данных Access.

Достоинством Access является интегрированность этой программы с Excel, Word и другими программами пакета Office.

Данные, созданные в разных приложениях, входящих в этот пакет, легко импортируются и экспортируются из одного приложения в другое.

Access - это, прежде всего, система управления базами данных (СУБД). Как и другие продукты этой категории, она предназначена для хранения и поиска данных, представления информации в удобном виде и автоматизации часто повторяющихся операций (таких, как ведение счетов, учет, планирование и т.п.). С помощью Access можно разрабатывать простые и удобные формы ввода данных, а также осуществлять обработку данных и выдачу сложных отчетов.

Access - мощное приложение Windows; впервые производительность СУБД органично сочетается с теми удобствами, которые имеются в распоряжении пользователей Microsoft Windows. Поскольку оба эти продукта - детища компании Microsoft, они прекрасно взаимодействуют между собой. Система Access работает под управлением Windows, так что при работе с ней пользователю доступны все преимущества Windows. Можно вырезать, копировать и вставлять данные из любого приложения Windows в Access и наоборот; можно создать проект формы в Access и вставить его в конструктор форм.

С помощью объектов OLE (Object Linking and Embedding - связывание и внедрение объектов) в Windows и компонентах Microsoft Office (Excel, Word, PowerPoint и Outlook) можно превратить Access в настоящую операционную среду баз данных. С помощью новых расширений для Internet можно создавать формы, которые будут напрямую взаимодействовать с данными из World Wide Web, и транслировать их в представление на языке HTML, обеспечивающее работу с такими продуктами, как Internet Explorer и Netscape Navigator.

При всем этом Access - не просто СУБД. Как реляционная СУБД Access обеспечивает доступ ко всем типам данных и позволяет использовать одновременно несколько таблиц базы данных. При этом можно существенно

упростить структуру данных, облегчая тем самым выполнение поставленных задач. Таблицу Access можно связать с данными, хранящимися на большой ЭВМ или на сервере. С другой стороны, можно использовать таблицы, созданные в среде Paradox или dBASE. Полученные результаты можно быстро и легко связать и объединить с данными из электронных таблиц Excel. Работая в среде Microsoft Office, пользователь получает в свое распоряжение полностью совместимые между собой Access и Word, Excel и PowerPoint.

Система Access - это набор инструментов конечного пользователя для управления базами данных. В ее состав входят конструкторы таблиц, форм, запросов и отчетов. Эту систему можно рассматривать и как среду разработки приложений. Используя макросы или модули для автоматизации решения задач, можно создавать ориентированные на пользователя приложения такими же мощными, как и приложения, написанные непосредственно на языках программирования. При этом они будут включать кнопки, меню и диалоговые окна. Программируя на языке VBA, можно создавать такие мощные программы, как сама система Access. По сути дела, многие инструментальные средства Access (например, мастера и конструкторы) написаны именно на VBA.

Мощность и доступность Access делают эту систему лучшей СУБД из представленных сегодня на рынке. Сначала познакомимся с Access на уровне конечного пользователя. Затем перейдем к более сложным элементам таким как элементы программирования на VBA и взаимодействия с Internet.

В Access в полной мере реализовано управление реляционными базами данных. Система поддерживает первичные и внешние ключи и обеспечивает целостность данных на уровне ядра (что предотвращает несовместимые операции обновления или удаления данных). Кроме того, таблицы в Access снабжены средствами проверки допустимости данных, предотвращающими некорректный ввод вне зависимости от того, как он осуществляется, а каждое поле таблицы имеет свой формат и стандартные описания, что существенно облегчает ввод данных. Access поддерживает все необходимые типы полей, в том числе текстовый, числовой, счетчик, денежный, дата/время, MEMO, логический, гиперссылка и поля объектов OLE. Если в процессе специальной обработки в полях не оказывается никаких значений, система обеспечивает полную поддержку пустых значений.

Реляционная обработка данных в Access за счет гибкой архитектуры системы способна удовлетворить любые потребности. При этом Access может использоваться как автономная СУБД в режиме файл-сервера или клиентского компонента таких продуктов, как SQL Server. Кроме того. Access поддерживает протокол ODBC (Open Database Connectivity), что позволяет подключаться к базам данных множества различных форматов, таких как SQL Server, Oracle, Sybase и даже DB/2 для больших ЭВМ фирмы IBM.

Система Access поддерживает обработку транзакций с гарантией их целостности. Кроме того, предусмотрена защита на уровне пользователя, что

позволяет контролировать доступ к данным отдельных пользователей и целых групп.

Контекстно-зависимая справка и Office Assistant

Организация справочной системы фирмы Microsoft - по-прежнему лучшая в отрасли, причем как для новичков, так и для опытных пользователей. В Access предусмотрена контекстно-зависимая справка, для получения которой достаточно нажать <F1>, и на экране сразу же появится справочная информация по тому вопросу, который интересует пользователя в текущий момент. При этом можно легко перейти к оглавлению справочной системы, конкретной информации, журналу предыдущих обращений и закладкам.

В Access компания Microsoft ввела Office Assistant (ассистента) и Screen Tips (всплывающие подсказки). Всплывающие подсказки содержат короткие объяснения чего-либо. В системе предусмотрена целая галерея из десяти возможных персонажей ассистентов (выбрать можно любого во своему вкусу). Если ассистент вам надоест, можете его отключить.

Простые в использовании мастера и конструкторы

Мастер (Wizard) может превратить часы работы в считанные минуты. Мастера задают наводящие вопросы относительно содержания, стиля и формата создаваемого объекта; затем они автоматически строят нужный объект. В составе Access около ста мастеров, помогающих конструировать базы данных, приложения, таблицы, формы, отчеты, диаграммы, почтовые наклейки, элементы управления и свойства. Допускается даже настройка мастеров для решения разных задач.

Импортирование, экспортирование и связывание внешних файлов

Access позволяет импортировать и экспортировать файлы многих известных форматов, включая dBASE, FoxPro, Excel, SQL Server, Oracle, Btrieve, многие текстовые форматы ASCII (в том числе с фиксированной длиной строки или заданным ограничителем), а также данные в формате HTML. В результате импортирования создается таблица Access; в результате экспортирования таблицы Access создается файл в заданном формате.

Связывание (ранее именовавшееся присоединением) означает, что можно использовать внешние данные без создания таблицы Access. Можно устанавливать подобную связь с данными dBASE, FoxPro, Excel, ASCII и SQL. Очень мощная возможность - связывание таблиц Access с их внешними таблицами с последующим совместным использованием; это относится к таблицам Access, dBASE, FoxPro и SQL Server.

Формы и отчеты WYSIWYG

Окна конструкторов форм и отчетов имеют одинаковый интерфейс и предоставляют пользователю много возможностей. Форма или отчет

конструируется по принципу WYSIWYG (What You See Is What You Get - что видишь, то и получишь). Добавляя очередной элемент управления, пользователь видит, как при этом изменяется создаваемая форма.

В формы и отчеты можно включать надписи, поля текстовых данных, переключатели, флажки, линии и прямоугольники, а также оформлять их, выделяя элементы цветом и тенью. Более того, можно включать целые рисунки, диаграммы, подформы и подотчеты. При этом все параметры представления данных остаются полностью подконтрольными пользователю. Формы могут занимать много страниц, а в отчетах может быть предусмотрено много уровней группировки данных и подведения итогов.

Формы и отчеты можно просматривать в режиме предварительного просмотра, обеспечивая взгляд "с высоты птичьего полета" путем изменения масштаба. В режиме конструирования отчет можно просматривать с фиктивными данными, чтобы не дожидаться обработки большого реального файла.

Конструктор отчетов - очень мощное средство, допускающее использование до десяти уровней группировки и сортировки. Благодаря ему существует возможность создания отчетов, демонстрирующих процентные и итоговые показатели, получить которые можно лишь за два прохода. Допускается создание многих типов отчетов, которые включают почтовые наклейки и списки рассылки почты.

Многотабличные запросы и отношения

Одна из самых мощных возможностей Access одновременно является и наиболее важной. Отношения позволяют связать таблицы графически. Можно даже связывать таблицы, представляющие файлы разных типов (например, таблицу Access и таблицу dBASE). После подобного связывания таблицы выступают уже как одно целое, и теперь можно строить запросы применительно к любым данным в них. Можно выбирать конкретные поля, определять порядок сортировки, создавать вычисляемые выражения и вводить критерии отбора нужных записей. Можно отображать результаты выполнения запроса в виде таблицы, формы или отчета. От пользователя не требуется предварительной установки связей: вместо этого достаточно войти в конструктор запросов (например, когда требуется построить определенный отчет).

Запросы применяют и в других случаях. Можно создавать запросы, которые обеспечивают вычисление итогов, отображение сгруппированных и построение новых таблиц. Запрос можно использовать даже для обновления данных в таблицах, удаления записей и добавления одной таблицы к другой.

Графики и диаграммы

В Access используется то же самое графическое приложение, что и в Microsoft Word, Excel, PowerPoint и Project. Оно позволяет создавать сотни

типов графиков и диаграмм, настраивая их, исходя из конкретных потребностей. Можно создавать гистограммы, линейчатые, круговые, поверхностные и другие диаграммы, причем как двухмерные, так и трехмерные. Их можно произвольно сопровождать текстом, оформлять разными цветами и узорами. Значения могут отображаться в столбцах или секторах круговых диаграмм. Можно разворачивать изображения диаграмм так, чтобы они воспроизводились под любым удобным углом зрения. Все это обеспечивает программа Access Graph.

Возможности DDE и OLE

С помощью DDE (Dynamic Data Exchange - динамический обмен данными) и OLE (Object Linking and Embedding - связывание и внедрение объектов) в формы и отчеты Access можно добавлять всевозможные новые объекты. Такими объектами могут быть звук, рисунки, диаграммы и даже видеоклипы. Можно внедрять объекты OLE (например, растровые изображения) или документы текстовых процессоров (Word или WordPerfect), или устанавливать связи с электронными таблицами Excel. Связывая эти объекты со своей базой данных, пользователь может создавать динамические формы и отчеты, а также использовать одну и ту же информацию в разных приложениях Windows.

Доступ к Internet

В Access теперь предусмотрены все возможности, обеспечивающие связь приложении с Internet/intranet. Одним щелчком кнопкой мыши можно сохранить таблицы, запросы, формы и отчеты в формате HTML. Соответствующий мастер позволяет даже новичку перенести коды HTML из объекта на Web-страницу, делая их доступными для использования всем, кто путешествует по Internet! Гиперссылки позволяют получать доступ к данным, которые размещены на Web-странице, прямо из форм Access.

Многие считают, что размещение данных на Web-страницах должно осуществляться Web-администраторами. Access с полной определенностью доказывает, что эта операция может быть с успехом выполнена любым пользователем. А поможет ему в этом мастер размещения на Web-странице, обеспечивающий преобразование выбранных объектов базы данных в формат HTML и перенос их уже в таком виде на Web-страницу. С помощью этого мастера можно создать статические либо динамические страницы, перенести их на Web-сервер, создать свою начальную страницу и даже использовать шаблоны для получения стандартного внешнего вида всех HTML-страниц!

Встроенные функции

Access содержит свыше ста функции (небольших встроенных программ, которые в результате выполнения возвращают значение), выполняющих множество разнообразных задач. Есть функции для манипулирования базами

данных, строками, числами в формате даты и времени, математические, деловые и финансовые. Их можно использовать для создания вычисляемых выражений в формах, отчетах и запросах.

Макросы: программирование без программирования

Для непрограммистов (или опытных пользователей, которые просто не желают программировать) в Access предусмотрены макросы. Они позволяют автоматизировать выполнение некоторых задач. Около пятидесяти макросов дают возможность манипулировать данными, создавать меню и диалоговые окна, открывать формы и отчеты, словом, автоматизировать выполнение практически любой задачи. С помощью макросов можно решить порядка 90% всех задач обработки данных.

Модули: Visual Basic for Applications программирование баз данных

Access - это серьезная среда разработки приложений с полнофункциональным языком программирования. Язык VBA (ранее известный как Access Basic) реализует объектно-ориентированный подход к программированию и позволяет программисту делать практически все, что только можно себе представить. Это мощный язык структурного программирования. Он является полностью расширяемым и поддерживает процедуры API в любых динамических библиотеках (DLL) операционных систем Windows 95 и Windows NT.

Полнофункциональная среда разработки поддерживает множество мощных современных возможностей: многооконный режим для редактирования и отладки, автоматическую проверку синтаксиса, контрольные точки, пошаговое выполнение и даже синтаксическую справку, отображающую на экране варианты вводимых команд.

Существуют два режима Access:

1) проектировочный - разрабатывается структура таблиц БД, создаются формы;
2) эксплуатационный (для пользователя) - пользователи наполняют БД данными и обслуживают их.

Объекты Access

Исходное окно Access 9x отличается простотой и лаконичностью. Шесть вкладок этого окна представляют шесть видов объектов, с которыми работает программа.

Таблицы - основные объекты базы данных. С ними мы уже знакомы. В них хранятся данные. Реляционная база данных может иметь много взаимосвязанных таблиц.

Запросы - это специальные структуры, предназначенные для обработки данных базы. С помощью запросов данные упорядочивают, фильтруют, отбирают, изменяют, объединяют, то есть обрабатывают.

Формы - это объекты, с помощью которых в базу вводят новые данные или просматривают имеющиеся.

Отчеты - это формы "наоборот". С их помощью данные выдают на принтер в удобном и наглядном виде.

Макросы - это *макрокоманды.* Если какие-то операции с базой производятся особенно часто, имеет смысл сгруппировать несколько команд в один макрос и назначить его выделенной комбинации клавиш.

Модули - это программные процедуры, написанные на языке Visual Basic. Если стандартных средств Access не хватает для удовлетворения особо изощренных требований заказчика, программист может расширить возможности системы, написав для этого необходимые модули.

Рассмотрев все достоинства и недостатки этих ИСР и их языков программирования, для разработки программного продукта была выбрана интегрированная среда разработки Borland Delphi и Microsoft Access.

2.2 Описание компонентов

Создавая данную программу, использованы различные компоненты, такие как button, combobox, edit, label, radiogroup, statictext, memo.

Простейшей и, пожалуй, наиболее часто используемой кнопкой является кнопка Button расположенная на странице библиотеки Standard.

Основное свойство кнопки с точки зрения внешнего вида - Caption (надпись). В надписях кнопок можно предусматривать использование клавиш ускоренного доступа, выделяя для этого один из символов надписи. Перед символом, который должен соответствовать клавише ускоренного доступа, ставится символ амперсанта "&". Этот символ не появляется в надписи, а следующий за ним символ оказывается подчеркнутым. Тогда пользователь может вместо щелчка на кнопке нажать в любой момент клавишу Alt совместно с клавишей выделенного символа.

Основное событие любой кнопки - OnClick, возникающее при щелчке на ней. Именно в обработчике этого события записываются операторы, которые должны выполняться при щелчке пользователя на кнопке. Помимо этого есть еще ряд событий, связанных с различными манипуляциями клавишами и кнопками мыши.

Свойство Cancel, если его установить в true, определяет, что нажатие пользователем клавиши Esc будет эквивалентно нажатию на данную кнопку. Это свойство целесообразно задавать равным true для кнопок "Отменить" в различных диалоговых окнах, чтобы можно было выйти из диалога, нажав на эту кнопку или нажав клавишу Esc. Свойство Default, если его установить в true, определяет, что нажатие пользователем клавиши ввода.

Enter будет эквивалентно нажатию на данную кнопку, даже если данная кнопка в этот момент не находится в фокусе.

Правда, если в момент нажатия Enter в фокусе находится другая кнопка, то все-таки сработает именно кнопка в фокусе.

Из методов, присущих кнопкам, имеет смысл отметить один - Click. Выполнение этого метода эквивалентно щелчку на кнопке, т.е. вызывает событие кнопки OnClick. Этим можно воспользоваться, чтобы продублировать какими-то другими действиями пользователя щелчок на кнопке. Пусть, например, вы хотите, чтобы при нажатии пользователем клавиши с символом "C" или "c" в любой момент работы с приложением выполнялись операции, предусмотренные в обработчике события OnClick кнопки Button1. Поскольку неизвестно, какой компонент будет находиться в фокусе в момент этого события, надо перехватить его на уровне формы. Такой перехват осуществляется, если установить свойство формы KeyPreview в true. Тогда в обработчике события формы OnKeyPress можно написать оператор

```
if (key='C' or key='c') then Button1.Click.
```

Если пользователь ввел символ "C" или "c", то в результате будет выполнен обработчик щелчка кнопки Button1.

ComboBox. Стиль изображения этого компонента определяется свойством Style, которое может принимать следующие основные значения:

- csDropDown - выпадающий список со строками одинаковой высоты и с окном редактирования, позволяющим пользователю вводить или редактировать текст;

- csSimple - развернутый список со строками одинаковой высоты и с окном редактирования, позволяющим пользователю вводить или редактировать текст;

- csDropDownList - выпадающий список со строками одинаковой высоты, не содержащий окна редактирования.

Выбор пользователя или введенный им текст можно определить по значению свойства Text. Если же надо определить индекс выбранного пользователем элемента списка, то можно воспользоваться свойством ItemIndex. Если начальное значение не задано, то в момент запуска приложения пользователь не увидит в окне компонента одно из возможных значений списка и, вероятнее всего, не очень поймет, что с этим окном надо делать.

Если в окне проводилось редактирование данных, то ItemIndex = -1. По этому признаку можно определить, что редактирование проводилось.

Свойство MaxLength определяет максимальное число символов, которые пользователь может ввести в окно редактирования.

Если MaxLength = 0, то число вводимых символов не ограничено.

Edit. В компоненте Edit вводимый и выводимый текст содержится в свойстве Text. Это свойство можно устанавливать в процессе проектирования или задавать программно. Выравнивание текста, как это имело место в метках и панелях, невозможно. Перенос строк тоже невозможен. Текст, не помещающийся по длине в окно, просто сдвигается и пользователь может перемещаться по нему с помощью курсора. Свойство AutoSize в окнах редактирования имеет смысл, отличный от смысла аналогичного свойства меток: автоматически подстраивается под размер текста только высота, но не ширина окна.

Окна редактирования снабжены многими функциями, свойственными большинству редакторов. Например, в них предусмотрены типичные комбинации "горячих" клавиш: Ctrl-C - копирование выделенного текста в буфер обмена Clipboard (команда Copy), Ctrl-X - вырезание выделенного текста в буфер Clipboard (команда Cut), Ctrl-V - вставка текста из буфера Clipboard в позицию курсора (команда Paste), Ctrl-Z - отмена последней команды редактирования.

Edit можно использовать и просто как компоненты отображения текста. Для этого надо установить в true его свойство ReadOnly и целесообразно установить AutoSelect в false. В этом случае пользователь не сможет изменять отображаемый текст и окно редактирования становится подобным меткам. При использовании окон редактирования для вывода, ввода и редактирования чисел необходимо использовать функции взаимного преобразования строк и чисел. Для вывода это описанные при рассмотрении меток функции FloatToStr и IntToStr. При вводе это функции StrToFloat - преобразование строки в значение с плавающей запятой, и StrToInt - преобразование строки в целое значение. Если вводимый текст не соответствует числу (например, содержит недопустимые символы), то функции преобразования генерируют исключение EConvertError. Поэтому в программе необходимо предусмотреть обработку этого исключения. Например:

```
var A: integer;
try
 A := StrToInt(Edit1.Text);
 ...{операторы, использующие переменную A}
except on EConvertError do
 ShowMessage('Вы ввели ошибочное число, повторите ввод').
```

Этот код обеспечивает сообщение пользователю об ошибке ввода и предотвращает ошибочные вычисления. Впрочем, это не лучший вариант предотвратить ошибочный ввод, поскольку пользователь узнает о своей ошибке только после того, как программа пытается использовать введенные данные. Лучше, если пользователь просто не сможет ввести неправильные символы.

Например, если вы хотите, чтобы пользователь мог вводить в окно редактирования Edit только цифры и символ точки, вы можете в обработчик события OnKeyPress этого компонента вставить оператор:

```
if not (Key in ['0'..'9', ',']) then Key := #0.
```

Этот оператор подменит все символы, кроме цифр и запятой, нулевым символом, который не занесется в текст окна Edit.

Свойство MaxLength определяет максимальную длину вводимого текста. Если MaxLength = 0, то длина текста не ограничена. В противном случае значение MaxLength указывает максимальное число символов, которое может ввести пользователь.

Свойство PasswordChar позволяет превращать окно редактирования в окно ввода пароля. По умолчанию значение PasswordChar равно #0 - нулевому символу. В этом случае это обычное окно редактирования. Но если в свойстве указать иной символ (например, символ звездочки "*"), то при вводе пользователем текста в окне будут появляться именно эти символы, а не те, которые вводит пользователь. Тем самым обеспечивается секретность ввода пароля.

Label, StaticText. Для отображения различных надписей на форме используются в основном компоненты Label, StaticText (появившийся только в Delphi 3) и Panel. Первые два из этих компонентов - метки, специально предназначенные для отображения текстов.

Тексты, отображаемые в перечисленных компонентах, определяются значением их свойства Caption. Его можно устанавливать в процессе проектирования или задавать и изменять программно во время выполнения приложения. Например:

```
Label1.Caption := 'Новый текст'.
```

Если требуется отобразить числовую информацию, можно воспользоваться функциями FloatToStr и IntToStr, переводящими соответственно числа с плавающей запятой и целые в строку. Для формирования текста, состоящего из нескольких фрагментов, можно использовать операцию "+", которая для строк означает их склеивание (конкатенацию). Например, если в программе имеется целая переменная I, отображающая число сотрудников некоторой организации, то вывести в метку Label1 информацию об этом можно оператором:

```
Label1.Caption := 'Число сотрудников: '+IntToStr(I).
```

Во всех компонентах цвет фона определяется свойством Color, а цвет надписи - подсвойством Color свойства Font.

Для метки Label цвет и шрифт - единственно доступные элементы оформления надписи. Компонент StaticText имеет кроме того свойство BorderStyle, определяющее рамку текста - бордюр.

Размер меток Label и StaticText определяется также свойством AutoSize. Если это свойство установлено в true, то вертикальный и горизонтальный размеры компонента определяются размером надписи. Если же AutoSize равно

false, то выравнивание текста внутри компонента определяется свойством Alignment, которое позволяет выравнивать текст по левому краю, правому краю или центру клиентской области метки.

В метке Label имеется свойство Wordwrap - допустимость переноса слов длинной надписи, превышающей длину компонента, на новую строчку. Чтобы такой перенос мог осуществляться, надо установить свойство WordWrap в true, свойство AutoSize в false (чтобы размер компонента не определялся размером надписи) и сделать высоту компонента такой, чтобы в нем могло поместиться несколько строк.

В метке StaticText перенос длинного текста осуществляется автоматически, если значение AutoSize установлено в false и размер компонента достаточен для размещения нескольких строк. Для того, чтобы в StaticText осуществлялся перенос при изменении пользователем размеров окна, надо осуществлять описанную выше перерисовку компонента методом Repaint в обработчике события формы OnResize.

RadioGroup. Радиокнопки образуют группы взаимосвязанных индикаторов, из которых может быть выбран только один. Они используются для выбора пользователем одной из нескольких взаимоисключающих альтернатив, например, отдела, в котором работает сотрудник, или пола сотрудника. Впрочем, радиокнопки могут использоваться и для отображения аналогичных данных. В этом случае управление кнопками осуществляется программно.

RadioGroup - это панель, которая может содержать регулярно расположенные столбцами и строками радиокнопки. Надпись в левом верхнем углу панели определяется свойством Caption. А надписи кнопок и их количество определяются свойством Items, имеющим тип TStrings. Щелкнув на кнопке с многоточием около этого свойства в окне Инспектора Объектов, вы попадете в редактор списков строк. В нем вы можете занести надписи, которые хотите видеть около кнопок, по одной в строке. Сколько строчек вы запишете - столько и будет кнопок.

Кнопки, появившиеся в панели после задания значений Items, можно разместить в несколько столбцов (не более 17), задав свойство Columns. По умолчанию Columns = 1, т.е. кнопки размещаются друг под другом.

Определить, какую из кнопок выбрал пользователь, можно по свойству ItemIndex, которое показывает индекс выбранной кнопки. Индексы, как всегда в Delphi, начинаются с 0. По умолчанию ItemIndex = -1, что означает отсутствие выбранной кнопки. Если вы хотите, чтобы в момент начала выполнения приложения какая-то из кнопок была выбрана (это практически всегда необходимо), то надо установить соответствующее значение ItemIndex во время проектирования. Если вы используете радиокнопки не для ввода, а для отображения данных, устанавливать значение ItemIndex можно программно во время выполнения приложения.

Компонент RadioGroup очень удобен, но не свободен от некоторых недостатков. Его хорошо использовать, если надписи кнопок имеют примерно одинаковую длину и если число кнопок в каждом столбце (при размещении их в нескольких столбцах) одинаково.

Memo. В компоненте Memo формат (шрифт, его атрибуты, выравнивание) одинаков для всего текста и определяется свойством Font. Если вы сохраните в файле текст, введенный или отредактированный пользователем, то будет создан текстовый файл, содержащий только символы и не содержащий элементов форматирования. При последующем чтении этого файла в Memo формат будет определяться текущим состоянием свойства Font компонента Memo, а не тем, в каком формате ранее вводился текст.

Свойства Alignment и WordWrap имеют тот же смысл, что, например, в метках, и определяют выравнивание текста и допустимость переноса длинных строк. Установка свойства ReadOnly в true задает текст только для чтения. Свойство MaxLength определяет максимальную длину вводимого текста. Если MaxLength = 0, то длина текста не ограничена. Свойства WantReturns и WantTab определяют допустимость ввода пользователем в текст символов перевода строки и табуляции.

Свойство ScrollBars определяет наличие полос прокрутка текста в окне. По умолчанию ScrollBars = ssNone, что означает их отсутствие. Пользователь может в этом случае перемещаться по тексту только с помощью курсора. Можно задать свойству ScrollBars значения ssHorizontal, ssVertical или ssBoth, что будет соответственно означать наличие горизонтальной, вертикальной или обеих полос прокрутки.

Свойства TabCount и Tab имеют смысл при вводе текста только при значении свойства компонента WantTabs = true. Это свойство разрешает пользователю вводить в текст символ табуляции. Если WantTabs = false, то нажатие пользователем клавиши табуляции просто переключит фокус на очередной компонент и символ табуляции в текст не введется.

Основное свойство окна memo - Lines, содержащее текст окна в виде списка строк и имеющее тип TStrings. Начальное значение текста можно установить в процессе проектирования, нажав кнопку с многоточием около свойства Lines в окне Инспектора Объектов. Перед вами откроется окно редактирования списков строк. Вы можете редактировать или вводить текст непосредственно в этом окне, или нажать кнопку CodeEditor и работать в обычном окне Редактора Кода. В этом случае, завершив работу с текстом, выберите из контекстного меню, всплывающего при щелчке правой кнопкой мыши, команду Close Page и ответьте утвердительно на вопрос, хотите ли вы сохранить текст в соответствующем свойстве окна редактирования.

Во время выполнения приложения вы можете заносить текст в окно редактирования с помощью методов свойства Lines типа TStrings. Этот тип широко используется в свойствах многих компонентов и его описание вы

можете найти в во встроенной справке Delphi. Здесь коротко укажем только на его основные свойства и методы, используемые в свойстве Lines.

Весь текст, представленный одной строкой типа String, внутри которой используются разделители типа символов возврата каретки и перевода строки, содержится в свойстве Text.

Доступ к отдельной строке текста вы можете получить с помощью свойства Strings[Index: Integer]. Индексы, как и везде в Delphi, начинаются с 0. Так что Memo1.Lines.Strings[0] - это текст первой строки. Учтите, что если окно редактирования изменяется в размерах при работе с приложением и свойство WordWrap = true, то индексы строк будут изменяться при переносах строк, так что в этих случаях индекс мало, о чем говорит.

Свойство только для чтения Count указывает число строк в тексте.

Для очистки текста в окне надо выполнить процедуру Clear. Этот метод относится к самому окну, а не к его свойству Lines.

Для занесения новой строки в конец текста окна редактирования можно воспользоваться методами Add или Append свойства Lines. Для загрузки текста из файла применяется метод LoadFromFile. Сохранение текста в файле осуществляется методом SaveToFile.

Компоненты, используемые при разработке в Delphi, встроены в среду разработки приложений и представляют из себя набор типов объектов, используемых в качестве фундамента при строительстве приложения.

Этот костяк называется Visual Component Library (VCL). В VCL есть такие стандартные элементы управления, как строки редактирования, статические элементы управления, строки редактирования со списками, списки объектов. Еще имеются такие компоненты, которые ранее были доступны только в библиотеках третьих фирм: табличные элементы управления, закладки, многостраничные записные книжки. Все объекты разбиты на страницы по своей функциональности и представленны в палитре компонент.

VCL содержит специальный объект, предоставлющий интерфейс графических устройств Windows, и позволяющий разработчикам рисовать, не заботясь об обычных для программирования в среде Windows деталях.

Ключевой особенностью Delphi является возможность не только использовать визуальные компоненты для строительства приложений, но и создание новых компонент. Такая возможность позволяет разработчикам не переходить в другую среду разработки, а наоборот, встраивать новые инструменты в существующую среду. Кроме того, можно улучшить или полностью заменить существующие по умолчанию в Delphi компоненты.

Здесь следует отметить, что обычных ограничений, присущих средам визуальной разработки, в Delphi нет. Сам Delphi написан при помощи Delphi, что говорит об отсутствии таких ограничений.

Классы объектов построены в виде иерархии, состоящей из абстрактных, промежуточных, и готовых компонент. Разработчик может пользоваться готовыми компонентами, создавать собственные на основе абстрактных или

промежуточных, а также создавать собственные объекты. Рассмотрим некоторые из них.

TMainMenu позволяет поместить главное меню в программу. При помещении TMainMenu на форму это выглядит, как просто иконка. Иконки данного типа называют *невизуальным компонентом*, поскольку они невидимы во время выполнения программы.

TPopupMenu позволяет создавать всплывающие меню. Этот тип меню появляется по щелчку правой кнопки мыши на объекте, к которому привязано данное меню. У всех видимых объектов имеется свойство PopupMenu, где и указывается нужное меню. Создается PopupMenu аналогично главному меню.

TLabel служит для отображения текста на экране. Можно изменить шрифт и цвет метки, если дважды щелкнуть на свойство Font в Инспекторе Объектов. Это легко сделать и во время выполнения программы, написав всего одну строчку кода.

TEdit - стандартный управляющий элемент Windows для ввода. Он может быть использован для отображения короткого фрагмента текста и позволяет пользователю вводить текст во время выполнения программы.

TMemo - иная форма TEdit. Подразумевает работу с большими текстами. TMemo может переносить слова, сохранять в ClipBoard фрагменты текста и восстанавливать их, и другие основные функции редактора. TMemo имеет ограничения на объем текста в 32Кб, это составляет 10-20 страниц (есть подобные компоненты, где этот предел снят).

TButton позволяет выполнить какие-либо действия при нажатии кнопки во время выполнения программы. В Delphi все делается очень просто. Поместив TButton на форму, по двойному щелчку можно создать заготовку обработчика события нажатия кнопки.

TCheckBox отображает строку текста с маленьким окошком рядом. В окошке можно поставить отметку, которая означает, что что-то выбрано.

TRadioButton позволяет выбрать только одну опцию из нескольких.

TListBox нужен для показа прокручиваемого списка. Классический пример ListBox’а в среде Windows - выбор файла из списка в пункте меню File | Open многих приложений. Названия файлов или директорий и находятся в ListBox’е.

TComboBox во многом напоминает ListBox, за исключением того, что позволяет водить информацию в маленьком поле ввода сверху ListBox. Есть несколько типов ComboBox, но наиболее популярен спадающий вниз (drop-down combo box), который можно видеть внизу окна диалога выбора файла.

TScrollbar - полоса прокрутки, появляется автоматически в объектах редактирования, ListBox’ах при необходимости прокрутки текста для просмотра.


TGroupBox используется для визуальных целей и для указания Windows, каков порядок перемещения по компонентам на форме (при нажатии клавиши TAB).


TRadioGroup используется аналогично TGroupBox, для группировки объектов TRadioButton.


TPanel - управляющий элемент, похожий на TGroupBox, используется в декоративных целях. Чтобы использовать TPanel, можно просто поместить его на форму и затем положите другие компоненты на него. Теперь при перемещении TPanel будут передвигаться и эти компоненты. TPanel используется также для создания линейки инструментов и окна статуса.


TBitBtn - кнопка вроде TButton, однако на ней можно разместить картинку (glyph). TBitBtn имеет несколько предопределенных типов (bkClose, bkOK и др), при выборе которых кнопка принимает соответствующий вид. Кроме того, нажатие кнопки на модальном окне приводит к закрытию окна с соответствующим модальным результатом.


TSpeedButton - кнопка для создания панели быстрого доступа к командам (SpeedBar). Пример - SpeedBar слева от Палитры Компонент в среде Delphi. Обычно на данную кнопку помещается только картинка (glyph).


TTabSet - горизонтальные закладки. Обычно используется вместе с TNoteBook для создания многостраничных окон. Название страниц можно задать в свойстве Tabs.


TNoteBook - используется для создания многостраничного диалога, на каждой странице располагается свой набор объектов. Используется совместно с TTabSet.


TTabbedNotebook - многостраничный диалог со встроенными закладками, в данном случае - закладки сверху.


TOutline - используется для представления иерархических отношений связанных данных. Например - дерево директорий.


TStringGrid - служит для представления текстовых данных в виде таблицы. Доступ к каждому элементу таблицы происходит через свойство Cell.


TDrawGrid - служит для представления данных любого типа в виде таблицы. Доступ к каждому элементу таблицы происходит через свойство CellRect.


TImage - отображает графическое изображение на форме. Воспринимает форматы BMP, ICO, WMF. Если картинку подключить во время дизайна программы, то она прикомпилируется к EXE файлу.


TShape - служит для отображения простейших графических объектов на форме: окружность, квадрат и т.п.


TBevel - элемент для рельефного оформления интерфейса.


THeader - элемент оформления для создания заголовков с

изменяемыми размерами для таблиц.


TScrollBox - позволяет создать на форме прокручиваемую область с размерами большими, нежели экран. На этой области можно разместить свои объекты.


TTimer - таймер, событие OnTimer периодически вызывается через промежуток времени, указанный в свойстве Interval. Период времени может составлять от 1 до 65535 мс.


TPaintBox - место для рисования. В обработчики событий, связанных с мышкой передаются относительные координаты мышки в TPaintBox, а не абсолютные в форме.


TFileListBox - специализированный ListBox, в котором отображаются файлы из указанной директории (св-во Directory). На названия файлов можно наложить маску, для этого служит св-во Mask. Кроме того, в св-ве FileEdit можно указать объект TEdit для редактирования маски.


TDirectoryListBox - специализированный ListBox, в котором отображается структура директорий текущего диска. В св-ве FileList можно указать TFileListBox, который будет автоматически отслеживать переход в другую директорию.


TDriveComboBox - специализированный ComboBox для выбора текущего диска. Имеет свойство DirList, в котором можно указать TDirectoryListBox, который будет отслеживать переход на другой диск.


TFilterComboBox - специализированный ComboBox для выбора маски имени файлов. Список масок определяется в свойстве Filter. В свойстве FileList указывается TFileListBox, на который устанавливается маска.

С помощью последних четырех компонент (TFileListBox, TDirectoryListBox, TDriveComboBox, TFilterComboBox) можно построить свой собственный диалог выбора файла, причем для этого не потребуется написать ни одной строчки кода.

TOLEContainer - контейнер, содержащий OLE объекты. Поддерживается OLE 2.02


TDDEClientConv,TDDEClientItem, TDDEServerConv, TDDEServerItem - 4 объекта для организации DDE. С помощью этих объектов можно построить приложение как DDE-сервер, так и DDE-клиент.


TChartFX - деловая графика. Компонент позволяет строить всевозможные графики и гистограммы.

Формы, модули и метод разработки "Two-Way Tools"

Формы - это объекты, в которые помещаются другие объекты для создания пользовательского интерфейса любого приложения. Модули состоят

из кода, который реализует функционирование приложения, обработчики событий для форм и их компонент.

Информация о формах хранится в двух типах файлов - *.dfm* и *.pas*, причем первый тип файла - двоичный - хранит образ формы и ее свойства, второй тип описывает функционирование обработчиков событий и поведение компонент. Оба файла автоматически синхронизируются Delphi, так что если добавить новую форму проект, связанный с ним файл *.pas* автоматически будет создан, и его имя будет добавлено в проект.

Такая синхронизация и делает Delphi *two-way-инструментом*, обеспечивая полное соответствие между кодом и визуальным представлением. Как только добавляется новый объект или код, Delphi устанавливает т.н. *"кодовую синхронизацию"* между визуальными элементами и соответствующими им кодовыми представлениями.

Two-way tools - однозначное соответствие между визуальным проектированием и классическим написанием текста программы Это означает, что разработчик всегда может видеть код, соответствующий тому, что он построил при помощи визуальных инструментов и наоборот.

Визуальный построитель интерфейсов (Visual User-interface builder) дает возможность быстро создавать клиент-серверные приложения визуально, просто выбирая компоненты из соответствующей палитры. В процессе построения приложения разработчик выбирает из палитры компонент готовые компоненты как художник, делающий крупные мазки кистью. Еще до компиляции он видит результаты своей работы - после подключения к источнику данных их можно видеть отображенными на форме, можно перемещаться по данным, представлять их в том или ином виде.

Масштабируемые средства для построения баз данных

Мощность и гибкость Delphi при работе с базами данных основана на низкоуровневом *ядре* - процессоре баз данных *Borland Database Engine* (BDE). Его интерфейс с прикладными программами называется *Integrated Database Application Programming Interface* (IDAPI). В принципе, сейчас не различают эти два названия (BDE и IDAPI) и считают их синонимами. BDE позволяет осуществлять доступ к данным как с использованием традиционного record-ориентированного (навигационного) подхода, так и с использованием set-ориентированного подхода, используемого в SQL-серверах баз данных. Кроме BDE, Delphi позволяет осуществлять доступ к базам данных, используя технологию (и, соответственно, драйверы) Open DataBase Connectivity (ODBC) фирмы Microsoft. Но, как показывает практика, производительность систем с использованием BDE гораздо выше, чем оных при использовании ODBC. ODBC драйвера работают через специальный "ODBC socket", который позволяет встраивать их в BDE.

Все инструментальные средства баз данных Borland - Paradox, dBase, Database Desktop - используют BDE. Все особенности, имеющиеся в Paradox или dBase, “наследуются” BDE, и поэтому этими же особенностями обладает и Delphi.

Библиотека объектов содержит набор визуальных компонент, значительно упрощающих разработку приложений для СУБД с архитектурой клиент-сервер. Объекты инкапсулируют в себя нижний уровень - Borland Database Engine.

Предусмотрены специальные наборы компонент, отвечающих за доступ к данным, и компонент, отображающих данные. Компоненты доступа к данным позволяют осуществлять соединения с БД, производить выборку, копирование данных, и т.п.

Компоненты визуализации данных позволяют отображать данные виде таблиц, полей, списков. Отображаемые данные могут быть текстового, графического или произвольного формата.

Таблицы сохраняются в базе данных. Некоторые СУБД сохраняют базу данных в виде нескольких отдельных файлов, представляющих собой таблицы (в основном, все локальные СУБД), в то время как другие состоят из одного файла, который содержит в себе все таблицы и индексы (InterBase). Например, таблицы dBase и Paradox всегда сохраняются в отдельных файлах на диске. Директорий, содержащий dBase *.DBF* файлы или Paradox *.DB* файлы, рассматривается как база данных. Другими словами, любой директорий, содержащий файлы в формате Paradox или dBase, рассматривается Delphi как единая база данных. Для переключения на другую базу данных нужно просто переключиться на другой директорий. InterBase сохраняет все таблицы в одном файле, имеющем расширение *.GDB*, поэтому этот файл и есть база данных InterBase.

Объекты БД в Delphi основаны на SQL и включают в себя полную мощь Borland Database Engine. В состав Delphi также включен Borland SQL Link, поэтому доступ к СУБД Oracle, Sybase, Informix и InterBase происходит с высокой эффективностью. Кроме того, Delphi включает в себя локальный сервер Interbase для того, чтобы можно было разработать расширяемые на любые внешние SQL-сервера приложения в офлайновом режиме. Разработчик в среде Delphi, проектирующий информационную систему для локальной машины (к примеру, небольшую систему учета медицинских карточек для одного компьютера), может использовать для хранения информации файлы формата .dbf (как в dBase или Clipper) или .db (Paradox). Если же он будет использовать локальный InterBase for Windows 4.0 (это локальный SQL-сервер, входящий в поставку), то его приложение безо всяких изменений будет работать и в составе большой системы с архитектурой клиент-сервер.

Масштабируемость на практике - одно и то же приложение можно использовать как для локального, так и для более серьезного клиент-серверного вариантов.

Настраиваемая среда разработчика

После запуска Delphi в верхнем окне горизонтально располагаются иконки палитры компонент. Если курсор задерживается на одной из иконок, под ней в желтом прямоугольнике появляется подсказка.

Из этой палитры компонент можно выбирать компоненты, из которых можно строить приложения. Компоненты включают в себя как визуальные, так и логические компоненты. Такие вещи, как кнопки, поля редактирования - это визуальные компоненты; а таблицы, отчеты - это логические.

Поскольку в Delphi программа строится визуальным образом, все эти компоненты имеют свое графическое представление в поле форм для того, чтобы можно было бы ими соответствующим образом оперировать. Но для работающей программы видимыми остаются только визуальные компоненты. Компоненты сгруппированы на страницах палитры по своим функциям. К примеру, компоненты, представляющие Windows “common dialogs” все размещены на странице палитры с названием “Dialogs”.

Delphi позволяет разработчикам настроить среду для максимального удобства. Можно легко изменить палитру компонент, инструментальную линейку, а также настраивать выделение синтаксиса цветом.

В Delphi можно определить свою группу компонент и разместить ее на странице палитры, а если возникнет необходимость, перегруппировать компоненты или удалить неиспользуемые.

Интеллектуальный редактор. Редактирование программ можно осуществлять, используя запись и исполнение макросов, работу с текстовыми блоками, настраиваемые комбинации клавиш и цветовое выделение строк.

Графический отладчик. Delphi обладает мощнейшим, встроенным в редактор графическим отладчиком, позволяющим находить и устранять ошибки в коде. Можно установить точки останова, проверить и изменить переменные, при помощи пошагового выполнения в точности понять поведение программы. Если же требуются возможности более тонкой отладки, можно использовать отдельно доступный Turbo Debugger, проверив ассемблерные инструкции и регистры процессора.

Инспектор объектов. Этот инструмент представляет из себя отдельное окно, где вы можете в период проектирования программы устанавливать значения свойств и событий объектов (Properties & Events).

Менеджер проектов. Дает возможность разработчику просмотреть все модули в соответствующем проекте и снабжает удобным механизмом для

управления проектами. Менеджер проектов показывает имена файлов, время/дату выбранных форм и пр. Можно немедленно попасть в текст или форму, просто щелкнув мышкой на соответствующее имя.

Навигатор объектов. Показывает библиотеку доступных объектов и осуществляет навигацию по приложению. Можно посмотреть иерархию объектов, прекомпилированные модули в библиотеке, список глобальных имен вашего кода.

Дизайнер меню. Можно создавать меню, сохранить созданные в виде шаблонов и затем использовать в их в любом приложении.

Эксперты. Это набор инструментальных программ, облегчающих проектирование и настройку Ваших приложений. Есть возможность подключать самостоятельно разработанные эксперты. Потенциально это та возможность, при помощи которой третьи фирмы могут расширять Delphi CASE-инструментами, разработанными специально для Delphi. Включает в себя:

1) эксперт форм, работающих с базами данных;
2) эксперт стилей и шаблонов приложений;
3) эксперт шаблонов форм.

В состав RAD Pack входит эксперт для преобразования ресурсов, изготовленных в Borland Pascal 7.0, в формы Delphi. Уже появились эксперты, облегчающие построение DLL и даже написание собственных экспертов.

Интерактивная обучающая система. Позволяет более полно освоить Delphi. Она являются не просто системой подсказок, а показывает возможности Delphi на самой среде разработчика.

3 Реализация проекта

3.1 Создание БД

Для отображения данных из таблицы используется компонент DataSource с вкладки Data Access палитры компонентов. Сначала поместим на форму компонент ADOConnection с вкладки ADO палитры компонентов. Настроим соединение с сервером, которое должно быть указано в свойстве Connectionstring (Рисунок 3.1).

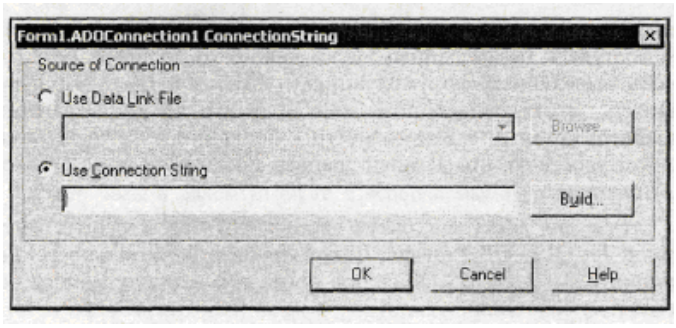

Рисунок 3.1 - Соединение с сервером

Далее щелкаем мышью по кнопке Build. Перед нами откроется еще одно окно (Рисунок 3.2).

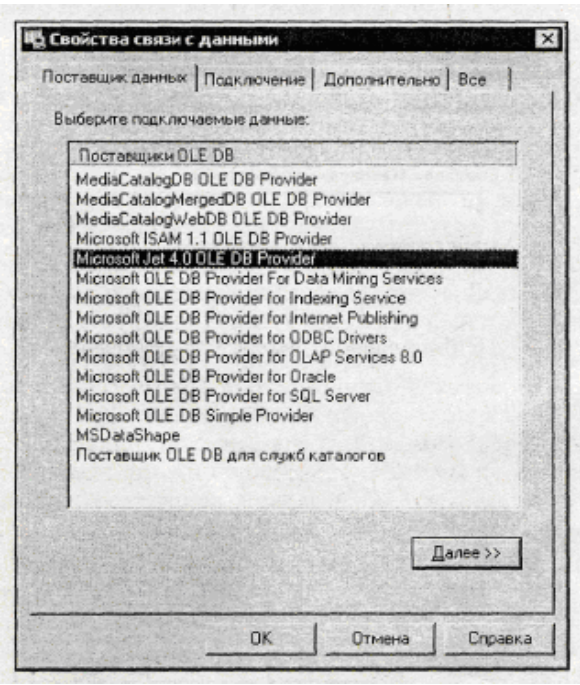

Рисунок 3.2 - Соединение с сервером

Выберем Microsoft Jet 4.0 OLE DB Provider. После этого нажмите кнопку Далее (Next). Вид вкладки Подключение (Connection) зависит от выбранного драйвера. В нашем случае она должна выглядеть так (Рисунок 3.3).

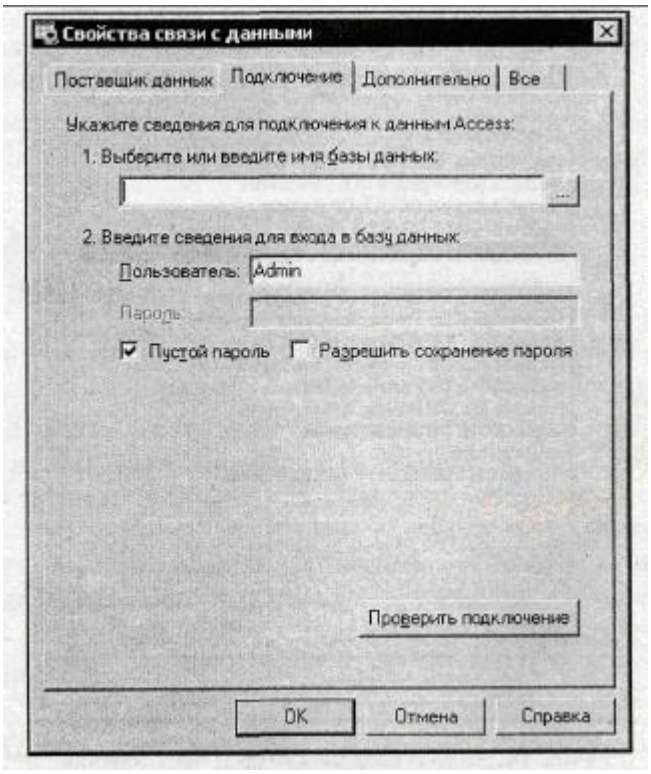

Рисунок 3.3 - Вкладка "Подключение"

В строке "Выберем или введем имя базы данных (Select or enter a database name)", надо ввести имя базы данных (при необходимости и путь). Если база данных будет располагаться в той же директории, что и исполняемый файл приложения, то путь указывать не надо. После выбора базы данных, нажмем кнопку Проверить подключение (Test Connection), чтобы протестировать соединение. Если все указано правильно, то должно появиться сообщение Тестирование соединения прошло удачно (Test connection succeeded). Все, можно нажать ОК, чтобы закрыть окно создания строки подключения. Затем еще раз нажать ОК, чтобы закрыть окно редактора строки подключения.

В свойствах компонента ADOConnection отключите свойство LoginPrompt, выставив его в false. Это нужно для того, чтобы при каждом обращении к базе не происходил вызов окна ввода пароля. Далее выставим свойство Connected в true, чтобы произошло соединение с базой.

Для получения доступа к созданной таблице поместим на форму компонент ADOTaЫe с вкладки ADO палитры компонентов. Щелкаем по выпадающему списку в свойстве Connection и выберите там единственный пункт ADOConnection1. В свойстве TabieName нужно выбрать имя таблицы.

Все, таблица и соединение указаны, можно подключатся. Для этого выставьте свойство Active В true.

Для отображения данных из таблицы надо установить на форму компонент DataSource с вкладки Data Access палитры компонентов. Теперь этому компоненту надо указать, какую именно таблицу он должен отображать. Для этого в свойстве DataSet нужно из выпадающего списка выбрать компонент ADOTable, который связан с нашей таблицей.

Все приготовления готовы, можно приступать к реальному отображению данных. Самый простой способ отобразить таблицу - установить компонент DBGrid. Этот компонент - сетка, которая может отображать данные в виде таблицы. В этом же компоненте можно добавлять, удалять и редактировать строки таблицы.

В моей программе существует 4 Базы данных, в каждой из которой содержится разное количество таблиц. Ниже приведены некоторые из них.

В файле базы данных "Аксессуары" содержится таблица "Носители информации"(Рисунок 3.4).

	Наименование товара	Срок гарантии	Цена в тенге	Колличество
1	1024Mb Flash-card type Micro SD	30 Дней	1264	5
2	1024Mb Flash-card type Mini SD	30 Дней	1510	5
3	2048Mb Flash-card type Micro+Mini+SD	30 Дней	1659	7
4	Card Reader SD/MMC	10 Дней	1580	4
5	DVD+R Ricoh 16x 4,7Gb (10 - по 50, 50 - по 45)		57	500
6	Milti Card Reader USB 2.0	30 Дней	2054	4
7	USB Flash Drive 4GB	30 Дней	1896	15
8	USB Flash Drive SP 16Gb	30 Дней	3002	8

Рисунок 3.4 - База данных "Аксессуары",
таблица "Носители информации"

В файле базы данных "Комплектующие" содержится таблица "Процессоры" (Рисунок 3.5).

№	Наименование товара	Срок гарантии	Цена в тенге	Колличество
1	Celeron-D336+2.8GHz 256KB OEM	2 Года	4898	10
4	Celeron-D347 3.06GHz 512KB OEM	1 Год	6162	5
5	Intel Core 2 Duo E7300 2,66GHz, OEM	2 Года	17969	2
6	Intel Core 2 Quad Q8200 2,3GHz, BOX	2 Года	31758	2
7	Pentium DualCore E2200 2,2GHz, OEM	1 Год	10117	3
8	Pentium DualCore E5300 2,6GHz, OEM	2 Года	12561	4

Рисунок 3.5 - База данных "Комплектующие", таблица "Процессоры"

В файле базы данных "Периферийные устройства" содержится таблица "Клавиатуры" (Рисунок 3.6).

№	Наименование товара	Срок гарантии	Цена в тенге	Колличество
1	Keyboard Genius KB-06XE eng / rus / kaz,Black,USB	3 Дня	667	2
2	KB-2015 Intex, black PS\2	10 Дней	948	3
3	KB-2015 Intex, black PS\2	10 Дней	790	3
4	Keyboard Defender KM-4110S Virtuoso	10 Дней	2607	2
5	Keyboard Genius Comfy KB-200 PS\2	10 Дней	1185	4
6	Keyboard Genius Luxmate Scroll, USB+PS/2	10 Дней	2291	3
7	Keyboard Genius Slimstar 320 USB	10 Дней	2212	2
8	Keyboard+mouse Genius Slimstar C100 PS/2	10 Дней	1896	5
9	Keyboard+Mouse USB BTC 8190 Black URF (радио)	10 Дней	6083	2

Рисунок 3.6 - База данных "Периферийные устройства", таблица "Клавиатуры"

В файле базы данных "Разное" содержится таблица "Ремонт" (Рисунок 3.7).

Талон№	ФИО	Домашний тел	Рабочий телеф	Сотовый телеф	Вид изделия	Детали сданны	Вид неисправ	Дата
1	Якушин Андре	31556	25368	87051661139	Ноутбук	Драйвера	Восстановлени	02.06.2009
2	Кажбулатов Ти	2712047	956390	87017343947	Принтер	Шнуры питания	Замена термог	03.06.2009

Рисунок 3.7 - База данных "Разное", таблица "Ремонт"

3.2 Организация диалоговых окон для управления данными

Шаг 1. Для начала я создал форму "Проверки пользователя". В данной форме есть выбор между двумя пользователями: "User", "Admin". При выборе пользователя "User" возможности работы в программе уменьшаются, пользователь не сможет изменять Базу Данных, а при выборе пользователя "Admin" появляется полный доступ к функциям программы (Рисунок 3.8).

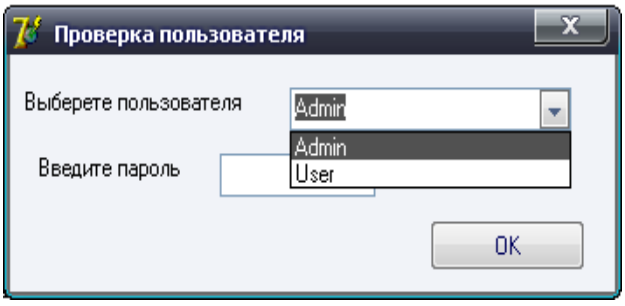

Рисунок 3.8 - Проверка пользователя

Компоненты, установленные на этой форме представлены в таблице 3.1.

Т а б л и ц а 3 . 1 - Компоненты формы "Проверка пользователя"

Объект	Имя	Свойства	Значения
Label	Метка	Name	Label1
		Caption	Выберете пользователя
		Color	clBtnFace
		Font	(TFont)
MaskEdit	Скрытое текстовое поле	Name	MaskEdit1
		Text	Admin
		Color	clBtnFace
		Font	(TFont)
ComboBox	Комбинированный список	Name	ComboBox1
		Items	(TStrings)
		Color	clBtnFace
		Font	(TFont)
Button	Кнопка	Name	Button1
		Caption	OK
		Color	clBtnFace
		Font	(TFont)
XPManifest	Оформление	Name	XPManifest1
		Tag	0

Шаг 2. Далее я создал основное меню (Рисунок 3.9).

Рисунок 3.9 - Основное меню

Компоненты, установленные на этой форме представлены в таблице 3.2.

Таблица 3.2 - Компоненты формы "Меню"

Объект	Имя	Свойства	Значения
Image	Рисунок	Name	Image1
		Picture	(None)
		Color	clBtnFace
		Font	(TFont)
Memo	Поле	Name	Memo"
		Text	
		Color	clBtnFace
		Font	(TFont)
Button	Кнопка	Name	Button1
		Caption	Продажа компьютеров
		Color	clBtnFace
		Font	(TFont)
XPManifest	Оформление	Name	XPManifest1
		Tag	0

Шаг 3. Теперь я создал форму "Продажа компьютеров", на которой отображается список готовых компьютеров (Рисунок 3.10).

Рисунок 3.10 - Форма "Продажа компьютеров"

Компоненты, установленные на этой форме представлены в таблице 3.3.

Т а б л и ц а 3 . 3 - Компоненты формы "Продажа компьютеров"

Объект	Имя	Свойства	Значения
DBGrid	Таблица	Name	DBGrid1
		DataSource	DataSource1
		Color	clBtnFace
		Font	(TFont)
Image	Рисунок	Name	Image1
		Picture	(None)
		Color	clBtnFace
		Font	(TFont)
DataSource	Отображение данных	Name	DataSource1
		DataSet	ADOTable1
		AutoEdit	True
		Tag	0
ADOConnection	Связь с БД	Name	ADOConnection
		LoginPromt	False
		Connected	True
		Provider	Microsoft.Jet.OLEDB.4.0
ADOTable	Управление таблицами	Name	ADOTable1
		Connection	ADOConnection1
		TableName	Компы
		Active	True
XPManifest	Оформление	Name	XPManifest1
		Tag	0

Шаг 4. При нажатии на кнопку "Продажа комплектующих" открывается форма, на которой я создал объекты, с помощью которых отображается список всех комплектующих, содержащихся в БазеДанных (Рисунок 3.11). Формы "Продажа периферии" и "Продажа аксессуаров" будут аналогичны этой форме.

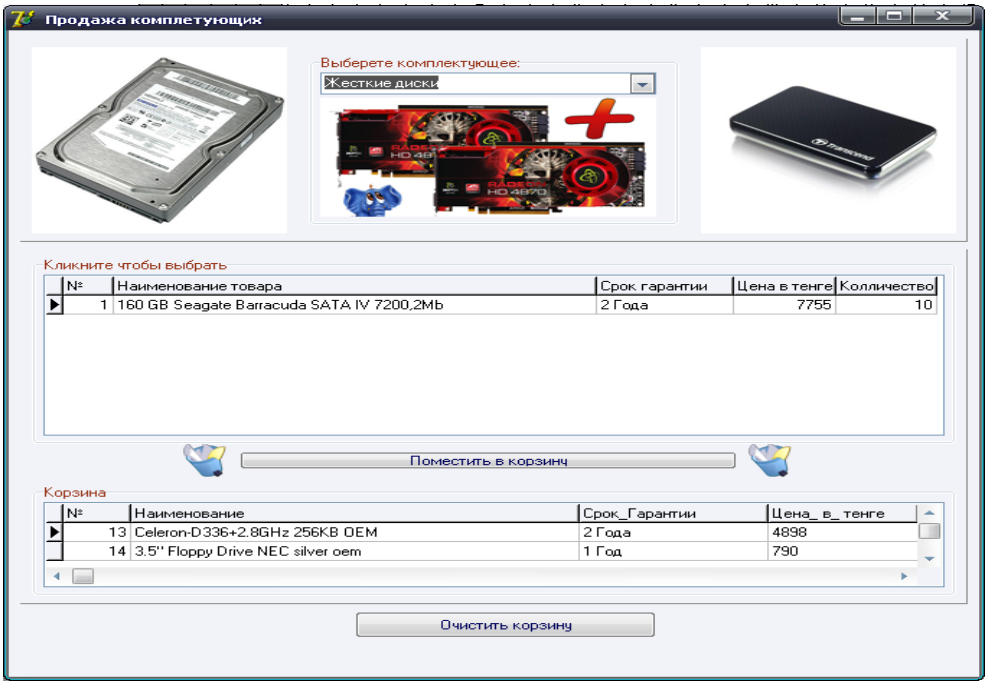

Рисунок 3.11 - Список всех комплектующих, содержащихся в Базе Данных

Компоненты, установленные на этой форме представлены в таблице 3.4.

Таблица 3.4 - Компоненты формы "Продажа комплектующих"

Объект	Имя	Свойства	Значения
DBGrid	Таблица	Name	DBGrid1
		DataSource	DataSource1
		Color	clBtnFace
		Font	(TFont)
Image	Рисунок	Name	Image1
		Picture	(None)
		Color	clBtnFace
		Font	(TFont)
DataSource	Отображение данных	Name	DataSource1
		DataSet	ADOTable1
		AutoEdit	True
		Tag	0
ADOConnection	Связь с БД	Name	ADOConnection
		LoginPromt	False
		Connected	True
		Provider	Microsoft.Jet.OLEDB.4.0
ADOTable	Управление таблицами	Name	ADOTable1
		Connection	ADOConnection1

Окончание таблицы 3.4

Объект	Имя	Свойства	Значения
ADOTable	Управление таблицами	TableName	Компы
		Active	True
Button	Кнопка	Name	Button1
		Caption	OK
		Color	clBtnFace
		Font	(TFont)
ComboBox	Комбинированный список	Name	ComboBox1
		Items	(TStrings)
		Color	clBtnFace
		Font	(TFont)
XPManifest	Оформление	Name	XPManifest1
		Tag	0

Шаг 5. При нажатии на кнопку "Регистрация на ремонт" открывается форма, на которой я создал список "Ремонтируемых изделий" (Рисунок 3.12), когда и кем они были оставлены на ремонт. Здесь же имеется поиск по талону, и распечатка "квитанции на ремонт".

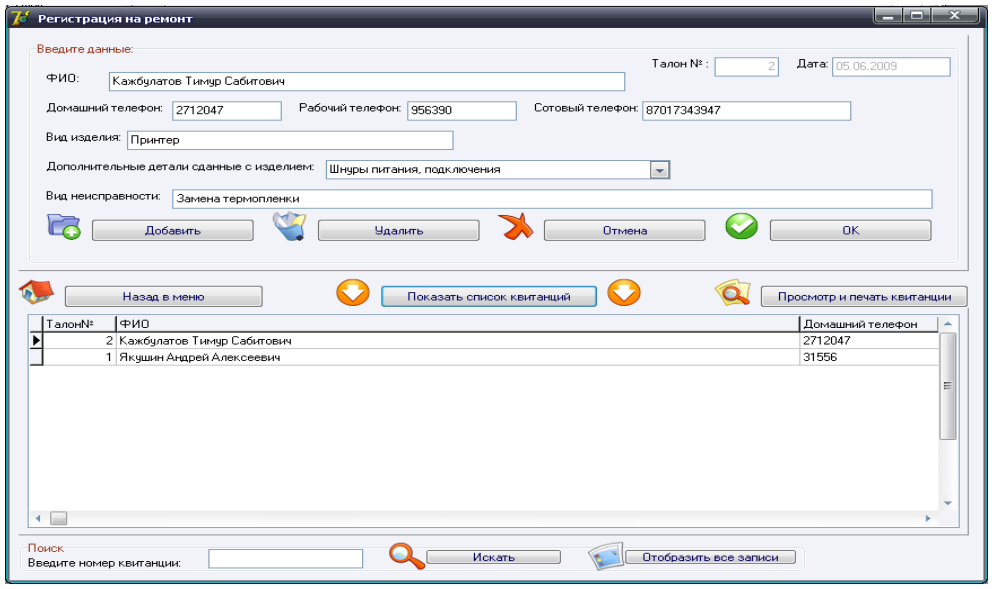

Рисунок 3.12 - Список "Ремонтируемых изделий"

Компоненты, установленные на этой форме представлены в таблице 3.5.

Т а б л и ц а 3 . 5 - Компоненты формы "Регистрация на ремонт"

Объект	Имя	Свойства	Значения
DBGrid	Таблица	Name	DBGrid1
		DataSource	DataSource1
		Color	clBtnFace
		Font	(TFont)
Image	Рисунок	Name	Image1
		Picture	(None)
		Color	clBtnFace
		Font	(TFont)
DataSource	Отображение данных	Name	DataSource1
		DataSet	ADOTable1
		AutoEdit	True
		Tag	0
ADOConnection	Связь с БД	Name	ADOConnection
		LoginPromt	False
		Connected	True
		Provider	Microsoft.Jet.OLEDB. 4.0
ADOTable	Управление таблицами	Name	ADOTable1
		Connection	ADOConnection1
		TableName	Компы
		Active	True
Button	Кнопка	Name	Button1
		Caption	OK
		Color	clBtnFace
		Font	(TFont)
DBComboBox	Комбинированный список	Name	DBComboBox1
		DataSource	DataSource1
		Color	clBtnFace
		Font	(TFont)
Label	Метка	Name	Label1
		Caption	ФИО:
		Color	clBtnFace
		Font	(TFont)
DBEdit	Текстовое поле	Name	DBEdit 1
		DataSource	DataSource1
		Color	clBtnFace
		Font	(TFont)

Окончание таблицы 3.5

Объект	Имя	Свойства	Значения
ADOQuery	Управление таблицами	Name	ADOQuery1
		Connection	ADOConnection1
		TableName	Компы
		Active	True
Excel	Связь с Excel	Name	Excel
		ConnectCind	ckRunningOrNew
XPManifest	Оформление	Name	XPManifest1
		Tag	0

Шаг 6. Создал форму "Заправка картриджей" (Рисунок 3.13), в которой можно зарегестрировать картридж пришедший на заправку, и осуществаить поиск.

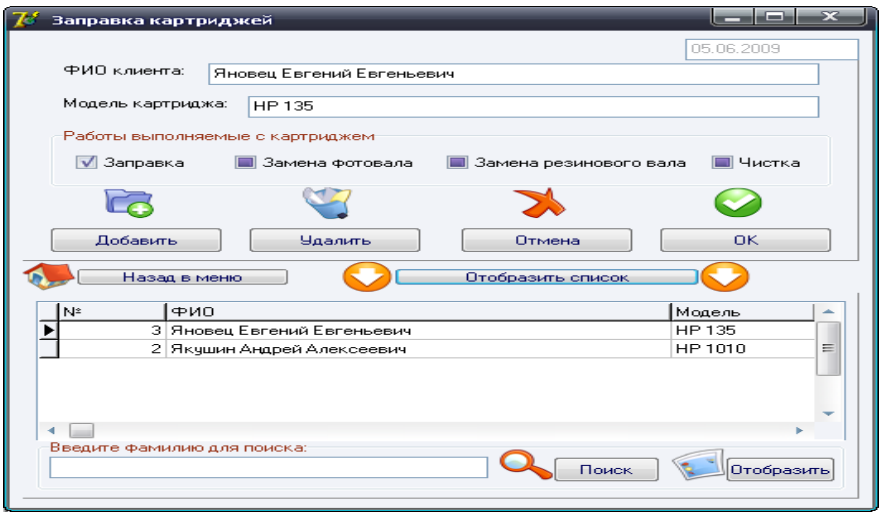

Рисунок 3.13 - Форма "Заправка картриджей"

Компоненты, установленные на этой форме представлены в таблице 3.6.

Таблица 3.6 - Компоненты формы " Заправка картриджей "

Объект	Имя	Свойства	Значения
DBGrid	Таблица	Name	DBGrid1
		DataSource	DataSource1
		Color	clBtnFace
		Font	(TFont)
Image	Рисунок	Name	Image1
		Picture	(None)

Окончание таблицы 3.6

Объект	Имя	Свойства	Значения
Image	Рисунок	Color	clBtnFace
		Font	TFont
DataSource	Отображение данных	Name	DataSource1
		DataSet	ADOTable1
		AutoEdit	True
		Tag	0
ADOConnection	Связь с БД	Name	ADOConnection
		LoginPromt	False
		Connected	True
		Provider	Microsoft.Jet.OLEDB.4.0
ADOTable	Управление таблицами	Name	ADOTable1
		Connection	ADOConnection1
		TableName	Компы
		Active	True
Button	Кнопка	Name	Button1
		Caption	ОК
		Color	clBtnFace
		Font	(TFont)
DBCheckBox	Метка	Name	DBCheckBox1
		DataSource	DataSource1
		Color	clBtnFace
		Font	(TFont)
Label	Метка	Name	Label1
		Caption	ФИО:
		Color	clBtnFace
		Font	(TFont)
DBEdit	Текстовое поле	Name	DBEdit 1
		DataSource	DataSource1
		Color	clBtnFace
		Font	(TFont)
ADOQuery	Управление таблицами	Name	ADOQuery1
		Connection	ADOConnection1
		TableName	Компы
		Active	True
XPManifest	Оформление	Name	XPManifest1
		Tag	0

Шаг 7. И наконец были созданы формы для добавления данных в Базу Данных. Они все похожи и в качестве примера я приведу одну (Рисунок 3.14).

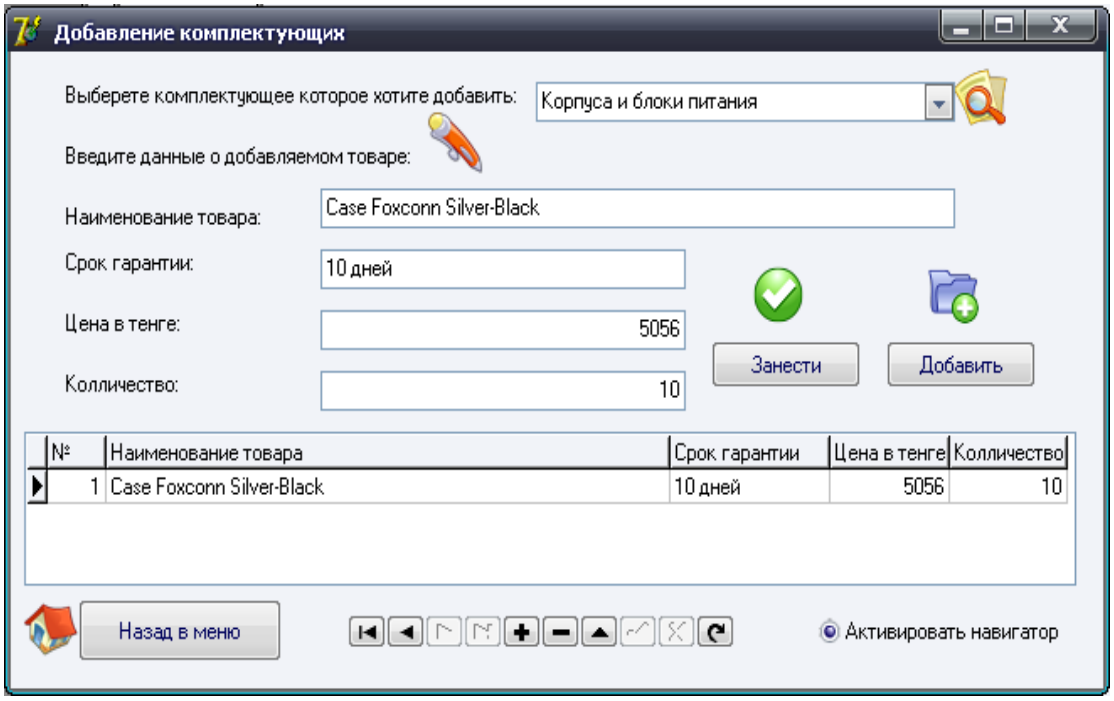

Рисунок 3.14 - Форма для добавления данных в Базу Данных

Компоненты, установленные на этой форме представлены в таблице 3.7.

Таблица 3.7 - Компоненты формы " Заправка картриджей "

Объект	Имя	Свойства	Значения
DBGrid	Таблица	Name	DBGrid1
		DataSource	DataSource1
		Color	clBtnFace
		Font	(TFont)
Image	Рисунок	Name	Image1
		Picture	(None)
		Color	clBtnFace
		Font	(TFont)
DataSource	Отображение данных	Name	DataSource1
		DataSet	ADOTable1
		AutoEdit	True
		Tag	0
ADOConnection	Связь с БД	Name	ADOConnection
		LoginPromt	False
		Connected	True

Окончание таблицы 3.7

Объект	Имя	Свойства	Значения
ADOConnection	Связь с БД	Provider	Microsoft.Jet.OLEDB.4.0
ADOTable	Управление таблицами	Name	ADOTable1
		Connection	ADOConnection1
		TableName	Компы
		Active	True
Button	Кнопка	Name	Button1
		Caption	OK
		Color	clBtnFace
		Font	(TFont)
ComboBox	Комбинированный список	Name	ComboBox1
		Items	(TStrings)
		Color	clBtnFace
		Font	(TFont)
Label	Метка	Name	Label1
		Caption	Цена в тенге:
		Color	clBtnFace
		Font	(TFont)
DBEdit	Текстовое поле	Name	DBEdit 1
		DataSource	DataSource1
		Color	clBtnFace
		Font	(TFont)
DBNavigator	Редактирование таблицы	Name	DBNavigator1
		DataSource	DataSource1
		Color	clBtnFace
		Font	(TFont)
XPManifest	Оформление	Name	XPManifest1
		Tag	0

3.3 Организация производства

3.3.1 Руководство программиста

Руководство системного программиста должно содержать следующие разделы:

1) Общие сведения о программном продукте. Дипломный проект "Разработка автоматизирование рабочего места специалита отдела продаж" был создан на языке программирования Delphi7. Программа работает только при наличии операционной программы Windows XP. Язык программирования - Delphi7. Программы Microsoft Office 2003.

2) Структура. Данная программа состоит из главного меню, называемая "Меню". Программа состоит из 17 форм. На главной форме программы, установлено меню программы, которое позволяет выбирать формы для регистрирования данных, просмотра данных и т.д. Связь между таблицами данных осуществляется с помощью диалоговых окон и программного кода.

3) Настройка. Файлы базы данных должны храниться, в папке программы "PC".

4) Проверка. Чтобы проверить работоспособность программы, нужно занести данные в каждую из форм, потом открыть базу в Access, и просмотреть выполнилось ли занесение.

Методы использованные в моей программе:

- Hide - закрыть форму;
- Show - открыть форму;
- Add Item -добавить в список;
- Clear - очистить список;
- Close - закрыть;
- Save - сохранить.

5) Переменные, используемые в программе.

По количеству:

- форм (Form) - 17;
- кнопок (Command Button) - 53;
- полей ввода (Edit) - 4;
- надписей (Label) - 97;
- комбинированных списков (Combo Box) - 8;
- полей ввода (DbEdit) - 44;
- флажок (DbCheck) - 4;
- комбинированных списков (DbCombo Box) - 1;
- рамка (Frame) - 13;
- оформление (XPManifest) - 17;
- редактирование таблицы (DBNavigator) - 4;
- управление таблицами (ADOTable) - 13;
- связь с БД (ADO Conection) - 13;

-отображение данных (Data Source) - 13;
-рисунок (Image) - 103.

6) Размер программы: 32,4 Mb.

3.3.2 Руководство пользователя.

Установка

Данная программа не требует установки. Для работы с ней необходимо просто скопировать папку  на свой компьютер.

Запуск

Для запуска данного программного продукта необходимо щелкнуть по значку  программы.

После этого откроется главное меню программы (Рисунок 3.15).

Рисунок 3.15 - Главное окно

Для добавления комплектующих нужно открыть форму "Добавление комплектующих" (Рисунок 3.16). Для заполнения данных выберете в комбинированном списке комплектующее, которое желаете добавить, далее введите данные о комплектующем и нажмите "Занести". Если нужно добавить еще, нажмите на кнопку "Добавить".

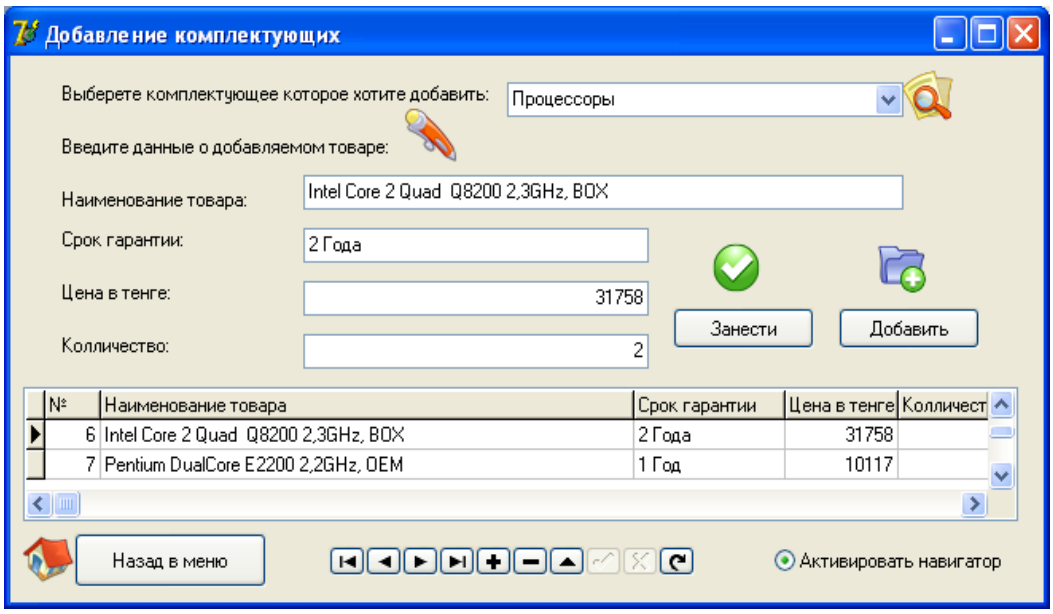

Рисунок 3.16 - Форма "Добавление комплектующих"

С помощью этой же формы просматривается наличие товара на складе.

Для регистрации компьютера или другого изделия на ремонт, нужно открыть форму "Регистрация на ремонт" (Рисунок 3.17). Здесь нужно ввести данные клиента и нажать на кнопку "ОК". Так же здесь есть список всех предыдущих клиентов, регистрирующихся ранее на ремонт. Среди них можно выполнить поиск по номеру квитанции. На форме есть кнопка "Просмотр и печать квитанции", если щелкнуть по ней, то перед вами откроется лист Excel с квитанцией.

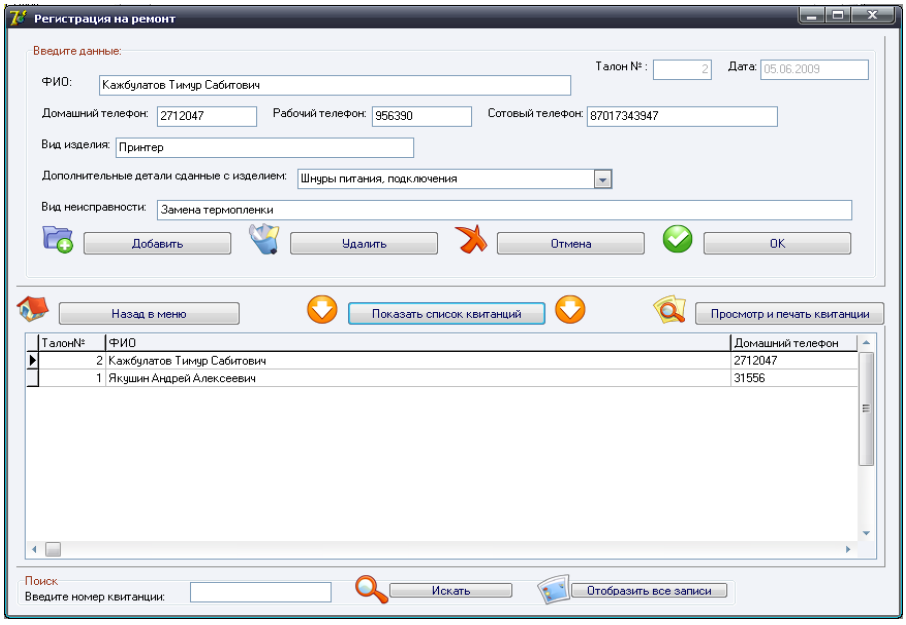

Рисунок 3.17 - Форма "Регистрация на ремонт"

Чтобы зарегистрировать картридж оставленный на заправку нужно воспользоваться формой "Заправка картриджей" (Рисунок 3.18). Здесь вы вводите данные картридже и выбираете, какие работы будут выполнены мастером при заправке.

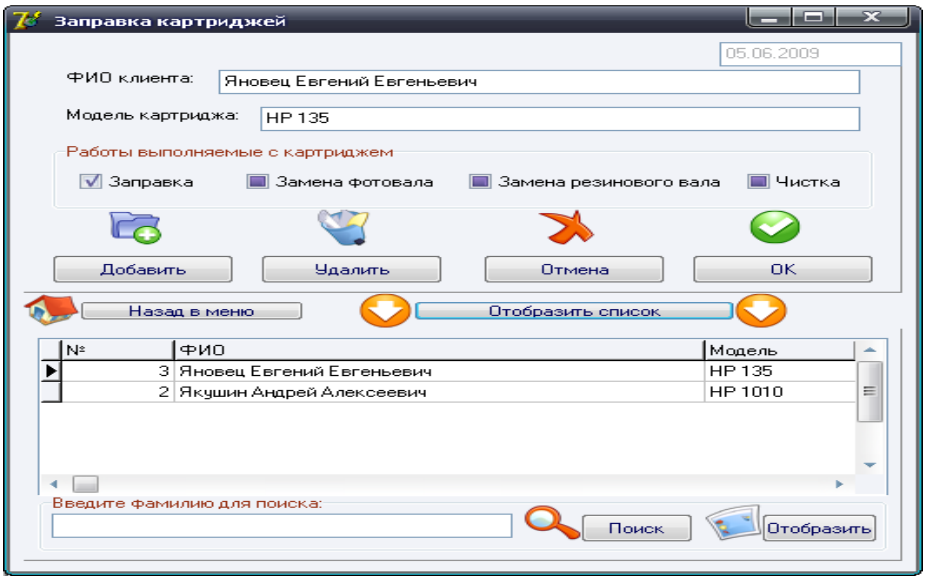

Рисунок 3.18 - Форма "Заправка картриджей"

4 Экономический анализ

4.1 Основные понятия экономики и рыночных отношениях

Фундаментальной проблемой для экономических отношений и практики был и остается способ согласования действий участников общественного производства. От выбора того или иного способа зависят, в конечном счете, суть и особенности функционирования различных социально-экономических систем. Причем сам этот выбор весьма ограничен. По существу он сводится к давно сформулированной экономической наукой альтернативе: "план" или "рынок". Откуда же вытекает такая жесткая постановка вопроса?

Дело в том, что труд человека, производящего товары, имеет двойственный характер. С одной стороны, это частный труд, ибо товаропроизводитель работает на свой страх и риск, занимаясь вроде бы сугубо индивидуальным делом. С другой стороны, этот частный труд должен непременно получить общественное признание. А это происходит лишь в процессе обмена товара на рынке.

Отсюда и возникает фундаментальный вопрос экономической теории и практики, который можно сформулировать следующим образом: каков тот общественный механизм, который заставляет работника выполнять работу, нужную для удовлетворения общественных потребностей, причем с высоким качеством и эффективностью?

Эта задача может быть решена только двумя принципиально разными способами:

- “Административный” - предполагает руководство экономикой из единого центра с помощью прямых команд - что производить, в каких количествах, куда и по какой цене поставлять. История показала, что создание такого, на первый взгляд, простого механизма сталкивается с не преодолимыми трудностями;
- “Рыночный ” - основа на системе экономических стимулов и санкций, которые исходят не от руководящего органа , а от самого потребителя. Именно потребитель “голосует” своей тенге “за” или “против” данного товара, тем самым, признавая или отрицая общественную значимость заключенного в нем труда.

Рыночная экономика формирует и соответствующий ее сути тип личности - энергичной, инициативной, способной к творчеству и оправданному риску. Вместе с тем субъекты рыночных отношений далеко не идеальны. Им свойствен эгоизм, предпочтение собственных интересов интересам общества и т.д. А потому цивилизованная рыночная экономика немыслима без хорошо разработанных тонких инструментов социальной регуляции, правовых и общественных институтов, призванных смягчать, обуздывать крайности рыночных отношений.

Однако и в этом случае рыночная экономика не страдает от застоя. Развитие идет стихийно, но чрезвычайно бурно, иногда сопровождаясь резкими социальными сдвигами и периодическими кризисами.

4.2 Расчет затрат на разработку информационных технологий

Под информационными технологиями понимаются экономические информационные системы (ЭИС), программные продукты (ПП), информационные базы данных и т.д.

Расчет полных затрат на разработку проектного решения в виде информационных технологий ($С_{пі}$) осуществляется по формуле

$$С_{пі} = З_{фот} + З_{сзі} + М_і + А + Р_{мі} + П_{зі} + Р_{ні} \quad (4.1)$$

где $З_{фот}$ - общий фонд оплаты труда разработчиков, тенге;

$З_{сзі}$ - отчисления по социальному налогу, тенге;

$М_і$ - затраты на материалы, тенге;

А - амортизация;

$Р_{мі}$ - затраты, связанные с эксплуатацией техники, тенге;

$П_{зі}$ - прочие затраты, тенге;

$Р_{ні}$ - накладные расходы, тенге.

Размер фонда оплаты труда разработчиков ($З_{ФОТ}$) рассчитывается по формуле

$$З_{ФОТ} = З_{оі} + З_{ді} \quad (4.2)$$

где $З_{оі}$ - основная заработная плата, тенге;

$З_{ді.}$ - дополнительная заработная плата, тенге.

Затраты на оплату труда зависят от объема и трудоемкости разработки программного обеспечения.

Общий объем (V_0) программного продукта определяется исходя из количества и объема функции, реализуемых программой

$$V_0 = \sum_{j=1}^{n} V_i \quad (4.3)$$

где V_i - объем отдельной функции ПО;

n - общее число функций.

Исходя из "приложения Б" уточненный объем ПО включает в себя следующие номера функции: 101, 102, 103, 104, 107, 109, 202, 203, 204, 206, 208, 401, 402, 403, 707.

Тогда уточненный объем ПО будет равен

$V_0 \approx 9000$.

Общая трудоемкость небольших проектов рассчитывается по формуле

$$T_0 = T_н * K_c * K_m * K_н \quad (6.4)$$

где $T_н$- нормативная трудоемкость;

$К_с$ - коэффициент, учитывающий сложность ПО;

$К_m$ - поправочный коэффициент, учитывающий степень использования при разработке стандартных модулей;

$К_н$ - коэффициент, учитывающий степень новизны ПО.

Посредством коэффициента сложности учитываются затраты труда, связанные со сложностью ПП. На основе данных таблицы 4.1 коэффициент сложности составляет 0,18.

Т а б л и ц а 4.1 - Дополнительные коэффициенты сложности ПО

Характеристика ПО	Значения $К_с$
1. Функционирование ПО в р7асширенной операционной среде (связь с другими ПО)	0,08
2. Интерактивный доступ	0,06
3. Обеспечение хранения, ведения и поиска данных в сложных структурах	0,07
4. Наличие у ПО одновременно нескольких характеристик по табл.Г4.1, приложение Г 4.1 2 характеристики 4.2 3 характеристики 4.3 Свыше 3-х характеристик	 0,12 0,18 0,26

Коэффициент, учитывающий степень использования при разработке ПО стандартных модулей. Степень использования в разрабатываемом ПО стандартных модулей определяется их удельным весом в общем объеме проектируемого продукта. В данном дипломном проекте степень охвата реализуемых функций разрабатываемого ПО стандартными модулями, типовыми программами и ПО до 20%, следовательно исходя из таблицы 4.2 $К_с$ = 0,9.

Т а б л и ц а 4.2 - Значения поправочного коэффициента, учитывающего использование стандартных модулей типовых программ и ПО

Степень охвата реализуемых функций разрабатываемого ПО стандартными модулями, типовыми программами и ПО	Значения
1. От 60% и выше	0,6
2. От 40% до 60%	0,7
3. От 20% до 40%	0,8
4. До 20%	0,9
5. Типовые программы и ПО не используемые для реализации функций разрабатываемого ПО	1,0

Поправочный коэффициент, учитывающий новизну разрабатываемого ПО ($К_н$) определяется на основе данных представленных в таблице 4.3 и составляет 1,0.

Т а б л и ц а 4.3 - Поправочные коэффициенты, учитывающие новизну ПО($К_н$)

Категория Новизны	Степень новизны	Использование		Значение $К_н$
		На основе нового типа ПК	В среде новой ОС	
А	Принципиально новые ПО, не имеющие доступных аналогов	+ - + -	+ + - -	1,75 1,6 1,2 1,0
Б	ПО, являющиеся развитием определенного параметрического ряда ПО	+ - +	+ - -	1,0 0,9 0,8
В	ПО, являющиеся развитием определенного параметрического ряда ПО, разработанных для ранее освоенных типов конфигурации ПК и ОС	-	-	0,7

Нормативная трудоемкость ПО ($Т_н$) определяется на основе принятого в расчет V_y и категории сложности, которая уточняется с учетом сложности и

новизны проекта и степени использования стандартных модулей при разработке.

В соответствии с этим, согласно укрупненным нормам времени на разработку ПО ($T_н$) в зависимости от уточненного объема ПО (V_0) и группы сложности (Приложение В): объем ПО(строки исходного кода, LOC) 9000, категория сложности ПО 2-я - $T_н$ =240, категория сложности ПО 37.

Следовательно $T_о$ будет равно

$$T_0 = 240 * 0,18 * 0,9 * 1 = 38,88(чел./дн.)$$

Численность исполнителей проекта ($Ч_р$) рассчитывается по формуле

$$Ч = T_0 / (T_р * Ф_{эф}) \quad (6.5)$$

где $Ф_{эф}$ - эффективный фонд времени работы одного работника в течение года (дн.);

$T_о$ - общая трудоемкость разработки проекта (чел./дн.);

$T_р$ - срок разработки проекта (лет).

Срок разработки проекта ($T_р$) определяется по формуле

$$Ч_р = T_0 / (T_р * Э_{эф}) \quad (6.6)$$

где $Ч_р$ - плановое число разработчиков.

Эффективный фонд времени работы одного работника ($Ф_{эф}$) рассчитывается по формуле

$$Ф_{эф} = Д_г - Д_п - Д_в - Д_о \quad (6.7)$$

где $Д_г$ - количество дней в году;

$Д_п$ - количество праздничных дней в году;

$Д_в$ - количество выходных дней в году;

$Д_о$ - количество дней отпуска.

В соответствии производственным календарем на 2014 год: $Д_г$ - 365; $Д_п$ = 14; $Д_в$ - 103; $Д_о$ - 10, то по формуле (6.7) получим

$$Ф_{эф} = 365 - 14 - 103 - 10 = 228 \text{ дней}$$

Плановое число разработчиков $Ч_р$ = 1, следовательно по формуле (6.6)

$$T_р = 38,88 / (1 * 228) = 0,17 \text{ года} = 62 \text{ дней}$$

Таким образом, согласно произведенным расчетам и в соответствие с формулой (6.5)

Ч = 38,88 / (0,17 * 228) = 1 чел

Основная заработная плата исполнителей на конкретное ПО рассчитывается по формуле

$$З_{оі} = \sum Т_{чі} * Т_ч * Ф_п * К \quad (6.8)$$

где n - количество исполнителей, занятых разработкой конкретного ПО;
$Т_{чі}$ - часовая тарифная ставка i-го исполнителя (тыс.тенге);
$Ф_п$ - плановый фонд рабочего времени i-го исполнителя (дней), 62 дня;
$Т_ч$ - количество часов работы в день (час), 8 часов;
К - коэффициент премирования, составляет 1,2.

По данным о специфике и сложности выполняемых функций составляется штатное расписание группы специалистов-исполнителей, участвующих в разработке ПО, с определением образования, специальности, квалификации и должности (Таблица 4.4).

Т а б л и ц а 4.4 - Сведения по работникам, задействованным в проекте

Специалист - Исполнитель	Количество, человек	Заработная плата в месяц, тенге
Программист	1	60 000
Итого		60 000

Часовая тарифная ставка рассчитывается путем деления месячной тарифной ставки на установленную при 40-часовой недельной норме рабочего времени расчетную среднемесячную норму рабочего времени в часах ($Ф_р$)

$$Т_ч = Т_н / Ф_р \quad (6.9)$$

где $Т_ч$- часовая тарифная ставка (тыс.тенге);
$Т_м$- месячная тарифная ставка (тыс.тенге).

Таким образом

$$Т_ч = \frac{60\,000}{168} = 357{,}1 \text{ тенге/час}$$

По формуле (6.8) можно рассчитать основную заработную плату исполнителей

$З_{оі}$ = 357,1 * 8 * 62* 1,2 = 212546 тенге

Дополнительная заработная плата составляет 10% от основной заработной платы и рассчитывается по формуле

$$З_{ді} = З_{оі} * Н_д/100 \quad (6.10)$$

где $Н_д$ - коэффициент дополнительной заработной платы разработчиков 22%

$З_{ді}$ =212 546 * 0,22 = 46760 тенге

Социальный налог составляет 11% (ст. 358 п. 1 НК РК) от дохода работника, и рассчитывается по формуле

$$З_{сзі} = (ФОТ-ПО) * 11\% \quad (6.11)$$

где ПО - пенсионные отчисления, которые составляют 10% от ФОТ и социальным налогом не облагаются

$$ПО = ФОТ * 10\% \quad (6.12)$$

Таким образом

ПО=212 546* 0,1 = 21 254,6 тенге,

$З_{сзі}$ = (212 546- 21 254,6) * 0,11 = 21 042 тенге

Затрат на материалы определяются по формуле

$$М_і = (З_{осн} * Н_{мз}) / 100\% \quad (6.13)$$

где $Н_{мз}$ - норма расхода материалов от основной заработной платы (3-5%)

$М_і$ = 212 546 * 0,042 = 8 926,9тенге

Амортизационные отчисления производятся по установленным нормам амортизации, выражаются, в процентах к балансовой стоимости оборудования и рассчитываются по формуле (6.14)

$$А = (С_{обор} * Н_А * N) / (365 * 100) \quad (6.14)$$

где $Н_а$ - норма амортизации (25 %);

$С_{обор.}$- первоначальная стоимость оборудования;

N - фактический срок эксплуатации оборудования, 60 дней.

Данные по стоимости оборудования представлены в таблице 4.5.

Т а б л и ц а 4.5 - Стоимость оборудования одного ПК с периферией

Наименование	Цена с НДС, тенге	Цена без НДС, тенге
Монитор	32000	28160
Материнская плата	13000	11264
Процессор	30000	26400
Видеокарта	23000	20240
HDD	12000	10560
DVD-RW	6500	5720
CPU Cooler	5500	4840
Операционная система	8000	7040
Клавиатура	1000	880
Мышь	1000	880
Принтер	8000	7040
Итого	140000	123024

$С_{обор.}$ = 139800 тенге

Тогда, согласно формуле (6.14)

$$А = \frac{139\,800 \cdot 25 \cdot 62}{365 \cdot 100} = 5936{,}7$$

Расходы по статье "Машинное время" ($Р_{мі}$) включают оплату машинного времени, необходимого для разработки и отладки ПО, которое определяется по нормативам (в машино-часах) на 100 строк исходного кода ($Н_{мв}$) машинного времени в зависимости от характера решаемых задач и типа ПК

$$Р_{мі} = Ц_{мі} * (V_{оі}/100) * Н_{мв} \qquad (6.15)$$

где $Ц_{мі}$- цена одного машино-часа (тыс.тенге);

$V_{оі}$- общий объем ПО (строк исходного кода);

$Н_{мв}$- норматив расхода машинного времени на отладку 100 строк исходного кода (машино-часов).

Согласно данным из Приложения Д: наименование подсистемы АС и ДОС - 1 и средний расход машинного времени, ч/100 сток кода составляет 12.

Таким образом

$Р_{Мі}$ = 357,1 * (9 000 / 100) * 11 = 353529 тенге

Расходы по статье "Прочие затраты" ($П_{зі}$) на конкретное ПО включают затраты на приобретение и подготовку специальной научно-технической информации и специальной литературы. Определяются по нормативу, разрабатываемому в целом по организации, в процентах к основной заработной плате

$$П_{зі} = З_{оі} * Н_{пз} / 100 \quad (6.16)$$

где $Н_{пз}$ - норматив прочих затрат в целом по организации в (%), в дипломной работе нужно брать 30% .

Подставляем все данные в формулу (6.20) получаем

$П_{зі}$= 212 546 * 0,3 = 63763,8 тенге

Затраты по статье "Накладные расходы" ($Р_{ні}$), связанные с необходимостью содержания аппарата управления, вспомогательных хозяйств и опытных(экспериментальных) производств, а также с расходами на общехозяйственные нужды ($Р_{ні}$), относятся на конкретное ПО по нормативу ($Н_{рн}$) в процентном отношении к основной заработной плате исполнителей. Норматив устанавливается в целом по организации

$$Р_{ні} = З_{оі} \times Н_{рн} / 100\% \quad (6.17)$$

где $Р_{ні}$- накладные расходы на конкретную ПО (тыс.тенге);

$Н_{рн}$- норматив накладных расходов в целом по организации в (%), в дипломной работе нужно брать 70% .

Подставляем все данные в формулу (6.21) получаем

$Р_{ні}$= 212 546 * 0,7 = 148782,2 тенге,

$С_{пі}$ = 814739,2 тенге

Результаты выполненных расчетов представлены в таблице 4.6.

Т а б л и ц а 4.6 - Затраты на разработку

Затраты на разработку	Условное обозначение	Значение, тенге	В процентах от общей суммы
Фонд оплаты труда	$З_{ФОТ}$	212546	26,09
Социальный налог	$З_{сзі}$	21254	2,61
Материалы	$М_і$	8926	1,1
Амортизация	А	5936	0,73
Машинное время	$Р_{мі}$	353529	43,39
Прочие затраты	$П_{зі}$	63763	7,83
Накладные расходы	$Р_{ні}$	148782	18,25
Итого		814739	100,0

4.3 Расчет цены программного продукта

Расчет цены ПП в организационно-экономической части дипломной работы предлагается производить следующим образом:

- если ПП разработан одной организаций по заказу другой и не предназначен для тиражирования, то затраты на разработку ПП считаются его себестоимостью, и при формировании цены применяется затратный метод;
- если ПП предназначен для тиражирования, то конечная цена определяется путем экспертных оценок на основании ценностного подхода с учетом текущих цен конкурентов (если существуют аналогичные ПП).

Расчет цены ПП, который разработан одной организаций по заказу другой и не предназначен для тиражирования, осуществляется по формуле

$$Ц_{ПП} = З_{РПР} + П_п + НДС \quad (6.27)$$

где $Ц_{ПП}$ - цена программного продукта, тенге;

$З_{РПР}$ - затраты на разработку проектного решения, в данном случае программного продукта, тенге;

$П_п$ - планируемая прибыль, тенге;

НДС - налог на добавленную стоимость, тенге.

Планируемая прибыль рассчитывается по формуле

$$П_п = З_{РПР} + R_{НПП} \quad (6.28)$$

где $R_{НПП}$ - нормативная рентабельность ПП, определяемая организацией.

НДС, начисленный на ПП, определяется следующим образом

$$НДС = (З_{РПР} + П_п)* k_{НДС} \quad (6.29).$$

где $k_{НДС}$ - ставка налога на добавленную стоимость.

Подставляем все значения в формулы (6.27) - (6.29) и получаем.

По формуле (6.28) учитывая, что $З_{РПР} = С_{пi}$, $R_{НПП}$ - это процент рентабельности по отношению к себестоимости составляет 20%

$R_{НПП} = С_{пi} * 0,20 = 814\,739,2 * 0,20 = 162947,84$ тенге,

$П_п = 814739,2 + 162\,947,84 = 977687,04$ тенге

Подставив данные в формуле (6.29) получаем

НДС = (814739,2 + 977687,04) * 0,12 = 215091,15 тенге

Подставив данные в формуле (6.27) получаем

$Ц_{ПП} = 814739,2 + 977687,04 + 215\,091,15 = 2007517,39$ тенге

4.4 Расчет показателей сравнительной экономической эффективности (ожидаемого годового экономического эффекта от внедрения ИС, срока окупаемости и коэффициента экономической эффективности)

Величина ожидаемого годового экономического эффекта от внедрения ИС рассчитывается по формуле

$$Э_г = Э_{уг} - К \times E_н \qquad (6.30)$$

где $Э_г$ - ожидаемый годовой экономический эффект, тенге;

$Э_{уг}$ - ожидаемая условно-годовая экономия, тенге;

К -капитальные вложения, тенге;

$E_н$ - нормативный коэффициент экономической эффекта от внедрения капитальных вложений.

Нормативный коэффициент экономической эффективности капитальных вложений определяется по формуле

$$E_н = 1 / T_н \qquad (6.31)$$

где $T_н$ - нормативный срок окупаемости капитальных вложений, лет.

Нормативный срок окупаемости капитальных вложений, принимается исходя из срока морального старения технических средств и проектных решений ИС ($T_н$ = 1,2,3 ... n), для программных продуктов срок окупаемости принимаем равным 4 года.

Ожидаемая условно-годовая экономия определяется по формуле

$$Э_{уг} = С_1 - С_2 \quad (6.32)$$

где $Э_{уг}$ и $Э_{уг}$ - показатели текущих затрат по базовому и внедряемому вариантам, тенге

$Э_{уг}$ = 4260000- 2150000 =2110000

Капитальные вложения равны

К = 814 739,2 тенге

$T_н$ = 4 года, $E_н$ = 1 / 4 = 0,25

Величина ожидаемого годового экономического эффекта от внедрения ИС равна

$Э_г$ = 2110000 - 814739,2 * 0,25 = 1 906 315,2

Расчетный коэффициент экономической эффективности капитальных вложений составляет

$$E_р = Э_{уг} / К \quad (6.33)$$

где $E_р$ - расчетный коэффициент экономической эффективности капитальных вложений;

$Э_{уг}$ - ожидаемая условно-годовая экономия, тенге;

К - капитальные вложения на создание системы, тенге.

Расчетный срок окупаемости капитальных вложений составляет

$$T_р = 1 / E_р \quad (6.34)$$

где $E_р$ - коэффициент экономической эффективности капитальных вложений.

$E_р$ = 2110000 / 814 739,2 = 2,59,

$T_р$ = 1 / 2,59 = 0,39 лет = 4,68 мес.

Результаты выполненных расчетов сводятся в таблицу 4.7.

Т а б л и ц а 4.7 - Показатели сравнительной экономической эффективности от внедрения программного продукта

Наименование показателей	Значение
Условная годовая экономия затрат, тенге	2 110 000
Коэффициент экономической эффективности капитальных вложений (E_p)	2,59
Срок окупаемости капитальных вложений (T_p), лет	0,39

Анализируя результаты внедрения программы "Разработка автоматизирование рабочего места специалита отдела продаж" по обеспечению компьютерной техникой можно сделать следующие выводы, что эта программа поможет:

1) ускорить доставку информации к центру управления производством;
2) сократить затраты на обслуживание этой системы;
3) сократить обслуживающий персонал;
4) своевременно изменить технологию производства;
5) своевременно доставить информацию по изменению выполнения производственных процессов;
6) своевременно сообщить изменения в директивных документах;
7) ускорить изменения в достижениях науки, техники и передового опыта.

5 Безопасность жизнедеятельности

5.1 Анализ потенциально-опасных вредных факторов, воздействующих на обслуживающий персонал во время работы

5.1.1 Специфика организации рабочих мест при работе за компьютером

Применение оргтехники освобождает человека от непроизводительной работы, связанной с обработкой информации, изменяет характер его труда. Однако при этом увеличивается доля умственного и нервно-напряженного труда, возрастает психоэмоциональная нагрузка. При значительной трудовой нагрузке, нерациональной организации работы и неблагоприятных факторах производственной среды быстро снижается работоспособность операторов ЭВМ, уменьшается производительность труда и ухудшается качество работы.

Опасный производственный фактор - производственный фактор, воздействие которого на сотрудника в условиях производства приводит к травме или другому внезапному резкому ухудшению здоровья.

Вредный производственный фактор - производственный фактор, воздействие которого на работающего в условиях производства приводит к заболеванию или снижению работоспособности. В зависимости от уровня и продолжительности воздействия вредный производственный фактор может стать опасным.

Опасность компьютера для здоровья состоит в том, что все вредные факторы дают о себе знать спустя некоторое время: гиподинамия, длительно повторяющиеся однообразные движения, электромагнитное и прочее излучение, снижение остроты зрения, нервные расстройства, шум. Для снижения этих воздействий в офисе предприятия ИП "Адилет" рабочие места с ПК размещены в изолированных помещениях. Все виды оборудования имеют гигиенический сертификат. Конструкция оборудования его дизайн, эргономические параметры обеспечивают надежно и комфортное с считывание отображаемой информации. Конструкция рабочий мебели обеспечивает возможность индивидуальной регулировки, соответственно росту работающего.

Производственная санитария - это система организационных мероприятий и технических средств, предотвращающих или уменьшающих воздействие на работающих вредных производственных факторов. Оздоровление воздушной среды проводится при помощи вентиляции помещений, так как при работе ПК возникают повышенное содержание вредных веществ в воздухе. Для защиты от внешнего шума на окнах офисных помещений установлены стеклопакеты. На предприятии соблюдаются режим труда и отдыха осуществляется контроль за трудовым процессом. Психофизиологические факторы предприятия отражают

напряженность и тяжесть труда, так как морально-психологический климат в коллективе. Экономические факторы на предприятии включают в себя систему оплаты и стимулирование труда.

Микроклимат производственных помещений - климат внутренней среды этих помещений, который определяется действующими на организм человека сочетаниями температуры, влажности и скорости движения воздуха, а также температурой окружающих поверхностей.

Эти параметры микроклимата каждый в отдельности и в различных сочетаниях оказывают огромное влияние на ход физиологических процессов в организме и, в конечном счете, на самочувствие и работоспособность человека.

Температура воздуха производственных помещений зависит от температур атмосферного воздуха и окружающих поверхностей, а последние в свою очередь - от времени года и характера технологии производства соответственно.

В помещениях ИП "Адилет" температура воздуха считается оптимальной 20 - 22^{o}C.

В офисных помещениях предприятия ИП "Адилет" микроклимат обеспечивается при помощи центрального отопления и кондиционирования воздуха. Скорость движения воздуха регулируется при помощи естественной вентиляции.

Естественная вентиляция осуществляется путем открытия дверей и окон.

Обеспечение влажности на предприятии осуществляется путем влажной уборки и озеленения помещений и кондиционированием.

Важную роль играет планировка рабочего места, которая должна удовлетворять требованиям удобства выполнения работ и экономии энергии и времени работника, рационального использования производственных площадей и удобства обслуживания оргтехники, соблюдения правил охраны труда.

Общие требования к компоновке рабочего места оператора компьютера регламентируются в Санитарных правилах и нормах "Гигиенические требования к видео дисплейным терминалам, персональным электронно-вычислительным машинам и организации работы".

Освещенность

90% информации человек получает через органы зрения. Свет оказывает положительное влияние на обмен веществ, сердечно-сосудистую систему, нервно-психическую сферу. Рациональное освещение способствует повышению производительности труда, его безопасности. При недостаточном освещении и плохом его качестве происходит быстрое утомление зрительных анализаторов, повышается травматизм. Высокая яркость освещение вызывает явление ослепления, нарушение функции глаза.

5.1.2 Безопасные приемы и меры профилактики при работе на ПК

Электромагнитное излучение

Компьютер испускает электромагнитное излучение. Мониторы соответствующие одному из стандартов ТСО не представляют опасности для человека по уровню электромагнитного излучения. Стандарт электромагнитного излучение на расстоянии 30 см от экрана ограничивается 200nT. Факс в предприятии на расстоянии 30 см создает поле в 300-600 nT, а копировальная машина - 2100-3100 nT. Всемирная организация здравоохранения подтвердила отсутствие взаимосвязи между воздействием электромагнитного излучения компьютеров и заболеваемостью пользователей ПК раковыми, сердечно-сосудистыми и другими заболеваниями.

Ухудшение зрения

С применением компьютеров стал выявляться "компьютеро-зрительный синдром" (Computer vision syndrome), который характеризуется следующими заболеваниями:

1) снижение остроты зрения;
2) нарушение аккомодации;
3) двоение предметов;
4) быстрое утомление при чтении,
5) жжение в глазах, чувство "песка;
6) боли в области глазниц и лба;
7) покраснение глазных яблок.

При работе за компьютером глаза находятся в постоянном напряжении. Связанно это с тем что, монитор является источником света, что не привычно для человеческого глаза, так как за долгое время эволюции он приспособился для восприятия отраженного от объектов света. Еще изображение на мониторе отличается от естественного тем, что оно имеет значительно меньший контраст, который еще больше уменьшается за счет внешнего освещения. Изображение состоит из дискретных точек - пикселей, которые имеют не ступенчатый, а плавный перепад яркости к фону.

Изображение на мониторе мерцающее (с определенной частотой), которое очень раздражает глаза. Поэтому необходимо периодически делать перерывы (отдых для глаз), чаще мигайте, чтобы глаза не были сухими (это актуально и при использовании TFT мониторов, просто при их использовании утомление наступает позже).

Большую нагрузку орган зрения испытывает при наборе текстов. Связано это с тем, что пользователь постоянно переводит взгляд с экрана на текст, а при отсутствии навыков "слепой печати" еще и на клавиатуру. Все это находиться на разном расстоянии и по-разному освещено. Снизить такую нагрузку помогают специальные клипсы, которые крепятся на монитор и удерживают лист бумаги рядом с экраном. Используйте такое приспособление.

Насчет расстояния до монитора - было уже сказано выше. Уменьшать его нельзя, для того чтобы не увеличивать воздействие излучений монитора. Сильно увеличивать расстояние тоже нельзя. Если надо будет всматриваться в изображение то это вызовет напряжение глаз. Не следует стремится к высоким разрешениям. Для 15 дюймовых мониторов оптимальное разрешение 800 на 600 точек, для 17" - 1024 на 768.

Гиподинамия

Профилактика: больше двигайтесь, чаще устраивайте перерывы. Каждые 1-1,5 часа прерывайтесь на 5-10-минут. Во время перерыва, в зависимости от нахождения рабочего места, можете выйти на улицу, поднимитесь по лестнице на другой этаж, сделайте несколько наклонов вперед, сомкните руки на затылке и одновременно, руками тяните голову вперед, а головой, наоборот, пытайтесь откинуться назад.

5.1.3 Электробезопасность рабочих мест

Проходя через организм человека электрический ток оказывает следующие воздействия: термическое действие - проявляющееся в ожогах отдельных частей тела, нагреве до высоких температур кровеносных сосудов, крови, нервов, сердца, мозга, что вызывает серьезное расстройство органов; электролитическое действие - разложение органической жидкости (лимфы и крови) с нарушением ее состава; механическое действие - (динамическое) расслоение, разрыв тканей организма (мышц сердца, сосудов) в результате электрически-динамического эффекта; мгновенного взрывоподобного образования пара от перегретой током тканевой жидкости и крови; биологическое - раздражение живых тканей организма; нарушение внутренних биоэлектрических процессов, протекающих в нормально-действующем организме.

Несоблюдение правил техники безопасности может привести ко всем вышеуказанным последствиям. Для предотвращения возможности поражения работающих электрическим током на предприятии ИП "Адилет" проводятся следующие мероприятия: каждый компьютер защищен специальным кожухом сделанный из пластмассы; внутри системного блока находятся комплектующие, которые покрыты специальным диэлектриком, шлейфы покрыты резиновой поверхностью, провода исходящие из блока питания также заизолированы; применяется защитное заземление на все электроустановки; при работе за ПК проверяют исправность розеток и соединительных проводов; два раза в год измеряют сопротивление изоляции проводов и сопротивление заземления. Эти мероприятия осуществляются энергетической службой предприятия.

Одно из основных задач предприятия ИП "Адилет" это обучение вопросам охраны труда рабочих и служащих. Это важнейшее

профилактическое мероприятие по предупреждению несчастных случаев и профессиональных заболеваний на производстве. Обучение охране труда на предприятии предусматривает следующие инструктажи

- Вводный инструктаж проводится со всеми вновь принятыми работниками, в том числе и временными работниками, а также направленными на производственную практику студентами и учащимися. Вводный инструктаж по технике безопасности получают независимо от характера производства, квалификации и стажа работы. По окончании вводного инструктажа каждому работнику выдается под расписку инструкция по технике безопасности, относящаяся к тому виду работы, на которую он будет направлен. Инструктаж проводится по типовой инструкции директор предприятия.

- Первичный инструктаж на рабочем месте проводят со всеми вновь принятыми на предприятие, а также с работниками, переведенными из другого подразделения или теми, кому поручено новая работа. Проводит его по типовой программе главный специалист-инструктор предприятия ИП "Адилет".

- Внеплановый инструктаж проводятся при несчастных случаях на предприятии и при изменениях технологического процесса, замене или модернизации оборудования, при грубых нарушениях работниками правил техники безопасности.

- Повторный инструктаж проводит главный специалист-инструктор предприятия ИП "Адилет" со всеми работниками не реже одного раза в месяц.

Вводный инструктаж регистрируется в журнале регистрации вводного инструктажа, все остальные инструктажи в журнале регистрации инструктажа на рабочем месте. При регистрации указывается дату инструктажа, его вид ставят свои подписи лица, проводившие инструктаж и лицо, слушавшие его. Подпись давшего инструктажа означает допуск к работе, и соответствует ответственности за это, подпись получившего - обязанность строго выполнять все требования охраны труда.

5.1.4 Пожаробезопасность рабочих мест

Пожарной безопасностью называется такое состояние объекта, при котором с установленной вероятностью исключается возможность возникновения и развития пожара и воздействие на людей опасных факторов пожара, а также обеспечивается защита материальных ценностей.

Причины возникновение пожара:

1) наличие неисправного электрооборудования;
2) неосторожное обращения с огнём;
3) курения в пожароопасных помещениях;
4) электробытовые приборы.

Одной из вероятных причин пожара в помещении предприятия может быть короткое замыкание, влекущее за собой возгорание электропроводки. Для

его предупреждения вся вычислительная техника, а также прочие электрические устройства оборудованы плавкими предохранителями, а на входе электросети предусмотрен автомат защиты. На предприятии предусмотрено наличие в пределах досягаемости первичных средств тушения пожара (огнетушителей) для локализации огня собственными средствами до приезда команды пожарной охраны. Разработан план экстренной эвакуации персонала при возникновении загорания.

В офисном здании где находится предприятие ИП "Адилет" имеется противопожарная сигнализация, которая предупреждает рабочий персонал в случае пожара. В каждом помещении вывешены "Инструкции по пожарной безопасности". На улице предусмотрены пожарные гидранты.

Для борьбы с небольшими возгораниями, в здании предприятия установлен пожарный щит, а на каждом этаже ящик с огнетушителями. Пожарный щит содержит

1) ящик с песком вместимостью $1м^3$, укомплектованный двумя совковыми лопатами;
2) 4 лома;
3) 6 багров;
4) 2 топора;
5) 2 порошковых огнетушителей ОП - 250.

Пожарная профилактика - это комплекс организационных и технических мероприятий, направленных на обеспечение безопасности людей, на предотвращение пожаров, ограничений его распространения, а так же на создание условий для успешного тушения пожаров.

Действия персонала при возникновении пожара:

1) вызвать пожарную службу по телефону 01;
2) во избежание повторного возгорания или несчастного случая отключить электросеть здания;
3) обеспечить эвакуацию персонала согласно плану эвакуации;
4) обеспечить возможность подъезда машин спец. служб.

5.2 Расчетная часть

5.2.1 Расчет освещения

Обосновываю систему освещения и тип светильника, выбираю люминесцентные лампы, которые создают в производственных помещениях искусственный свет, приближающийся к естественному, более экономичны и имеют большой срок службы. Выбираем светильник типа PRS/S 2*36 .

Выбираю высоту подвеса светильника H=3 м.

Выбираю оптимальное значение расстояние между светильниками Lопт=1,60 м

Определяю наивыгоднейшее расстояние между светильниками

L= 1,60 * 3=4,8

Принимаю схему размещения светильников (симметричное)
Определяю число рядов светильников

m=12 / 4,8=2,5=3 ряда

Нахожу расстояние от стен до светильника

Lст. = 1,2 (м)

Рассчитываю расстояние между рядами

La= (12-2*1,2) / (3-1)=4,8

Вычислить расстояние между светильником в ряду

$L_в=4,8^2/4,8=4,8$

Нахожу число светильников в ряду

n_1= ((14-2*1,2) / 4,8) +1 = 3

Уточняем расстояние между светильниками в ряду

$L_в$=(14-2*1,2)/(3-1)=5,8

Общее число светильников

n =3*3=9

Определяю индекс помещения

i=(14*12)/(3*(14+12)=2,15

По таблицам нахожу коэффициент отражения стен и потолка

$P_{потолка}$=70%, коэффициент отражения рабочей поверхности

$P_{раб.пов.}$=30%

Коэффициент использования осветительной установки типа PRS/S η=50%=0,5.

По нормам освещенности принимаю

E_{min}=50 люкс/м^2

Находим по таблице коэффициент неравномерности распределения светового потока z=1,1

Из таблиц выбираю коэффициент запаса K=1,5 и определяю расчетный световой поток 1-ой лампы

$Ф_{пр}$= (50 * 168 *1,5 * 1,1) / (9 * 0,5)=3080 лм

Рассчитываю установленную мощность

$P_{уст}$ =80*9=720 Вт

$P_{лампы}$=80 Вт

Определяю удельную мощность

$P_{уд}$=720 : 168=4,3 Вт/м^2

Схема распределения светильников в помещении представлен на рисунке 5.1.

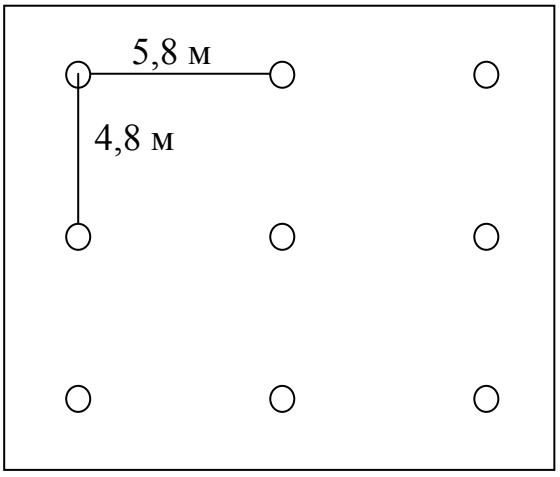

Рисунок 5.1 - Схема распределения светильников в помещении

Вывод: при заданной площади помещения и с учетом специфики проведения производственных работ в нем было рассчитано необходимое

количество электроосветительных приборов для поддержки необходимого нормируемого уровня освещенности. При расчетах было выбрано 9 светильников с лампой накалывания напряжением 220В, мощностью 720Вт. Выявлено, что условия освещенности приемлемы для данного помещения.

5.2.2 Расчет уровня шума от персонального компьютера

Основным источником шума является системный блок ПК. Шум издают такие механические устройства, как жесткий диск, CD/DVD приводы, а так же кулеры (вентиляторы). Принтер и сканер тоже относятся к источникам шума, но ввиду того что они используются крайне редко, рассматриваться не будут.

Жесткий диск. В системе используется один жесткий диск, со скоростью вращения шпинделя 7200 об/мин. Согласно паспорту максимальный уровень шума не превышает 25 дБ при бездействии, и 32 дБ во время дисковой активности (чтение/запись/поиск).

CD/DVD приводы. В системе установлено два привода - DVD-RW. Максимальный шум из него при максимальных оборотах вращения шпинделя не должен превышать 40-45 дБ, к счастью в нынешние времена пользоваться этими приводами приходится крайне редко. Установленный FDD привод не используется и отключен.

Системы активного охлаждения. Для обеспечения циркуляции воздуха внутри системного блока и качественного охлаждения компонентов системы, в корпусе насчитывается 6 кулеров разных размеров и скоростей вращения, более подробное описание ниже.

- Кулер на процессоре - 1шт., Ø 60мм, 2700об/мин, 22дБА.
- Кулер на северном мосту материнской платы - 1шт., Ø 50мм, 4500об/мин, 23дБА.
- Кулер видеокарты - 1шт., Ø 40мм, скорость вращения и уровень шума неизвестны, но даже при выключенном компьютере и других источниках шума, шум этого кулера едва слышим.
- Кулер блока питания - 1шт., Ø 80мм, 4000об/мин, 33дБА.
- Корпусные кулеры - 2шт., 80мм, 2500об/мин, 28дБА.

Суммарный шум будет равен

$$L_{сум} = 10 * lg(10^{32/10} + 10^{22/10} + 10^{23/10} + 10^{33/10} + 10^{28/10}) = 36{,}6 \text{дБА}$$

Вывод: в компьютерном помещении уровень шума не должно превышать 50дБА. По полученным подсчетам, уровень шума, производимый компьютерами в помещении не превышает максимально допустимый уровень шума.

Заключение

Моя программа "Разработка автоматизированного рабочего места специалиста отдела продаж " позволяет выводить цену и количество товара на складе, вводить в базу определенный товар его сумму и количество, печать квитанции на ремонт, печать гарантийного талона на приобретенный товар.

Программа создана на языке программирования Delphi 7, что намного облегчает работу.

1) Сокращает время и увеличивает трудоспособность.
2) Качественно и надежно хранит информацию.
3) Программа позволяет также обновлять базу данных.
4) Программа позволяет заполнять выходные документы.
5) Программа позволяет хранить гарантийные талоны.

Моя программа позволяет ежедневно принимать заказ всех покупателей.

Кроме того: уменьшается время на поиски и добавления товара, облегчается труд продавца, все покупатели будут довольны быстрым обслуживанием, автоматически будет подсчитываться заказ на определённую сумму в зависимости от наименования и количества заказываемого оборудования.

Список использованной литературы

1. Епанишников А. М., Епанишников В. А. Программирование в среде Turbo Pascal 7.0 М.: Диалог МИФИ, 1998.

2. Зуев В. А. Turbo Pascal 6.0, 7.0 М.: Веста. Радио и связь, 1998.

3. Коцюбинский А.О., Грошев С.В. Язык программирования Delphi 5. - М.: Издательство Триумф, 1999.

4. Леонтьев В. Delphi 5. - М.: Олма-Пресс, 1999.

5. Моисеев А. Object Pascal: Спб, 2000.

6. Немнюгин С.А. Программирование: Питер, 2000.

7. В.В.Старлова. Программирование. Питер, 1999.

8. Гофман В. Работа с базами данных в Delphi. БХВ-Петербург 2001.

9. Фаронов В. Программирование баз данных в Delphi 7. - С-Пб: Питер 2003.

10. А.М. Вендров. Проектирование программного обеспечения экономических информационных систем: Учебник. - М.: Финансы и статистика, 2005. - 352 с.

11. Смирнова Г.Н. и др. Проектирование экономических информационных систем: Учебник - М. Финансы и статистка, 2001.

12. А.М. Вендров. Проектирование программного обеспечения экономических информационных систем: Учебное пособие. - М.: Финансы и статистика, 2002.

13. Венчковский Л.Б. Разработка сложных программных изделий.

14. Буч Г. Объектно - ориентированный анализ.

15. Базиян, Менахем и др. Visual FoxPro 6. - М.: Издательский дом "Вильямс", 1999. - 928 с.

16. Сайт http://www.citforum.ru.

18. СниП РК 3.02 - 04 - 2009. Административные и бытовые здания раздел Противопожарная безопасность.

19. С.А. Алибаева. Дипломное проектирование: Методические указания (для студентов всех форм обучения направления. - 652400 - Радиоэлектроника и телекоммуникации). - Алматы: АИЭС, 2001.

20. Фирменный стандарт. Работы учебные. общие требования к построению, изложению, оформлению и содержанию. СТ НАО 56023-1910-01-2009. - Алматы: АИЭС, 2014.

Приложение А

Листинг программы

Процедура для связи формы с таблицами БД

```
procedure TForm10.FormCreate(Sender: TObject);
begin
form10.ComboBox1.Items.Add('Корпуса и блоки питания');
form10.ComboBox1.Items.Add('Материнские платы');
form10.ComboBox1.Items.Add('Процессоры');
form10.ComboBox1.Items.Add('Память ОЗУ');
form10.ComboBox1.Items.Add('Жесткие диски');
form10.ComboBox1.Items.Add('Флоппи-дисководы');
form10.ComboBox1.Items.Add('Приводы CD-ROM, CD-RW, DVD');
form10.ComboBox1.Items.Add('Видеокарты');
form10.ComboBox1.Items.Add('Вентиляторы, системы охлаждения');
form10.Image3.Picture.LoadFromFile('Icons\Images\Komplect\xfx20487020x2.bmp');
form10.Image4.Picture.LoadFromFile('Icons\ico\comic_48\0034.ico');
form10.Image5.Picture.LoadFromFile('Icons\ico\comic_48\0034.ico');
end;

procedure TForm10.ComboBox1Change(Sender: TObject);
begin
if form10.ComboBox1.Text='Корпуса и блоки питания' then
begin
form10.Image1.Picture.LoadFromFile('Icons\Images\Komplect\Keus.bmp');
form10.Image2.Picture.LoadFromFile('Icons\Images\Komplect\BP.bmp');
form10.ADOTable1.Active:=false;
form10.ADOTable1.TableName:='Корпуса и БП';
form10.ADOTable1.Active:=true;
form10.DBGrid1.DataSource:=DataSource1;
form10.ADOTable1.Fields.FieldByName('№').DisplayWidth:=5;
form10.ADOTable1.Fields.FieldByName('Наименование товара').DisplayWidth:=55;
form10.ADOTable1.Fields.FieldByName('Срок гарантии').DisplayWidth:=15;
form10.ADOTable1.Fields.FieldByName('Цена в тенге').DisplayWidth:=10;
form10.ADOTable1.Fields.FieldByName('Колличество').DisplayWidth:=10;
//////////////////////
DBEdit6.DataField:='№';
DBEdit6.DataSource:=DataSource1;
DBEdit7.DataField:='Наименование товара';
DBEdit7.DataSource:=DataSource1;
DBEdit8.DataField:='Срок гарантии';
DBEdit8.DataSource:=DataSource1;
DBEdit9.DataField:='Цена в тенге';
DBEdit9.DataSource:=DataSource1;
DBEdit10.DataField:='Колличество';
DBEdit10.DataSource:=DataSource1
end;
```

```
if form10.ComboBox1.Text='Материнские платы' then
begin
form10.Image1.Picture.LoadFromFile('Icons\Images\Komplect\Mam1.bmp');
form10.Image2.Picture.LoadFromFile('Icons\Images\Komplect\Mam2.bmp');
form10.ADOTable1.Active:=false;
form10.ADOTable1.TableName:='Матплаты';
form10.ADOTable1.Active:=true;
form10.DBGrid1.DataSource:=DataSource1;
form10.ADOTable1.Fields.FieldByName('№').DisplayWidth:=5;
form10.ADOTable1.Fields.FieldByName('Наименование товара').DisplayWidth:=55;
form10.ADOTable1.Fields.FieldByName('Срок гарантии').DisplayWidth:=15;
form10.ADOTable1.Fields.FieldByName('Цена в тенге').DisplayWidth:=10;
form10.ADOTable1.Fields.FieldByName('Колличество').DisplayWidth:=10;
//////////////////////
DBEdit6.DataField:='№';
DBEdit6.DataSource:=DataSource1;
DBEdit7.DataField:='Наименование товара';
DBEdit7.DataSource:=DataSource1;
DBEdit8.DataField:='Срок гарантии';
DBEdit8.DataSource:=DataSource1;
DBEdit9.DataField:='Цена в тенге';
DBEdit9.DataSource:=DataSource1;
DBEdit10.DataField:='Колличество';
DBEdit10.DataSource:=DataSource1
end;

if form10.ComboBox1.Text='Процессоры' then
begin
form10.Image1.Picture.LoadFromFile('Icons\Images\Komplect\Proc1.bmp');
form10.Image2.Picture.LoadFromFile('Icons\Images\Komplect\Proc2.bmp');
form10.ADOTable1.Active:=false;
form10.ADOTable1.TableName:='Процы';
form10.ADOTable1.Active:=true;
form10.DBGrid1.DataSource:=DataSource1;
form10.ADOTable1.Fields.FieldByName('№').DisplayWidth:=5;
form10.ADOTable1.Fields.FieldByName('Наименование товара').DisplayWidth:=55;
form10.ADOTable1.Fields.FieldByName('Срок гарантии').DisplayWidth:=15;
form10.ADOTable1.Fields.FieldByName('Цена в тенге').DisplayWidth:=10;
form10.ADOTable1.Fields.FieldByName('Колличество').DisplayWidth:=10;
//////////////////////
DBEdit6.DataField:='№';
DBEdit6.DataSource:=DataSource1;
DBEdit7.DataField:='Наименование товара';
DBEdit7.DataSource:=DataSource1;
DBEdit8.DataField:='Срок гарантии';
DBEdit8.DataSource:=DataSource1;
DBEdit9.DataField:='Цена в тенге';
DBEdit9.DataSource:=DataSource1;
```

```
DBEdit10.DataField:='Колличество';
DBEdit10.DataSource:=DataSource1
end;
if form10.ComboBox1.Text='Память ОЗУ' then
begin
form10.Image1.Picture.LoadFromFile('Icons\Images\Komplect\OZY1.bmp');
form10.Image2.Picture.LoadFromFile('Icons\Images\Komplect\OZY2.bmp');
form10.ADOTable1.Active:=false;
form10.ADOTable1.TableName:='Память';
form10.ADOTable1.Active:=true;
form10.DBGrid1.DataSource:=DataSource1;
form10.ADOTable1.Fields.FieldByName('№').DisplayWidth:=5;
form10.ADOTable1.Fields.FieldByName('Наименование товара').DisplayWidth:=55;
form10.ADOTable1.Fields.FieldByName('Срок гарантии').DisplayWidth:=15;
form10.ADOTable1.Fields.FieldByName('Цена в тенге').DisplayWidth:=10;
form10.ADOTable1.Fields.FieldByName('Колличество').DisplayWidth:=10;
//////////////////////
DBEdit6.DataField:='№';
DBEdit6.DataSource:=DataSource1;
DBEdit7.DataField:='Наименование товара';
DBEdit7.DataSource:=DataSource1;
DBEdit8.DataField:='Срок гарантии';
DBEdit8.DataSource:=DataSource1;
DBEdit9.DataField:='Цена в тенге';
DBEdit9.DataSource:=DataSource1;
DBEdit10.DataField:='Колличество';
DBEdit10.DataSource:=DataSource1
end;

if form10.ComboBox1.Text='Жесткие диски' then
begin
form10.Image1.Picture.LoadFromFile('Icons\Images\Komplect\Hard1.bmp');
form10.Image2.Picture.LoadFromFile('Icons\Images\Komplect\Hard2.bmp');
form10.ADOTable1.Active:=false;
form10.ADOTable1.TableName:='HDD';
form10.ADOTable1.Active:=true;
form10.DBGrid1.DataSource:=DataSource1;
form10.ADOTable1.Fields.FieldByName('№').DisplayWidth:=5;
form10.ADOTable1.Fields.FieldByName('Наименование товара').DisplayWidth:=55;
form10.ADOTable1.Fields.FieldByName('Срок гарантии').DisplayWidth:=15;
form10.ADOTable1.Fields.FieldByName('Цена в тенге').DisplayWidth:=10;
form10.ADOTable1.Fields.FieldByName('Колличество').DisplayWidth:=10;
//////////////////////
DBEdit6.DataField:='№';
DBEdit6.DataSource:=DataSource1;
DBEdit7.DataField:='Наименование товара';
DBEdit7.DataSource:=DataSource1;
DBEdit8.DataField:='Срок гарантии';
```

```
DBEdit8.DataSource:=DataSource1;
DBEdit9.DataField:='Цена в тенге';
DBEdit9.DataSource:=DataSource1;
DBEdit10.DataField:='Колличество';
DBEdit10.DataSource:=DataSource1
end;
if form10.ComboBox1.Text='Флоппи-дисководы' then
begin
form10.Image1.Picture.LoadFromFile('Icons\Images\Komplect\Flop1.bmp');
form10.Image2.Picture.LoadFromFile('Icons\Images\Komplect\Flop2.bmp');
form10.ADOTable1.Active:=false;
form10.ADOTable1.TableName:='Флопы';
form10.ADOTable1.Active:=true;
form10.DBGrid1.DataSource:=DataSource1;
form10.ADOTable1.Fields.FieldByName('№').DisplayWidth:=5;
form10.ADOTable1.Fields.FieldByName('Наименование товара').DisplayWidth:=55;
form10.ADOTable1.Fields.FieldByName('Срок гарантии').DisplayWidth:=15;
form10.ADOTable1.Fields.FieldByName('Цена в тенге').DisplayWidth:=10;
form10.ADOTable1.Fields.FieldByName('Колличество').DisplayWidth:=10;
//////////////////////
DBEdit6.DataField:='№';
DBEdit6.DataSource:=DataSource1;
DBEdit7.DataField:='Наименование товара';
DBEdit7.DataSource:=DataSource1;
DBEdit8.DataField:='Срок гарантии';
DBEdit8.DataSource:=DataSource1;
DBEdit9.DataField:='Цена в тенге';
DBEdit9.DataSource:=DataSource1;
DBEdit10.DataField:='Колличество';
DBEdit10.DataSource:=DataSource1
end;

if form10.ComboBox1.Text='Приводы CD-ROM, CD-RW, DVD' then
begin
form10.Image1.Picture.LoadFromFile('Icons\Images\Komplect\CD1.bmp');
form10.Image2.Picture.LoadFromFile('Icons\Images\Komplect\CD2.bmp');
form10.ADOTable1.Active:=false;
form10.ADOTable1.TableName:='Приводы';
form10.ADOTable1.Active:=true;
form10.DBGrid1.DataSource:=DataSource1;
form10.ADOTable1.Fields.FieldByName('№').DisplayWidth:=5;
form10.ADOTable1.Fields.FieldByName('Наименование товара').DisplayWidth:=55;
form10.ADOTable1.Fields.FieldByName('Срок гарантии').DisplayWidth:=15;
form10.ADOTable1.Fields.FieldByName('Цена в тенге').DisplayWidth:=10;
form10.ADOTable1.Fields.FieldByName('Колличество').DisplayWidth:=10;
//////////////////////
DBEdit6.DataField:='№';
DBEdit6.DataSource:=DataSource1;
```

```
DBEdit7.DataField:='Наименование товара';
DBEdit7.DataSource:=DataSource1;
DBEdit8.DataField:='Срок гарантии';
DBEdit8.DataSource:=DataSource1;
DBEdit9.DataField:='Цена в тенге';
DBEdit9.DataSource:=DataSource1;
DBEdit10.DataField:='Колличество';
DBEdit10.DataSource:=DataSource1
end;
if form10.ComboBox1.Text='Видеокарты' then
begin
form10.Image1.Picture.LoadFromFile('Icons\Images\Komplect\Video1.bmp');
form10.Image2.Picture.LoadFromFile('Icons\Images\Komplect\Video2.bmp');
form10.ADOTable1.Active:=false;
form10.ADOTable1.TableName:='Видеокарты';
form10.ADOTable1.Active:=true;
form10.DBGrid1.DataSource:=DataSource1;
form10.ADOTable1.Fields.FieldByName('№').DisplayWidth:=5;
form10.ADOTable1.Fields.FieldByName('Наименование товара').DisplayWidth:=55;
form10.ADOTable1.Fields.FieldByName('Срок гарантии').DisplayWidth:=15;
form10.ADOTable1.Fields.FieldByName('Цена в тенге').DisplayWidth:=10;
form10.ADOTable1.Fields.FieldByName('Колличество').DisplayWidth:=10;
//////////////////////
DBEdit6.DataField:='№';
DBEdit6.DataSource:=DataSource1;
DBEdit7.DataField:='Наименование товара';
DBEdit7.DataSource:=DataSource1;
DBEdit8.DataField:='Срок гарантии';
DBEdit8.DataSource:=DataSource1;
DBEdit9.DataField:='Цена в тенге';
DBEdit9.DataSource:=DataSource1;
DBEdit10.DataField:='Колличество';
DBEdit10.DataSource:=DataSource1
end;

if form10.ComboBox1.Text='Вентиляторы, системы охлаждения' then
begin
form10.Image1.Picture.LoadFromFile('Icons\Images\Komplect\Ohlazhd1.bmp');
form10.Image2.Picture.LoadFromFile('Icons\Images\Komplect\Ohlazhd2.bmp');
form10.ADOTable1.Active:=false;
form10.ADOTable1.TableName:='Вентеляторы';
form10.ADOTable1.Active:=true;
form10.DBGrid1.DataSource:=DataSource1;
form10.ADOTable1.Fields.FieldByName('№').DisplayWidth:=5;
form10.ADOTable1.Fields.FieldByName('Наименование товара').DisplayWidth:=55;
form10.ADOTable1.Fields.FieldByName('Срок гарантии').DisplayWidth:=15;
form10.ADOTable1.Fields.FieldByName('Цена в тенге').DisplayWidth:=10;
form10.ADOTable1.Fields.FieldByName('Колличество').DisplayWidth:=10;
```

```
//////////////////////
DBEdit6.DataField:='№';
DBEdit6.DataSource:=DataSource1;
DBEdit7.DataField:='Наименование товара';
DBEdit7.DataSource:=DataSource1;
DBEdit8.DataField:='Срок гарантии';
DBEdit8.DataSource:=DataSource1;
DBEdit9.DataField:='Цена в тенге';
DBEdit9.DataSource:=DataSource1;
DBEdit10.DataField:='Колличество';
DBEdit10.DataSource:=DataSource1
end;
end;

procedure TForm10.Button1Click(Sender: TObject);
var
kol:integer;
begin
ADOTable2.Insert;
DBEdit1.Text:='1';
DBEdit2.Text:=DBEdit7.Text;
DBEdit3.Text:=DBEdit8.Text;
DBEdit4.Text:=DBEdit9.Text;
ADOTable2.Post;
kol:=StrToInt(DBEdit10.Text);
dec(kol);
DBEdit10.Text:=IntToStr(kol);
end;

procedure TForm10.Button2Click(Sender: TObject);
begin
form10.ADOTable2.Delete;
end;
end.
```

Процедура поиска квитанции по ее номеру

```
procedure TForm13.Button8Click(Sender: TObject);
var
a:STRING;
begin
a:='WHERE Талон№ LIKE "%'+Form13.Edit1.Text+'%"';
ADOQuery1.SQL.Strings[2]:=a;
ADOQuery1.Active:=TRUE;
end;

procedure TForm13.Button9Click(Sender: TObject);
var
```

```
a:STRING;
begin
a:='WHERE Талон№ LIKE "%'+''+'%"';
ADOQuery1.SQL.Strings[2]:=a;
ADOQuery1.Active:=TRUE;
end;
```

Процедура создания листа MS Excel с квитанцией

```
procedure TForm13.Button7Click(Sender: TObject);
begin
//Создаем новую рабочую книгу с одним листом
excel.SheetsInNewWorkbook[0]:=1;
excel.Workbooks.Add(EmptyParam,0);

//Устанавливаем шрифт для всех колонок
excel.Columns.Font.Size:=10;

//Заголовок прайс листа
excel.Cells.Item[3,5].value:='"SUPPORT"';
excel.Cells.Item[3,5].font.size:=14;
excel.Cells.Item[3,5].font.bold:= True;

excel.Cells.Item[5,8].value:='дом:';
excel.Cells.Item[6,8].value:='раб:';
excel.Cells.Item[7,8].value:='сот:';
excel.Cells.Item[5,9].value:=form13.DBEdit2.Text;
excel.Cells.Item[6,9].value:=form13.DBEdit3.Text;
excel.Cells.Item[7,9].value:=form13.DBEdit4.Text;
excel.Cells.Item[7,9].NumberFormat:=0;
excel.Cells.Item[7,2].value:='ФИО:';
excel.Cells.Item[7,3].value:=form13.DBEdit1.Text;
excel.Cells.Item[7,3].font.underline:= True;
excel.Cells.Item[7,7].value:='тел:';
excel.Cells.Item[9,2].value:='№';
excel.Cells.Item[9,4].value:='Наименование изделия';
excel.Cells.Item[10,2].value:='1';
excel.Cells.Item[10,4].value:=form13.DBEdit5.Text;
excel.Cells.Item[9,2].HorizontalAlignment:=xlCenter;
excel.Cells.Item[10,2].HorizontalAlignment:=xlCenter;
excel.Cells.Item[12,3].value:='Дата';
excel.Cells.Item[12,5].value:='Вид неисправности:';
excel.Cells.Item[13,3].value:=form13.DBEdit8.Text;
excel.Cells.Item[13,5].value:=form13.DBEdit6.Text;
excel.Cells.Item[15,2].value:='Дополнительны детали сданные с изделием:';
excel.Cells.Item[15,2].font.size:=14;
excel.Cells.Item[16,2].value:=form13.DBComboBox1.Text;
excel.Cells.Item[16,2].font.size:=14;
```

```
excel.Cells.Item[18,5].value:='ВНИМАНИЕ!!!';
excel.Cells.Item[18,5].font.size:=16;
excel.Cells.Item[18,5].font.bold:=true;
excel.Cells.Item[19,2].value:='Компания не несет ответственность за информацию';
excel.Cells.Item[19,2].font.size:=14;
excel.Cells.Item[19,2].font.bold:=true;
excel.Cells.Item[20,2].value:='содержащуюся в оборудовании, сданном на проверку!!!';
excel.Cells.Item[20,2].font.size:=14;
excel.Cells.Item[20,2].font.bold:=true;
excel.Cells.Item[22,2].value:=' ';
excel.Cells.Item[22,2].font.underline:=true;
excel.Cells.Item[23,2].value:=' Подпись';
excel.Cells.Item[23,5].value:=' М.П.';
excel.Cells.Item[23,5].font.size:=16;
excel.Cells.Item[23,5].font.bold:=true;
excel.Cells.Item[23,8].value:='Талон№';
excel.Cells.Item[23,8].font.size:=12;
excel.Cells.Item[23,9].value:=form13.DBEdit7.Text;
excel.Cells.Item[23,9].font.size:=16;
excel.Cells.Item[23,9].font.bold:=true;
excel.Cells.Item[23,9].HorizontalAlignment:=xlCenter;
//Заголовок прайс листа
excel.Cells.Item[28,5].value:='"SUPPORT"';
excel.Cells.Item[28,5].font.size:=14;
excel.Cells.Item[28,5].font.bold:= True;
//Телефоны
excel.Cells.Item[30,8].value:='раб:';
excel.Cells.Item[31,8].value:='сот:';
excel.Cells.Item[32,8].value:='сот:';
excel.Cells.Item[30,9].value:='8(274) 2-53-68';
excel.Cells.Item[30,9].NumberFormat:=0;
excel.Cells.Item[31,9].value:='8-777-245-43-33';
excel.Cells.Item[31,9].NumberFormat:=0;
excel.Cells.Item[32,9].value:='8-701-121-21-33';
excel.Cells.Item[32,9].NumberFormat:=0;
excel.Cells.Item[32,2].value:='ФИО:';
excel.Cells.Item[32,3].value:='Магазин "Support"';
excel.Cells.Item[32,3].font.underline:= True;
excel.Cells.Item[32,7].value:='тел:';
excel.Cells.Item[34,2].value:='№';
excel.Cells.Item[34,4].value:='Наименование изделия';
excel.Cells.Item[35,2].value:='1';
excel.Cells.Item[35,4].value:=form13.DBEdit5.Text;
excel.Cells.Item[34,2].HorizontalAlignment:=xlCenter;
excel.Cells.Item[35,2].HorizontalAlignment:=xlCenter;
excel.Cells.Item[37,3].value:='Дата';
excel.Cells.Item[37,5].value:='Вид неисправности:';
excel.Cells.Item[38,3].value:=form13.DBEdit8.Text;
```

```
excel.Cells.Item[38,5].value:=form13.DBEdit6.Text;
excel.Cells.Item[40,2].value:='Дополнительны детали сданные с изделием:';
excel.Cells.Item[40,2].font.size:=14;
excel.Cells.Item[41,2].value:=form13.DBComboBox1.Text;
excel.Cells.Item[41,2].font.size:=14;
excel.Cells.Item[43,5].value:='ВНИМАНИЕ!!!';
excel.Cells.Item[43,5].font.size:=16;
excel.Cells.Item[43,5].font.bold:=true;
excel.Cells.Item[44,2].value:='Компания не несет ответственность за информацию';
excel.Cells.Item[44,2].font.size:=14;
excel.Cells.Item[44,2].font.bold:=true;
excel.Cells.Item[45,2].value:='содержащуюся в оборудовании, сданном на проверку!!!';
excel.Cells.Item[45,2].font.size:=14;
excel.Cells.Item[45,2].font.bold:=true;
excel.Cells.Item[47,2].value:=' ';
excel.Cells.Item[47,2].font.underline:=true;
excel.Cells.Item[48,2].value:=' Подпись';
excel.Cells.Item[48,5].value:=' М.П.';
excel.Cells.Item[48,5].font.size:=16;
excel.Cells.Item[48,5].font.bold:=true;
excel.Cells.Item[48,8].value:='Талон№';
excel.Cells.Item[48,8].font.size:=12;
excel.Cells.Item[48,9].value:=form13.DBEdit7.Text;
excel.Cells.Item[48,9].font.size:=16;
excel.Cells.Item[48,9].font.bold:=true;
excel.Cells.Item[48,9].HorizontalAlignment:=xlCenter;

//Показываем окно Excel
excel.Visible[0]:= True;
end;
end.
```