МИНОБРНАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Ижевский государственный технический университет имени М.Т. Калашникова»
Кафедра АСОИУ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к выпускной квалификационной работе
по направлению 09.03.01 «Информатика и вычислительная техника»
на тему «Разработка автоматизированного рабочего места менеджера по финансовым продуктам АО «Связной Логистика»

Выполнил
студент гр. Б07-782-2зт А.С. Буракова

Руководитель
к.т.н., доцент кафедры АСОИУ М.Н. Мокроусов

Нормоконтроль
старший преп. каф. АСОИУ Н. В. Соболева

И.о. зав. кафедрой
д.т.н., профессор О.В. Малина

Ижевск 2016

РЕФЕРАТ

Пояснительная записка к выпускной квалификационной работе на тему «Разработка автоматизированного рабочего места менеджера по финансовым продуктам АО «Связной Логистика» изложена на 58 страницах, содержит 16 рисунков, 1 таблицу, 10 источников литературы, 2 приложения.

Ключевые слова работы: автоматизация, учет, информационная поддержка, система.

Целью работы является повышение эффективности деятельности торговой точки.

Во введении описывается актуальность выбранной темы работы, а также общие задачи, которые необходимо решить в процессе создания программы.

В первой главе представлен обзор существующих систем предприятия и сведения об объекте автоматизации.

Во второй главе описывается математическая постановка задачи формирования отчетов.

В третьей главе пошагово описано проектирование разрабатываемой системы.

В четвертой главе представлены результаты тестирования системы.

Разработаны 2 приложения. В приложении А представлено руководство пользователя, Приложение Б содержит текст работы программы.

В результате выполнения работы был разработан программный продукт, автоматизирующий процессы выдачи и обслуживания кредитных продуктов, позволяющий сократить время работы менеджера по финансовым продуктам при обслуживании клиентов. Разработанная система успешно прошла апробацию и может быть внедрена на предприятии.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Данная работа направлена на разработку программного модуля для менеджера по финансовым продуктам компании АО «Связной Логистика», предназначенного для учета фактов выдачи и погашения кредитов. Кредитование является одним из важнейших направлений для компании, так как в настоящее время большую часть товара клиенты приобретают именно в кредит. Так же для компании важно отслеживать состояние выданных кредитов для оперативного реагирования в случае образования дебиторских задолженностей, которые могут привести к значительным потерям прибыли.

С учетом этих аспектов выделим основные функции, которые должна выполнять разрабатываемая система:

- учет клиентов;
- ведение справочника банков;
- учет выданных кредитов и карт;
- построение графика погашения кредита;
- формирование выходных документов;
- выбор кредитных договоров по заданным критериям;
- формирование отчетов по задолженностям и просрочкам платежей.

Автоматизация деятельности менеджера поможет повысить производительность труда, сократить время на поиск нужной информации, минимизировать убытки.

Целью работы является получение практических навыков в проектировании автоматизированного рабочего места специалиста.

В связи с поставленной целью были выявлены следующие задачи:

- собрать подробную информацию о деятельности компании и объекте автоматизации;
- провести анализ существующей системы предприятия, выявить ее преимущества и недостатки;
- разработать функциональную модель и структуру системы;
- выбрать технические и программные средства разработки системы;
- разработать базу данных системы;
- разработать алгоритм работы программы;
- разработать эргономичный интерфейс программы;
- провести тестирование системы;
- разработать руководство пользователя.

1 Сведения о предприятии

«Связной» (АО «Связной Логистика») – крупнейший в России мультиканальный ретейлер федерального масштаба – начал свою работу в России в 1995 году. На сегодняшний день открыто 3000 магазинов «Связной» на территории России, которые ежедневно посещают более 1.7 миллионов человек.

В магазинах «Связной», помимо мобильных телефонов и услуг операторов сотовой связи, можно купить ноутбуки, планшетные компьютеры, фотоаппараты и видеокамеры, «умные» и спортивные часы, электронные книги и многое другое.

Также в торговых точках, принадлежащих сети «Связной», можно получить финансовые услуги: оформить кредит, кредитную или дебетовую карты.

1.1 Сведения об объекте автоматизации

Итак, объект автоматизации – магазин «Связной». Магазин осуществляет выдачу кредитных и дебетовых карт, потребительских кредитов на следующих условиях:

1. кредитная карта:
   - сумма от 3000 рублей до 300000 рублей;
   - льготный период до 55 дней;
   - срок кредита до 36 месяцев;
   - ежемесячный платеж 8% от остатка суммы задолженности.
2. потребительский кредит:
   - сумма от 1500 рублей до 70000 рублей;
   - срок кредита от 4 месяцев до 24 месяцев;
   - сумма первоначального взноса – по желанию клиента;
   - сумма ежемесячного платежа формируется равными долями на весь срок кредита;
   - банки – «Ренессанс Кредит», «Хоум кредит энд Финанс банк», «ОТП Банк», «Альфа-Банк», «Лето Банк», «Тинькофф Кредитные Системы»;
   - процентная ставка от 11.5% до 85% годовых;
   - штраф за просрочку 1% от суммы задолженности за каждый день просрочки;
   - допускается досрочное погашение.
3. дебетовая карта:

- проценты на остаток собственных средств – 8%;
- Cashback за любые покупки – 1%.

Для оформления любого кредитного продукта обязательно требуется паспорт РФ и наличие постоянной регистрации на территории РФ. Для неработающих пенсионеров дополнительно требуется пенсионное удостоверение. Также по желанию клиента возможно предоставление дополнительных документов (СНИЛС, ИНН и т.д.).

Требования к заемщикам:

- возраст от 18 лет до 60 лет;
- наличие постоянного источника дохода (зарплата, пенсия);
- стаж на последнем месте работы от 3 месяцев;
- отсутствие в черном списке БКИ.

Можно выделить следующие этапы выдачи кредитного продукта:

- определение вида кредитного продукта, необходимого заемщику;
- сбор всех необходимых данных о заемщике (паспортные данные, контактные данные, сведения об источниках дохода и т.д.);
- анализ кредитной заявки и вынесение решения (положительного или отрицательного);
- если решение положительное, то подготовка документов;
- подписание документов с клиентом.

Также можно выделить этапы последующего обслуживания выданных кредитов:

- хранение информации о совершенных платежах;
- формирование отчетов по просрочкам и задолженностям платежей;
- формирование отчетов по остатку по кредиту;
- закрытие кредитного договора, подготовка документов.

В случае выдачи кредитных и дебетовых карт ведется только учет фактов выдачи и последующее обслуживание не осуществляется.

Теперь все выше сказанное удобнее будет представить в виде схем. Они представлены на рисунках 1.1–1.4.

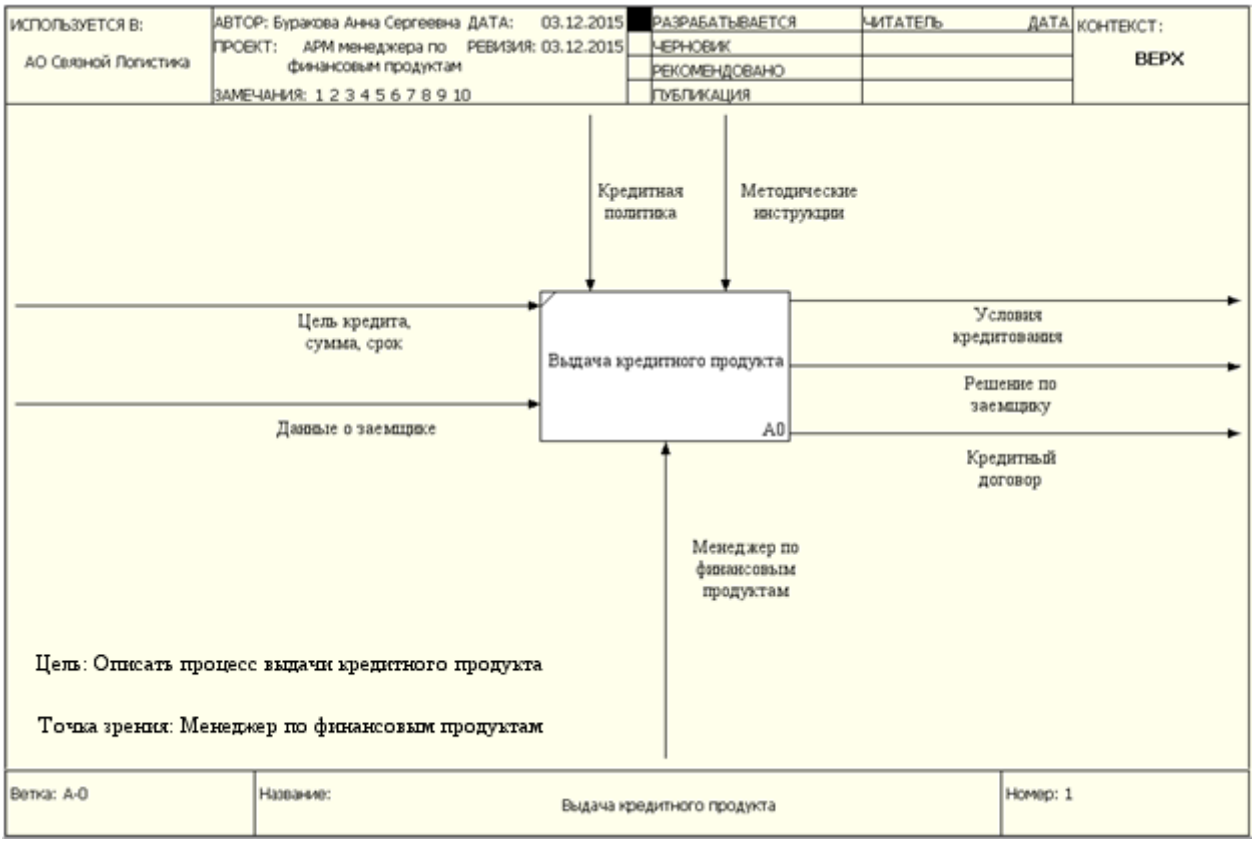

Рисунок 1.1 - Контекстная диаграмма процесса выдачи кредитного продукта

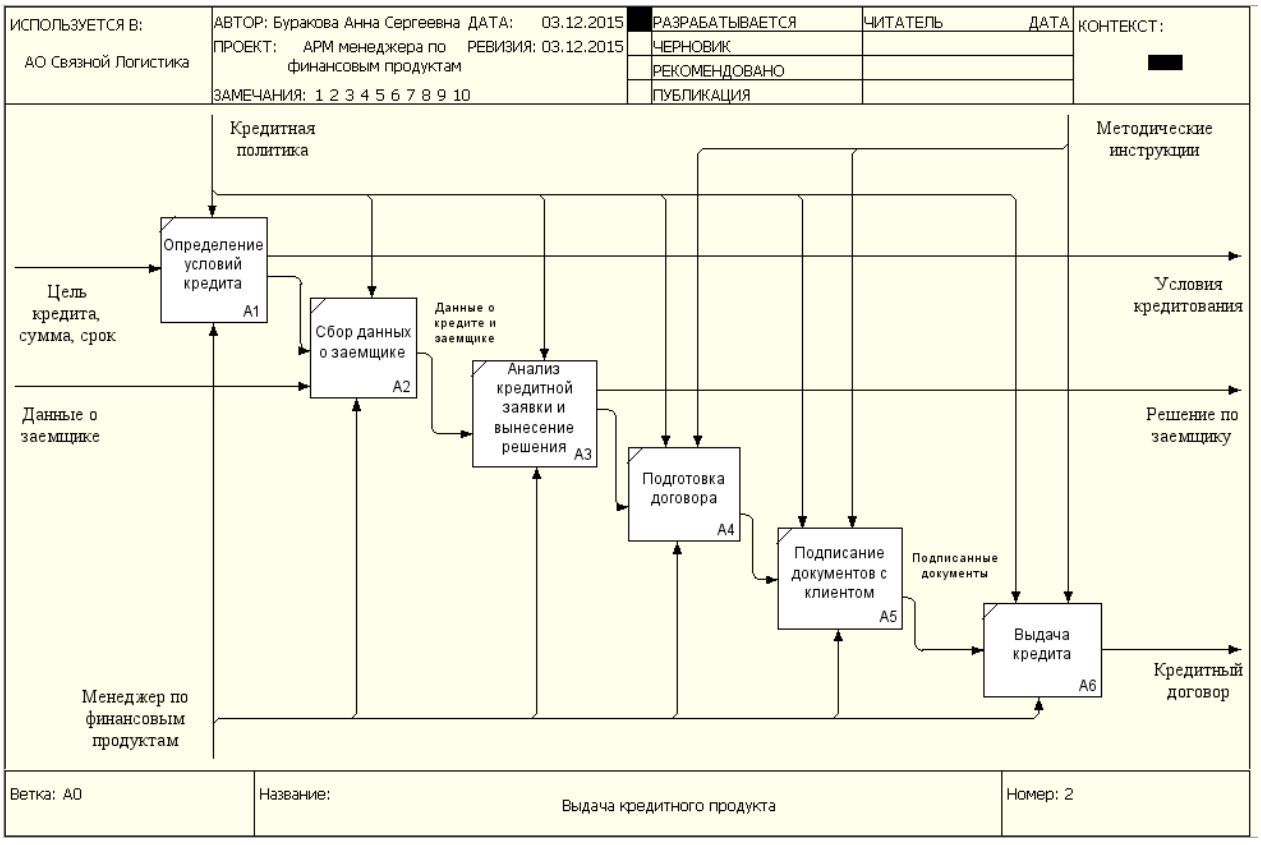

Рисунок 1.2 – Диаграмма декомпозиции процесса выдачи кредитного продукта

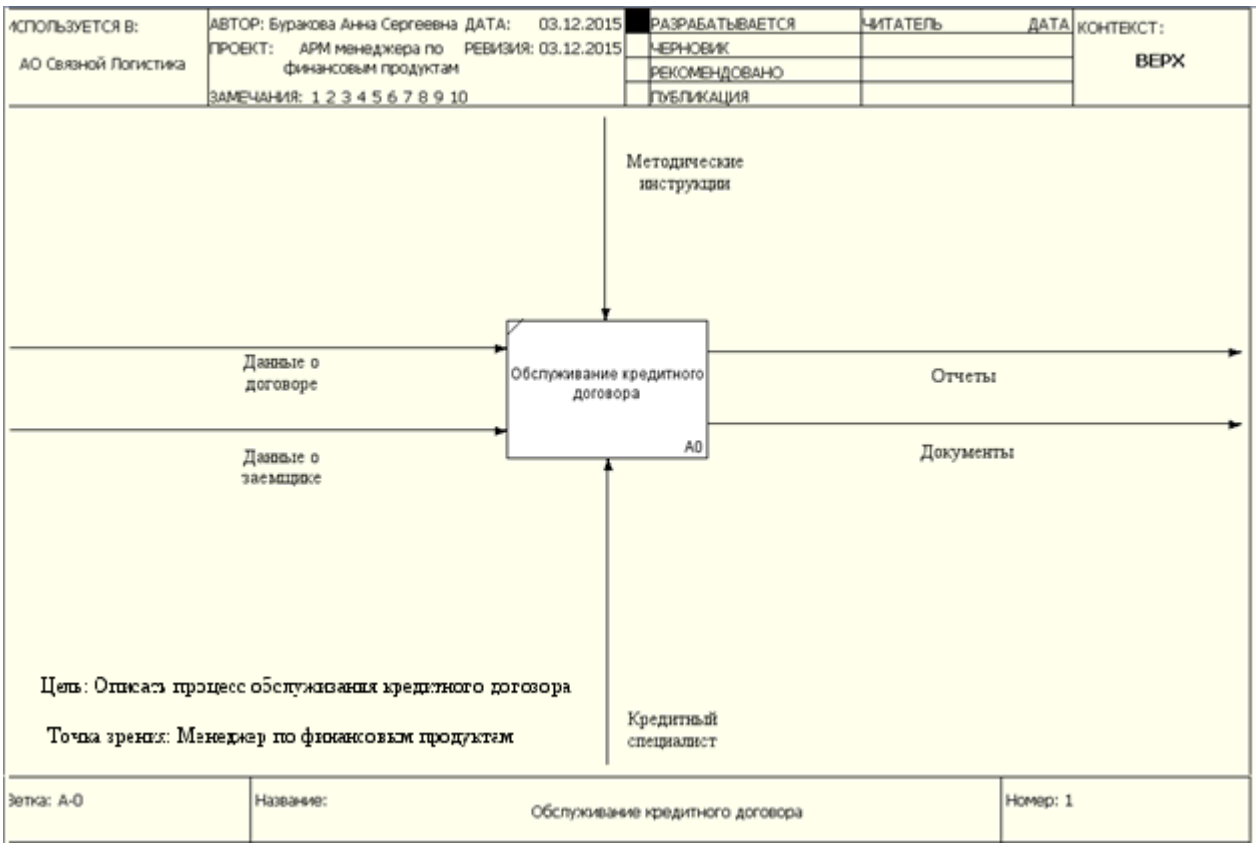

Рисунок 1.3 - Контекстная диаграмма процесса обслуживания кредитного договора

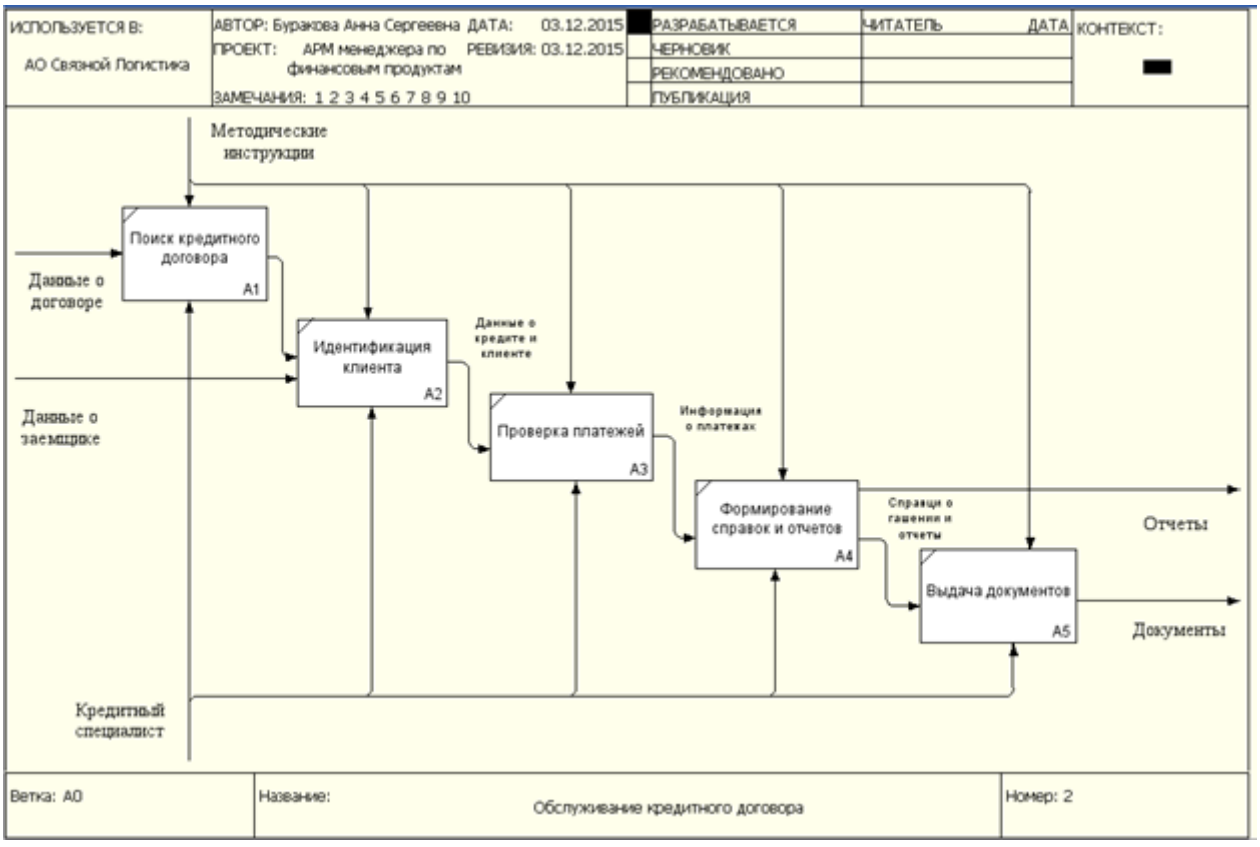

Рисунок 1.4 – Диаграмма декомпозиции процесса обслуживания кредитного договора

1.2 Обзор существующих систем предприятия

В настоящее время компания «Связной» для оформления кредитов использует программу под названием «Связной Брокер». Данная программа разработана самой компанией и предназначена строго для внутреннего использования. В рамках кредитной заявки «Связной Брокер» обеспечивает выполнение следующих основных операций:

- автоматизация деятельности всех торговых точек компании;
- информационная поддержка таких кредитных продуктов, как предварительный расчет сумм первоначального взноса и сумм ежемесячных платежей;
- ввод кредитной заявки, контроль правильности заполнения документов (первичная верификация);
- расчет суммы кредита с учетом суммы первоначального взноса, а также платежей, включаемых в сумму кредита (страховых премий);
- прикрепление графической информации к кредитной заявке (графические образы документов, удостоверяющие личность клиента);
- передача заявки на рассмотрение в банки;
- формирование и печать документов;
- формирование пакета документов на основе одобренной заявки в составе: кредитного договора, договора на открытие банковского счета, графика платежей, списка пунктов приема платежей по кредиту.

На этом функционал программы заканчивается. Последующее обслуживание не реализовано. Как говорилось ранее для компании важно отслеживать состояние выданных кредитов на отсутствие дебиторской задолженности, а так же владеть информацией о состоянии каждого кредитного договора и в случае обращения заемщика предоставить ему все необходимые данные. На данный момент такой возможности нет. С помощью программы «Связной Брокер» менеджер по финансовым услугам создает заявку на кредит, озвучивает предварительные условия кредитования клиенту, и в случае одобрения и согласия самого заемщика с условиями кредитования, печатает комплект документов и подписывает их вместе с клиентом. При обращении заемщика кредитный специалист не может предоставить ни данных о платежах, ни об остатке задолженности, что приводит к конфликтным ситуациям и вследствие чего к негативному отношению клиента к компании.

1.3 Выводы по главе

В результате проведенного анализа было принято решение разработать программный модуль, который будет выполнять функции используемой на данный момент программы «Связной Брокер», а так же функции, которые сейчас недоступны, но при этом имеют высокую важность в процессе развития компании. Разрабатываемая программа предназначена для автоматизации деятельности менеджера по финансовым продуктам в части учета клиентов и выданных кредитов, формирования отчетов и печати выходных документов.

2 Математическая модель формирования отчетов по выданным кредитам

Расчет количества дефолтных договоров.

Задача расчета количества дефолтных договоров заключается в выборке из базы данных с помощью специального запроса и вычислении итогов[1].

Входные данные:

$$\boldsymbol{Kd}=\{Kd_i / \ i=0,\ldots,L\},$$

$$\boldsymbol{Kd_i}=\langle nomer_i,\ data_i,\ fio_i\rangle,$$

где *Kd* – множество выданных кредитов;

L – количество кредитов

$nomer_i$ – номер договора;

$data_i$ – дата договора;

fio_i – фамилия, имя, отчество клиента.

DateStart – дата начала расчетного периода;

DateFin – дата окончания расчетного периода;

Status – статус рассчитываемых договоров;

NomerK – номер рассчитываемых договоров.

Выходные данные:

Множество дефолтных договоров за расчетный период:

$$KdDPeriod \subset Kd.$$

Количество дефолтных договоров за расчетный период:

$$KdDCount.$$

Алгоритм (метод):

1. Установить $KdDPeriod = 0.$

2. Если для множества *Kd* существует такое *i*, что номер договора Kd_i совпадает с заданным и дата исполнения лежит внутри заданного периода, то такой договор помещается в искомое множество *KdDPeriod*:

$$\exists i (Nomer_i = NomerK) \wedge (DataStart \le data_i < DataFin) \Rightarrow (Kd_i \in KdDPeriod) \wedge (KdDPeriod \cup Kd_i).$$
$$i=1$$

3. Количество дефолтных договоров вычисляется с помощью функции:

Count(KdDPeriod, KdDCount).

Разработанный алгоритм позволяет сформировать отчет, отображающий количество дефолтных договоров за заданный период времени. Задача решается по мере выдачи кредитов за любой срок.

3 Проектирование системы

3.1 Выбор и обоснование выбора программных и технических средств

Обоснование выбора СУБД.

На сегодняшний день известно более двух десятков форматов данных настольных СУБД, однако наиболее популярными следует признать Access, Paradox, FoxPro и dBase. Рассмотрим каждую из этих СУБД в отдельности.

dBase и Visual dBase.

Первая промышленная версия СУБД dBase – dBase II (принадлежащая тогда компании Ashton-Tate, приобретенной позже компанией Borland) появилась в начале 80-х годов. Благодаря простоте в использовании, нетребовательности к ресурсам компьютера и, что не менее важно, грамотной маркетинговой политике компании-производителя этот продукт приобрел немалую популярность [2].

Хранение данных в dBase основано на принципе «одна таблица — один файл» (эти файлы обычно имеют расширение *.dbf). MEMO-поля и BLOB-поля (доступные в поздних версиях dBase) хранятся в отдельных файлах (обычно с расширением *.dbt). Индексы для таблиц также хранятся в отдельных файлах. При этом в ранних версиях этой СУБД требовалась специальная операция реиндексирования для приведения индексов в соответствие с текущим состоянием таблицы [2].

Формат данных dBase является открытым, что позволило ряду других производителей заимствовать его для создания dBase-подобных СУБД, частично совместимых с dBase по форматам данных. Например, весьма популярная некогда СУБД FoxBase (разработанная Fox Software, Inc. и ныне принадлежащая Microsoft) использовала формат данных dBase для таблиц, однако форматы для хранения MEMO-полей и индексов были своими собственными, несовместимыми с dBase.

После покупки dBase компанией Borland этот продукт, получивший впоследствии название Visual dBase, приобрел набор дополнительных возможностей, характерных для средств разработки этой компании и для имевшейся у нее другой настольной СУБД – Paradox. Среди этих возможностей были специальные типы полей для графических данных, поддерживаемые индексы, хранение правил ссылочной целостности внутри самой базы данных, а также возможность манипулировать данными других форматов, в частности серверных СУБД, за счет использования BDE API и SQL Links [2].

В настоящее время Visual dBase принадлежит компании dBase, Inc. Его последняя версия — Visual dBase 7.5 имеет следующие возможности:

- средства манипуляции данными dBase и FoxPro всех версий;
- ядро доступа к данным Advantage Database Server фирмы Extended Systems и ODBC-драйвер для доступа к данным этой СУБД;
- средства визуального построения запросов.

Paradox.

В конце 80-х — начале 90-х годов Paradox, принадлежавший тогда компании Borland International, был весьма популярной СУБД, в том числе и в нашей стране, где он одно время занимал устойчивые позиции на рынке средств разработки настольных приложений с базами данных.

Принцип хранения данных в Paradox сходен с принципами хранения данных в dBase — каждая таблица хранится в своем файле (расширение *.db), MEMO- и BLOB-поля хранятся в отдельном файле (расширение *.md), как и индексы (расширение *.px).

Однако, в отличие от dBase, формат данных Paradox не является открытым, поэтому для доступа к данным этого формата требуются специальные библиотеки. Например, в приложениях, написанных на C или Pascal, использовалась некогда популярная библиотека Paradox Engine, ставшая основой Borland Database Engine (BDE). Эта библиотека используется ныне в приложениях, созданных с помощью средств разработки Borland (Delphi, C++Builder), в некоторых генераторах отчетов (например, Crystal Reports) и в самом Paradox. Существуют и ODBC-драйверы к базам данных, созданным различными версиями этой СУБД [1].

Microsoft FoxPro и Visual FoxPro.

FoxPro ведет свое происхождение от настольной СУБД FoxBase фирмы Fox Software. Разрабатывая FoxBase в конце 80-х годов, эта компания преследовала цель создать СУБД, функционально совместимую с dBase с точки зрения организации файлов и языка программирования, но существенно превышающую ее по производительности. Одним из способов повышения производительности являлась более эффективная организация индексных файлов, нежели в dBase, – по формату индексных файлов эти две СУБД несовместимы между собой [2].

По сравнению с аналогичными версиями dBase, FoxBase и более поздняя версия этого продукта, получившая название FoxPro, предоставляли своим пользователям несколько более широкие возможности, такие как использование деловой графики, генерация кода приложений, автоматическая генерация документации к приложениям и т.д.

Впоследствии этот продукт был приобретен компанией Microsoft. Его последние вер-

сии (начиная с версии 3.0, выпущенной в 1995 году) получили название Visual FoxPro. С каждой новой версией этот продукт оказывался все более и более интегрирован с другими продуктами Microsoft, в частности с Microsoft SQL Server, – в состав Visual FoxPro в течение нескольких последних лет входят средства переноса данных FoxPro в SQL Server и средства доступа к данным этого сервера из Visual FoxPro и созданных с его помощью приложений [3].

Последняя версия этого продукта — Visual FoxPro 6.0, доступна и отдельно, и как составная часть Microsoft Visual Studio 6.0. Отличительной особенностью этой настольной СУБД от двух рассмотренных выше является интеграция этого продукта с технологиями Microsoft, в частности поддержка COM (Component Object Model — компонентная объектная модель, являющаяся основой функционирования 32-разрядных версий Windows и организации распределенных вычислений в этой операционной системе), интеграция с Microsoft SQL Server, возможности создания распределенных приложений, основанных на концепции Windows DNA (Distributed interNet Applications) [3].

Visual Fox Pro 6.0 предоставляет следующие возможности [3]:

- средства создания COM-объектов и объектов для Microsoft Transaction Server, позволяющих создавать масштабируемые многозвенные приложения для обработки данных;
- средства доступа к данным серверных СУБД, базирующиеся на использовании OLE DB (набор COM-интерфейсов, позволяющий осуществить унифицированный доступ к данным из разнообразных источников, в том числе из не реляционных баз данных и иных источников, например Microsoft Exchange);
- средства доступа к данным Microsoft SQL Server и Oracle, включая возможность создания и редактирования таблиц, триггеров, хранимых процедур;
- средства отладки хранимых процедур Microsoft SQL Server;
- средство визуального моделирования компонентов и объектов, являющиеся составными частями приложения – Visual Modeller;
- средство для управления компонентами приложений, позволяющее осуществлять их повторное использование.

Итак, тенденции развития этого продукта очевидны: из настольной СУБД Visual FoxPro постепенно превращается в средство разработки приложений в архитектуре «клиент/сервер» и распределенных приложений в архитектуре Windows DNA. Впрочем, эти тенденции в определенной степени характерны для всех наиболее популярных настольных СУБД: и dBase, и Paradox также позволяют осуществлять доступ к наиболее популярным серверным СУБД.

Microsoft Access.

В отличие от Visual FoxPro, фактически превратившегося в средство разработки приложений, Access ориентирован в первую очередь на пользователей Microsoft Office, в том числе и не знакомых с программированием. Это, в частности, проявилось в том, что вся информация, относящаяся к конкретной базе данных, а именно таблицы, индексы (естественно, поддерживаемые), правила ссылочной целостности, бизнес-правила, список пользователей, а также формы и отчеты хранятся в одном файле, что в целом удобно для начинающих пользователей.

Microsoft Access – это функционально полная реляционная СУБД. В ней предусмотрены все необходимые вам средства для определения и обработки данных, а также для управления ими при работе с большими объемами информации [4].

В состав Access входят [4]:

- средства манипуляции данными Access и данными, доступными через ODBC (последние могут быть «присоединены» к базе данных Access);
- средства создания форм, отчетов и приложений; при этом отчеты могут быть экспортированы в формат Microsoft Word или Microsoft Excel, а для создания приложений используется Visual Basic for Applications, общий для всех составных частей Microsoft Office;
- средства доступа к данным серверных СУБД через OLE DB;
- средства создания клиентских приложений для Microsoft SQL Server;
- средства администрирования Microsoft SQL Server.

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 [4].

Access специально спроектирован для создания многопользовательских приложений, где файлы базы данных являются разделяемыми ресурсами в сети. 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.

Основываясь на изложенных выше данных, в качестве СУБД разрабатываемой системы был выбран Microsoft Access.

Обоснование выбора языка программирования.

Бурное развитие вычислительной техники, потребность в эффективных средствах разработки программного обеспечения привели к появлению систем программирования, ориентированных на так называемую «быструю разработку», среди которых можно выделить Borland Delphi и Microsoft Visual Basic. В основе систем быстрой разработки (RAD-систем, Rapid Application Development – среда быстрой разработки приложений) лежит технология визуального проектирования и событийного программирования, суть которой заключается в том, что среда разработки берет на себя большую часть рутинной работы, оставляя программисту работу по конструированию диалоговых окон и функций обработки событий [5].

Delphi – это среда быстрой разработки, в которой в качестве языка программирования используется язык Delphi. Язык Delphi – строго типизированный объектно-ориентированный язык, в основе которого лежит хорошо знакомый программистам Object Pascal.

Delphi – мощная система визуального объектно-ориентированного проектирования, позволяющая решать множество задач, в частности [5]:

- создавать законченные приложения для Windows самой различной направленности, от чисто вычислительных и логических, до графических и мультимедиа;
- быстро создавать (даже начинающим программистам) профессионально выглядящий оконный интерфейс для любых приложений;
- создавать мощные системы работы с локальными и удаленными базами данных;
- создавать справочные системы (файлы .hlp) для своих приложений.
- и многое другое.

Мечта программистов о среде программирования, в которой бы простота и удобство сочетались с мощью и гибкостью, стала реальностью с появлением среды Delphi. Она обеспечивала визуальное проектирование пользовательского интерфейса, имела развитый объектно-ориентированный язык Object Pascal (позже переименованный в Delphi) и уникальные по своей простоте и мощи средства доступа к базам данных. Язык Delphi по возможностям значительно превзошел язык Basic и даже в чем-то язык C++, но при этом он оказался весьма надежным и легким в изучении (особенно в сравнении с языком C++). В результате, среда Delphi позволила программистам легко создавать собственные компоненты и строить из них профессиональные программы. Среда оказалась настолько удачной, что по запросам любителей C++ была позже создана среда C++Builder - клон среды Delphi на основе языка C++ (с расширенным синтаксисом). Среда Delphi стала, по сути, одним из лучших средств программирования для операционной системы Windows [5].

Общая продуктивность любых инструментов создания программного обеспечения определяется следующими пятью важнейшими аспектами [5]:

- качеством визуальной среды разработки;
- скоростью работы компилятора и быстродействием откомпилированных программ;
- мощностью языка программирования и его сложностью;
- гибкостью и масштабируемостью используемой архитектуры баз данных;
- наличием поддерживаемых средой разработки шаблонов проектирования и использования.

Безусловно, существует еще немало важных факторов, например, вопросы установки, документация, поддержка сторонних производителей и т.д. Тем не менее, можно считать, что этой упрощенной модели вполне достаточно для объяснения, почему имеет смысл остановить свой выбор на Delphi.

3.2 Разработка базы данных

В проектируемой модели использовалась логико-физическая модель, описанная далее.

Логическое проектирование.

В разрабатываемой системе можно выделить следующие сущности: Журнал, Клиент, Кредит, Образование, СемПоложение, СтатусКредита, СтрахКомпания, Банк, Тип операции.

ER-диаграмма системы на логическом уровне представлена на рисунке 3.1.

Данные в БД должны обладать свойством целостности.[9] Под целостностью данных понимается корректность данных, и их непротиворечивость в любой момент времени. Поддержание целостности базы данных может рассматриваться как защита данных от неверных изменений или разрушения (этот вопрос не относится к незаконным изменениям и разрушениям, которые являются проблемой безопасности).

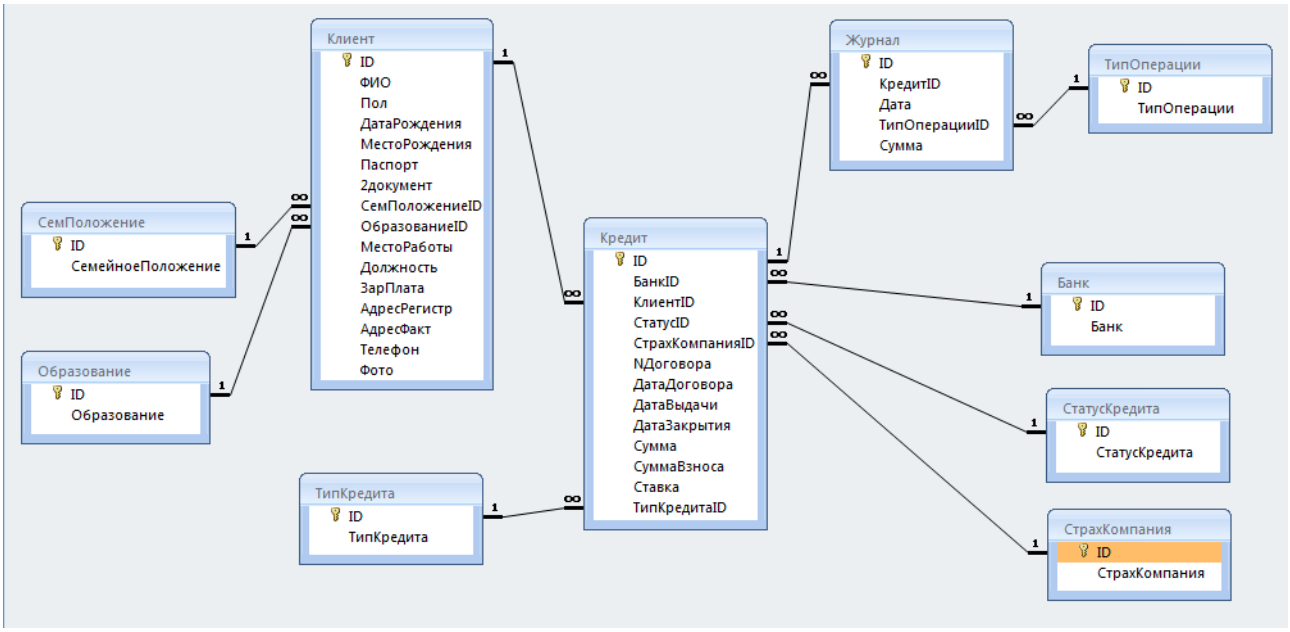

Рисунок 3.1 –ER-диаграмма системы на логическом уровне

В разрабатываемой структуре БД учтены основные правила целостности. Каждая сущность идентифицируется уникальным ключом, и разработана система внешних ключей. База данных не содержит несогласованных значений внешних ключей, то есть при работе с записями происходит каскадное обновление связанных полей и каскадное удаление связанных записей[6].

Физическое проектирование.

Физическая модель данных зависит от конкретной СУБД. В физической модели содержится информация обо всех объектах БД. Поскольку стандартов на объекты БД не существует, физическая модель зависит от конкретной реализации СУБД. Следовательно, одной и той же логической модели могут соответствовать несколько разных физических моделей. Если в логической модели не имеет значения, какой конкретно тип данных имеет атрибут, то в физической модели важно описать всю информацию о конкретных физических объектах.

В качестве СУБД выбрана Microsoft Access. Физическое описание модели удобнее всего представить в виде таблиц.[10] База данных проекта содержит таблицы, названия которых соответствуют именам сущностей инфологической модели. Структура базы данных описана в таблице 3.1.

Таблица 3.1 – Описание таблиц базы данных

Наименование таблицы	Наименование поля	Тип поля	Назначение таблицы
Журнал	ID	AutoNumber	Таблица хранит в себе данные об операциях, производимых с выданным кредитом: гашение, вынос на просрочку, закрытие, дата операции, и сумму в случае операции гашения.
	КредитID	Long Integer	
	Дата	Date/Time	
	ТипОперацииID	Long Integer	
	Сумма	Currency	
Клиент	ID	AutoNumber	Таблица хранит в себе все необходимые персональные данные о клиенте, предоставленные для оформления кредита.
	ФИО	Text(70)	
	Пол	Да/Нет	
	ДатаРождения	Date/Time	
	МестоРожд	Text(100)	
	Паспорт	Text(255)	
	2документ	Text(100)	
	СемПоложениеID	Long Integer	
	ОбразованиеID	Long Integer	
	МестоРаботы	Text(100)	
	Должность	Text(100)	
	ЗарПлата	Currency	
	АдресРегистр	Text(250)	
	АдресФакт	Text(250)	
	Телефон	Text(20)	
	Фото	OLE Object	
СтрахКомпания	ID	AutoNumber	Таблица хранит справочные данные о страховых компаниях.
	СтрахКомпания	Text(255)	
СтатусКредита	ID	AutoNumber	Таблица содержит справочную информацию о возможных статусах кредитных договоров.
	СтатусКредита	Text(12)	
Тип операции	ID	AutoNumber	Таблица содержит справочную информацию о возможных операциях по кредитным договорам.
	Тип операции	Text(50)	
Банк	ID	AutoNumber	Таблица хранит справочные данные о банках.
	Банк	Text(255)	
	Тип операции	Text(50)	

Продолжение таблицы 3.1

Наименование таблицы	Наименование поля	Тип поля	Назначение таблицы
Кредит	ID	AutoNumber	Таблица содержит все данные о выданном кредите и кредитном договоре.
	БанкID	Long Integer	
	КлиентID	Long Integer	
	СтатусID	Long Integer	
	СтрахКомпанияID	Long Integer	
	NДоговора	Text(15)	
	ДатаДог	Date/Time	
	ДатаЗакрыт	Date/Time	
	Сумма	Currency	
	СуммаВзноса	Currency	
	Ставка	Single	
Образование	ID	AutoNumber	Таблица содержит справочную информацию о видах образования.
	Образование	Text(50)	
СемПоложение	ID	AutoNumber	Таблица содержит справочную информацию о семейном положении.
	СемейноеПоложение	Text(50)	

Сбор информации осуществляется следующим образом: справочная информация вносится сразу после внедрения проекта и в будущем заносится по мере поступления сведений. Оперативная информация вносится в базу данных по мере поступления соответствующих документов и сведений.

3.3 Разработка пользовательского интерфейса

Пользователь имеет возможность выбора функций системы, применяя кнопочное и пиктографическое меню. Пользователь видит перед собой содержимое базы данных в виде экранного документа, в котором значения реквизитов (полей) отвечают наименованиями из его предметной области согласно заданию проекта, а не условным обозначениями полей базы данных.

Взаимодействие с пользователем осуществляется посредством экранных форм. На рисунке 3.3 представлена главная экранная форма. Граф переходов экранных форм представлен на рисунке 3.4.

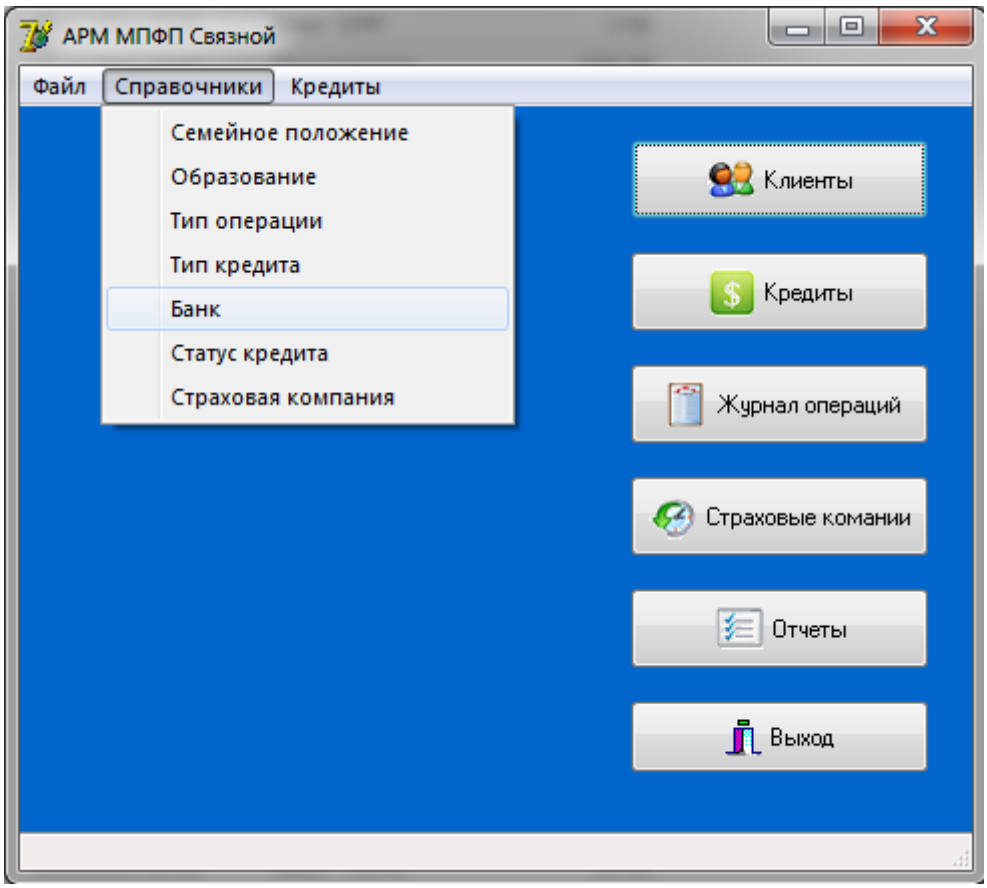

Рисунок 3.3 – Главная экранная форма

Рисунок 3.4– Граф перехода экранных форм

Текст программы в разрабатываемом приложении обрабатывает информацию, поступающую от пользователя, а также производит операции с базой данных (Приложение Б).

3.4 Описание работы системы

После запуска программы пользователь вызывает необходимую ему в данный момент экранную форму, на которой он вводит оперативные или справочные данные, либо формирует необходимые отчеты. После введения данных или формирования отчета, пользователь может завершить работу с программой, либо открыть другую экранную форму для дальнейшей работы. Подробная инструкция для пользователя представлена в руководстве (Приложение А).

Алгоритмы работы программы являются стандартными алгоритмами работы с базой данных. В основном все алгоритмы работы связаны с вводом данных от пользователя, проверке введенной информации на предмет нарушения целостности данных и занесение введенной информации в саму базу, если введенные сведения не нарушают целостности. [7,8]

Алгоритмы по редактированию данных, занесению их в базу, удаление из базы данных, а также алгоритмы, осуществляющие вывод информации из базы данных также являются стандартными. Приблизительный алгоритм работы программы представлен на рисунке 3.5.

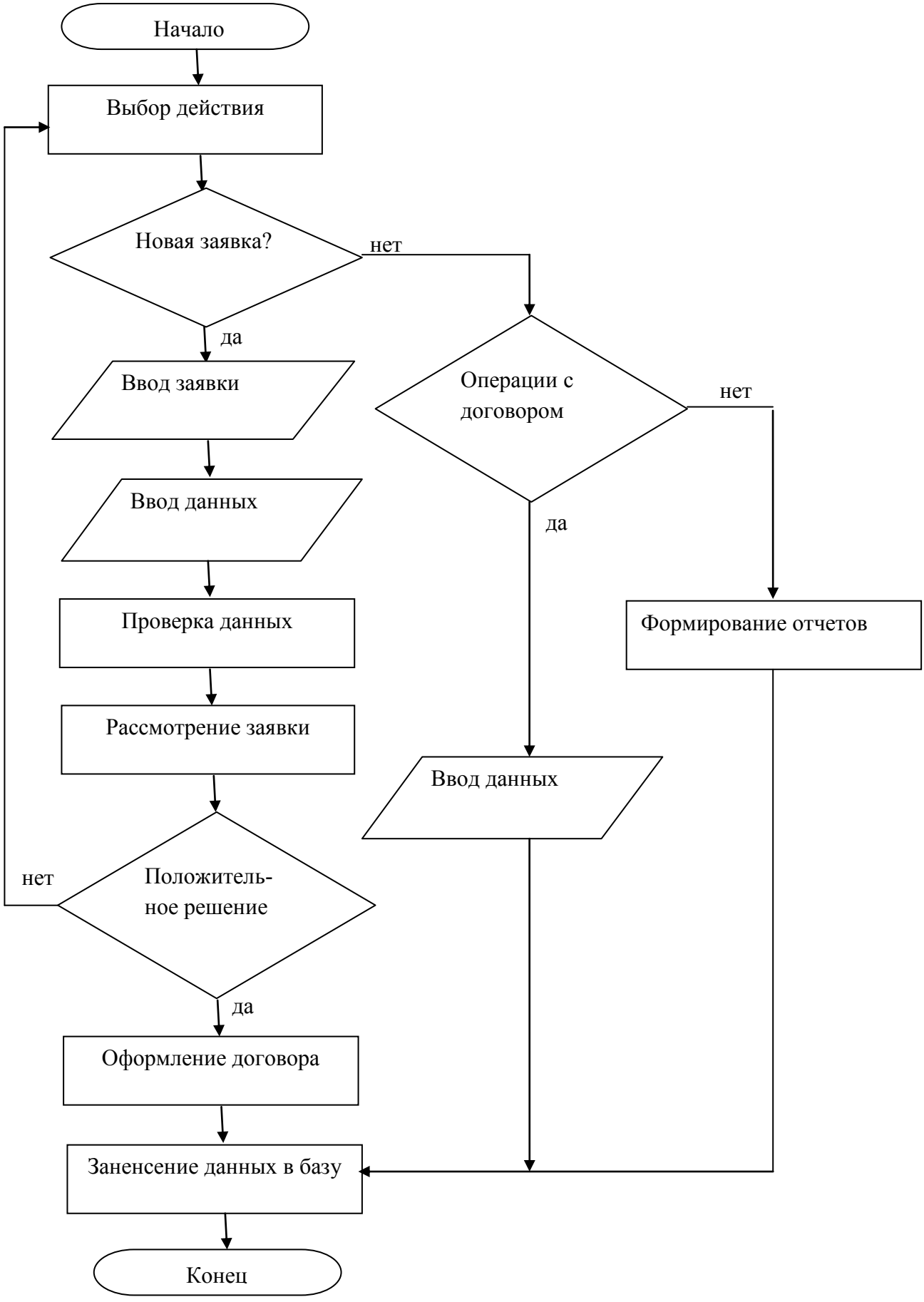

Рисунок 3.5– Приблизительный алгоритм работы программы

3.5 Техническое обеспечение системы

Техническое обеспечение автоматизированного рабочего места менеджера по финансовым продуктам – это средства вычислительной техники, входящие в системный блок и являющиеся составными частями персонального компьютера, на базе которого реализуется автоматизированное рабочее место, а также внешние устройства. Эти средства и устройства обеспечивают автоматизацию выполнения задач специалиста по вводу и обработке данных в различных форматах представления.

В состав средств вычислительной техники и внешних устройств входят:

- персональный компьютер стандартной конфигурации;
- USB-токен Rutoken. Применяется для защищенного документооборота в системе "клиент – банк". Rutoken - персональное устройство доступа к информационным ресурсам, полнофункциональный аналог смарт-карты, выполненный в виде usb-брелока;
- периферийные устройства, обеспечивающие ввод-вывод информации: эргономичный монитор, классическая клавиатура, оптическая светодиодная мышь, многофункциональное устройство.
- технические средства телекоммуникационного доступа и связи, поддерживающие работоспособность автоматизированного рабочего места специалиста – подключение по локальной сети внутри предприятия.

Техническое обеспечение автоматизированного рабочего менеджера по финансовым продуктам соответствует должностным обязанностям, которые выполняет специалист.

4 Тестирование системы

В результате тестирования программы были получены следующие отчеты:

- отчет по всем кредитным договорам, оформленным через ОТП банк за период с 01.11.2015г. по 30.11.2015г.(Рисунок 4.1);
- отчет по всем выданным кредитам и картам за период с 01.10.2015г. по 30.11.2015г.(Рисунок 4.2);
- отчет по закрытым договорам за период с 30.11.2015г. по 01.02.2016г.(Рисунок 4.3);
- отчет по клиентам, оформившим кредит в период с 01.11.2015г. по 30.11.2015г.(Рисунок 4.4);
- отчет по договорам с просрочкой платежей (Рисунок 4.5).

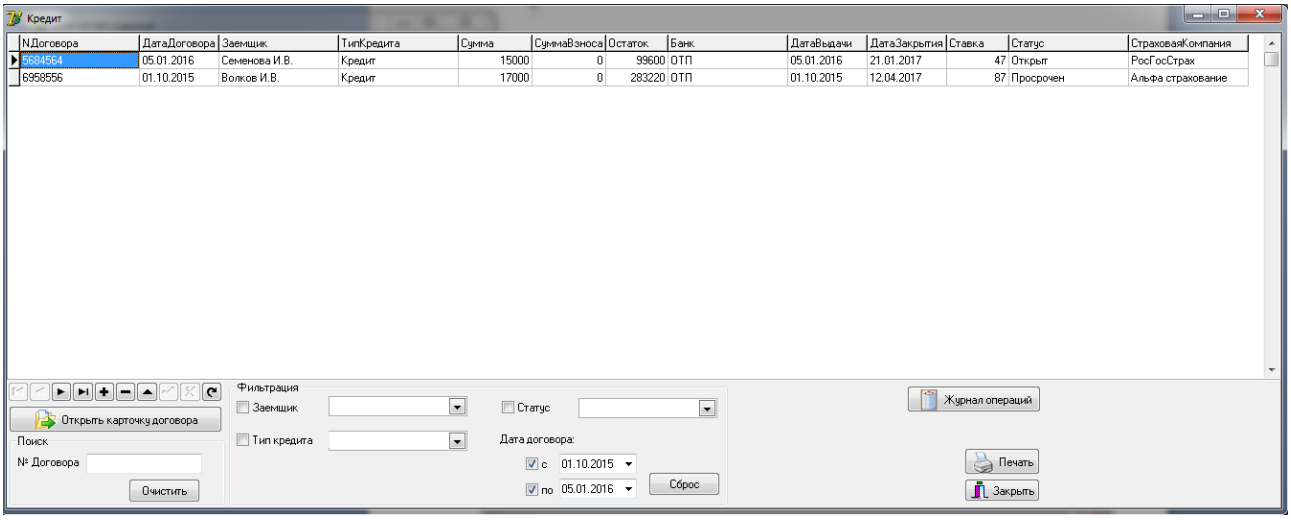

Рисунок 4.1 – Отчет по договорам ОТП банка за период с 01.11.2015г. по 30.11.2015г.

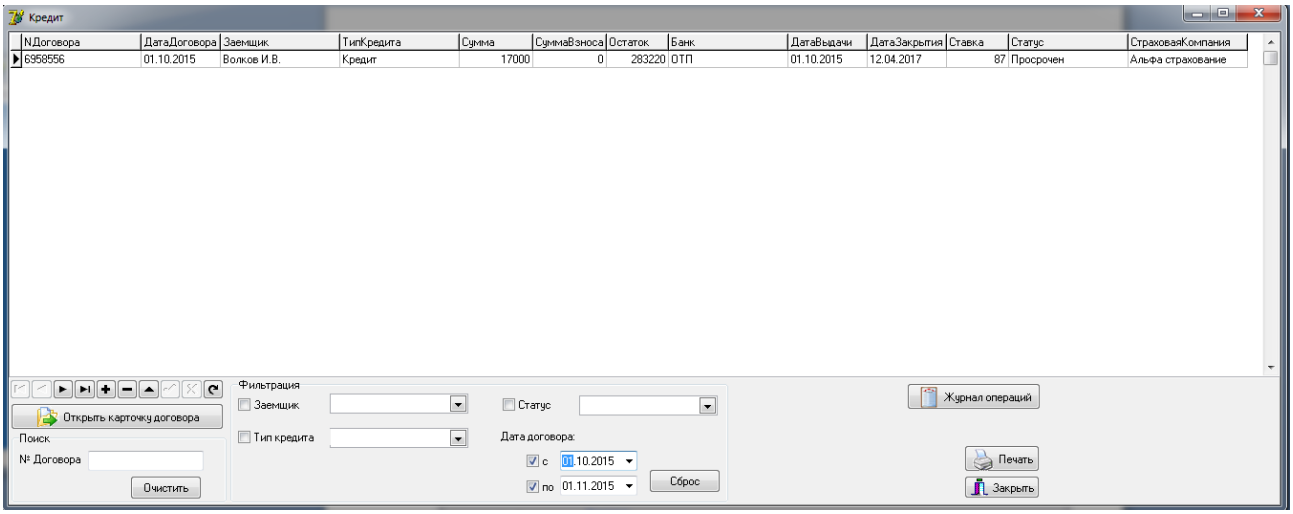

Рисунок 4.2 – Отчет по оформленным договорам за период с 01.10.2015г. по 30.11.2015г.

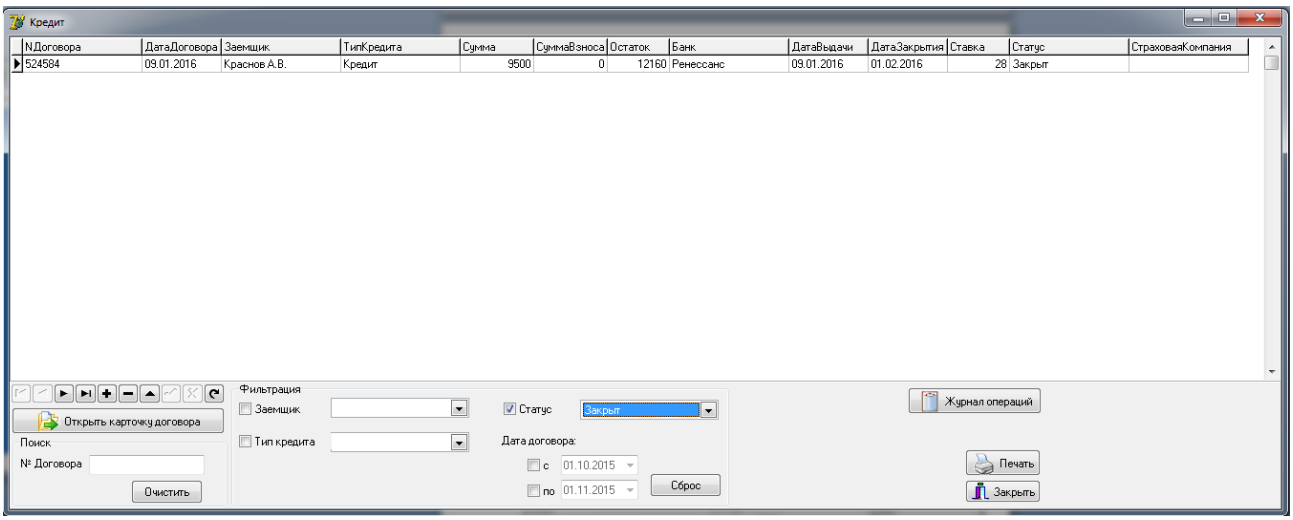

Рисунок 4.3 – Отчет по закрытым договорам за период с 01.11.2015г. по 01.02.2016г.

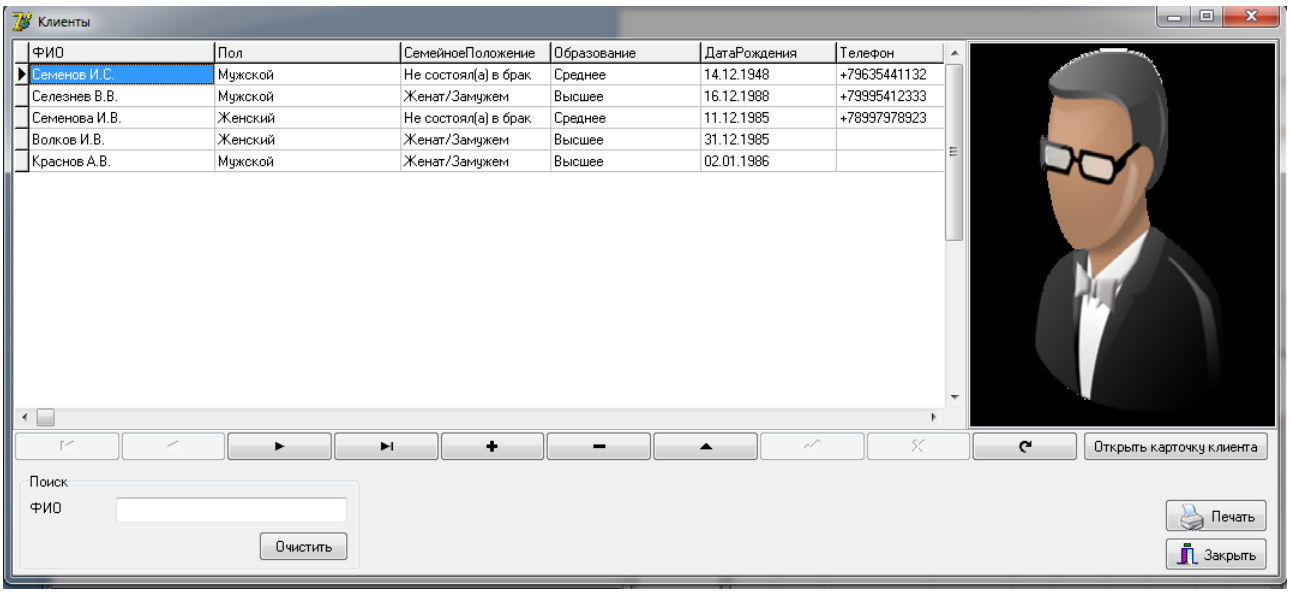

Рисунок 4.4 – Отчет по клиентам, оформившим кредит в период с 01.11.2015г. по 30.11.2015г.

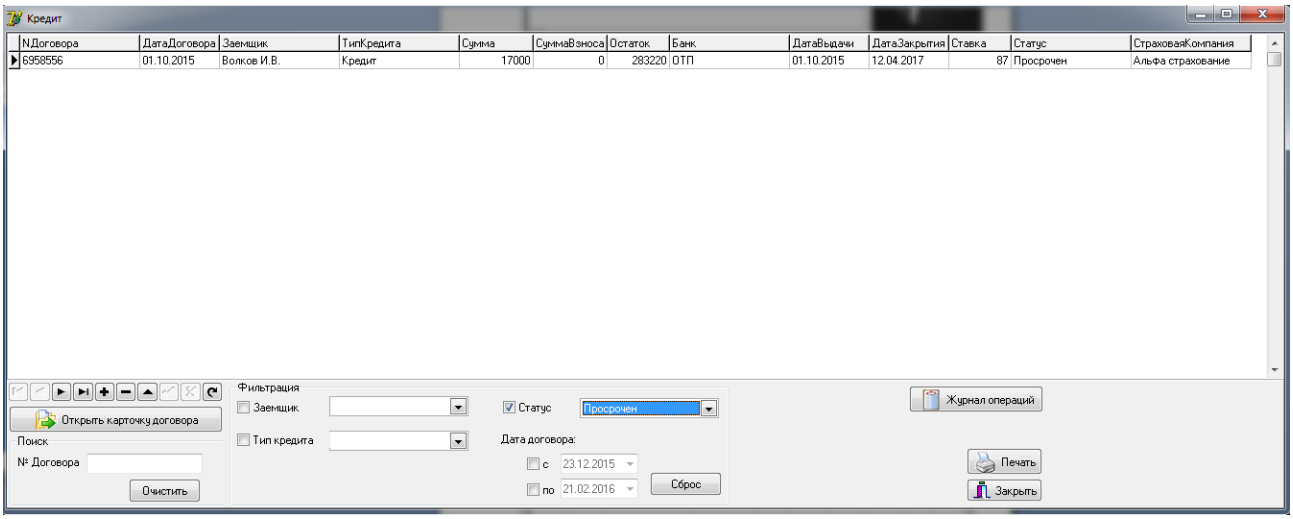

Рисунок 4.5 – Отчет по договорам с просрочкой платежей

Печатные формы отчетов представлены на рисунках 4.6-4.8.

NДоговора	ДатаДоговора	Заемщик	ТипКредита	Сумма	СуммаВзноса	Остаток	Банк	ДатаВыдачи	ДатаЗакрытия	Ставка	Статус	СтраховаяКомпания
5684564	05.01.2016	Семенова И.В.	Кредит	15 000,00 ₽	0,00 ₽	99600	ОТП	05.01.2016	21.01.2017	47	Открыт	РосГосСтрах
6958556	01.10.2015	Волков И.В.	Кредит	17 000,00 ₽	0,00 ₽	283220	ОТП	01.10.2015	12.04.2017	87	Просрочен	Альфа страхование
78999	04.05.2015	Краснов А.В.	Карта	7 000,00 ₽	0,00 ₽	16100	Тинькофф	04.05.2015	04.03.2016	13	Открыт	
524584	09.01.2016	Краснов А.В.	Кредит	9 500,00 ₽	0,00 ₽	12160	Ренессанс	09.01.2016	01.02.2016	28	Закрыт	
5588888	13.01.2016	Селезнев В.В.	Кредит	32 590,00 ₽	0,00 ₽	697426	Хоум Кредит энд Фина	13.01.2016	13.01.2018	85	Открыт	РосГосСтрах

Рисунок 4.6 – Отчет по договорам.

ФИО	Пол	Образование	ДатаРождения	МестоРождения	Паспорт	2документ	МестоРаботы	Должность
Семенов И.С.	Мужской	Среднее	14.12.1948	гор ижевск	3225 556666		ООО Тандер	Консультант
Селезнев В.В.	Мужской	Высшее	16.12.1988	гор. Москва	5236 788955		ИП Кольцо	Продавец
Семенова И.В.	Женский	Среднее	11.12.1985	Сызрань	1231236456	3123123123	-	-
Волков И.В.	Женский	Высшее	31.12.1985	г43				
Краснов А.В.	Мужской	Высшее	02.01.1986	г.Пенза				

Рисунок 4.7 – Отчет по клиентам

NДоговора	ТипОпераций	Дата	Сумма
78999	Гашение	27.12.2015	1 500,00 ₽
6958556	Просрочка	09.01.2016	8 000,00 ₽
5588888	Гашение	07.01.2016	750,00 ₽
524584	Гашение	17.12.2015	13 000,00 ₽

Рисунок 4.8 – Отчет по операциям

Полученные отчеты полностью удовлетворяют условиям их формирования. Программа работает корректно. В процессе тестирования ошибок и сбоев не обнаружено.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы был проведен анализ деятельности менеджера по финансовым продуктам АО «Связной Логистика», рассмотрены основные этапы выдачи и последующего обслуживания кредитного продукта.

В ходе работы были разработаны, созданы и отлажены все компоненты системы и проведена следующая работа:

- описана существующая система;
- выявлены недостатки в текущей системе;
- обоснована необходимость создания новой системы;
- определены цели, задачи учета кредитов, а также функциональные возможности разрабатываемой системы;
- разработана функциональная модель и представлено ее описание;
- продуман пользовательский интерфейс;
- проведено тестирование системы;
- разработано руководство пользователя.

В результате была разработана и отлажена новая система учета кредитов. Она обеспечивает выполнение следующих основных функций:

- учет клиентов;
- учет выданных кредитов;
- выполнение операций с кредитами: открытие, гашение, закрытие;
- формирование отчетов по выданным кредитам, просрочкам платежей по кредитам, задолженностям;
- формирование выходных документов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Ермилов В.В., Исенбаева Е.Н., Кучина Т.Л., Кучуганов В.Н., Соболева Н.В., Соловьева А.Н. Под редакцией В.Н. Кучуганова. Методические указания по оформлению математического раздела курсовых и дипломных проектов. – Ижевск: Издательство ИжГТУ, 2008 г. – 50 с.
2. Кузнецов С.Д. Основы баз данных. – 2-е изд. – М.: Интернет-Университет Информационных Технологий; БИНОМ. Лаборатория знаний, 2007. – 484 с.
3. Вальвачев А.Н., Сурков К.А., Сурков Д.А., Четырько Ю.М. Программирование на языке Delphi. Учебное пособие. – 2005.
4. Вендров А.М. CASE–технологии. Современные методы и средства проектирования информационных систем. – М.: Финансы и статистика, 2000.
5. Коннолли Т., Бегг К. Базы данных. Проектирование, реализация и сопровождение. Теория и практика = Database Systems: A Practical Approach to Design, Implementation, and Management. — 3-е изд. — М.: Вильямс, 2003. — 1436 с.
6. Гарсиа-Молина Г., Ульман Дж., Уидом Дж. Системы баз данных. Полный курс. – М.: Вильямс, 2003. – 1088 с.
7. ГОСТ 19.701– 90.Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения. 26 с. (Единая система программной документации).
8. ГОСТ 2.701 – 2008. Схемы. Виды и типы. Общие требования к выполнению. М. : Стандартинформ, 2009. 16 с. (Единая система конструкторской документации).
9. ГОСТ Р 50.1.028 – 2001. Информационные технологии поддержки жизненного цикла продукции. Методология функционального моделирования. М.: Госстандарт России. 2001. 49 с.
10. Кучуганов В.Н., Соболева Н.В. - Методические указания по оформлению курсовых работ, курсовых и дипломных проектов для студентов специальностей 230102 «Автоматизированные системы обработки информации и управления», 230104 «Системы автоматизированного проектирования», направления 230100 «Информатика и вычислительная техника». Форма обучения очная и заочная. — Ижевск: ГОУ ВПО ИжГТУ, 2010.

Приложение А

(обязательное)

Руководство пользователя

Для запуска программы следует дважды нажать значок на рабочем столе под названием Credit.exe. После загрузки на экране появляется главная форма программы. Из меню главной формы можно получить доступ к любой другой форме.

Для редактирования данных необходимо выбрать соответствующий пункт меню «Справочники» главного меню программы. Далее принципы работы с любой из этих форм одинаковы и соответствуют описанному ниже.

Добавление, удаление или сохранение данных на всех формах производится посредством соответствующих кнопок на формах, снабженных всплывающими подсказками. Для добавления новой записи следует нажать кнопку «+», после чего появится новая пустая строка в таблице данных на форме. Для изменения данных следует выбрать необходимую строку в таблице и ввести в нее новые значения полей. После добавления или изменения следует нажать кнопку с дискетой для сохранения введенных данных, либо кнопку «х» для удаления введенных данных. Для удаления выбранной в таблице на форме записи следует нажать кнопку «х».

Для того чтобы распечатать данные на форме, следует нажать кнопку с изображением принтера (при ее наличии), после чего на экране появится сформированный отчет в формате MS Excel, который можно либо просто просмотреть, либо отправить на печать.

Данные на формах можно отсортировать по большинству отображаемых столбцов в таблицах щелчком мыши по заголовку выбранного столбца.

Для просмотра, редактирования или добавления нового кредита следует нажать кнопку «Кредиты» или выбрать соответствующий пункт меню, после чего на экране появится одноименная форма. Выбранный действующий или только что заведенный договор можно открыть в отдельной форме двойным щелчком мыши по выбранному договору или нажатием кнопки «Открыть карточку договора».

С кредитом можно произвести следующие операции соответствующими кнопками на форме «Кредитный договор»:

– выдача кредита;

– закрытие кредита;

– гашение;

– вынесение кредита на просрочку.

Все операции, проводимые с кредитным договором, сохраняются в журнале операций. Форма «Журнал операций» открывается при нажатии одноименной кнопки на главной форме или выборе соответствующего пункта главного меню. Отобранные операции также можно выгрузить в MS Excel и распечатать.

Приложение Б

(обязательное)

Листинг программы

```
program Credit;

uses
  Forms,
  UnMain in 'UnMain.pas' {frmMain},
  UnPay in 'UnPay.pas' {frmPay},
  UnFamStatus in 'UnFamStatus.pas' {frmFamStatus},
  UnEducation in 'UnEducation.pas' {frmEducation},
  UnBank in 'UnBank.pas' {frmBank},
  UnCreditStatus in 'UnCreditStatus.pas' {frmCreditStatus},
  UnCompany in 'UnCompany.pas' {frmCompany},
  UnDM in 'UnDM.pas' {frmDM: TDataModule},
  UnTypePay in 'UnTypePay.pas' {frmTypePay},
  UnClient in 'UnClient.pas' {frmClient},
  UnProfile in 'UnProfile.pas' {frmProfile},
  UnCredit in 'UnCredit.pas' {frmCredit},
  UnTypeCredit in 'UnTypeCredit.pas' {frmTypeCredit},
  UnAgrm in 'UnAgrm.pas' {frmAgrm},
  UnReport in 'UnReport.pas' {frmReport};

{$R *.res}

begin
  Application.Initialize;
  Application.CreateForm(TfrmMain, frmMain);
  Application.CreateForm(TfrmPay, frmPay);
  Application.CreateForm(TfrmFamStatus, frmFamStatus);
  Application.CreateForm(TfrmEducation, frmEducation);
  Application.CreateForm(TfrmBank, frmBank);
  Application.CreateForm(TfrmCreditStatus, frmCreditStatus);
  Application.CreateForm(TfrmCompany, frmCompany);
  Application.CreateForm(TfrmTypePay, frmTypePay);
  Application.CreateForm(TfrmClient, frmClient);
  Application.CreateForm(TfrmProfile, frmProfile);
  Application.CreateForm(TfrmCredit, frmCredit);
  Application.CreateForm(TfrmTypeCredit, frmTypeCredit);
  Application.CreateForm(TfrmDM, frmDM);
  Application.CreateForm(TfrmAgrm, frmAgrm);
  Application.CreateForm(TfrmReport, frmReport);
  Application.Run;
end.


unit UnAgrm;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, DBCtrls, Mask, ComCtrls;

type
  TfrmAgrm = class(TForm)
    Button1: TButton;
    Button2: TButton;
```

```
    Label1: TLabel;
    DBEdit1: TDBEdit;
    Label2: TLabel;
    Label3: TLabel;
    DBLookupComboBox1: TDBLookupComboBox;
    Label4: TLabel;
    DBLookupComboBox2: TDBLookupComboBox;
    Label5: TLabel;
    DBEdit3: TDBEdit;
    Label6: TLabel;
    DBLookupComboBox3: TDBLookupComboBox;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    DBEdit6: TDBEdit;
    Label10: TLabel;
    DBLookupComboBox4: TDBLookupComboBox;
    Label11: TLabel;
    DBLookupComboBox5: TDBLookupComboBox;
    DateTimePicker3: TDateTimePicker;
    DateTimePicker1: TDateTimePicker;
    DateTimePicker2: TDateTimePicker;
    procedure Button2Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure DateTimePicker3Change(Sender: TObject);
    procedure DateTimePicker1Change(Sender: TObject);
    procedure DateTimePicker2Change(Sender: TObject);
    procedure DateTimePicker3DropDown(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmAgrm: TfrmAgrm;

implementation

uses UnDM;

{$R *.dfm}

procedure TfrmAgrm.Button2Click(Sender: TObject);
begin
Close;
end;

procedure TfrmAgrm.Button1Click(Sender: TObject);
begin
frmDM.tbCredit.Post;
Close;
end;

procedure TfrmAgrm.DateTimePicker3Change(Sender: TObject);
begin
frmDM.tbCredit.FieldValues['ДатаДоговора'] :=DateToStr(DateTimePicker3.Date);
end;

procedure TfrmAgrm.DateTimePicker1Change(Sender: TObject);
begin
frmDM.tbCredit.FieldValues['ДатаВыдачи'] :=DateToStr(DateTimePicker1.Date);
end;

procedure TfrmAgrm.DateTimePicker2Change(Sender: TObject);
begin
frmDM.tbCredit.FieldValues['ДатаЗакрытия'] :=DateToStr(DateTimePicker2.Date);
```

```
end;

procedure
TfrmAgrm.DateTimePicker3DropDown(Sender:
TObject);
begin
frmDM.tbCredit.Edit;
end;

end.


unit UnBank;

interface

uses
  Windows, Messages, SysUtils, Variants,
Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, DBCtrls, Grids, DBGrids;

type
  TfrmBank = class(TForm)
    DBGrid1: TDBGrid;
    DBNavigator1: TDBNavigator;
    procedure DBGrid1TitleClick(Column:
TColumn);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmBank: TfrmBank;

implementation

uses UnDM;

{$R *.dfm}

procedure
TfrmBank.DBGrid1TitleClick(Column: TColumn);
begin

frmDM.tbBank.Sort:='['+Column.FieldName+']';
end;

end.


unit UnClient;

interface

uses
  Windows, Messages, SysUtils, Classes,
Graphics, Controls, Forms, Dialogs,
  Word97, OleServer, StdCtrls, Word2000,
ExtCtrls, ComObj, DdeMan, Clipbrd,
  WordXP, ExcelXP, Grids, Excel2000, Vari-
ants, DB, ADODB, DBGrids, DBCtrls,
  ComCtrls, Buttons;

type
  TfrmClient = class(TForm)
    DBGrid1: TDBGrid;
    Panel1: TPanel;
    DBNavigator1: TDBNavigator;
    Button3: TButton;
    GroupBox1: TGroupBox;
    Label2: TLabel;
    Edit2: TEdit;
    Button4: TButton;
    Panel2: TPanel;
    DBImage1: TDBImage;
    DateTimePicker1: TDateTimePicker;
    OpenDialog1: TOpenDialog;
    Button1: TBitBtn;
    Button2: TBitBtn;
    procedure Button4Click(Sender: TObject);
    procedure Edit2Change(Sender: TObject);
    procedure
```

```
DBGrid1DrawColumnCell(Sender: TObject; const
Rect: TRect;
                DataCol: Integer; Column: TColumn;
State: TGridDrawState);
              procedure DBGrid1ColExit(Sender:
TObject);
              procedure DBGrid1KeyPress(Sender:
TObject; var Key: Char);
              procedure DateTimePicker1Change(Sender:
TObject);
              procedure
DateTimePicker1DropDown(Sender: TObject);
              procedure DBImage1Click(Sender:
TObject);
              procedure Button3Click(Sender: TObject);
              procedure DBGrid1TitleClick(Column:
TColumn);
              procedure DBGrid1DblClick(Sender:
TObject);
              procedure Button1Click(Sender: TObject);
              procedure Button2Click(Sender: TObject);
             private
              { Private declarations }
             public
              { Public declarations }
             end;

           var
            frmClient: TfrmClient;

           implementation

           uses UnDM, UnProfile;

           {$R *.dfm}

           procedure TfrmClient.Button4Click(Sender:
TObject);
           begin
           Edit2.Clear;
           end;

           procedure TfrmClient.Edit2Change(Sender:
TObject);
           begin
           if Edit2.Text <> '' then
           begin
            frmDM.tbClient.Filter:='ФИО LIKE
*'+Edit2.Text+'*';
            frmDM.tbClient.Filtered:=True;
           end
           else
           begin
            frmDM.tbClient.Filtered:=False;
           end


           end;

           procedure
TfrmClient.DBGrid1DrawColumnCell(Sender:
TObject;
             const Rect: TRect; DataCol: Integer; Col-
umn: TColumn;
             State: TGridDrawState);
           begin

             if (gdFocused in State) then
             begin
              if (Column.Field.FieldName =
'ДатаРождения') then
               with DateTimePicker1 do
               begin
                Left := Rect.Left + DBGrid1.Left + 1;
                Top := Rect.Top + DBGrid1.Top + 1;
                Width := Rect.Right - Rect.Left + 2;
                Width := Rect.Right - Rect.Left + 2;
                Height := Rect.Bottom - Rect.Top + 2;
                Visible := True;
                if (DBGrid1.SelectedField.Value=Null)
then
                 Date:=SysUtils.Date()-30*365
                else
                 Date:=DBGrid1.SelectedField.Value;
```

```
    end;
  end;

end;

procedure
TfrmClient.DBGrid1ColExit(Sender: TObject);
begin
  if DBGrid1.SelectedField.FieldName =
'ДатаРождения' then
    DateTimePicker1.Visible := False;

end;

procedure
TfrmClient.DBGrid1KeyPress(Sender: TObject; var
Key: Char);
begin
  if (key = Chr(9)) then Exit; // код табуляции
  if (DBGrid1.SelectedField.FieldName =
'ДатаРождения') then
  begin
    DateTimePicker1.SetFocus;
    SendMessage(DateTimePicker1.Handle,
WM_Char, word(Key), 0);
  end;

end;

procedure
TfrmClient.DateTimePicker1Change(Sender:
TObject);
begin
  if DBGrid1.DataSource.State in [dsEdit,
dsInsert] then

DBGrid1.SelectedField.Value:=DateToStr(DateTimePi
cker1.Date);

end;

procedure
TfrmClient.DateTimePicker1DropDown(Sender:
TObject);
begin
DBGrid1.DataSource.Edit;
end;

procedure
TfrmClient.DBImage1Click(Sender: TObject);
begin
if (OpenDialog1.Execute) then
 begin
 frmDM.tbClient.Edit;

frmDM.tbClientDSDesigner12.LoadFromFile(OpenDi
alog1.FileName);
 end
end;

procedure TfrmClient.Button3Click(Sender:
TObject);
begin
frmDM.tbClient.Edit;
if frmDM.tbClient.RecordCount=0 then
 frmDM.tbClient.Insert;
if
(frmDM.tbClient.FieldValues['ДатаРождения']=Null)
then
 begin
 frmProfile.DateTimePicker3.Date:=Date()-
30*365;

frmDM.tbClient.FieldValues['ДатаРождения']
:=DateToStr(frmProfile.DateTimePicker3.Date);
 end
else

frmProfile.DateTimePicker3.Date:=frmDM.tbClient.Fi
eldValues['ДатаРождения'];
frmProfile.Show;
end;

procedure
```

```
TfrmClient.DBGrid1TitleClick(Column: TColumn);
        begin
        if
(Column.FieldName<>'СемейноеПоложение') and
(Column.FieldName<>'Образование')  then

frmDM.tbClient.Sort:='['+Column.FieldName+']';
        end;

        procedure
TfrmClient.DBGrid1DblClick(Sender: TObject);
        begin
        Button3.Click;
        end;

        procedure TfrmClient.Button1Click(Sender:
TObject);
        var
         i,j: integer;
         Excel: Variant;
         WorkbookName: string;
        begin
         Excel:=CreateOleObject('Excel.Application');
// для остальных
          try
          Excel.SheetsInNewWorkbook:=1;
          Excel.WorkBooks.Add;
          WorkbookName := GetCurrentDir +
'\Клиенты.xls';
          Ex-
cel.ActiveWorkbook.SaveAs(WorkbookName);
//Открытие книги
          Excel.WorkSheets[1].Select;

          for i:=0 to frmDM.tbClient.FieldList.Count-6
do    //вывод шапки таблицы
          begin
          Ex-
cel.Cells[1,i+1]:=frmDM.tbClient.FieldList.Fields[i].D
isplayName;
          Ex-
cel.WorkSheets[1].Columns.Item[i+1].ColumnWidth:=
frmDM.tbClient.FieldList.Fields[i].DisplayWidth;
          frmDM.tbClient.First;
          j:=2;
          while not frmDM.tbClient.Eof do
//вывод данных таблицы
           begin
           Ex-
cel.Cells[j,i+1]:=frmDM.tbClient.FieldValues[frmDM.
tbClient.FieldList.Fields[i].FieldName];
           frmDM.tbClient.Next;
           j:=j+1;
           end;
          end;
          frmDM.tbClient.First;

          Excel.Range['A1:M'+IntToStr(j-1)].Select;
          Excel.Selection.Borders.LineStyle:=1;



          Excel.ActiveWorkbook.Save;     //сохранить
документ  Excel
          Excel.WindowState := -4137;    //развернуть
на весь экран
          Excel.Visible := True;         //вывести на эк-
ран


          except
           Excel.Quit; //закрыть, если произошла
ошибка работы с Excel
          end;


        end;

        procedure TfrmClient.Button2Click(Sender:
TObject);
        begin
        Close;
        end;

        end.
```

```
unit UnCompany;

interface

uses
  Windows, Messages, SysUtils, Variants,
Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, DBCtrls, Grids, DBGrids;

type
  TfrmCompany = class(TForm)
    DBGrid1: TDBGrid;
    DBNavigator1: TDBNavigator;
    procedure DBGrid1TitleClick(Column:
TColumn);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmCompany: TfrmCompany;

implementation

uses UnDM;

{$R *.dfm}

procedure
TfrmCompany.DBGrid1TitleClick(Column:
TColumn);
begin

frmDM.tbCompany.Sort:='['+Column.FieldName+']';
end;

end.
```

```
unit UnCredit;

interface

uses
  Windows, Messages, SysUtils, Classes,
Graphics, Controls, Forms, Dialogs,
  Word97, OleServer, StdCtrls, Word2000,
ExtCtrls, ComObj, DdeMan, Clipbrd,
  WordXP, ExcelXP, Grids, Excel2000, Vari-
ants, DB, ADODB, DBGrids, DBCtrls,
  ComCtrls, Buttons;

type
  TfrmCredit = class(TForm)
    DBGrid1: TDBGrid;
    Panel1: TPanel;
    DBNavigator1: TDBNavigator;
    GroupBox2: TGroupBox;
    CheckBox1: TCheckBox;
    CheckBox2: TCheckBox;
    CheckBox3: TCheckBox;
    Label2: TLabel;
    CheckBox4: TCheckBox;
    DateTimePicker1: TDateTimePicker;
    DateTimePicker2: TDateTimePicker;
    CheckBox5: TCheckBox;
    DBLookupComboBox1:
TDBLookupComboBox;
    DBLookupComboBox2:
TDBLookupComboBox;
    DBLookupComboBox3:
TDBLookupComboBox;
    Button4: TButton;
    Label1: TLabel;
    Edit1: TEdit;
    Button5: TButton;
    DateTimePicker3: TDateTimePicker;
    DateTimePicker4: TDateTimePicker;
    DateTimePicker5: TDateTimePicker;
    Button1: TBitBtn;
    Button3: TBitBtn;
```

```
            Button6: TBitBtn;
            Button2: TBitBtn;
            GroupBox1: TGroupBox;
            procedure Button4Click(Sender: TObject);
            procedure CheckBox1Click(Sender:
TObject);
            procedure CheckBox2Click(Sender:
TObject);
            procedure CheckBox3Click(Sender:
TObject);
            procedure Button5Click(Sender: TObject);
            procedure CheckBox4Click(Sender:
TObject);
            procedure CheckBox5Click(Sender:
TObject);
            procedure
DateTimePicker3DropDown(Sender: TObject);
            procedure DateTimePicker3Change(Sender:
TObject);
            procedure
DateTimePicker3KeyPress(Sender: TObject; var Key:
Char);
            procedure DBGrid1ColExit(Sender:
TObject);
            procedure
DateTimePicker4KeyPress(Sender: TObject; var Key:
Char);
            procedure
DateTimePicker5KeyPress(Sender: TObject; var Key:
Char);
            procedure DateTimePicker4Change(Sender:
TObject);
            procedure DateTimePicker5Change(Sender:
TObject);
            procedure
DateTimePicker4DropDown(Sender: TObject);
            procedure
DateTimePicker5DropDown(Sender: TObject);
            procedure
DBGrid1DrawColumnCell(Sender: TObject; const
Rect: TRect;
              DataCol: Integer; Column: TColumn;
State: TGridDrawState);
            procedure
DBLookupComboBox1Click(Sender: TObject);
            procedure DBGrid1TitleClick(Column:
TColumn);
            procedure DBGrid1DblClick(Sender:
TObject);
            procedure Button1Click(Sender: TObject);
            procedure BitBtn4Click(Sender: TObject);
            procedure Button6Click(Sender: TObject);
            procedure Button3Click(Sender: TObject);
          private
            { Private declarations }
          public
            { Public declarations }
          end;

        var
          frmCredit: TfrmCredit;

        implementation

        uses UnDM, UnAgrm, UnPay;

        {$R *.dfm}

        procedure TfrmCredit.Button4Click(Sender:
TObject);
        begin
        Edit1.Clear;
        CheckBox1.Checked:=False;
        CheckBox2.Checked:=False;
        CheckBox3.Checked:=False;
        CheckBox4.Checked:=False;
        CheckBox5.Checked:=False;
        DateTimePicker1.Enabled:=False;
        DateTimePicker2.Enabled:=False;

        frmDM.tbCredit.Filtered:=False;
        end;

        procedure
```

```
TfrmCredit.CheckBox1Click(Sender: TObject);
        begin
        if CheckBox1.Checked then
         begin
         DBLookupComboBox1.Enabled:=True;

DBLookupComboBox1.KeyValue:=frmDM.tbClient.F
ieldValues['ID'];
         end
        else
         begin
          DBLookupComboBox1.Enabled:=False;
          DBLookupComboBox1.KeyValue:=Null;
         end;

        frmCredit.DBLookupComboBox1Click(Sende
r);
        end;

        procedure
TfrmCredit.CheckBox2Click(Sender: TObject);
        begin
        if CheckBox2.Checked then
         begin
         DBLookupComboBox2.Enabled:=True;

DBLookupComboBox2.KeyValue:=frmDM.tbTypeCr
edit.FieldValues['ID'];
         end
        else
         begin
          DBLookupComboBox2.Enabled:=False;
          DBLookupComboBox2.KeyValue:=Null;
         end;

        frmCredit.DBLookupComboBox1Click(Sende
r) ;
        end;

        procedure
TfrmCredit.CheckBox3Click(Sender: TObject);
        begin
        if CheckBox3.Checked then
         begin
         DBLookupComboBox3.Enabled:=True;

DBLookupComboBox3.KeyValue:=frmDM.tbCreditSt
atus.FieldValues['ID'];
         end
        else
         begin
          DBLookupComboBox3.Enabled:=False;
          DBLookupComboBox3.KeyValue:=Null;
         end;

        frmCredit.DBLookupComboBox1Click(Sende
r) ;
        end;

        procedure TfrmCredit.Button5Click(Sender:
TObject);
        begin
        Edit1.Clear;
        end;

        procedure
TfrmCredit.CheckBox4Click(Sender: TObject);
        begin
        if CheckBox4.Checked then
         DateTimePicker1.Enabled:=True
        else
         DateTimePicker1.Enabled:=False;

        frmCredit.DBLookupComboBox1Click(Sende
r) ;
        end;

        procedure
TfrmCredit.CheckBox5Click(Sender: TObject);
        begin
        if CheckBox5.Checked then
         DateTimePicker2.Enabled:=True
        else
         DateTimePicker2.Enabled:=False;
```

```
        frmCredit.DBLookupComboBox1Click(Sende
r) ;
        end;

        procedure
TfrmCredit.DateTimePicker3DropDown(Sender:
TObject);
        begin
        DBGrid1.DataSource.Edit;
        end;

        procedure
TfrmCredit.DateTimePicker3Change(Sender:
TObject);
        begin
          if DBGrid1.DataSource.State in [dsEdit,
dsInsert] then

DBGrid1.SelectedField.Value:=DateToStr(DateTimePi
cker3.Date);

        end;

        procedure
TfrmCredit.DateTimePicker3KeyPress(Sender:
TObject;
          var Key: Char);
        begin
          if (key = Chr(9)) then Exit; // код табуляции
          if (DBGrid1.SelectedField.FieldName =
'ДатаДоговора') then
          begin
            DateTimePicker3.SetFocus;
            SendMessage(DateTimePicker3.Handle,
WM_Char, word(Key), 0);
          end;

        end;

        procedure
TfrmCredit.DBGrid1ColExit(Sender: TObject);
        begin
          if DBGrid1.SelectedField.FieldName =
'ДатаДоговора' then
            DateTimePicker3.Visible := False;
          if DBGrid1.SelectedField.FieldName =
'ДатаВыдачи' then
            DateTimePicker4.Visible := False;
          if DBGrid1.SelectedField.FieldName =
'ДатаЗакрытия' then
            DateTimePicker5.Visible := False;
        end;

        procedure
TfrmCredit.DateTimePicker4KeyPress(Sender:
TObject;
          var Key: Char);
        begin
          if (key = Chr(9)) then Exit; // код табуляции
          if (DBGrid1.SelectedField.FieldName =
'ДатаВыдачи') then
          begin
            DateTimePicker4.SetFocus;
            SendMessage(DateTimePicker4.Handle,
WM_Char, word(Key), 0);
          end;
        end;

        procedure
TfrmCredit.DateTimePicker5KeyPress(Sender:
TObject;
          var Key: Char);
        begin
          if (key = Chr(9)) then Exit; // код табуляции
          if (DBGrid1.SelectedField.FieldName =
'ДатаЗакрытия') then
          begin
            DateTimePicker5.SetFocus;
            SendMessage(DateTimePicker5.Handle,
WM_Char, word(Key), 0);
          end;
        end;
```

```
        procedure
TfrmCredit.DateTimePicker4Change(Sender:
TObject);
        begin
          if DBGrid1.DataSource.State in [dsEdit,
dsInsert] then

DBGrid1.SelectedField.Value:=DateToStr(DateTimePi
cker4.Date);

        end;

        procedure
TfrmCredit.DateTimePicker5Change(Sender:
TObject);
        begin
          if DBGrid1.DataSource.State in [dsEdit,
dsInsert] then

DBGrid1.SelectedField.Value:=DateToStr(DateTimePi
cker5.Date);
        end;

        procedure
TfrmCredit.DateTimePicker4DropDown(Sender:
TObject);
        begin
        DBGrid1.DataSource.Edit;
        end;

        procedure
TfrmCredit.DateTimePicker5DropDown(Sender:
TObject);
        begin
        DBGrid1.DataSource.Edit;
        end;

        procedure
TfrmCredit.DBGrid1DrawColumnCell(Sender:
TObject;
          const Rect: TRect; DataCol: Integer; Col-
umn: TColumn;
          State: TGridDrawState);
        begin
        if (gdFocused in State) then
        begin
          if (Column.Field.FieldName =
'ДатаДоговора') then
          with DateTimePicker3 do
          begin
            Left := Rect.Left + DBGrid1.Left + 1;
            Top := Rect.Top + DBGrid1.Top + 1;
            Width := Rect.Right - Rect.Left + 2;
            Width := Rect.Right - Rect.Left + 2;
            Height := Rect.Bottom - Rect.Top + 2;
            Visible := True;
            if (DBGrid1.SelectedField.Value=Null)
then
              Date:=SysUtils.Date()
            else
              Date:=DBGrid1.SelectedField.Value;
          end;
          if (Column.Field.FieldName =
'ДатаВыдачи') then
          with DateTimePicker4 do
          begin
            Left := Rect.Left + DBGrid1.Left + 1;
            Top := Rect.Top + DBGrid1.Top + 1;
            Width := Rect.Right - Rect.Left + 2;
            Width := Rect.Right - Rect.Left + 2;
            Height := Rect.Bottom - Rect.Top + 2;
            Visible := True;
            if (DBGrid1.SelectedField.Value=Null)
then
              Date:=SysUtils.Date()
            else
              Date:=DBGrid1.SelectedField.Value;
          end;
          if (Column.Field.FieldName =
'ДатаЗакрытия') then
          with DateTimePicker5 do
          begin
            Left := Rect.Left + DBGrid1.Left + 1;
            Top := Rect.Top + DBGrid1.Top + 1;
```

```
            Width := Rect.Right - Rect.Left + 2;
            Width := Rect.Right - Rect.Left + 2;
            Height := Rect.Bottom - Rect.Top + 2;
            Visible := True;
            if (DBGrid1.SelectedField.Value=Null)
then
              Date:=SysUtils.Date()+365
            else
              Date:=DBGrid1.SelectedField.Value;
          end;
        end;

      end;

      procedure
TfrmCredit.DBLookupComboBox1Click(Sender:
TObject);
      begin

      frmDM.tbCredit.Filter:='NДоговора <> 0';

      if (frmCredit.Edit1.Text<>'') then

frmDM.tbCredit.Filter:=frmDM.tbCredit.Filter+' and
NДоговора LIKE *'+frmCredit.Edit1.Text+'*';
      if (frmCredit.CheckBox1.Checked=true) and
(frmDM.tbClient.FieldValues['ID']<>null) then

frmDM.tbCredit.Filter:=frmDM.tbCredit.Filter+' and
КлиентID LIKE '+string
(DBLookupComboBox1.KeyValue);
      if (frmCredit.CheckBox2.Checked=true) and
(frmDM.tbTypeCredit.FieldValues['ID']<>null) then

frmDM.tbCredit.Filter:=frmDM.tbCredit.Filter+' and
ТипКредитаID LIKE '+string
(DBLookupComboBox2.KeyValue);
      if (frmCredit.CheckBox3.Checked=true) and
(frmDM.tbCreditStatus.FieldValues['ID']<>null) then

frmDM.tbCredit.Filter:=frmDM.tbCredit.Filter+' and
СтатусID LIKE '+string
(DBLookupComboBox3.KeyValue);
      if (frmCredit.DateTimePicker1.Enabled=True)
then

frmDM.tbCredit.Filter:=frmDM.tbCredit.Filter+' and
ДатаДоговора >=
'+DateToStr(frmCredit.DateTimePicker1.Date);
      if (frmCredit.DateTimePicker2.Enabled=True)
then

frmDM.tbCredit.Filter:=frmDM.tbCredit.Filter+' and
ДатаДоговора <=
'+DateToStr(frmCredit.DateTimePicker2.Date);

      frmDM.tbCredit.Filtered:=True;
      end;

      procedure
TfrmCredit.DBGrid1TitleClick(Column: TColumn);
      begin
      if (Column.FieldName<>'Заемщик') and
(Column.FieldName<>'ТипКредита') and (Col-
umn.FieldName<>'Банк')
and(Column.FieldName<>'Статус') and (Col-
umn.FieldName<>'СтраховаяКомпания') then

frmDM.tbCredit.Sort:='['+Column.FieldName+']';
      end;

      procedure
TfrmCredit.DBGrid1DblClick(Sender: TObject);
      begin
      Button3.Click;
      end;

      procedure TfrmCredit.Button1Click(Sender:
TObject);
      var
       i,j: integer;
       Excel: Variant;
       WorkbookName: string;
      begin
```

```
Excel:=CreateOleObject('Excel.Application');
// для остальных
try
Excel.SheetsInNewWorkbook:=1;
Excel.WorkBooks.Add;
WorkbookName := GetCurrentDir +
'\Кредиты.xls';
Ex-
cel.ActiveWorkbook.SaveAs(WorkbookName);
//Открытие книги
Excel.WorkSheets[1].Select;

for i:=0 to frmDM.tbCredit.FieldList.Count-
7 do  //вывод шапки таблицы
begin
Ex-
cel.Cells[1,i+1]:=frmDM.tbCredit.FieldList.Fields[i].D
isplayName;
Ex-
cel.WorkSheets[1].Columns.Item[i+1].ColumnWidth:=
frmDM.tbCredit.FieldList.Fields[i].DisplayWidth;
frmDM.tbCredit.First;
j:=2;
while not frmDM.tbCredit.Eof do
//вывод данных таблицы
 begin
 Ex-
cel.Cells[j,i+1]:=frmDM.tbCredit.FieldValues[frmDM.
tbCredit.FieldList.Fields[i].FieldName];
 frmDM.tbCredit.Next;
 j:=j+1;
 end;
end;
frmDM.tbCredit.First;

Excel.Range['A1:M'+IntToStr(j-1)].Select;
Excel.Selection.Borders.LineStyle:=1;


Excel.ActiveWorkbook.Save;  //сохранить
документ  Excel
Excel.WindowState := -4137;  //развернуть
на весь экран
Excel.Visible := True;  //вывести на эк-
ран

except
 Excel.Quit; //закрыть, если произошла
ошибка работы с Excel
end;

end;

procedure TfrmCredit.BitBtn4Click(Sender:
TObject);
begin
Close;
end;

procedure TfrmCredit.Button6Click(Sender:
TObject);
begin
frmPay.CheckBox1.Checked:=false;
frmPay.CheckBox1.Checked:=true;
frmPay.Show;
end;

procedure TfrmCredit.Button3Click(Sender:
TObject);
begin
frmDM.tbCredit.Edit;
if frmDM.tbCredit.RecordCount=0 then
 frmDM.tbCredit.Insert;

if
(frmDM.tbCredit.FieldValues['ДатаДоговора']=Null)
then
 begin
 frmAgrm.DateTimePicker3.Date:=Date();

frmDM.tbCredit.FieldValues['ДатаДоговора']
:=DateToStr(frmAgrm.DateTimePicker3.Date);
 end
else
```

```
frmAgrm.DateTimePicker3.Date:=frmDM.tbCredit.Fie
ldValues['ДатаДоговора'];

        if
(frmDM.tbCredit.FieldValues['ДатаВыдачи']=Null)
then
          begin
          frmAgrm.DateTimePicker1.Date:=Date();
          frmDM.tbCredit.FieldValues['ДатаВыдачи']
:=DateToStr(frmAgrm.DateTimePicker1.Date);
          end
        else

frmAgrm.DateTimePicker1.Date:=frmDM.tbCredit.Fie
ldValues['ДатаВыдачи'];

        if
(frmDM.tbCredit.FieldValues['ДатаЗакрытия']=Null)
then
          begin
          frmAgrm.DateTimePicker2.Date:=Date();

frmDM.tbCredit.FieldValues['ДатаЗакрытия']
:=DateToStr(frmAgrm.DateTimePicker2.Date+365);
          end
        else

frmAgrm.DateTimePicker2.Date:=frmDM.tbCredit.Fie
ldValues['ДатаЗакрытия'];

        frmAgrm.Show;
        end;

        end.


        unit UnCreditStatus;

        interface

        uses
          Windows, Messages, SysUtils, Variants,
Classes, Graphics, Controls, Forms,
          Dialogs, ExtCtrls, DBCtrls, Grids, DBGrids;

        type
          TfrmCreditStatus = class(TForm)
            DBGrid1: TDBGrid;
            DBNavigator1: TDBNavigator;
            procedure DBGrid1TitleClick(Column:
TColumn);
          private
            { Private declarations }
          public
            { Public declarations }
          end;

        var
          frmCreditStatus: TfrmCreditStatus;

        implementation

        uses UnDM;

        {$R *.dfm}

        procedure
TfrmCreditStatus.DBGrid1TitleClick(Column:
TColumn);
        begin
        frmDM.tbCreditStatus.Sort:='['+Column.Field
Name+']';
        end;

        end.


        unit UnDM;

        interface

        uses
          SysUtils, Classes, DB, ADODB, Variants,
```

```
DateUtils;

    type
     TfrmDM = class(TDataModule)
      ADOConnection1: TADOConnection;
      tbFamStatus: TADOTable;
      dsFamStatus: TDataSource;
      tbEducation: TADOTable;
      dsEducation: TDataSource;
      dsTypePay: TDataSource;
      tbTypePay: TADOTable;
      tbBank: TADOTable;
      dsBank: TDataSource;
      tbCreditStatus: TADOTable;
      dsCreditStatus: TDataSource;
      tbCredit: TADOTable;
      dsCredit: TDataSource;
      tbClient: TADOTable;
      dsClient: TDataSource;
      dsPay: TDataSource;
      tbPay: TADOTable;
      tbClientID: TAutoIncField;
      tbClientDSDesigner: TWideStringField;
      tbClientDSDesigner3: TDateTimeField;
      tbClientDSDesigner5: TWideStringField;
      tbClientDSDesigner22: TWideStringField;
      tbClientID2: TIntegerField;
      tbClientID3: TIntegerField;
      tbClientDSDesigner6: TWideStringField;
      tbClientDSDesigner7: TWideStringField;
      tbClientDSDesigner8: TBCDField;
      tbClientDSDesigner9: TWideStringField;
      tbClientDSDesigner10: TWideStringField;
      tbClientDSDesigner11: TWideStringField;
      tbClientDSDesigner12: TBlobField;
      tbClientField: TStringField;
      tbClientDSDesigner4: TWideStringField;
      tbEducationID: TAutoIncField;
      tbEducationDSDesigner:
TWideStringField;
      tbFamStatusID: TAutoIncField;
      tbFamStatusDSDesigner:
TWideStringField;
      tbTypePayID: TAutoIncField;
      tbTypePayDSDesigner: TWideStringField;
      tbBankID: TAutoIncField;
      tbBankDSDesigner: TWideStringField;
      tbCreditStatusID: TAutoIncField;
      tbCreditStatusDSDesigner:
TWideStringField;
      tbPayID2: TIntegerField;
      tbPayDSDesigner: TDateTimeField;
      tbPayID3: TIntegerField;
      tbPayDSDesigner2: TBCDField;
      tbPayField2: TStringField;
      tbCreditID: TAutoIncField;
      tbCreditID2: TIntegerField;
      tbCreditID3: TIntegerField;
      tbCreditID4: TIntegerField;
      tbCreditID5: TIntegerField;
      tbCreditN: TWideStringField;
      tbCreditDSDesigner3: TBCDField;
      tbCreditDSDesigner4: TBCDField;
      tbCreditDSDesigner5: TIntegerField;
      tbPayField3: TStringField;
      tbPayID: TAutoIncField;
      tbCompany: TADOTable;
      dsCompany: TDataSource;
      tbCompanyID: TAutoIncField;
      tbCompanyDSDesigner: TWideStringField;
      tbCreditDSDesigner: TDateTimeField;
      tbCreditDSDesigner6: TDateTimeField;
      tbCreditField: TStringField;
      tbCreditID6: TIntegerField;
      tbTypeCredit: TADOTable;
      dsTypeCredit: TDataSource;
      tbTypeCreditID: TAutoIncField;
      tbTypeCreditDSDesigner:
TWideStringField;
      tbCreditField2: TStringField;
      tbCreditField3: TStringField;
      tbCreditField4: TStringField;
      tbCreditField5: TStringField;
      tbCreditDSDesigner2: TDateTimeField;
```

```
    tbClientDSDesigner2: TWideStringField;
    tbClientField2: TStringField;
    ADOQuery1: TADOQuery;
    ADOQuery1S: TBCDField;
    tbCreditField6: TCurrencyField;
    procedure tbPayAfterPost(DataSet:
TDataSet);
    procedure tbPayAfterInsert(DataSet:
TDataSet);
    procedure tbCreditCalcFields(DataSet:
TDataSet);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmDM: TfrmDM;

implementation

uses UnCredit;

{$R *.dfm}

procedure TfrmDM.tbPayAfterPost(DataSet:
TDataSet);
begin


  tbCredit.Locate('NДоговора',
tbPay.FieldValues['NДоговора'], []);
  tbCredit.Edit;
  if
(tbPay.FieldValues['ТипОперацииID']<>Null) then
  case tbPay.FieldValues['ТипОперацииID'] of
  1:
    begin
    tbCredit.FieldValues['СтатусID']:=2;
    end;
  2: begin
    tbCredit.FieldValues['СтатусID']:=1;
    end;
  3:
    begin
    tbCredit.FieldValues['СтатусID']:=2;
    end;
  4: begin
    tbCredit.FieldValues['СтатусID']:=3;
    end;
  end;
  tbCredit.Post;
  //tbCredit.Active:=False;
  //tbCredit.Active:=True;
  end;

  procedure TfrmDM.tbPayAfterInsert(DataSet:
TDataSet);
  begin
  tbPay.FieldValues['Дата']:=Date();
  end;

  procedure
TfrmDM.tbCreditCalcFields(DataSet: TDataSet);
  var
  Sum: currency;
  MonthCount: integer;
  begin

  if tbCredit.State = dsCalcFields  then
  begin

  Sum:=0;
  ADOQuery1.Parameters[0].Name;
  ADOQuery1.Parameters[0].Value:=tbCredit.F
ieldValues['ID'];
  ADOQuery1.Active:=False;
  ADOQuery1.Active:=True;
  Sum:=ADOQuery1S.Value;
  //((date1.Year - date2.Year) * 12) +
date1.Month - date2.Month

  MonthCount:=(YearOf(tbCredit.FieldValues['
```

```
ДатаЗакрытия'])-
YearOf(tbCredit.FieldValues['ДатаВыдачи']))*12+Mo
nthOf(tbCredit.FieldValues['ДатаЗакрытия'])-
MonthOf(tbCredit.FieldValues['ДатаВыдачи']);

        tbCredit.FieldValues['СуммаВзноса']:=Sum;
        if
(tbCredit.FieldValues['Сумма']<>Null)and(tbCredit.Fi
eldValues['Ставка']<>Null)and(tbCredit.FieldValues['
СуммаВзноса']<>Null) then
        tbCredit.FieldValues['Остаток']:=tbCredit.Fie
ldValues['Сумма']+tbCredit.FieldValues['Сумма']*tbC
redit.FieldValues['Ставка']*MonthCount/100-
tbCredit.FieldValues['СуммаВзноса'];
        end;

        end;

        end.


        unit UnEducation;

        interface

        uses
          Windows, Messages, SysUtils, Variants,
Classes, Graphics, Controls, Forms,
          Dialogs, ExtCtrls, DBCtrls, Grids, DBGrids;

        type
          TfrmEducation = class(TForm)
            DBGrid1: TDBGrid;
            DBNavigator1: TDBNavigator;
            procedure DBGrid1TitleClick(Column:
TColumn);
          private
            { Private declarations }
          public
            { Public declarations }
          end;

        var
          frmEducation: TfrmEducation;

        implementation

        uses UnDM;

        {$R *.dfm}

        procedure
TfrmEducation.DBGrid1TitleClick(Column:
TColumn);
        begin

frmDM.tbEducation.Sort:='['+Column.FieldName+']';
        end;

        end.


        unit UnFamStatus;

        interface

        uses
          Windows, Messages, SysUtils, Variants,
Classes, Graphics, Controls, Forms,
          Dialogs, ExtCtrls, DBCtrls, Grids, DBGrids;

        type
          TfrmFamStatus = class(TForm)
            DBGrid1: TDBGrid;
            DBNavigator1: TDBNavigator;
            procedure DBGrid1TitleClick(Column:
TColumn);
          private
            { Private declarations }
          public
            { Public declarations }
          end;

        var
```

```
  frmFamStatus: TfrmFamStatus;

implementation

uses UnDM;

{$R *.dfm}

procedure
TfrmFamStatus.DBGrid1TitleClick(Column:
TColumn);
begin

frmDM.tbFamStatus.Sort:='['+Column.FieldName+']';
end;

end.



unit UnMain;

interface

uses
  Windows, Messages, SysUtils, Variants,
Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Menus, DBCtrls, XPMan,
Buttons, ImgList, ComCtrls;

type
  TfrmMain = class(TForm)
    MainMenu1: TMainMenu;
    N1: TMenuItem;
    N2: TMenuItem;
    N3: TMenuItem;
    N4: TMenuItem;
    N5: TMenuItem;
    N6: TMenuItem;
    N7: TMenuItem;
    N8: TMenuItem;
    N9: TMenuItem;
    N10: TMenuItem;
    N11: TMenuItem;
    N12: TMenuItem;
    N13: TMenuItem;
    N14: TMenuItem;
    XPManifest1: TXPManifest;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    BitBtn3: TBitBtn;
    BitBtn4: TBitBtn;
    BitBtn5: TBitBtn;
    StatusBar1: TStatusBar;
    BitBtn6: TBitBtn;
    procedure N4Click(Sender: TObject);
    procedure N5Click(Sender: TObject);
    procedure N6Click(Sender: TObject);
    procedure N7Click(Sender: TObject);
    procedure N8Click(Sender: TObject);
    procedure N9Click(Sender: TObject);
    procedure N11Click(Sender: TObject);
    procedure N10Click(Sender: TObject);
    procedure N13Click(Sender: TObject);
    procedure N12Click(Sender: TObject);
    procedure N14Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
    procedure BitBtn4Click(Sender: TObject);
    procedure BitBtn5Click(Sender: TObject);
    procedure BitBtn6Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmMain: TfrmMain;

implementation

uses UnDM, UnBank, UnClient, UnCompany,
UnCreditStatus, UnEducation,
```

```
  UnFamStatus, UnPay, UnProfile,
UnTypePay, UnCredit, UnTypeCredit,
  UnReport;

{$R *.dfm}

procedure TfrmMain.N4Click(Sender:
TObject);
begin
frmDM.tbFamStatus.Active:=False;
frmDM.tbFamStatus.Active:=True;
frmDM.tbFamStatus.Filtered:=False;
frmFamStatus.Show;
end;

procedure TfrmMain.N5Click(Sender:
TObject);
begin
frmDM.tbEducation.Active:=False;
frmDM.tbEducation.Active:=True;
frmDM.tbEducation.Filtered:=False;
frmEducation.Show;
end;

procedure TfrmMain.N6Click(Sender:
TObject);
begin
frmDM.tbTypePay.Active:=False;
frmDM.tbTypePay.Active:=True;
frmDM.tbTypePay.Filtered:=False;
frmTypePay.Show;
end;

procedure TfrmMain.N7Click(Sender:
TObject);
begin
frmDM.tbBank.Active:=False;
frmDM.tbBank.Active:=True;
frmDM.tbBank.Filtered:=False;
frmBank.Show;
end;

procedure TfrmMain.N8Click(Sender:
TObject);
begin
frmDM.tbCreditStatus.Active:=False;
frmDM.tbCreditStatus.Active:=True;
frmDM.tbCreditStatus.Filtered:=False;
frmCreditStatus.Show;
end;

procedure TfrmMain.N9Click(Sender:
TObject);
begin
frmDM.tbCompany.Active:=False;
frmDM.tbCompany.Active:=True;
frmDM.tbCompany.Filtered:=False;
frmCompany.Show;
end;

procedure TfrmMain.N11Click(Sender:
TObject);
begin
frmDM.tbPay.Active:=False;
frmDM.tbPay.Active:=True;
frmDM.tbPay.Filtered:=False;
frmPay.Show;
end;

procedure TfrmMain.N10Click(Sender:
TObject);
begin
frmDM.tbCredit.Active:=False;
frmDM.tbCredit.Active:=True;
frmCredit.DateTimePicker1.Date:=Date()-30;
frmCredit.DateTimePicker2.Date:=Date()+30;
frmCredit.Button4.Click;
frmCredit.Show;
end;

procedure TfrmMain.N13Click(Sender:
TObject);
begin
frmDM.tbTypeCredit.Active:=False;
```

```
        frmDM.tbTypeCredit.Active:=True;
        frmDM.tbTypeCredit.Filtered:=False;
        frmTypeCredit.Show;
        end;

        procedure TfrmMain.N12Click(Sender:
TObject);
        begin
        frmClient.Button4.Click;
        frmDM.tbClient.Active:=False;
        frmDM.tbClient.Active:=True;
        frmDM.tbClient.First;
        frmClient.Show;
        end;

        procedure TfrmMain.N14Click(Sender:
TObject);
        begin
        Close;
        end;

        procedure TfrmMain.BitBtn1Click(Sender:
TObject);
        begin
        N12.Click;
        end;

        procedure TfrmMain.BitBtn2Click(Sender:
TObject);
        begin
        N10.Click;
        end;

        procedure TfrmMain.BitBtn3Click(Sender:
TObject);
        begin
        frmDM.tbPay.Active:=False;
        frmDM.tbPay.Active:=True;
        frmPay.Button1.Click;
        frmPay.Show;
        end;

        procedure TfrmMain.BitBtn4Click(Sender:
TObject);
        begin
        N9.Click;
        end;

        procedure TfrmMain.BitBtn5Click(Sender:
TObject);
        begin
        Close;
        end;

        procedure TfrmMain.BitBtn6Click(Sender:
TObject);
        begin
        frmReport.Show;
        end;

        end.


        unit UnPay;

        interface

        uses
          Windows, Messages, SysUtils, Classes,
Graphics, Controls, Forms, Dialogs,
          Word97, OleServer, StdCtrls, Word2000,
ExtCtrls, ComObj, DdeMan, Clipbrd,
          WordXP, ExcelXP, Grids, Excel2000, Vari-
ants, DB, ADODB, DBGrids, DBCtrls,
          ComCtrls, Buttons;

        type
          TfrmPay = class(TForm)
            DBGrid1: TDBGrid;
            Panel1: TPanel;
            DBNavigator1: TDBNavigator;
            GroupBox1: TGroupBox;
            Button1: TButton;
            CheckBox1: TCheckBox;
```

```
            DBLookupComboBox1:
TDBLookupComboBox;
            DBLookupComboBox2:
TDBLookupComboBox;
            CheckBox2: TCheckBox;
            Button3: TBitBtn;
            Button2: TBitBtn;
            DateTimePicker4: TDateTimePicker;
            procedure Button1Click(Sender: TObject);
            procedure CheckBox1Click(Sender:
TObject);
            procedure CheckBox2Click(Sender:
TObject);
            procedure
DBLookupComboBox1Click(Sender: TObject);
            procedure DBGrid1TitleClick(Column:
TColumn);
            procedure Button3Click(Sender: TObject);
            procedure Button2Click(Sender: TObject);
            procedure DBGrid1ColExit(Sender:
TObject);
            procedure
DBGrid1DrawColumnCell(Sender: TObject; const
Rect: TRect;
              DataCol: Integer; Column: TColumn;
State: TGridDrawState);
            procedure DateTimePicker4Change(Sender:
TObject);
            procedure
DateTimePicker4DropDown(Sender: TObject);
            procedure
DateTimePicker4KeyPress(Sender: TObject; var Key:
Char);
          private
            { Private declarations }
          public
            { Public declarations }
          end;

        var
          frmPay: TfrmPay;

          implementation

          uses UnDM, UnCredit;

          {$R *.dfm}

          procedure TfrmPay.Button1Click(Sender:
TObject);
          begin
          CheckBox1.Checked:=False;
          CheckBox2.Checked:=False;
          frmDM.tbPay.Filtered:=False;
          end;

          procedure TfrmPay.CheckBox1Click(Sender:
TObject);
          begin
          if CheckBox1.Checked then
           begin
           DBLookupComboBox1.Enabled:=True;

DBLookupComboBox1.KeyValue:=frmDM.tbCredit.F
ieldValues['ID'];
           end
          else
           begin
            DBLookupComboBox1.Enabled:=False;
            DBLookupComboBox1.KeyValue:=Null;
           end;

          DBLookupComboBox1Click(Sender);
          end;

          procedure TfrmPay.CheckBox2Click(Sender:
TObject);
          begin
          if CheckBox2.Checked then
           begin
           DBLookupComboBox2.Enabled:=True;

DBLookupComboBox2.KeyValue:=frmDM.tbTypePa
y.FieldValues['ID'];
```

```
        end
       else
        begin
         DBLookupComboBox2.Enabled:=False;
         DBLookupComboBox2.KeyValue:=Null;
        end;

       DBLookupComboBox1Click(Sender);
       end;

       procedure
TfrmPay.DBLookupComboBox1Click(Sender:
TObject);
       begin
       frmDM.tbPay.Filter:='ID <> 0';

       if (frmPay.CheckBox1.Checked=true) and
(frmDM.tbCredit.FieldValues['ID']<>null) then
        frmDM.tbPay.Filter:=frmDM.tbPay.Filter+'
and КредитID LIKE
'+IntToStr(frmDM.tbCredit.FieldValues['ID']);
       if (frmPay.CheckBox2.Checked=true) and
(frmDM.tbTypePay.FieldValues['ID']<>null) then
        frmDM.tbPay.Filter:=frmDM.tbPay.Filter+'
and ТипОперацииID LIKE
'+IntToStr(frmDM.tbTypePay.FieldValues['ID']);
       frmDM.tbPay.Filtered:=True;
       end;

       procedure
TfrmPay.DBGrid1TitleClick(Column: TColumn);
       begin
       if (Column.FieldName<>'NДоговора') and
(Column.FieldName<>'ТипОпераций')  then

frmDM.tbPay.Sort:='['+Column.FieldName+']';
       end;

       procedure TfrmPay.Button3Click(Sender:
TObject);
       var
        i,j: integer;
        Excel: Variant;
        WorkbookName: string;
       begin
        Excel:=CreateOleObject('Excel.Application');
// для остальных
        try
        Excel.SheetsInNewWorkbook:=1;
        Excel.WorkBooks.Add;
        WorkbookName := GetCurrentDir +
'\Журнал операций.xls';
        Ex-
cel.ActiveWorkbook.SaveAs(WorkbookName);
//Открытие книги
        Excel.WorkSheets[1].Select;

        for i:=0 to frmDM.tbPay.FieldList.Count-4
do    //вывод шапки таблицы
        begin
        Ex-
cel.Cells[1,i+1]:=frmDM.tbPay.FieldList.Fields[i].Disp
layName;
        Ex-
cel.WorkSheets[1].Columns.Item[i+1].ColumnWidth:=
frmDM.tbPay.FieldList.Fields[i].DisplayWidth;
        frmDM.tbPay.First;
        j:=2;
        while not frmDM.tbPay.Eof do       //вывод
данных таблицы
         begin
         Ex-
cel.Cells[j,i+1]:=frmDM.tbPay.FieldValues[frmDM.tb
Pay.FieldList.Fields[i].FieldName];
         frmDM.tbPay.Next;
         j:=j+1;
         end;
        end;
        frmDM.tbPay.First;

        Excel.Range['A1:D'+IntToStr(j-1)].Select;
        Excel.Selection.Borders.LineStyle:=1;
```

```
            Excel.ActiveWorkbook.Save;     //сохранить документ  Excel
            Excel.WindowState := -4137;    //развернуть на весь экран
            Excel.Visible := True;         //вывести на экран

            except
              Excel.Quit; //закрыть, если произошла ошибка работы с Excel
            end;
          end;

          procedure TfrmPay.Button2Click(Sender: TObject);
          begin
          Close;
          end;

          procedure TfrmPay.DBGrid1ColExit(Sender: TObject);
          begin
            if DBGrid1.SelectedField.FieldName = 'Дата' then
              DateTimePicker4.Visible := False;
          end;

          procedure TfrmPay.DBGrid1DrawColumnCell(Sender: TObject; const Rect: TRect;
            DataCol: Integer; Column: TColumn; State: TGridDrawState);
          begin
            if (gdFocused in State) then
            begin
              if (Column.Field.FieldName = 'Дата') then
              with DateTimePicker4 do
              begin
                Left := Rect.Left + DBGrid1.Left + 1;
                Top := Rect.Top + DBGrid1.Top + 1;
                Width := Rect.Right - Rect.Left + 2;
                Width := Rect.Right - Rect.Left + 2;
                Height := Rect.Bottom - Rect.Top + 2;
                Visible := True;
                if (DBGrid1.SelectedField.Value=Null) then
                  Date:=SysUtils.Date()
                else
                  Date:=DBGrid1.SelectedField.Value;
              end;
            end;
            end;

          procedure TfrmPay.DateTimePicker4Change(Sender: TObject);
          begin
            if DBGrid1.DataSource.State in [dsEdit, dsInsert] then

DBGrid1.SelectedField.Value:=DateToStr(DateTimePicker4.Date);
          end;

          procedure TfrmPay.DateTimePicker4DropDown(Sender: TObject);
          begin
          DBGrid1.DataSource.Edit;
          end;

          procedure TfrmPay.DateTimePicker4KeyPress(Sender: TObject; var Key: Char);
          begin
            if (key = Chr(9)) then Exit; // код табуляции
            if (DBGrid1.SelectedField.FieldName = 'Дата') then
              begin
                DateTimePicker4.SetFocus;
                SendMessage(DateTimePicker4.Handle, WM_Char, word(Key), 0);
          end;
          end;
```

```
end.


unit UnProfile;

interface

uses
  Windows, Messages, SysUtils, Variants,
Classes, Graphics, Controls, Forms,
  Dialogs, DBCtrls, StdCtrls, Mask, ComCtrls;

type
  TfrmProfile = class(TForm)
    DBImage1: TDBImage;
    Label1: TLabel;
    DBEdit1: TDBEdit;
    Label2: TLabel;
    DBLookupComboBox1:
TDBLookupComboBox;
    Label3: TLabel;
    DBLookupComboBox2:
TDBLookupComboBox;
    Label4: TLabel;
    Label5: TLabel;
    DBEdit3: TDBEdit;
    Label6: TLabel;
    DBEdit4: TDBEdit;
    Label7: TLabel;
    DBEdit5: TDBEdit;
    Label8: TLabel;
    DBEdit6: TDBEdit;
    Label9: TLabel;
    DBEdit7: TDBEdit;
    Label10: TLabel;
    DBEdit8: TDBEdit;
    Label11: TLabel;
    DBEdit9: TDBEdit;
    Label12: TLabel;
    DBEdit10: TDBEdit;
    Label13: TLabel;
    DBEdit11: TDBEdit;
    Button1: TButton;
    Button2: TButton;
    DateTimePicker3: TDateTimePicker;
    procedure Button2Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure DBImage1Click(Sender:
TObject);
    procedure DateTimePicker3Change(Sender:
TObject);
    procedure
DateTimePicker3DropDown(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frmProfile: TfrmProfile;

implementation

uses UnDM, UnClient;

{$R *.dfm}

procedure TfrmProfile.Button2Click(Sender:
TObject);
begin
Close;
end;

procedure TfrmProfile.Button1Click(Sender:
TObject);
begin
frmDM.tbClient.Post;
Close;
end;

procedure
TfrmProfile.DBImage1Click(Sender: TObject);
begin
```

```
        if (frmClient.OpenDialog1.Execute) then
         begin
         frmDM.tbClient.Edit;

frmDM.tbClientDSDesigner12.LoadFromFile(frmClie
nt.OpenDialog1.FileName);
         end
        end;

        procedure
TfrmProfile.DateTimePicker3Change(Sender:
TObject);
        begin
        frmDM.tbClient.FieldValues['ДатаРождения'
] :=DateToStr(DateTimePicker3.Date);
        end;

        procedure
TfrmProfile.DateTimePicker3DropDown(Sender:
TObject);
        begin
        frmDM.tbClient.Edit;
        end;

        end.



        unit UnReport;

        interface

        uses
          Windows, Messages, SysUtils, Variants,
Classes, Graphics, Controls, Forms,
          Dialogs, StdCtrls;

        type
         TfrmReport = class(TForm)
           Button1: TButton;
           Button2: TButton;
           Button3: TButton;
           Button4: TButton;
           procedure Button4Click(Sender: TObject);
           procedure Button1Click(Sender: TObject);
           procedure Button2Click(Sender: TObject);
           procedure Button3Click(Sender: TObject);
         private
           { Private declarations }
         public
           { Public declarations }
         end;

        var
         frmReport: TfrmReport;

        implementation

        uses UnCredit, UnDM;

        {$R *.dfm}

        procedure TfrmReport.Button4Click(Sender:
TObject);
        begin
        frmDM.tbTypeCredit.Locate('ТипКредита','К
арта',[]);
        frmCredit.CheckBox1.Checked:=False;
        frmCredit.CheckBox3.Checked:=False;
        frmCredit.CheckBox2.Checked:=True;
        frmCredit.CheckBox2Click (Sender);
        frmCredit.Show;
        end;

        procedure TfrmReport.Button1Click(Sender:
TObject);
        begin
        frmDM.tbTypeCredit.Locate('ТипКредита','К
редит',[]);
        frmCredit.CheckBox1.Checked:=False;
        frmCredit.CheckBox3.Checked:=False;
        frmCredit.CheckBox2.Checked:=True;
        frmCredit.CheckBox2Click (Sender);
        frmCredit.Show;
        end;
```

```
        procedure TfrmReport.Button2Click(Sender:
TObject);
        begin
        frmDM.tbCreditStatus.Locate('СтатусКредит
a','Просрочен',[]);
        frmCredit.CheckBox1.Checked:=False;
        frmCredit.CheckBox2.Checked:=False;
        frmCredit.CheckBox3.Checked:=True;
        frmCredit.CheckBox3Click (Sender);
        frmCredit.Show;
        end;

        procedure TfrmReport.Button3Click(Sender:
TObject);
        begin
        frmCredit.CheckBox1.Checked:=True;
        frmCredit.CheckBox2.Checked:=False;
        frmCredit.CheckBox3.Checked:=False;
        frmCredit.CheckBox1Click (Sender);
        frmCredit.Show;
        end;

        end.


        unit UnTypeCredit;

        interface

        uses
          Windows, Messages, SysUtils, Variants,
Classes, Graphics, Controls, Forms,
          Dialogs, ExtCtrls, DBCtrls, Grids, DBGrids;

        type
          TfrmTypeCredit = class(TForm)
            DBGrid1: TDBGrid;
            DBNavigator1: TDBNavigator;
            procedure DBGrid1TitleClick(Column:
TColumn);
          private
            { Private declarations }
          public
            { Public declarations }
          end;

        var
          frmTypeCredit: TfrmTypeCredit;

        implementation

        uses UnDM;

        {$R *.dfm}

        procedure
TfrmTypeCredit.DBGrid1TitleClick(Column:
TColumn);
        begin

frmDM.tbTypeCredit.Sort:='['+Column.FieldName+']';
        end;

        end.


        unit UnTypePay;

        interface

        uses
          Windows, Messages, SysUtils, Variants,
Classes, Graphics, Controls, Forms,
          Dialogs, ExtCtrls, DBCtrls, Grids, DBGrids;

        type
          TfrmTypePay = class(TForm)
            DBGrid1: TDBGrid;
            DBNavigator1: TDBNavigator;
            procedure DBGrid1TitleClick(Column:
TColumn);
          private
            { Private declarations }
```

```
    public
      { Public declarations }
    end;

var
    frmTypePay: TfrmTypePay;

implementation

uses UnDM;

{$R *.dfm}

procedure
TfrmTypePay.DBGrid1TitleClick(Column: TColumn);
begin

frmDM.tbTypePay.Sort:='['+Column.FieldName+']';
end;

end.
```