МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
Белорусский национальный технический университет
Автотракторный факультет
Кафедра «Экономика и логистика»

ЭЛЕКТРОННЫЙ УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС ПО УЧЕБНОЙ ДИСЦИПЛИНЕ

«ПРИКЛАДНЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ В ЛОГИСТИКЕ»

для специальности
1-27 02 01 «Транспортная логистика (по направлениям)»,
направления специальности
1-27 02 01-01 «Транспортная логистика (автомобильный транспорт)»

Составитель: Стефанович Н.В.

Минск ◊ БНТУ ◊ 2022

Перечень материалов

Пояснительная записка

Цели данного ЭУМК – повышение эффективности организации учебного процесса с использованием дистанционных технологий; представление возможности студентам заниматься самообразованием, пользуясь комплектом учебно-методических материалов по курсу «Прикладные информационные системы в логистике».

ЭУМК содержит четыре раздела: теоретический, практический, контроля знаний и вспомогательный раздел. В теоретическом разделе представлен лекционный материал в соответствии с основными разделами и темами учебной программы. Практический раздел включает лабораторный практикум. Раздел контроля знаний включает экзаменационные вопросы, а также задания для выполнения расчетно-графической и контрольной работы, предусмотренной программой дисциплины. Вспомогательный раздел содержит выдержки из учебной программы дисциплины, список рекомендуемой литературы.

Материалы учебно-методического комплекса представлены в формате *PDF*. Учебные материалы структурированы по разделам. ЭУМК содержит перекрестные гиперссылки, позволяющие оперативно найти необходимый материал, перейти к нужной теме. Предусматривается навигация по разделам через закладки *PDF* формата, обеспечивающая возможность быстрого поиска требуемой информации и быстрый возврат к предыдущей информации.

Открытие ЭУМК производится посредством запуска файла *PISvL.pdf*.

ТЕОРЕТИЧЕСКИЙ РАЗДЕЛ

Содержание

РАЗДЕЛ I. ПРЕДМЕТНАЯ ОБЛАСТЬ, МЕСТО И РОЛЬ ИНФОРМАЦИОННЫХ СИСТЕМ В ЛОГИСТИКЕ

Тема 1.1 Информационная логистика

Основной тенденцией в совершенствовании современных процессов управления является признание приоритетности его информационной сущности.

Для повышения эффективности в логистике активно применяются информационные технологии. В качестве наиболее полезных из них следует выделить оборудование для обработки, хранения и связи, всевозможное программное обеспечение. С точки зрения эффективности любые применяемые информационные технологии должны обеспечивать точную, доступную, надежную, гибкую, интегрированную и своевременную информацию.

Информационный поток – это поток сообщений в речевой, документной (бумажной и электронной) и других формах, сопутствующий материальному или сервисному потоку в рассматриваемой логистической системе и предназначенный в основном для реализации управляющих воздействий.

Информационные потоки, возникающие при внешних воздействиях на соответствующую среду, переносят информацию (сообщения) от ее источников к ее потребителям. Эти потоки могут иметь значение для оперативного управления и выработки стратегических решений, а могут соответствовать материальным и управлять ими. Различие скоростей материальных и информационных потоков может при наличии соответствия приводить к временному сдвигу между ними.

Для обработки информационных потоков современные логистические системы имеют в своем составе информационный *логистический центр*. Задача такого центра – накопление получаемых данных и их прагматическая фильтрация дли решения логистических задач. При этом связь центра с источниками информации может быть односторонней, двусторонней и многосторонней. Современные логистические системы используют последний способ связи. Таким образом, логистика оперирует многочисленными показателями и характеристиками информационных потоков:

- номенклатурой передаваемых сообщений, типами данных, документами, массивами данных;
- интенсивностью и скоростью передачи данных;
- специальными характеристиками (пропускной способностью информационных каналов, защитой от несанкционированного доступа).

Между информационным и материальным потоком отсутствует изоморфность (однозначное соответствие, синхронность во времени возникновения). Как правило, информационный поток либо опережает материальный, либо отстает от него. В частности, само зарождение материального потока обычно является следствием информационных потоков в ходе, например, переговоров по сделкам купли-продажи товаров, составления контрактов. Типичным является наличие нескольких информационных потоков, сопровождающих материальный поток.

Информационные потоки в логистике образуются в виде потоков массивов электронных данных, определенным образом оформленных бумажных документов, а также в виде потоков, состоящих из обоих этих типов квантов информации.

К такой информации относятся:

- телефонограммы и факсы;
- накладные, поступающие вместе с товаром;
- информация о поступлении и размещении грузов на складах;
- данные о транспортных тарифах и о возможных маршрутах и типах транспорта;
- изменения в динамических моделях состояния запасов;
- библиотеки управляющих программ для технологического оборудования с числовым программным управлением и каталоги этих библиотек;
- различная нормативно-справочная производственная информация;

- изменения в динамических моделях рынка и в его сегментировании;
- текущие сведения о производственных мощностях, поставщиках и продуцентах;
- изменения в динамических моделях портфеля заказов;
- текущие сведения о незавершенном производстве;
- данные о планах выпуска;
- текущие данные о складах;
- данные об объемах и видах готовой продукции, фактическом сбыте продукции потребителям, финансовых потоках.

Информационный поток определяется следующими параметрами:

1. Источником возникновения.
2. Направлением движения, либо адресатом.
3. Скоростью передачи.
4. Общим объемом.

В практической деятельности скорость информационного потока может определяться числом документов или строк во всех документах, передаваемых или обрабатываемых в единицу времени. Соответственно общий объем информационного потока может измеряться общим числом передаваемых или обрабатываемых документов, суммарным числом содержащихся в них строк.

Информационный поток может функционировать в том же направлении, что и соответствующий материальный поток, либо он может быть направлен навстречу «своему» материальному потоку. Направление информационного потока может в ряде случаев не иметь ничего общего с направлением движения соответствующего материального потока. Например, комплектующие изделия поступают от продуцента на входной склад, а соответствующие счета – в бухгалтерию. Если удовлетворяются заказы на поставку сырья, материалов и комплектующих, информационный поток, образованный этими заказами, оформленными в виде документов, направлен в сторону, противоположную соответствующему материальному потоку. Он возникает раньше этого материального потока. Иными словами, этот информационный поток предваряет инициированный им материальный поток.

Фактуры, накладные и необходимая эксплуатационная документация образуют информационный поток, двигающийся в том же направлении, что и соответствующий материальный поток и одновременно с ним.

Информационный поток, двигающийся навстречу материальному, может быть не только предваряющим, но и отстающим. Например, поток информации, образованный документами о результатах приемки или отказе в приемке груза, различными претензиями, гарантийными документами.

Таким образом, информационные потоки могут опережать, отставать или быть синхронными с соответствующими материальными потоками.

Каждый их этих типов информационных потоков может двигаться в том же направлении, что и соответствующий материальный поток, быть встречным ему или же двигаться в не совпадающем с ним направлении.

Каждый тип информационного потока характеризуется своим сочетанием этих двух качеств. Соответственно можно назвать следующие

- опережающие с совпадающим направлением;
- опережающие встречные;
- опережающие, различающиеся по направлению;
- синхронные с совпадающим направлением;
- синхронные встречные;
- синхронные, различающиеся по направлению;
- отстающие с совпадающим направлением;
- отстающие встречные;
- отстающие, различающиеся по направлению.

Таким образом, разнообразные информационные потоки являются теми связями, которые объединяют в единое целое различные функциональные подсистемы. В каждой из этих функциональных подсистем реализуются материальные потоки, соответствующие целям, обеспечиваемым этими подсистемами. Информационные потоки объединяют эти подсистемы в единое целое, так что отдельные цели каждой подсистемы подчиняются общей цели всего производственно-сбытового комплекса. Именно это является основной концепцией логистики.

С обработкой увеличивающегося объема информации, необходимой для планирования и контроля логистических мероприятий, а также с развитием коммуникаций и компьютеризацией деятельности связана *информационная логистика* — одна из вспомогательных подсистем логистики. Она представляет собой управление информационным потоком на предприятии и в его окружении с целью использования информации для регулирования экономических процессов.

Предприятиями могут быть использованы следующие *информационные технологии*:

- управление данными (data managment — DM);
- электронный обмен данными (electronic data interchange — EDI);
- штриховое кодирование (bar coding— BC);
- искусственный интеллект/экспертные системы (artificial inteligence/expert systems — AI/ES);
- дистанционный доступ и коммуникации (remote access and communication — RA&C).

Управление данными представляет собой процесс накопления и систематизации в необходимом объёме данных с целью доступа к ним целевых пользователей в нужное время. Современные информационные технологии ориентированы не на локально организованные данные, а на базы данных, представляющие собой специально организованное хранение информационных ресурсов в виде интегрированной совокупности, предназначенной для многоцелевого использования и модификации различными пользователями.

Рисунок 1.1 – Современные информационные технологии в логистике

Эти совокупности работают под управлением СУБД — системы управления базой данных, основное назначение которой, наряду с управлением данными, обеспечение доступа к ним, организация и связь с пользователем.

Использование информационных технологий значительно сокращает и ускоряет путь перемещения продукции от производителя к потребителю. При этом большое значение в минимизации движения товара имеет быстрая передача информации как внутри предприятия, так и во внешней среде.

Электронный обмен данными. В зависимости от финансовой ситуации фирмам необходимо внедрять сначала внутреннюю систему *обмена данных* при помощи локальных информационных источников с широким использованием средств EDI — для исключения бумажного обращения документации, а затем при финансовом росте интегрировать ее с Internet для широкого доступа к возможным клиентам.

Штриховое кодирование — один из видов автоматической идентификации, при котором используется метод оптического считывания информации, обозначающей товар в виде комбинации параллельных темных штрихов и светлых полос согласно определенной системе. Главной задачей обозначения товаров штрих-кодами является рационализация продажи и распределения товаров, независимо от страны их происхождения, места сбыта и расположения складского хозяйства.

Дистанционный доступ к коммуникации базируется на использовании спутниковой связи и современных коммуникаций, обеспечивающих

аудиосвязь в режиме реального времени и позволяющих предприятиям отдаленные рынки сделать частью одной сети распределения.

Программы искусственного интеллекта в первую очередь используются при принятии заказа и обслуживании покупателей. Основное их преимущество в адаптировании общения с заказчиком по телефону к реальной ситуации при персональной продаже. При этом менеджер с помощью компьютера получает подсказки о ценовых скидках, возможностях доставки, предложениях замены при отсутствии необходимого товара на предприятии, перечне регулярно покупаемых товаров.

Информационные системы в логистике предполагают быструю адекватную реакцию на требование рынка, слежение за временем доставки, оптимизацию функций в целях качественной доставки и своевременного снабжения и другое. Логистические информационные системы подразделяются на группы:

Плановые информационные системы. Эти системы создаются на административном уровне управления и служат для принятия долгосрочных решений стратегического характера. Среди решаемых задач: создание и оптимизация звеньев логистической цепи; управление условно-постоянными данными; планирование производства; общее управление запасами; управление резервами.

Диспозитивные информационные системы. Эти системы служат для обеспечения отлаженной работы логистических систем: детальное управление запасами; распоряжение транспортом; отбор грузов по заказам и их комплектование, учёт отправляемых грузов.

Исполнительные информационные системы. Создаются на уровне административного или оперативного управления. Обработка информации в этих системах производится в темпе, определяемом скоростью её поступления в ЭВМ. Это так называемый режим работы в реальном масштабе времени, который позволяет получать необходимую информацию о движении грузов в текущий момент времени и своевременно выдавать соответствующие административные и управляющие воздействия на объект управления. Этими системами могут решаться разнообразные задачи, связанные с контролем материальных потоков, оперативным управлением обслуживания производства, управлением перемещениями.

Тема 1.2 Информационная инфраструктура логистики

Современный электронный бизнес представляет собой непрерывную оптимизацию товаров и услуг определенной организации, а также производственных связей за счет применения цифровых технологий и использование Internet как первичного средства коммуникации. Таким образом, технологии электронного бизнеса — это системы, которые объединяют посредством совместной работы и координации действий разных организаций, вовлекаемых в общие производственно-торговые

процессы. При этом системы электронного бизнеса (СЭБ) выполняют не только операции купли-продажи, но и сопровождения процессов стимулирования спроса на товары и услуги, автоматизацию административных функций, связанных с продажей и обработкой заказа, а также с усовершенствованием обмена информацией между партнерами.

Модель систем электронного бизнеса представлена на рисунке 1.2. Наиболее перспективным Internet-проектом является B2B-сегмент (business-to-business, или предприятие—предприятие), а именно создание корпоративных систем электронного бизнеса (для работы с поставщиками, посредниками), а также отраслевых торговых площадок, Internet-бирж. Обмену коммерческой информацией между B2B-площадками помогают E2E-порталы (exchange-to-exchange, или обмен—обмен).

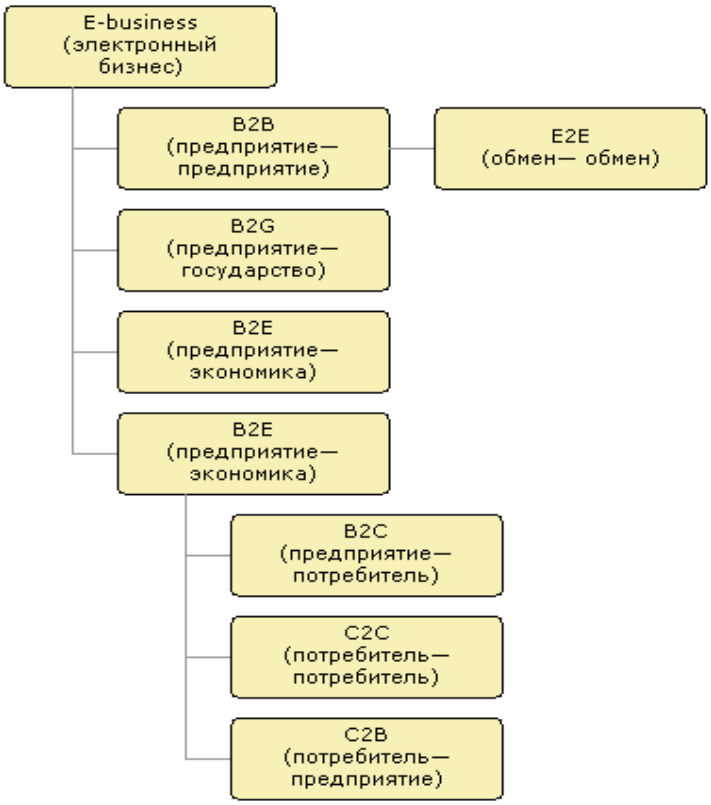


Рисунок 1.2 – Модель систем электронного бизнеса

Другой электронный сегмент B2G (business-to-goverment, или предприятие—государство) предназначен для торговых операций между предприятием и государством (любым правительственным органом).

Выходу предприятия на международные рынки способствует B2E-сегмент (business-to-economy, или предприятие—экономика).

Электронная торговля (E-commerce), как часть электронного бизнеса, представляет собой использование Internet-технологий для поддержки торговых операций между партнерами, одна или две стороны из которых являются конечными потребителями (организацией—потребителем).

B2C-сайты (business-to-consumer, или предприятие—потребитель) представляют собой продажу предприятием товаров непосредственно потребителям.

Электронная торговля типа C2C (consumer-to-consumer, или потребитель—потребитель) представляет собой продажи товара одними потребителями другим. При C2B (consumer-to-business, или потребитель—предприятие) посетители в условиях реального времени могут принимать участие в виртуальном аукционе инвестиционных товаров и задавать требования к товару (размер, цвет, материал, атрибуты).

Таким образом, реализация логистических информационных технологий с одной стороны — это долгосрочная инвестиция для предприятия; с другой — оптимизация его управленческой и финансово-хозяйственной деятельности и в конечном итоге снижение дорожно-транспортных расходов, сокращение производственного цикла и производственного брака, увеличение оборачиваемости средств в расчетах, снижение издержек, связанных с отгрузкой товаров, в большей мере удовлетворение потребности покупателей по принципу «jfu» (just for you — только для Вас) и выполнение поставок по принципу «jit» (just-in-time — точно, своевременно).

Особо следует выделить логистическую информацию, которая составляет важнейший стратегический ресурс логистики в модели «поставщик — потребитель». Использование для ее обработки вычислительной техники позволяет снизить издержки благодаря более эффективному управлению информационными потоками, увеличению их скорости и координации. Понятие «информационный ресурс» рассматривается в качестве экономической категории.

Интенсивное развитие и широкое внедрение современных логистических информационных технологий привело к созданию корпоративных систем управления предприятиями (КСУП).

Сегодня в мире распространение получили три основные концепции КСУП — планирование ресурсов производства (Manufacturing Resource Planning — MRP), планирование ресурсов предприятия (Enterprise Resource Planning — ERP) и планирование ресурсов, синхронизированное с потребителем (Consumer Synchronized Resource Planning — CSRP).

Первая концепция представляет собой методологию детального планирования производства, включая учет заказов, планирование загрузки производственных мощностей и потребности во всех ресурсах производства (материалах, сырье, комплектующих, оборудовании, кадрах), планирование производственных расходов, моделирование производственного процесса, его учет, планирование выпуска готовой продукции, оперативное корректирование плана и производственных заданий. Развитие концепции MRP осуществляется по нескольким направлениям:

- *управление сложными производственными проектами,*
- *интегрированное управление мелкосерийным производством,*
- *управление сложными финансово-сбытовыми и производственными структурами,*

– *холдинговое управление.*

Кроме того, с ее помощью возможно решение и самостоятельных задач (например, управление складским хозяйством и отгрузкой товара).

Концепция ERP в свою очередь представляет собой методологию интегрированного управления всеми сферами деятельности предприятия, включая производственные мощности, материальные и нематериальные потоки. К общим функциям ERP-систем относится:

– *руководство предприятием,*
– *финансовая деятельность,*
– *функции поддержки (информационное и технологическое обеспечение, работа с кадрами, делопроизводство, юридическая деятельность),*
– *взаимодействие с территориальными структурными подразделениями.*

Если рассмотренные выше концепции ориентированы на внутреннюю организацию предприятия, то основой методологии CSRP является интегрирование заказчика или покупателя в КСУП и обеспечение полного жизненного цикла товара — от проектирования до послепродажного обслуживания. Ее сущность в том, что информация о клиентах и их запросах включается в процессы производственно-сбытового планирования предприятия, происходит переход от планирования производства до планирования удовлетворения потребностей клиентов. С указанной концепцией тесно связана технология управления взаимоотношениями с клиентами (Customer Relationship Management — CRM).

Продолжением развития концепции CSRP является стратегия эффективного обслуживания клиентов (Efficient Concumer Responce — ECR) в логистических каналах. Она охватывает управление определенной категорией продуктов, запасами и поставками, а также информацией. При этом:

- управление категорией продуктов заключается в сотрудничестве производителей, посредников и потребителей в процессе принятия решений в отношении ширины, глубины, насыщенности и сопоставимости товарного ассортимента. Участники логистической цепи уже в фазе разработки концепции и составления планов внедрения новых товаров совместно анализируют их влияние на реализацию целей партнеров, оценивают рентабельность новых товаров при учете всех затрат, определяют момент их внедрения на рынок и объем запасов. Совместно осуществляют также разработку стратегических программ продвижения товаров и оценивают их эффективность;
- управление запасами и поставками средств базируется на трансфере информации, постоянной инвентаризации запасов, автоматическом расчете заказов, мониторинге доступности товаров, а также на постоянном их перемещении от поставщика к розничному посреднику

или организации-потребителю при наименьшем количестве перевозок и простоев;

- управление информацией основывается на полном доступе каждого из партнеров ко всей торговой информации, необходимой с точки зрения продвижения данной категории товаров. Автоматическая их идентификация при помощи штрихового кодирования, регистрация ежедневной (или за определенный период) продажи, электронный обмен данными и составляет необходимый инструмент, позволяющий превратить цепи поставок в цепи, отвечающие потребностям клиентов.

Главной особенностью транспортной инфраструктуры является ее высокая технологическая зависимость.

Специфика транспортной отрасли - необходимость постоянного обмена информацией между очень удаленными друг от друга пунктами. Это обуславливает необходимость использования новейшего сетевого оборудования, технологий передачи данных. Обмен данными происходит между центрами обработки данных, использующими различное серверное оборудование, различные операционные системы, различные протоколы обмена данными.

Самые крупные компании отрасли используют серверные решения уровня мэйнфрейм. Современные технологии виртуализации и терминального доступа (VMWare, Citrix) позволяют сосредоточить все вычислительные мощности и системы хранения и резервного копирования данных в одном центральном центре обработки данных, позволяя разворачивать в удаленных офисах и филиалах лишь вспомогательную IT-инфраструктуру.

Необходимость в автоматизации испытывают провайдеры логистических услуг (PL-операторы), число которых растет. Отличие дистрибьютора от PL-оператора, прежде всего, состоит в максимальной гибкости последнего. Логистический провайдер оказывает услуги сразу нескольким компаниям и работает с разными группами товаров, имеющими различные сроки годности, условия хранения и оборачиваемость. При этом спектр услуг должен включать в себя не только хранение и доставку, но и упаковку, переупаковку, штучный подбор, формирование наборов. Помимо оказания услуг PL-оператор должен еще эффективно взаимодействовать со своими клиентами – быстро и максимально прозрачно для них и для себя выставлять стоимость услуг. Чтобы отвечать всем этим требованиям, ему недостаточно мощной WMS-системой, дающей возможность осуществлять функции приемки, отгрузки, кросс-докинга, штучного подбора и учитывать различные требования к хранению товаров, сроки годности и оборачиваемость товаров от разных поставщиков.

Специфика работы с несколькими клиентами состоит в том, что их информационные системы могут предоставлять данные в различных форматах и для их переформатирования необходимо время. Однако одной из ключевых потребностей логистического провайдера является точность и скорость предоставления информации. Информация о заказе и отгрузке,

полученная не вовремя, неизбежно ведет к несвоевременному получению товара клиентом, независимо от того, как эффективно настроена работа склада. Все это приводит PL-оператора к необходимости создавать дополнительную интеграционную платформу для обмена информацией со своими клиентами, что влечет за собой дополнительные инвестиции. Другая, менее затратная альтернатива решения этой проблемы – подключение всех компаний, участвующих в цепочке поставок, к единой системе обмена информацией (EDI).

РАЗДЕЛ II. КОРПОРАТИВНЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ (КИС) В ЛОГИСТИКЕ

Тема 2.1. Базы данных как основной элемент КИС

Корпоративная информационная система (КИС) - это открытая интегрированная автоматизированная система реального времени по автоматизации бизнес-процессов предприятия. Под КИС следует понимать в первую очередь систему, и затем только программное обеспечение. Но часто этот термин используется IT-специалистами в качестве объединяющего названия программных систем семейства CASE, ERP, CRM, MRP.

КИС должны обладать следующими функциями:

1. сбора и регистрации информационных ресурсов;
2. хранения, обработки и актуализации информационных ресурсов;
3. предоставления информационных ресурсов пользователям;
4. планирования, анализа, учета и контроля.

Для определения КИС используется термин "автоматизированные системы управления" (АСУ), который впервые появился в России в 1960-е гг. XX века в связи с применением компьютеров и информационных технологий в управлении экономическими объектами и процессами.

В настоящее время существует достаточно большое количество разновидностей КИС (например, SAP R/3, BAAN, Галактика, Парус, 1С: Предприятие).

Концепция построения КИС предусматривает наличие типовых компонентов:

1. Ядро системы, которое обеспечивает комплексную автоматизацию совокупности бизнес-приложений и содержит полный набор функциональных модулей для автоматизации задач управления;
2. Система автоматизации документооборота;
3. Вспомогательные инструментальные системы обработки информации (экспертные системы, системы подготовки и принятия решений) на базе хранилищ данных КИС;
4. Программно-технические средства системы безопасности КИС;
5. Сервисные коммуникационные приложения (электронная почта, программное обеспечение удаленного доступа);

6. Компоненты интернет/интранет для доступа к разнородным базам данных и информационным ресурсам, сервисным услугам;

7. Системы специального назначения - системы автоматизированного проектирования (САПР), автоматизированные системы управления технологическими процессами (АСУТП), банковские системы

7. Офисные программы - текстовый редактор, электронные таблицы, системы управления базами данных.

Ключевым компонентом КИС является СУБД.

Понятие и классификация СУБД. Иерархические, сетевые, реляционные базы данных

Впервые термин «реляционная модель данных» ввел сотрудник фирмы IBM доктор Кодд (математик по образованию), который предложил использовать для обработки данных аппарат теории множеств (объединение, пересечение, разность, декартово произведение) и показал, что любое представление данных сводится к совокупности двумерных таблиц особого вида, известного в математике как отношение (relation, англ.).

Реляционной является БД, в которой все данные организованы в виде набора двумерных таблиц и все операции над данными сводятся к операциям над этими таблицами.

Система управления базами данных (СУБД) с централизованной архитектурой

При использовании этой технологии база данных, СУБД и прикладная программа (приложение) располагаются на одном компьютере. Для такого способа организации не требуется поддержки сети и все сводится к автономной работе**.**

СУБД (архитектура «файл-сервер»)

База данных в виде набора файлов находится на жестком диске специально выделенного компьютера (файлового сервера). При этом существует локальная сеть, состоящая из клиентских компьютеров, на каждом из которых установлены СУБД и приложение для работы с БД.

СУБД инициирует обращения к данным, находящимся на файловом сервере, в результате которых часть файлов БД копируется на клиентский компьютер и обрабатывается. При необходимости (в случае изменения данных) данные отправляются назад на файловый сервер с целью обновления БД. Однако недостаточно развитый аппарат транзакций служит потенциальным источником ошибок в плане нарушения смысловой и ссылочной целостности информации при одновременном внесении изменений в одну и ту же запись.

СУБД с архитектурой «клиент-сервер»

Использование технологии «клиент – сервер» предполагает наличие некоторого количества компьютеров, объединенных в сеть, один из которых выполняет особые управляющие функции (является сервером сети) и разделяет функции приложения пользователя (называемого клиентом). При этом по сети "путешествуют" только те данные, которые необходимы

клиенту, за счет чего снижается нагрузка на сеть и существенно уменьшается сетевой трафик.

Трехзвенная (*клиент - сервер приложений - сервер базы данных*) представляет собой дальнейшее совершенствование технологии «клиент – сервер», в которой приложения разделены на три логических и физических уровня: уровень представления (пользовательский интерфейс), уровень приложения, на котором осуществляется обработка данных, и уровень данных, предназначенный для хранения и управления данными, относящимися к приложению.

По функциональному назначению СУБД делят на:

- системы оперативной обработки транзакций (OLTP-системы, Online Transaction Processing);
- системы делового анализа / хранилища данных (OLAP-системы, Online Analysis Processing).

Если первые это основа функционирования предприятия: принятие заказов клиентов, учет сырья, складской учет, учет оплаты продукции, или ведение в электронном виде небольших по объему транзакций (операций с денежными средствами: перевод, вывод или зачисление на счет). Транзакция - это логически завершенная банковская операция, заключающаяся в переводе определенной суммы денег со счета одного лица на счет другого.

СУБД делового анализа используются для принятия решений на основе сбора и анализа большого объема информации.

Распределенная система управления базой данных (распределенная СУБД) состоит из единой логической базы данных, разделенной на некоторое количество фрагментов. Каждый фрагмент базы данных сохраняется на одном или нескольких компьютерах, работающих под управлением отдельных СУБД и соединенных между собой сетью связи.

Приведем список современных СУБД:

1. Oracle DataBase
2. MySQL
3. IBM DB2
4. Microsoft SQL Server
5. Microsoft Access
6. PostgreSQL
7. MongoDB

Тема 2.2. Проектирование реляционных баз данных

Организация типичной СУБД и состав ее компонентов соответствует набору функций. Логически в современной СУБД можно выделить внутреннюю часть — ядро СУБД (Data Base Engine), компилятор языка баз данных (обычно SQL - Structured Query Language), подсистему поддержки времени выполнения, набор утилит.

Ядро СУБД отвечает за журнализацию и управление:

- данными во внешней памяти;
- буферами оперативной памяти;
- транзакциями.

Все функции взаимосвязаны, поэтому компоненты должны взаимодействовать по протоколам или определенным правилам.

Ядро СУБД является основной резидентной частью СУБД и обладает собственным интерфейсом, не доступным пользователю напрямую и используемым в программах, производимых компилятором SQL, и в утилитах БД, реализующих такие процедуры, которые накладно выполнять с использованием языка БД, например, загрузка БД, сбор статистики, глобальная проверка целостности.

Почти все продукты баз данных, созданные с 1970 года, основаны на подходе, который называют реляционным.

Основными понятиями реляционных баз данных являются тип данных, домен, атрибут, кортеж, первичный ключ и отношение.

В таблице 2.1 приведены термины, используемые на практике, и показан общий смысл этих понятий.

Таблица 2.1 Терминология БД

Теория БД	Практика	SQL Server
Отношение (Relation)	Таблица (Table)	Таблица (Table)
Кортеж (Tuple)	Запись (Record)	Строка (Row)
Атрибут (Attribute)	Поле (Field)	Столбец (Column)
Домен (Domain)	Общая совокупность допустимых значений	Количество столбцов
Степень отношения	Кардинальное число отношения	Количество строк

Понятие тип данных в реляционной модели данных полностью адекватно понятию типа данных в языках программирования. Обычно в современных реляционных БД допускается хранение символьных, числовых данных, битовых строк, специализированных числовых данных (таких как "деньги"), а также специальных "темпоральных" данных (дата, время, временной интервал).

При работе с таблицами в реляционных базах данных, желательно (необходимо), чтобы каждая таблица имела так называемый первичный ключ.

Первичный ключ – это поле, которое используется для обеспечения уникальности данных в таблице. Уникальность необходима во избежание неоднозначности, когда неизвестно к какой записи таблицы можно обратиться, если в таблице есть повторяющиеся записи (две записи имеют одинаковые значения во всех полях таблицы).

Базисные средства манипулирования реляционными данными Проектирование реляционных баз данных с использованием нормализации. Внутренняя организация реляционных СУБД (стр. 12-23).

Среди важных характеристик любой базы данных следует назвать производительность, надежность и простоту администрирования. Знание того, как большинство СУБД физически хранят данные во внешней памяти, представление о параметрах этого хранения и соответствующих методах доступа может помочь при проектировании баз данных, обладающих заданной производительностью.

Любая СУБД основывается на конкретном комплексном решении задач, связанных с организацией хранения и управления данными.

Организация баз данных.

Тема 2.3. Разработка приложений для работы с базой данных

Пользователи и система по-разному воспринимают работу базы данных и ее предназначение. Пользователь рассматривает ее в качестве таблицы, в которой содержится необходимая ему информация для выполнения тех или иных задач. Как системный компонент база данных представляет собой комплекс файлов, внутри которых расположены всевозможные связи и таблицы, к которым должен получить доступ пользователь.

На сегодня большинство компаний использует централизованную технологию обеспечения доступа к базам данных, которая предусматривает прямое взаимодействие сервера и клиента. В качестве последних выступают компьютеры пользователей, которых обслуживают базы данных, в то время как сервер представляет собой высокомощный компьютер, с помощью которого пользователи могут одновременно получить доступ к одной и той же информации, даже если одинаковые запросы будут отправлены с более чем тысячи компьютеров.

Пользователь или приложение создает запрос SQL-серверу, который выполняет его обработку и возврат вместе с запрошенными данными. Реализация этого запроса осуществляется на специальном языке, который одинаково понятен обеим сторонам.

Сервера могут отличаться друг от друга по следующим критериям:

- технология предоставления доступа к данным;
- каким образом осуществляется хранение информации;
- каким образом пользователь получает сведения с сервера при отправке соответствующего запроса.

Каждая из задач предусматривает использование определенного программного компонента:

- прикладные программы, с помощью которых пользователи могут получить доступ и визуализацию базы данных;
- интерфейс, через который предоставляется вся необходимая информация;
- специальное программное обеспечение для взаимодействия с файловой системой, с помощью которого обеспечивается надежное хранение данных на сервере.

Система управления базой данных представляет собой своеобразную программную прослойку, которая создается между взаимодействующими сторонами (клиентом и сервером), чтобы пользователи могли абстрагироваться от системного видения базы данных, но при этом не потеряли возможности нормально с ней взаимодействовать.

Грамотно построенная система должна обеспечивать:

- возможность сохранения всей необходимых данных на носители и ее извлечения по необходимости;
- возможность взаимодействия с данными, расположенными в серверной оперативной памяти;
- запись истории внесения корректировок в базу данных;
- поддержание запросов пользователей.

В зависимости от того, как обеспечена реализация указанных компонентов, системы управления базами данных разделяются на несколько типов. Помимо этого, они различаются по тому, каким образом внутри них обеспечивается связь данных.

База данных (БД) представляет собой совокупность взаимосвязанных, хранящихся вместе данных при наличии такой минимальной избыточности, которая допускает их эффективное использование для одного или нескольких приложений (задач). Данные и их описания (словарь данных) хранятся так, что они в значительной степени становятся независимыми от использующих их программ. Для добавления новых или модификации существующих данных, а также для поиска данных в базе данных применяется общий управляемый способ.

Современные системы идентификации и аутентификации пользователей позволяют с высокой степенью вероятности определить подлинность пользователя, либо удаленного узла. Данные системы предназначены для однозначного определения субъекта доступа и его полномочий по отношению к конкретному ресурсу.

Идентификация — это процедура распознавания субъекта по его идентификатору. В процессе регистрации субъект предъявляет системе свой идентификатор, и она проверяет его наличие в своей базе данных. Субъекты с идентификаторами известными системе считаются легальными (законными), остальные субъекты относятся к нелегальным.

Аутентификация пользователей — процедура проверки подлинности субъекта, позволяющая достоверно убедиться в том, что субъект, предъявивший свой идентификатор, на самом деле является именно тем субъектом, идентификатор которого он использует. Для этого он должен подтвердить факт обладания некоторой информацией, которая может быть доступна только ему одному (пароль, ключ).

Авторизация — процедура предоставления субъекту определенных прав доступа к ресурсам системы после прохождения им процедуры аутентификации. Для каждого субъекта в системе определяется набор прав, которые он может использовать при обращении к ее ресурсам.

Существующие системы аутентификации пользователей:

1) парольные системы (самый простой и распространенный способ);

2) системы PKI (криптографические сертификаты);

3) системы одноразовых паролей;

4) биометрические системы.

1) В основе механизмов аутентификации пользователей «лежат» ПАРОЛИ, поэтому данный способ наиболее распространенный. В силу своей «открытости», а также ужесточении требований к длине пароля, большим количеством программного обеспечения для взлома паролей, данная система является наиболее уязвимой.

Основными «проблемными» местами систем паролей являются:

- «сложность» для запоминания пароля конечным пользователем, как следствие — нарушение Политики безопасности организации;
- хранение паролей в «открытом» виде; «беззащитность» пароля при вводе;
- применение «нестойких» алгоритмов аутентификации и открытых каналов передачи данных;
- легкость взлома с помощь специального программного обеспечения — «взломщиков паролей», «клавиатурных шпионов», «кейлогеров»;
- канал передачи пароля — открытый, нешифрованный канал передачи данных.

2) Технология инфраструктуры открытых ключей позволяет проверять и удостоверять подлинность пользователя. Инфраструктура открытых ключей или PKI обеспечивает единую идентификацию, аутентификацию и авторизацию пользователей системы, приложений и процессов и вместе с этим гарантирует доступность, целостность и конфиденциальность информации. Инфраструктура PKI представляет собой систему цифровых сертификатов, носителями которых являются USB-ключи или смарт-карты.

При использовании индивидуального секретного пароля и средств криптографической защиты, цифровые сертификаты получают роль электронных паспортов. Использование в корпоративной сети технологии инфраструктуры открытых ключей значительно повышает безопасность всей сети в целом, так как позволяет отказаться от использования парольной аутентификации пользователей внутри, и обеспечивает безопасный доступ удаленных пользователей в систему.

Основные носители информации: USB-ключи; Смарт-карты.

Использование в корпоративной сети технологии PKI значительно повышает безопасность всей сети в целом, так как позволяет отказаться от использования парольной аутентификации пользователей внутри, а также обеспечивает безопасный доступ удаленных пользователей в систему. Пользователям не надо запоминать сложные пароли и периодически их менять — достаточно подключить электронный ключ или смарт-карту и ввести PIN-код.

3) Системы многофакторной аутентификации, основанные на технологии одноразовых паролей OTP является платформенно-независимым решением для аутентификации мобильных пользователей, которое

отличается крайней простотой в использовании, установке и администрировании.

Данная технология основана на том, что пароль пользователя не постоянен и изменяется с течением времени специальным устройством (аппаратным или программным) — токеном. Данное решение широко используется в системах удаленного доступа, в том числе системах клиент-банк, для аутентификации пользователей при доступе из недоверенной среды (Интернет-кафе, бизнес-центр).

OTP-токен — мобильное персональное устройство, принадлежащее определенному пользователю, генерирующее одноразовые пароли, используемые для аутентификации данного пользователя. OTP-токены имеют небольшой размер и выпускаются в виде: карманного калькулятора; брелока; смарт-карты; устройства, комбинированного с USB-ключом; специального программного обеспечения для карманных компьютеров.

3) Биометрические системы — это измеримые физиологические или поведенческие данные живого человека.

Некоторые биометрические данные уникальны для данного человека, и их можно использовать для установления личности или проверки декларируемых личных данных:

- для идентификации пользователя (вместо ввода имени пользователя);
- для однофакторной аутентификации пользователя;
- совместно с паролем или аутентификационным токеном (таким, как смарт-карта) для обеспечения двухфакторной аутентификации.

Биометрические данные делятся на группы:

1. Физиологические биометрические характеристики — основанные на данных, полученных путем измерения анатомических характеристик человека, таких, как, например, отпечаток пальца, форма лица или кисти, сетчатка глаза.

2. Поведенческие биометрические характеристики (динамические) — основанные на данных, полученных путем измерения действий человека. Характерной чертой для поведенческих характеристик является их протяженность во времени — измеряемое действие имеет начало, середину и конец. Например, голос.

Криптографическая защита данных

Криптография (иногда употребляют термин криптология) — область знаний, изучающая тайнопись (криптография) и методы ее раскрытия (криптоанализ). Криптография считается разделом математики.

Цель криптографической системы заключается в том, чтобы зашифровать осмысленный исходный текст (также называемый открытым текстом), получив в результате совершенно бессмысленный на взгляд шифрованный текст (шифртекст, криптограмма). Получатель, которому он предназначен, должен быть способен расшифровать («дешифровать») этот шифртекст, восстановив, таким образом, соответствующий ему открытый

текст. При этом противник (называемый также криптоаналитиком) должен быть неспособен раскрыть исходный текст.

Существует важное отличие между расшифрованием (дешифрованием) и раскрытием шифртекста. Пример простого алгоритма шифрования: Широко известным историческим примером криптосистемы является так называемый шифр Цезаря, который представляет из себя простую замену каждой буквы открытого текста третьей следующей за ней буквой алфавита (с циклическим переносом, когда это необходимо). Например, «A» заменялась на «D», «B» на «E», «Z» на «C».

Все методы шифрования можно разделить на две группы:

- шифры с секретным ключом (симметричная схема);
- шифры с открытым ключом (асимметричная схема).

Первый тип шифров подразумевает наличие некой информации (ключа), обладание которой позволяет как зашифровать, так и расшифровать сообщение. Шифры с открытым ключом подразумевают наличие двух ключей — открытого и закрытого; один используется для шифровки, другой для расшифровки сообщений.

Электронная подпись (ЭП)

ЭП — последовательность символов, полученная в результате криптографического преобразования исходной информации с использованием закрытого ключа ЭЦП, которая позволяет подтверждать целостность и неизменность этой информации, а также ее авторство при условии использования открытого ключа ЭП и его сертификата.

Цифровая подпись обеспечивает:

– Удостоверение источника документа. В зависимости от деталей определения «документа» могут быть подписаны такие поля как автор, внесённые изменения, метка времени.

– Защиту от изменений документа. При любом случайном или преднамеренном изменении документа (или подписи) изменится хэш, следовательно, подпись станет недействительной.

– Невозможность отказа от авторства. Так как создать корректную подпись можно лишь зная закрытый ключ, а он известен только владельцу, то владелец не может отказаться от своей подписи под документом.

Более мощным средством защиты данных от просмотра является их шифрование. Шифрование – это преобразование читаемого текста в нечитаемый текст, при помощи некоторого алгоритма; применяется для защиты уязвимых данных.

Процесс дешифрования восстанавливает данные в исходное состояние.

В целях контроля использования основных ресурсов СУБД во многих системах имеются средства установления прав доступа к объектам БД. Права доступа определяют возможные действия над объектами. Владелец объекта (пользователь, создавший объект), а также администратор БД имеют все права. Остальные пользователи к разным объектам могут иметь различные уровни доступа. Разрешение на доступ к конкретным объектам базы данных сохраняется в файле рабочей группы.

Файл рабочей группы содержит данные о пользователях группы и считывается во время запуска. Файл содержит следующую информацию: имена учетных записей пользователей, пароли пользователей, имена групп, в которые входят пользователи.

По отношению к таблицам могут предусматриваться следующие права доступа:

- просмотр (чтение) данных;
- изменение (редактирование) данных;
- добавление новых записей;
- добавление и удаление данных;
- изменение структуры таблицы.

К данным, имеющимся в таблице, могут применяться меры защиты по отношению к отдельным полям и отдельным записям.

Защита данных в полях таблиц предусматривает следующие уровни прав доступа:

- полный запрет доступа;
- только чтение;
- разрешение всех операций (просмотр, ввод новых значений, удаление и изменение).

По отношению к формам могут предусматриваться две основные операции:

- вызов для работы и проектирование (режим Конструктора). Запрет вызова Конструктора целесообразно выполнять для экранных форм готовых приложений, чтобы конечный пользователь случайно не изменил приложение;
- защита отдельных элементов. Например, некоторые поля исходной таблицы вообще могут отсутствовать или скрыты от пользователя, а некоторые поля - доступны для просмотра.

Отчеты во многом похожи на экранные формы. На отчеты аналогично может накладываться запрет на вызов средств их разработки.

К дополнительным средствам защиты БД можно отнести такие, которые нельзя прямо отнести к средствам защиты, но которые непосредственно влияют на безопасность данных. Их составляют следующие средства:

- встроенные средства контроля значений данных в соответствии с типами;
- повышения достоверности вводимых данных;
- обеспечения целостности связей таблиц;
- организации совместного использования объектов БД в сети.

Восстановление базы данных с помощью резервного копирования базы данных, с помощью журнала транзакций.

Поскольку данные, хранимые компьютерными средствами подвержены потерям и повреждениям, вызываемым разными событиями, важно обеспечить средства восстановления данных. Приведение базы данных точно в то состояние, которое существовало перед отказом не всегда возможно, но

процедуры восстановления базы данных могут привести ее в состояние, существовавшее незадолго до отказа.

Восстановление базы данных применяется при повреждениях, не позволяющих пользователю открыть базу данных или работать с ней. Одной из причин повреждения базы данных может быть воздействие компьютерных вирусов или наличие дефектов (физических или логических) на диске.

Повреждение базы данных может проявляться при попытке пользователя открыть, сжать, зашифровать или дешифровать БД.

Следующим способом обслуживания базы данных является резервное копирование. Основным назначением резервного копирования базы данных является предотвращение потери информации и реализуется путем одноразового или периодического копирования и архивирования наиболее ценной информации. Резервное копирование заключается в создании резервной копии базы данных и размещении на вспомогательных носителях информации.

Резервная копия может быть точной копией исходной БД или сжатой (архивной) копией.

Резервное копирование может осуществляться во время работы с БД (режим online) или в другое время. Копия может создаваться по инициативе оператора, либо автоматически в заданное время путем запуска соответствующей утилиты.

При организации резервного копирования администратор решает такие вопросы как:

- какие устройства выбрать для резервного копирования;
- когда и с какой частотой выполнять резервное копирование.

Важно периодически проверять корректность выполненного резервирования информации путем пробного восстановления.

Репликация (replication) – создание специальных копий (реплик) базы данных, с которыми пользователи могут работать одновременно на разных рабочих станциях.

Журнал транзакций БД — это особая часть БД, недоступная пользователям СУБД, в которую поступают записи обо всех изменениях основной части БД. Для эффективной реализации функции ведения журнала изменений БД необходимо обеспечить повышенную надежность хранения и поддержания в рабочем состоянии самого журнала. Иногда для этого в системе хранят несколько копий журнала. В разных СУБД изменения базы данных фиксируются в журнале на разных уровнях. Иногда запись в журнале соответствует какой-то операции изменения БД (например, операции удаления строки из таблицы реляционной БД), а иногда — минимальной внутренней операции модификации страницы внешней памяти. В некоторых схемах используются оба подхода одновременно.

Небольшую и несложную базу данных или приложение Access можно создать в СУБД Access без использования языков программирования SQL и Visual Basic. В СУБД Access имеется достаточно средств (различных мастеров и конструкторов) для визуального проектирования объектов базы

данных. Для решения некоторых задач автоматизации приложений Access можно использовать макросы вместо языка программирования Visual Basic (например, при создании кнопочной формы). Но создание коммерческих баз данных в СУБД Access невозможно без применения визуального языка программирования Visual Basic и языка запросов SQL.

Для автоматизации действий над объектами в Microsoft Access и в других приложениях Microsoft Office применяются макросы и модули.

Макросы - это небольшие программы на языке макрокоманд (языке сценариев). Модули - это объекты, содержащие программы на языке Visual Basic.

Основное назначение макросов и модулей — это создание удобного интерфейса приложения.

Стандарты DAO и ADO обеспечивают программиста похожим набором инструментов управления. DAO – это более ранний стандарт средств управления базой данных. Стандарт ADO представляет более мощные инструменты и его применение приводит к сокращению программного кода.

Тема 2.4. Введение в язык SQL

Для извлечения данных из базы данных используется язык SQL.

SQL (Structured Query Language) — это язык, предназначенный для программ управления базами данных. В 1986 г. ANSI и ISO официально приняли стандартное определение языка «Язык баз данных SQL». Новые версии стандарта были опубликованы в 1989, 1992, 1996, 1999, 2003, 2006, 2008, 2011 и в 2016 году. Последняя действующая редакция стандартов языка SQL – ISO/IEC 9075:2016.

В Ms Access SQL используется в каждом запросе.

Структурированный язык запросов SQL реализуется в следующих формах:

- интерактивный SQL;
- статический SQL;
- динамический SQL;
- встроенный SQL.

Выделяют следующие виды SQL запросов:

1) DDL (Data Definition Language) – язык определения данных. Задачей DDL запросов является создание БД и описание ее структуры. Запросами такого вида устанавливаются правила того, в каком виде различные данные будут размещаться в БД.

2) DML (Data Manipulation Language) – язык манипулирования данными. В число запросов этого типа входят различные команды, используя которые непосредственно производятся некоторые манипуляции с данными. DML-запросы необходимы для добавления изменений в уже внесенные данные, для получения данных из БД, для их сохранения, для обновления

различных записей и для их удаления из БД. В число элементов DML-обращений входит основная часть SQL операторов.

3) DCL (Data Control Language) – язык управления данными. Включает в себя запросы и команды, касающиеся разрешений, прав и других настроек СУБД.

4) TCL (Transaction Control Language) – язык управления транзакциями. Конструкции такого типа применяют чтобы управлять изменениями, которые производятся с использованием DML запросов. Конструкции TCL позволяют производить объединение DML запросов в наборы транзакций.

Например, простая инструкция SQL, извлекающая список фамилий контактов с именем Mary, может выглядеть следующим образом:

```
SELECT Last_Name
FROM Contacts
WHERE First_Name ="Mary";
```

Общий формат инструкции SQL:

```
SELECT <что выводится>
FROM <откуда>
WHERE <каким условиям должно отвечать>
GROUP BY <какие поля группируются>
HAVING <условие для сгруппированных данных>
ORDER BY <в каком порядке выводить данные>.
```

Операторы (команды), написанные на языке SQL, лишь указывают СУБД, какой результат должен быть получен, но не описывают процедуру получения этого результата. СУБД сама определяет способ выполнения команды пользователя.

Операторы языка SQL строятся с применением:

- зарезервированных ключевых слов;
- идентификаторов (имен) таблиц и столбцов таблиц;
- логических, арифметических и строковых выражений, используемых для формирования критериев поиска информации в БД и для вычисления значений ячеек результирующих таблиц;
- идентификаторов (имен) операций и функций, используемых в выражениях.

SQL — это следующие функциональные возможности:

- определение данных — дает возможность разработчику определять структуру хранения данных и отношения между элементами данных;
- выборка данных — дает возможность разработчику использовать любые данные из БД для своих целей;
- обработка информации — дает возможность разработчику делать с данными что угодно: добавлять, изменять, удалять и не только;

- управление доступом — дает возможность разработчику обезопасить информацию в БД от непреднамеренного использования;
- совместное применение данных — дает возможность нескольким разработчикам одновременно работать с информацией в одной БД, при этом организовывает работу таким образом, чтобы действия одного разработчика не навредили действиям другого разработчика;
- целостность данных — дает возможность разработчику обезопасить данные от случайного разрушения при отказе системы или случайных изменений.

Основные преимущества языка структурированных запросов SQL приведены на рисунке 2.1.

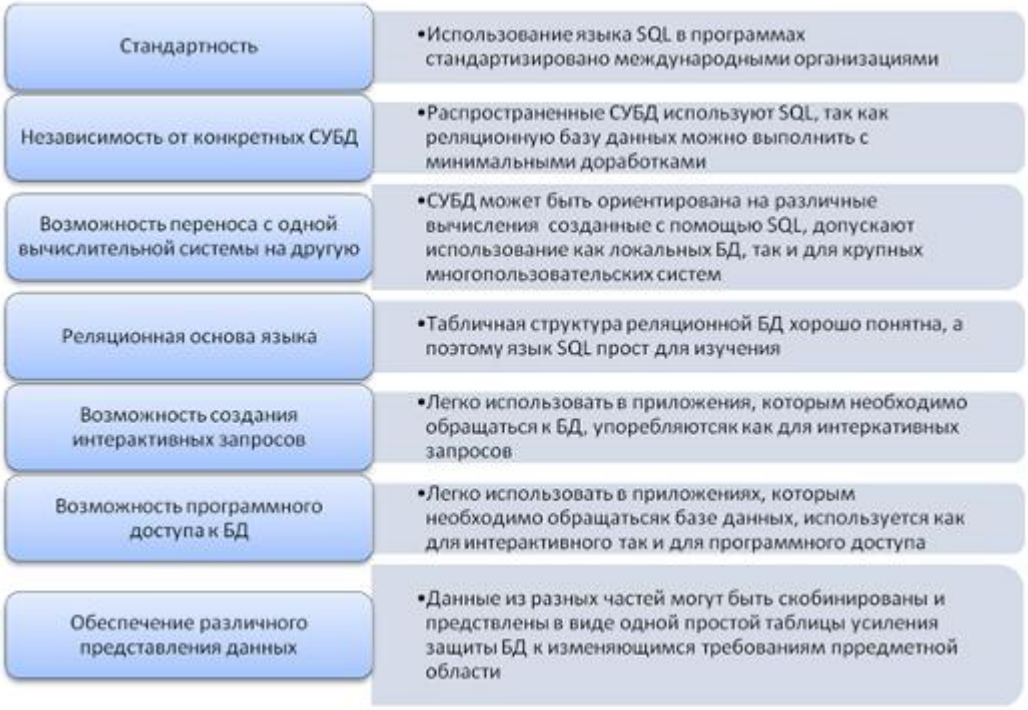


Рисунок 2.1 - Преимущества структурированного языка запросов

Язык универсален и обладает чётко определённой структурой за счёт устоявшихся стандартов. Взаимодействие с базами данных происходит быстро даже в ситуациях, когда объёмы данных велики (Big Data). Кроме того, эффективное управление возможно даже без особых познаний кода.

РАЗДЕЛ III. АВТОМАТИЗАЦИЯ ПРОЕКТИРОВАНИЯ ПРИКЛАДНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ В ЛОГИСТИКЕ

Тема 3.1. Проектный подход к разработке информационных систем

Система автоматизированного проектирования

Системы автоматизированного проектирования с поддержкой функций внедрения и связывания объектов OLE (Object Linking and Embedding)

используют современный метод передачи информации между приложениями. Двойной щелчок мыши по внедренному объекту автоматически загружает программу, в которой он был создан, и позволяет сразу отредактировать.

Технология обмена данными OLE.

Интерфейс программного продукта включает в себя:

- способы взаимодействия с внутренней частью программы (операционной системой, платформой, сервером);
- дизайн;
- доступные функции.

Допустим пользователь открывает программу Microsoft Word и видит лист, разметку, фон и другие элементы. Это внешнее оформление. Возможность ввести текст, изменить шрифт, откорректировать содержимое – это функционал. За кнопками скрывается внутренняя часть программного обеспечения, работа которой не видна пользователям.

Создание интуитивно понятного дизайна для пользователей – это одна из основных задач при разработке информационной системы.

Тема 3.2. Разработка прикладных программных систем

Процесс разработки прикладного программного обеспечения. Основные этапы создания прикладного программного обеспечения. Алгоритмизация. Технологии программирования: структурное, модульное, объектно-ориентированное, императивное, функциональное, параллельное программирование.

Тема 3.3. Проектирование интерфейса информационных систем в логистике

Интерфейс программного продукта и его проектирование.

Элементы графического интерфейса. Основные элементы управления.

Приведем расширенный список программ для автоматизации транспортно-логистической деятельности:

1. КиберЛог. Облачный сервис для управления транспортными перевозками, упрощающий взаимодействие между участниками бизнес-процессов. Позволяет отслеживать все этапы выполнения заявки – от заключения договора и формирования оферты до доставки груза.

При этом пользователю доступна автоматическая архивация данных и передача данных по протоколу SSL, имеется возможность добавить печать и подпись, предусмотрена интеграция с Интернет-банками.

2. ЯКурьер. CRM-система для оптимизации транспортного отдела.

Встроенный алгоритм построит оптимальный маршрут в соответствии весом, объемом и интервалом доставки. Водитель получит данный маршрут, контактные данные и комментарии к заказу.

Расширяет функционал данного программного обеспечения агрегатор доставок/биржа грузоперевозок, мониторинг транспорта, возможности осуществления таможенных процедур и оптимизации маршрутной доставки.

3. Мегалогист. Предназначен для комплексной автоматизации транспортной логистики и разработан на платформе 1С:Предприятие 8.

Перечислим основные преимущества:

- создание заданий и планирование маршрутов, мониторинг рейсов, анализ KPI (коэффициента эффективности выполнения перевозок) и рентабельности доставки;
- единовременная оплата лицензии;
- планирование маршрутов в ручном и автоматическом режиме;
- интеграция с онлайн-кассами;
- диспетчеризация и мониторинг транспорта.

4. 1С TMS Логистика. Программа для планирования и учета деятельности транспортных компаний, анализирующая многие процессы, включая документооборот и организацию мультимодальной перевозки.

«1С TMS Логистика» реализована на платформе «1С: Предприятие 8». Разработчиками предусмотрена интеграция в 1С ERP; возможности подключения дополнительных модулей; автоматическое планирование маршрутов и выгрузка заданий в АРМ экспедитора; формирование отчётов.

5. Forecast now! Система складского учета для прогнозирования спроса и управления товарными запасами.

Инновационный программный продукт отслеживает и анализирует историю продаж для каждого склада или торговой точки, оптимизирует и формирует заказы на основе прогноза спроса, работает с равномерным и редким вариативным спросом и предоставляет аналитическую отчетность для принятия стратегических решений.

6. Инструменты Логиста 24. Автоматизированный сервис для управления перевозками, основанный на искусственном интеллекте.

Пользователю необходимо загрузить в сервис готовый список заказов, созданный в приложении Ms Excel и запустить автопланирование. После завершения процесса обработки сервис возвратит набор маршрутных листов для каждой транспортной единицы.

7. Maxoptra. Максоптра - онлайн-система управления логистикой, позволяющая автоматически распределить задачи между исполнителями и спланировать наиболее быстрые маршруты без лишних затрат. Поддерживает интеграцию с системами ГЛОНАСС и GPS для точного отслеживания доставки и отсутствия незапланированных задержек.

Оптимальное распределение заказов между исполнителями в автоматическом режиме, ручная корректировка маршрутов и добавление новых заявок в расписание.

8. Умная логистика. CRM система для автоматизации работы компаний перевозчиков. Помимо браузерного решения доступны варианты для Windows, MacOS и iOS.

9. 4logist. Многофункциональный сервис для транспортных и логистических компаний, CRM для экспедиторов и логистов.

Приведенный список программ может служить ориентиром при выборе качественной системы автоматизации транспортной системы для обеспечения бесперебойного движения товаров по логистической цепочке.

Объединяя человеческий интеллект с искусственным, транспортно-логистические компании ускоряют решение любых оптимизационных задач, рационализируют планирование логистических процессов в области управления цепями поставок технологией искусственного интеллекта.

Оптимальный выбор перевозчика или расчет минимального расстояния перевозки может занять от 10 минут и более, когда это выполняется вручную. В случае передачи этого процесса искусственному интеллекту он может быть выполнен за считанные секунды.

Для доказательства возьмем решение задачи коммивояжера о нахождении оптимального маршрута передвижения между двумя заданными точками из 69 возможных.

Используем стандартное приложение, входящее в состав Ms Office: Ms Excel и встроенный в него язык программирования VBA. Изначально формируем два выпадающих списка предусматривающих выбор пунктов отправления и назначения.

Программируем кнопку «Рассчитать», используя Visual Basic, которая позволит запустить алгоритм для расчета в действие. Результаты выполненных расчетов вернутся в диапазон ячеек (столбец B, C и D) и определенную ячейку D2. (рисунок 3.1).

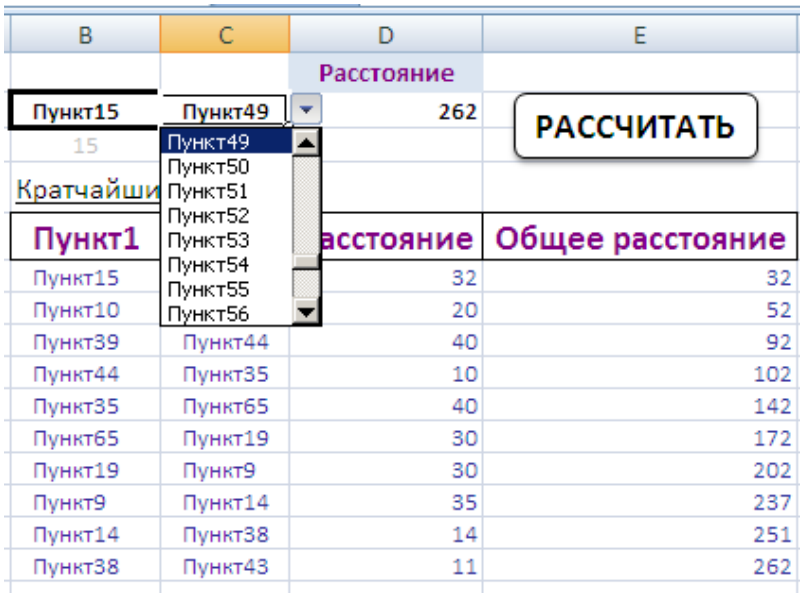


Рисунок 3.1 – Графическая интерпретация решения задачи с выбранным пунктом 15 для отправления груза и пунктом 49 для доставки

В данном случае алгоритм решения включает в себя перебор всех возможных вариантов для поиска кратчайшего пути в графе (shortest-paths, SP) - известная задача комбинаторной оптимизации, имеющая множество реальных приложений.

Тема 3.4. Основы технологии разработки программных средств

В современных условиях исследование и прогнозирование поведения транспортно-логистических систем на практике осуществляется посредством экономико-математического моделирования или описания логистических процессов в виде моделей.

Под моделью обычно понимается абстрактное или материальное отображение этой системы, которое может быть использовано вместо нее для изучения ее свойств и возможных вариантов поведения.

Большинство современных концепций и стратегий в области управления сетями поставок (Supply Chain Management, Efficient Consumer Response, Cross-Docking, Continuous Replenishment, Automatic Replenishment, Quick Response и Vendor Managed Inventory) имитируют течение управляемого процесса с последующим анализом результатов моделирования для выбора окончательного решения.

Производственные и управленческие процессы в нынешних условиях как никогда предполагают согласованность действий, а координация деятельности по всей цепочке поставок перерастает в новое качество – временную интеграцию.

Синхронизируются и интегрируются во времени не только процессы, рождаемые организациями и выходящие за ее пределы, но и генераторы, а также исполнители таких процессов, то есть сами организации. Это новое качество логистического взаимодействия участников логистической цепи поставок, которое сведено к согласованию ресурсов, применяемых для перемещения груза, с целевыми пространственно-временными интервалами.

В большинстве программируемых задач получается, что целью является перемещение груза между заданными точками пространства в заданном интервале времени, то есть время выступает либо как одна из целей задачи, если рассматривать время отдельно от пространства, либо как параметр цели, если задача рассматривается как пространственно-временная.

Оптимизация, которая будет выполнена с помощью заранее написанной программы, направлена на более рациональное использование или экономию имеющихся ресурсов. Самым простым будет создание гомоморфной модели, которая представляет собой, подобные отображаемому объекту отношения, характерные и важные для процесса моделирования.

Системы объектно-ориентированного программирования (ООП) дают возможность визуализировать процесс создания графического интерфейса разрабатываемого приложения. Сегодня ООП — самая распространенная методология программирования.

ООП базируется на трех основных принципах. Это инкапсуляция, полиморфизм и наследование.

Инкапсуляция - это понятие в ООП обозначающее защиту данных (сокрытие данных) от внешнего пользователя.

Для лучшего понимания, чтобы совершить звонок по сотовому телефону, вам необязательно знать, как работают сотовые сети, где расположены вышки связи, как у них организованно хранение данных. Все что вам нужно знать, чтобы совершить звонок по сотовому телефону - это что у вас должен быть номер того абонента, кому вы хотите позвонить.

Свойство инкапсуляции в ООП обозначает то, что необходимо дать пользователю программы лишь только доступ к интерфейсу.

Полиморфизм (polymorphism) — свойство, позволяющее использовать один и тот же интерфейс для различных действий.

Наследование — это процесс, посредством которого один объект может приобретать свойства другого. Точнее, объект может наследовать основные свойства другого объекта и добавлять к ним черты, характерные только для него.

В основе проектирования информационной системы лежит моделирование предметной области, при этом под моделью понимается некоторая система, имитирующая структуру и (или) ее функционирование.

Фундаментальным понятием ООП является Объект, представляющий собой некую сущность реального мира или концептуальную сущность с четко определенными границами и значением для системы. Объект может быть чем-то конкретным, например, студент Иванов И.И. или кассовый аппарат № 4, или концептуальным, например, банковская операция, заказ, зачетная сессия или ставка рефинансирования.

Каждый объект имеет три характеристики: состояние, поведение и индивидуальность.

Состояние – одно из условий, в котором объект может находиться, меняется во времени и определяется атрибутами и отношениями между объектами. Например, имеем систему регистрации успеваемости. Объект Студенты группы № 1 в системе регистрации успеваемости может находиться в одном из двух состояний: сдавали экзамен или не сдавали экзамен. Если сдавали, могут сдавать другой экзамен или могут быть переведены на следующий семестр, а если не сдавали, необходимо организовать экзамен.

Поведение определяет, как объект реагирует на запросы других объектов и что может делать сам объект. Поведение реализуется с помощью наборов операций для объекта. В системе регистрации успеваемости объект Студенты группы № 1 может иметь операции перевести на следующий курс или направить на экзамен.

Индивидуальность означает, что каждый объект уникален, даже если его состояние аналогично состоянию другого объекта. Например, студент Иванов И.И. и студент Петров П.П. – два объекта в системе регистрации

успеваемости. Хотя они оба являются студентами группы № 1, каждый из них уникален.

Класс – фундаментальное понятие ООП, представляющее собой описание группы объектов с общими свойствами (атрибутами), поведением (операциями), определяющим взаимодействие объектов с внешней средой, а также отношениями с другими объектами и семантикой. Класс – шаблон для создания объекта. Каждый объект является экземпляром конкретного класса и не может быть экземпляром нескольких классов.

В соответствии с традиционным подходом данные располагаются в базе данных, а поведением занимается собственно приложение. Объектно-ориентированный подход предполагает объединение некоторого количества данных и поведений, обрабатывающих эти данные.

Визуальное моделирование (visual modeling) – это процесс графического представления модели с помощью некоторого стандартного набора графических элементов.

Пользовательский интерфейс – совокупность программных и аппаратных средств, обеспечивающих взаимодействие пользователя с компьютером.

Модели пользовательского интерфейса:

- модель программиста;
- модель пользователя;
- программная модель.

Программист, разрабатывая пользовательский интерфейс, исходит из того, управление какими операциями ему необходимо реализовать, как это осуществить, не затрачивая ни существенных ресурсов компьютера, ни своих сил и времени. Его интересуют функциональность, эффективность, технологичность, внутренняя стройность и другие, не связанные с удобством пользователя характеристики программного обеспечения. Поэтому большинство интерфейсов существующих программ вызывают серьезные нарекания пользователей.

Пользовательская модель интерфейса – это совокупность обобщенных представлений конкретного пользователя или некоторой группы пользователей о процессах, происходящих во время работы программы или программной системы. Эта модель базируется на особенностях опыта конкретных пользователей, который характеризуется:

- уровнем подготовки в предметной области разрабатываемого программного обеспечения;
- интуитивными моделями выполнения операций в этой предметной области;
- уровнем подготовки в области владения компьютером;
- устоявшимися стереотипами работы с компьютером.

Для построения пользовательской модели необходимо изучить перечисленные выше особенности опыта предполагаемых пользователей программного обеспечения. С этой целью используют опросы, тесты и даже

фиксируют последовательность действий, осуществляемых в процессе выполнения некоторых операций.

Приведение в соответствие моделей пользователя и программиста, а также построение на их базе программной модели интерфейса задача непростая. Основа взаимодействия – диалоги.

Диалог – регламентированный обмен информацией между человеком и компьютером, осуществляемый в реальном масштабе времени и направленный на совместное решение конкретной задачи: обмен информацией и координация действий. Каждый диалог состоит из отдельных процессов ввода-вывода, которые физически обеспечивают связь пользователя и компьютера.

Обмен информацией осуществляется передачей сообщений и управляющих сигналов.

Виды сообщений:

- входные сообщения, которые генерируются человеком с помощью средств ввода информации;
- выходные сообщения, которые генерируются компьютером в виде текстов, звуковых сигналов и/или изображений и выводятся пользователю на экран монитора или другие устройства вывода информации.

Тип диалога определяет, кто из «собеседников» управляет процессом обмена информацией.

Различают два типа диалога:

- управляемые программой;
- управляемые пользователем.

Диалог, управляемый программой, предусматривает наличие жесткого, линейного или древовидного, т. е. включающего возможные альтернативные варианты, сценария диалога, заложенного в программное обеспечение. Такой диалог обычно сопровождают большим количеством подсказок, которые уточняют, какую информацию необходимо вводить на каждом шаге.

Диалог, управляемый пользователем, подразумевает, что сценарий диалога зависит от пользователя, который применяет систему для выполнения необходимых ему операций. При этом система обеспечивает возможность реализации различных пользовательских сценариев.

Интерфейсы строятся по технологии WIMP: W – Windows (окна), I – Icons (пиктограммы), M – Mouse (мышь), P - Pop-up (всплывающие или выпадающие меню). Основные элементы графических интерфейсов: окна, пиктограммы, комноненты ввода-вывода и мышь, которую используют в качестве указующего устройства и устройства прямого манипулирования объектами на экране.

Окна. Окно – прямоугольная, ограниченная рамкой область физического экрана. Окно может менять размеры и местоположение в пределах экрана.

5 категорий окон:

- основные окна (окна приложений);
- дочерние или подчиненные окна;
- окна диалога;

- информационные окна;
- окна меню.

Окно приложения Windows содержит: рамку, ограничивающую рабочую область окна, строку заголовка с кнопкой системного меню и кнопками выбора представления окна и выхода, строку меню, пиктографическое меню (панель инструментов), горизонтальные и вертикальные полосы прокрутки и строку состояния.

Дочернее окно Windows используют в многодокументных программных интерфейсах (MDI). Это окно не содержит меню. В строке заголовка – специальное имя, идентифицирующее связанный с ним документ или файл. Пиктограммы всех дочерних окон одинаковы.

Диалоговое окно Windows используют для просмотра и задания различных режимов работы, необходимых параметров или другой информации. Оно может содержать:

- строку заголовка с кнопкой системного меню;
- компоненты, обеспечивающие пользователю возможность ввода или выбора ответа;
- вспомогательные компоненты, обеспечивающие подсказку (поле просмотра или кнопка справки).

Ввод-вывод сообщения в VBA осуществляется в специальных диалоговых окнах. Данные диалоговые окна функционируют при помощи встроенных функций InputBox и MsgBox.

Функция InputBox имеет следующий шаблон написания:

ИмяПеременной = InputBox (Prompt, [Title], [Default], [XPos], [YPos], [HelpFile], [Context]),

где «Prompt (Сообщение)» — обязательный аргумент, так как именно он передает информацию в диалоговом окне, что конкретно необходимо от пользователя. Все остальные аргументы заполняются по желанию.

Фактически это может выглядеть так:

y = InputBox («Введите дату», «Форма заказа»)

То есть пользователю откроется окно с Title «Форма заказа», в том окне будет одно поле для ввода какого-то значения с названием «Введите дату» и две кнопки: «Ok», «Cancel».

Чтобы организовать вывод сообщения в VBA и диалогового окна, используют специальный оператор MsgBox. Шаблон оператора MsgBox:

MsgBox Prompt, [Buttons], [Title], [HelpFile], [Context]

Здесь аргумент Prompt (Сообщение) является обязательным, так как именно он задает выводимую информацию. Остальные аргументы применяются по желанию.

Пользовательская форма UserForm предоставляет пользователю возможность создавать диалоговые окна создаваемых приложений. Она

служит базой для пользовательского диалогового окна, на котором в зависимости от решаемой задачи размещаются требуемые элементы управления.

Текстовые окна (TextBox) являются основными элементами управления для ввода и вывода информации на форму. Используются для реализации диалога с пользователем путем ввода с клавиатуры исходной информации.

Тема 3.5. Конструирование прикладных информационных систем в логистике

Программа – это законченная последовательность команд (инструкций) языка программирования, описывающая алгоритм решения задачи. Программы на языке VBA создаются в виде процедур.

Инструкция – представляет собой операцию (отдельное действие), описание или определение.

Процедура – это именованная последовательность совместно выполняемых инструкций, заключенных между ключевыми словами Sub и End Sub.

Модуль – это набор описаний и процедур на языке VBA, собранных в одну программную единицу. Модули располагаются в проектах.

Проект – это набор всех программных модулей, связанных с документом пакета MS Office.

Структура программы — искусственно выделенные взаимодействующие части программы. Использование рациональной структуры устраняет проблему сложности разработки; делает программу понятной; повышает надежность работы программы при сокращении срока ее тестирования и сроков разработки.

Часто некоторую последовательность инструкций требуется повторить в нескольких местах программы. Чтобы не приходилось тратить время и усилия на копирование этих инструкций, в большинстве языков программирования предусматриваются средства для организации подпрограмм. Подпрограмма — некоторая последовательность инструкций, которая может вызываться в нескольких местах программы.

Разложение на подпрограммы необходимо как для документирования, так и для верификации программы.

Ключевое слово – это слово, которое является частью языка программирования VBA. К ключевым словам относятся: имена инструкций, типов данных, методов, свойств, операторов, встроенных констант, объектов и стандартных функций.

Данными (data) называются объекты, обрабатываемые программой.

Для хранения временных значений (данных) используются переменные. Переменной называется имя, определяющее область памяти для хранения величины, которая может изменяться во время работы программы.

Область видимости переменных – это область программы, где имя переменной считается доступным (видимым), а значит, возможен доступ к ее значению.

Ключевое слово Dim используется для объявления переменной на уровне процедуры. Переменная, объявленная на уровне процедуры, называется локальной. Такая переменная доступна только в той процедуре, где она объявлена. После выхода из процедуры, память, выделенная под переменную, высвобождается (т.е. теряется значение переменной).

Область видимости переменных определяет время жизни переменных. Время жизни переменной – время, в течение которого переменная может иметь значение. Локальные переменные имеют значение только во время выполнения процедуры, в которой они объявлены.

Тип данных (data type) – это характеристика определенного вида данных, которые VBA сохраняет и которыми может манипулировать.

Тип данных определяет:

- формат (размер) хранения данных;
- диапазон допустимых значений данных;
- операции, которые могут выполняться над данными.

Константа (const) – это значение в программе VBA, которое не меняется.

Объявление константы во многом напоминает объявление переменной. Однако в этот момент задается значение, которое уже нельзя изменить.

Формат объявления именованных констант:

Const <имяКонстанты> = <значение>

Для констант, как и для переменных, существует понятие область действия.

Выражение это формальное правило для вычисления некоторого значения. Выражение строится как совокупность операндов, объединенных знаками операций, выполнение которых приводит к вычислению значения.

В программах на VBA можно использовать стандартный набор операций над данными: имеются три основных типа операций:

- математические (или арифметические) – выполняются над числами, и их результатом являются числа;

- отношения – применяются не только к числам, и их результатом являются логические значения, например, x> y;

- логические – используются в логических выражениях и их результатом являются логические значения, например, Not x And y.

Порядок выполнения операций определяется расстановкой круглых скобок и приоритетом (старшинством) операций.

VBA - операторный язык. Это значит, что его программы (процедуры или функции) представляют последовательности операторов. В языке VBA можно выделить следующие группы операторов:

1. декларативные операторы, предназначенные для описания объектов, с которыми работает программа (типов переменных, констант и массивов);

2. операторы-комментарии;

3. операторы присваивания и изменения значений объектов;

4. операторы, управляющие ходом вычислений (условный, циклический, перехода).

Линейная программа должна состоять из следующих операторов: ввода данных, присваивания, вывода (печать) результатов расчета. Линейный вычислительный процесс сводится к последовательным вычислениям арифметических выражений, причем последовательность вычислений полностью соответствует порядку записи математических зависимостей в постановке задачи.

Если какие-либо действия должны происходить только при выполнении некоторого условия, то говорят, что такой процесс имеет разветвление. Выбор варианта действий обеспечивается условным оператором, который таким образом осуществляет управление в программе. Алгоритм, порядок выполнения действий в котором зависит от итогов проверки условия, называется алгоритмом разветвляющейся структуры.

При проверке любое условие может принимать только одно из двух значений: либо True (истина), когда условие выполняется, либо False (ложь), когда оно не выполняется. Проверка выполнимости условий в программах используется довольно часто и производится многими операторами, в которых само условие является их частью.

Операторы передачи управления применяются в программе для реализации безусловных алгоритмических конструкций. Они выполняют переход с одного участка программы на любой другой без какого-либо условия.

Переключатели в Visual Basic реализуются оператором Select Case, который позволяет сделать выбор из нескольких альтернативных вариантов в зависимости от значения условного выражения.

Программы циклической структуры - это такие программы, в которых какая-то группа операторов многократно повторяется. Эта группа операторов, оформленная специальным образом, называется циклом.

Количество повторений определяется либо параметрами цикла, либо условием, заданным вне цикла. Многократное повторение выполняется за счет передачи управления на начало этой группы операторов. Циклические алгоритмы применяются при решении задач на табулирование функций (составление таблицы значений функции), на вычисление суммы и произведений, при обработке массивов.

Существуют два основных типа циклов – циклы со счетчиком (с известным числом повторений) и циклы с условием. Циклы со счетчиком используют в тех случаях, когда необходимо выполнить некоторые действия определенное число раз. Циклы с условием применяются тогда, когда некоторые действия в программе должны повторяться до тех пор, пока выполняется определенное условие или до тех пор, пока не будет выполнено определенное условие.

Символьная информация в алгоритмах и программах описывается данными двух типов: символьным и литерным. Они отличаются друг от

друга тем, что значением символьной переменной является один символ, а литерной — строка символов.

Строка - это последовательность символов кодовой таблицы персональной ЭВМ. При использовании в выражениях строка-константа заключается в апострофы. Длина строки равняется количеству символов в этой строке и может изменяться от 0 до 255.

Для хранения больших объемов данных предусмотрены особые структуры в языках программирования – массивы.

Массив – это упорядоченное множество однотипных элементов.

Массив – это структурированный набор данных, который состоит из однородной, фиксированной по размеру и конфигурации совокупности элементов простой или составной структуры, упорядоченных по номерам.

Массивы бывают следующих видов:

1. Статические и константные массивы всегда имеют фиксированный размер.
2. Динамические массивы делают работу с данными более гибкой, так как не требуют предварительного определения хранимых объёмов данных, а позволяют регулировать размер массива в соответствии с реальными потребностями.
3. Гетерогенные массивы удобны как универсальная структура для хранения наборов данных произвольных типов.

Тема 3.6. Приложения с графическим пользовательским интерфейсом

Графический интерфейс пользователя (graphical user interface, GUI) — система средств для взаимодействия пользователя с электронными устройствами, основанная на представлении всех доступных пользователю системных объектов и функций в виде графических компонентов экрана (окон, значков, меню, кнопок, списков).

Выделяют два типа интерфейсов: SDI (Single Document Interface - однодокументный интерфейс) и MDI (Multi Document Interface - многодокументный интерфейс).

В VBA класс объектов UserForm (пользовательских форм) является основой для создания GUI разрабатываемого приложения. Форма представляет собой окно, в котором размещаются управляющие элементы. Необходимо отметить, что графический интерфейс проекта может включать в себя несколько форм. Такое многооконное приложение всегда имеет главную форму, которая появляется на экране в момент его запуска, а остальные загружаются по мере надобности.

Чтобы создать такое приложение, необходимо сначала создать проект и добавить в него необходимое количество форм.

Второй шаг при создании многооконного приложения – выбор загрузочного модуля, или формы, которая будет отображена первой (по умолчанию таковой является форма, созданная в проекте первой). Для этого

выбирается команда «Project Properties» меню «Project». В появившемся диалоговом окне на вкладке «General» в выпадающем списке «Startup Object» выбирается имя необходимой формы.

При этом необходимо помнить, что при старте программы отобразится только загрузочная форма. Для появления остальных окон необходимо использовать следующий оператор UserFormN. Show, где N – порядковый номер требующейся для открытия формы. При этом текущую форму можно закрыть. Можно для этой цели воспользоваться двумя способами:

- спрятать форму (метод Hide), например, UserForm(N -1).Hide

При этом форма останется в памяти. Потом при помощи метода Show можно будет опять ее вызвать в том же состоянии, в каком она была, или изменять ее и расположенные на ней элементы управления.

Если форма больше точно не потребуется, можно ее удалить из памяти при помощи команды Unload UserForm(N -1).

Элементы управления — это специализированные объекты, которые можно размещать на формах VBA (и непосредственно в документах), используемые для организации взаимодействия с пользователем. В VBA используют как стандартные элементы управления (CommandButton, CheckBox, OptionButton), так и нестандартные (например, Calendar). Элементы управления реагируют на события, которые генерирует пользователь (нажатие на кнопку, ввод значения, перемещение ползунка).

Добавление элементов управления на форму при помощи Toolbox. Для этого необходимо выбрать элемент управления в Toolbox и перетащить его на форму или (что более удобно) выделить элемент управления в Toolbox и затем на форме выделить ту область экрана, которую будет занимать этот элемент управления.

В форме UserForm содержатся вертикальные и горизонтальные направляющие, которые помогают выровнять добавленные в диалоговое окно элементы управления. При добавлении или перемещении элемент управления привязывается к направляющим, что облегчает упорядочение таких элементов в окне.

Каждый элемент управления разрабатывается для того, чтобы реагировать на определенные события (возникают в результате действий пользователя или генерируются программой Ms Excel) и характеризуется набором параметров, которые определяют внешний вид и поведение элемента управления.

Свойства служат для описания и задания характеристик объектов. Например, размера и цвета шрифта, положения формы на экране или состояние объекта (доступность, видимость), задания значений. Чтобы задать или изменить характеристику объекта, надо изменить значение его свойства. Синтаксис задания значения свойства следующий:

Объект.Свойство = Значение Свойства,

где: Объект - обозначает имя объекта; Свойство - имя свойства, которому присваивается значение, имя объекта отделяется от имени свойства точкой. Пример задания свойств объекту:

Cells(1, 1).Value = 2023 - поместить в ячейку A1 значение 2022.

Событие представляет собой действие, например, щелчок мышью, нажатие клавиши, для которого можно запрограммировать отклик, или реакцию объекта на произошедшее событие.

В языке программирования VBA для каждого объекта определен набор стандартных событий. Например, стандартное событие для объекта CommandButton (кнопка) - Click (щелчок мышью). Если пользователь нажимает на кнопку, то это событие. На это событие должен быть отклик, или выполнение какой-либо процедуры (программы). Такая процедура называется процедурой обработки события и имеет стандартное имя. Если такой отклик не создан, не написана соответствующая процедура, то система не будет реагировать на это событие. Для каждого события можно написать процедуру, которая будет срабатывать именно тогда, когда это событие произойдет. Особое значение понятие события имеет при написании процедур реакций пользовательской формы на изменения ее элементов.

Любая программа должна иметь простые и удобные средства для взаимодействия с ней. В графическом интерфейсе Windows для передачи команд приложению используются меню и панели управления. Именно меню позволяют избежать использования множества кнопок, располагаемых на листы и формы. Для создания меню и работы с ним VBA использует стандартные объекты MS Office.

В объектной модели MS Office каждая панель инструментов, меню, строка меню и даже подменю - это объекты типа CommandBar. Все объекты *CommandBar* объединены в семейство *CommandBars* (рисунок 3.2).

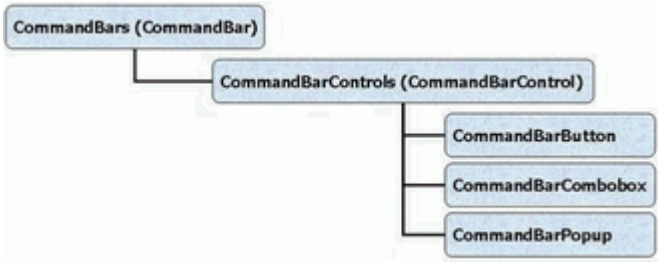


Рисунок 3.2 – Семейство *CommandBars*

Элементы управления, находящиеся на панели инструментов или в составе меню, представлены объектами типа *CommanBarControl*, которые объединены в семейство *CommandBarControls*, где *CommandBarButton* - кнопка или элемент меню, вызывающий выполнение команды; *CommandBarComboBox* - список, поле ввода или поле со списком; *CommandBarPopUp* - всплывающее или вложенное меню.

Меню - это набор пунктов, каждый из которых соответствует той или иной команде или действию, а панели инструментов - это обычные кнопки с пиктограммами, объединенные в единое целое и сгруппированные по функциональному назначению.

Создать новые панели инструментов и дополнительные пункты главного меню можно в любом приложении MS Office. Однако эти настройки действуют постоянно для всего приложения.

Программирование на VBA позволяет разработать для каждой из рабочих книг Ms Excel свой интерфейс. Например, исключить из меню и панелей Ms Excel кнопки и команды, не используемые в конкретной рабочей книге и добавить новые необходимые средства. Причем изменения в стандартный интерфейс Ms Excel будут внесены только на время работы с конкретной книгой, и по ее закрытии все вернется назад.

Существует достаточно простая возможность создания программы (макроса) на языке VBA с использованием MacroRecorder.

MacroRecorder – это транслятор, который переводит все действия пользователя с момента запуска MacroRecorder до окончания записи макроса на язык VBA.

ПРАКТИЧЕСКИЙ РАЗДЕЛ

Лабораторные работы №1-№9

Лабораторная работа № 10
РАЗРАБОТКА ПРЕЗЕНТАЦИИ БАЗЫ ДАННЫХ СРЕДСТВАМИ MICROSOFT POWER POINT

Цель работы:

1) изучить функциональные возможности программы подготовки презентаций MS Power Point;

2) приобрести практические навыки разработки презентации информационного продукта (базы данных) средствами MS Power Point.

Порядок выполнения работы:

1. Средствами MS PowerPoint подготовить презентацию базы данных (БД), разработанной в предыдущих лабораторных работах.

Для автоматизации работы при подготовке презентации можно воспользоваться Мастером автосодержания или выбрать один из стандартных шаблонов презентаций MS PowerPoint.

Презентация должна включать 10–15 слайдов со следующей информацией:

1) титульный слайд (1 слайд) – название БД, автор-разработчик;

2) краткое описание информационного продукта (2 слайда) – назначение, область применения, отличительные особенности (достоинства) разработанной БД;

3) подробное описание БД (10 слайдов) – структура таблиц (поля и типы данных), схема БД, запросы, формы, отчеты, макросы;

4) выводы (1 слайд) – достоинства применения разработанной БД;

5) слайд с контактными данными разработчика (1 слайд).

Настроить анимацию текста и объектов на слайдах. Эффекты назначаются по времени после предыдущего объекта (автоматически, не по щелчку мыши).

Настроить параметры показа презентации в режиме непрерывного цикла (до нажатия Esc). Сохранить презентацию как демонстрацию MS PowerPoint (тип файла Демонстрация).

Содержание отчета:

Скриншот полученной презентации в режиме сортировщика слайдов.

Контрольные вопросы:

1. Понятие «Презентация». Способы создания презентации в MS PowerPoint.
2. Основные форматы файлов MS PowerPoint и их особенности.
3. Назначение и особенности режимов просмотра MS PowerPoint. Способы переключения между ними.
4. Понятие «слайд». Виды слайдов. Способы создания (добавления), удаления, перемещения слайда.

5. Краткая характеристика атрибутов слайда: макет (разметка) слайда, шаблон оформления, цветовая схема слайда, фон слайда. Способы их изменения.

6. Как добавить мультимедийный объект на один слайд презентации? На каждый слайд презентации?

7. Как выполняется настройка анимации объектов слайда, порядок анимации? Особенности автоматической анимации и анимации по щелчку.

8. Как добавить на слайд управляющие кнопки и настроить переходы? Как настроить презентацию для управления только кнопками?

9. Как скрыть слайд при показе презентации? Как в процессе показа презентации выполнить переход к скрытому слайду?

Лабораторная работа № 11
СОЗДАНИЕ ПРИЛОЖЕНИЯ

Цель работы. Ознакомиться с технологией создания приложения в системе проектирования Visual Basic.

Теоретическая часть

Процесс создания Windows-приложения состоит из пяти этапов:

1. Постановка задачи – составление по возможности точного и понятного словесного описания того, как должно работать будущее приложение. Это описание должно объяснить, как будет выглядеть форма (окно) этого приложения, определить порядок и формат ввода и вывода информации.
2. Разработка интерфейса – создание экранной формы со всеми находящимися на этой форме объектами и свойствами этих объектов.
3. Программирование – определение того, какие события будут происходить в процессе работы приложения, составление алгоритмов процедур для этих событий и написание программы (программных кодов) этих процедур.
4. Отладка программы – устранение логических ошибок и выполнение тестовых задач.
5. Сохранение проекта и, если необходимо, компиляция – превращение проекта в исполняемое приложение, способное работать самостоятельно за пределами среды проектирования.

Порядок выполнения работы:

В качестве примера создадим приложение для расчета объема кузова автомобиля.

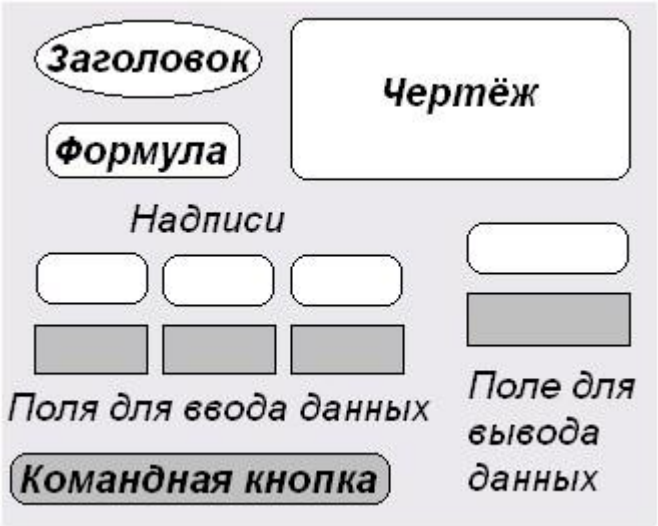
Исходными данными будут три размера кузова: ширина, длина, высота. После ввода исходных данных и нажатия кнопки "РАСЧЕТ" результат должен появиться в текстовом поле "Объем кузова".

Разработка интерфейса состоит из таких шагов:

- создание эскиза экранной формы;
- вход в среду проектирования Windows-приложений Visual Basic;
- создание экранной формы и установка значений свойств этой формы;
- создание на форме объектов управления и установка значений свойств этих объектов.

Как следует из постановки задачи, необходимо создать окно Windows-приложения, на котором будет 4 текстовых поля: 3 из них для ввода исходных данных и одно для вывода результата. Кроме текстовых полей на экранной форме должны быть пояснения в виде заголовка, надписей, расчетной формулы и небольшого чертежа. Сигналом для расчета по

формуле будет нажатие командной кнопки. Эскиз будущей формы показан на рисунке:



Порядок выполнения работы:

1. Запускается система проектирования Visual Basic. На мониторе появляется Главная панель проекта, на ней нужно открыть окна для создания экранной формы: TollBox (Окно инструментов), Form (Окно экранной формы), Properties (Окно свойств этой формы).

Эти и некоторые другие окна могут появиться на Главной панели и без нашего участия, поэтому возможна корректировка окон.

2. Создание экранной формы.

Прежде всего, устанавливаются значения ее размеров – свойства Width (Ширина) и Height (Высота). Эти свойства можно установить с помощью окна Properties, но можно это сделать, "растягивая" мышкой стороны экранной формы. При этом значения в таблице окна Properties будут изменяться автоматически. Положение формы на экране определяется свойствами Left (Левый край) и Top (Верхний край). Определим новое имя формы, новую надпись в строке заголовка формы и цвет фона формы. Этими свойствами являются соответственно: Name, Caption, BackColor. Значение свойства BackColor выбирается с помощью раскрывающейся панели с двумя закладками.

Далее на экранной форме создаются объекты управления и устанавливаются их свойства.

Заполнение экранной формы начинаем с установки объектов Label (Метка). Все надписи, заголовок и формула размещаются в нескольких таких объектах. В окне Панель элементов выбирается инструмент Label.

Для переноса на форму объекта Label щелкаем пиктограмму Label (Метка) в окне ToolBox (Панель элементов). Помещаем указатель мыши в то место экранной формы, где будет находиться левый верхний угол будущего объекта. При нажатой левой клавише мыши "протащим" указатель в то место, где будет находиться правый нижний угол будущего объекта. Это

процесс будет сопровождаться растяжением пунктирной рамки – границы создаваемого объекта.

Размеры и положение появившегося прямоугольника можно подкорректировать. Делается это так. Прямоугольник окружен восемью маленькими квадратиками – маркерами "Ухватив" какой-либо из этих маркеров мышью, можно изменить размеры объекта. Установив курсор внутри прямоугольника, с помощью мыши можно переместить объект в пределах экранной формы, не меняя его размеров. Создаем еще пять таких объектов.

Далее переносим на форму объекты Поле, CommandButton (Кнопка) и PictureBox (Рисунок).

Теперь наполним созданные объекты конкретным содержанием. Это содержание определяется установкой значений свойств перенесенных объектов.

Главной характеристикой объекта CommandButton (Командная кнопка) является не какое-нибудь свойство, а событие. Оно заключается в щелчке мышью по этой кнопке. Свойству (Caption) присваивается значение в виде слова "РАСЧЕТ".

В заключение устанавливается значение всего одного свойства объекта PictureBox (Рисунок) – свойство Picture. Этим значением должен быть графический файл с рисунком, находящийся на компьютере.

3. Составление алгоритма и написание программы – это второй и главный этап проектирования приложения в среде Visual Basic. В составляемом приложении есть только одно событие: щелчок мышью по командной кнопке. Именно это событие должно запустить программу вычисления.

Алгоритм решения задачи вычисления объема кузова следующий:

1. Ввести три числа: A, B, H.
2. Найти объем: V=A*B*H.
3. Вывести результат: число V.

Для написания программного кода и привязки его к событию Нажатие кнопки необходимо раскрыть Окно программного кода Code, которое открывается командой Code (Программа) в меню View.

Процедура – это фрагмент программного кода, с помощью которого решается какая-то локальная задача. Часто (но не всегда!) процедура вызывается событием. В рассматриваемом примере вычисление по формуле начинается после нажатия кнопки "РАСЧЕТ".

Из правого списка выбираем событие Click, из левого – объект CommandButton1. В Окне программного кода появляется заготовка процедуры, программы реакции на нажатие кнопки "РАСЧЕТ".

Первая строка программы начинается со слов Private Sub, а заканчивается программа словами End Sub; это служебные слова языка. Последовательность строк кода соответствует последовательным шагам алгоритма решения данной задачи. Знак "=" обозначает присваивание переменной определенного значения. Знаки "*" и "+" обозначают операции

умножения и сложения. Выражение TextBox1.Text обозначает значение свойства Text объекта TextBox1. Запись Val(X) означает, что значение переменной X преобразуется из строки символов в число, а запись Str (X) означает, что значение переменной X преобразуется из числа в строку символов.

Запускать программу можно:

- с помощью опции Run и команды Start Главной панели проекта;
- с помощью кнопки Start линейки инструментов Главной панели проекта;
- с помощью клавиши F5 клавиатуры.

Завершить работу программы можно тоже по-разному, например:

- с помощью кнопки End на линейке инструментов;
- с помощью стандартного элемента Windows – системной кнопки закрытия окна в правом верхнем углу окна приложения.

Отладка программы. Первая попытка запустить программу не всегда бывает успешной. Часто попытка запуска приводит к появлению сообщений системы Visual Basic об ошибках. В этом случае их нужно исправить – для этого Visual Basic предоставляет разнообразные средства отладки.

Сохранение экранной формы проекта в виде файлов. Когда программа отлажена, проверена, когда доведен интерфейс, выполняется заключительный этап – компилирование.

Лабораторная работа № 12
ПЕРЕМЕННАЯ И ЕЕ ЗНАЧЕНИЕ

Цель работы. Ознакомиться с типами переменных, их описанием в программе.

Теоретическая часть

Необходимость использования переменных в программировании возникает по той же причине, что и в математике, - для обозначения величин, которые часто меняют свое значение.

Понятие переменных является, пожалуй, самым главным понятием в каждом языке программирования. У переменной есть две характеристики: имя и значение. Имя переменной уникально и неизменно, а значение может меняться в процессе выполнения алгоритма. Переменные позволяют с помощью небольшого числа инструкций предусмотреть большое число шагов исполнителя. Кроме этого, одна программа может решать задачу для разных значений исходных величин.

Имя переменной – это строка символов, которая идентифицирует переменную в программе. Переменные создаются программистом при написании программного кода. Он же дает им имена. Имена переменных создаются по определенным правилам:

1. Первым символом имени должна быть буква.
2. Остальные символы – буквы и цифры (прописные и строчные буквы различаются).
3. В имени можно использовать знак "_", но нельзя использовать знак "." (точка). Точка в языке используется в синтаксических конструкциях.
4. Число символов в имени может достигать 255, но лучше работать с короткими именами.
5. Имя не должно быть ключевым словом Visual Basic, в противном случае фиксируется ошибка и на экране не отображается подсказка.

Значение переменной – это данные, которые хранятся и обрабатываются компьютером. Это выполняется по-разному и зависит от того, к какому типу принадлежат данные. Типом данных называется способ хранения и представления данных в компьютере. В языке Visual Basic переменная может принадлежать к одному из более чем десяти типов.

Вот некоторые из них, наиболее употребляемые:

- тип Byte. Короткое неотрицательное целое число, оно занимает 1 байт памяти, его значение меняется в пределах от 0 до 255; тип Integer. Целое число, оно занимает 2 байта памяти, его значение меняется в пределах от -32768 до 32767;
- тип Long. Длинное целое число. Значение переменной этого типа занимает 4 байта памяти и меняется в пределах от -2147483648 до 2147483647;

- тип Single. Десятичное число обычной точности, оно занимает 4 байта памяти, его значение меняется в пределах от 1.401298E-45 до 3.402823E+38 (по модулю);
- тип Double. Десятичное число двойной точности, занимает 8 байт памяти и меняется в пределах от 4.94065645842147E-324 до 1.79769313486232E+308 (по модулю);
- тип String. Переменная строкового типа и текстовая. Значением переменной этого типа является строка любых символов, длина которой может достигать двух миллиардов. Слева и справа строка обрамляется кавычками;
- тип Variant. Произвольное значение. Переменная этого типа может иметь любой размер. Но за это надо платить дорогую цену - объем памяти, занимаемой значением переменной этого типа, бывает разным, но не менее 16 байт!

Тип переменной может в программе и не объявляться. В этом случае по умолчанию он будет установлен самой системой Visual Basic и будет соответствовать типу Variant. Но хорошим тоном в программировании считается обязательное объявление типа каждой переменной.

Объявлять тип созданной переменной можно несколькими способами, самый распространенный - с использованием оператора определения переменной.

Пример.

```
Dim MyName As String
Dim Nmb As Integer
Dim AAA, BBB, CCC As Double
Dim X As Singlr, NmbX As Integer, XX As Double
```

Для присвоения переменной некоторого значения используется оператор присваивания. Оператор – это такая синтаксическая единица языка программирования, которая используется в программе для выполнения отдельного предписания. Операторы делятся на две категории. К первой относятся алгоритмические операторы, ко второй – функциональные

Алгоритмические операторы – это такие операторы, которые используются для организации последовательности выполняемых исполнителем действий. Важнейшие из них – операторы безусловных переходов, условные операторы, операторы циклов.

Функциональные операторы – это встроенные в язык функции и процедуры, с помощью которых производятся важные и распространенные действия, такие, как ввод данных, действия над числами. Любая программа состоит из последовательности операторов, которые записываются в соответствии со строгими синтаксическими правилами: компьютер не воспринимает программы, написанные с ошибками.

Оператор присваивания – один из самых распространенных. Синтаксическое правило для этого оператора выглядит так:

Имя Переменной = Значение Переменной

При выполнении оператора присваивания переменная, имя которой указано слева от знака равенства, получает значение, равное значению выражения, находящегося справа от знака равенства.

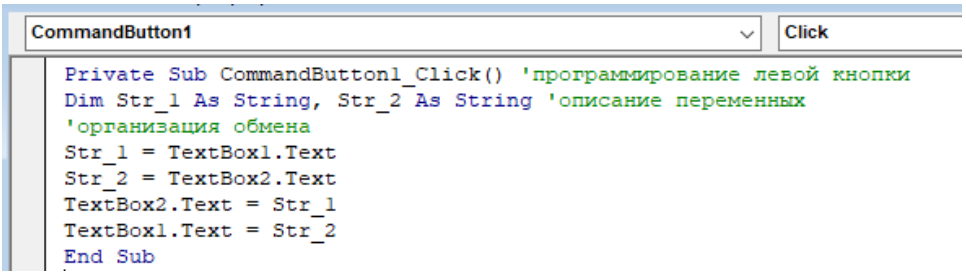
Порядок выполнения работы:

Рассмотрим пример построения Windows-приложения:

В приложении программируются два события: нажатие левой кнопки "Меняются надписи полей" и нажатие правой кнопки "Меняются надписи полей и фон".

Левая кнопка меняет местами содержание окон без изменения цвета фона, правая кнопка меняет местами содержание окон с изменением цвета фона.

Процедура, срабатывающая при нажатии левой кнопки, имеет вид:



Процедура, срабатывающая при нажатии правой кнопки, имеет вид:

```
Private Sub CommandButton2_Click()
Dim Str_1, Str_2 As String, Color_F, Color_L As Long
Str_1 = TextBox1.Text
Str_2 = TextBox2.Text
Color_F = TextBox1.BackColor
TextBox1.Text = Str_2
TextBox2.Text = Str_1
TextBox1.BackColor = TextBox2.BackColor
TextBox2.BackColor = Color_F
End Sub
```

Запустить приложение, проверить работу, правильность обмена.

Сохранить файлы проекта в папке "Организация_обмена".

Задания для самостоятельной работы

1. По заданному радиусу R определить длину окружности l, ее диаметр d и площадь круга S.

2. По заданному диаметру d и углу α определить радиус окружности R, длину дуги l и площадь сектора S.

3. По заданным трем сторонам прямоугольного параллелепипеда a, b, c определить площадь его боковой поверхности Sбок, площадь полной поверхности S и объем V.

4. По заданному радиусу R определить диаметр шара d, площадь его поверхности S и объем V.

5. По заданным радиусу основания R и высоте цилиндра H определить площадь его боковой поверхности Sбок, площадь полной поверхности S и объем V.

6. По заданному радиусу R и высоте шарового сегмента H определить площадь сегментной поверхности S, объем шарового сегмента V и объем шарового сектора Vсек.

7. По заданным радиусу основания R, высоте H и образующей L определить площадь боковой поверхности конуса Sбок, площадь его полной поверхности S и объем V.

8. По заданным радиусам оснований R, r, высоте H и образующей L определить площадь боковой поверхности усеченного конуса Sбок, площадь его полной поверхности S и объем V.

9. По заданным катетам прямоугольного треугольника a, b определить его гипотенузу c, периметр p и площадь S.

10. По заданным сторонам прямоугольника a, b определить квадрат его диагонали d2, периметр p и площадь S.

11. По заданному радиусу R описанной вокруг квадрата окружности определить его сторону a, периметр p и площадь S.

12. По заданному радиусу R описанной вокруг правильного треугольника окружности определить его сторону a, периметр p, площадь S.

Лабораторная работа № 13
ВЫРАЖЕНИЯ И ФУНКЦИИ

Цель работы. Изучить правила построения выражений. Ознакомиться с использованием функций в приложении.

Теоретическая часть

В операторе присваивания справа от знака "=" может быть расположено не только конкретное значение, но и выражение. При выполнении оператора присваивания во время работы программы это выражение вычисляется. Это означает, что по определенным правилам рассчитывается значение этого выражения, а затем это значение присваивается переменной. В состав выражений могут входить конкретные числа, переменные, строки, функции.

Чаще всего в операторе присваивания справа от знака "=" находится так называемое арифметическое выражение. Арифметическое выражение – это последовательность чисел, констант, переменных, функций и арифметических выражений, заключенных в круглые скобки, которые соединены между собой знаками арифметических операций. Значения арифметических выражений вычисляются по правилам, которые являются общеизвестными. Ниже приведена лишь таблица арифметических операций, используемых в языке Visual Basic.

Операция	Описание операции
A^B	Возведение A в степень B
-A	Перемена знака A
A*B	Умножение A на B
A/B	Деление A на B
A\B	Целочисленное деление A на B
A mod B	Деление по модулю A на B
A+B	Сложение A с B
A-B	Вычитание B из A

Переменные, входящие в выражение, должны иметь численные значения. Функции также должны иметь численные значения. Говорят, что функции возвращают определенные численные значения.

Понятие функции в языке близко понятию функции в математике. Функция – это правило, которое ставит в соответствие одному набору

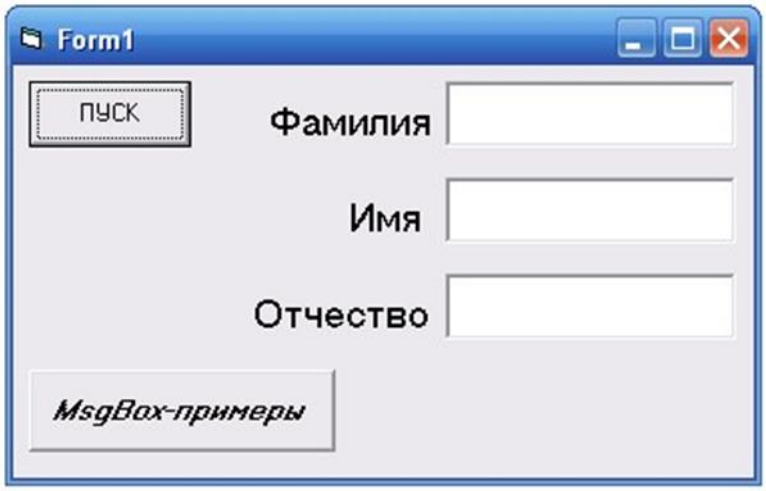
значений аргументов из области их допустимых значений ровно одно значение самой функции.

Системные функции. К системным функциям относятся функции, действие которых напрямую зависит от работы системы Windows. К таким функциям относятся две:

- функция InputBox – для ввода данных пользователем через системное окно;
- функция MsgBox – для выдачи сообщений пользователю через системное окно.

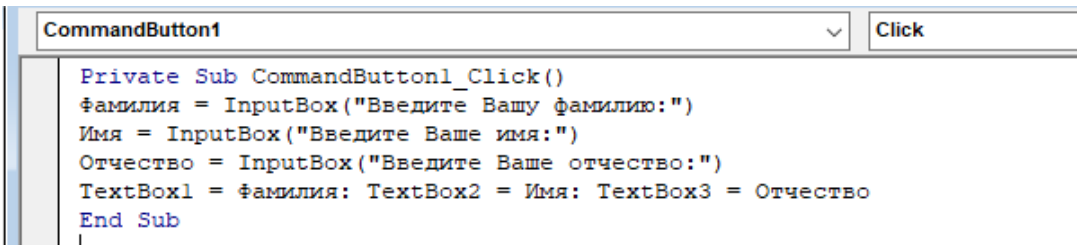
Порядок выполнения работы:

1. Рассмотрим пример построения Windows-приложения, в котором используются функции InputBox и MsgBox:

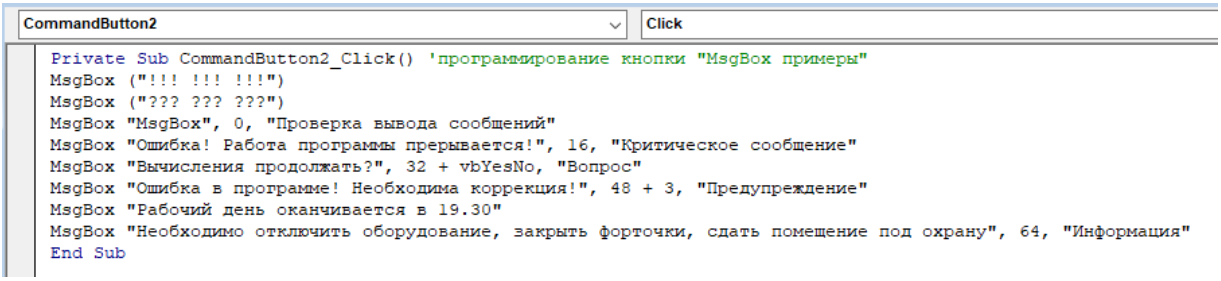


Программа состоит из двух частей. В первой части кнопка "ПУСК" через стандартные окошки ввода функции вводит фамилию, имя, отчество и отображает их в текстовых полях. Во второй части кнопка "MsgBox-примеры" последовательно выводит ряд сообщений разного вида.

Процедура, срабатывающая при нажатии кнопки "MsgBox-примеры", имеет вид:



Процедура, срабатывающая при нажатии кнопки "MsgBox-примеры", имеет вид:



2. Запустить приложение, проверить работу.
3. Дополнить приложение процедурой, срабатывающей, если щелчок левой кнопки мыши будет приходиться не по кнопкам формы. При этом на экране должно появляться сообщение-предупреждение "Ох! Да Вы промазали по кнопке!".
4. Откомпилировать приложение под именем "Предупреждение". Сохранить файлы проекта в папке "Ввод_данных".

Задания для самостоятельной работы

1. Ввести с клавиатуры произвольное строковое выражение и поместить его в выбранную ячейку текущего рабочего листа.
2. Отобразить содержимое любой ячейки рабочего листа в диалоговом окне MsgBox.
3. Разместить несколько вводимых с клавиатуры чисел в разных ячейках на одной строке (в одном столбце) текущего рабочего листа.
4. Скопировать содержимое некоторых ячеек с одного рабочего листа на другой.

Лабораторная работа № 14

ФУНКЦИИ РАБОТЫ СО СТРОКАМИ. ФИНАНСОВЫЕ ФУНКЦИИ

Цель работы. Ознакомиться с функциями обработки строк, основами программирования финансовых функций.

Теоретическая часть

Строка – это либо упорядоченная последовательность символов, либо пустая строка. Для обозначения строки используются кавычки:

“” – пример пустой строки;

“Программирование” – пример непустой строки.

Число символов строки называется длиной строки. Длина пустой строки равна нулю. Каждый символ строки имеет свою позицию – порядковый номер при счете слева направо. В VBA используется понятие подстроки – это вырезанный кусок из строки.

Две строки можно соединить в одну, такое действие называется конкатенацией или сложением строк:

Объединение_Строк = Строка1 + Строка2 + Строка3.

Можно применить знак конкатенации &. С его помощью можно соединить не только строки, но и числа. При этом числа будут сначала преобразованы в строки, и результат тоже будет строкой.

Существует несколько функций обработки строк, которые позволяют модифицировать, обрабатывать строки, выбирать информацию.

Функция определения длины строки:

Len(Строка)

В результате возвращается длина строки.

Функция выделения подстроки:

Mid(Строка, Позиция [,Длина])

В Строке выделяется и возвращается подстрока, начиная с заданной Позиции. Длину выделяемой подстроки можно не указывать - тогда будет возвращена подстрока от данной Позиции до конца Строки.

Функция поиска подстроки:

lnStr([Старт,] Строка, Подстрока)

В Строке ищется то место, где находится Подстрока. В результате возвращается позиция первого символа Подстроки. Если подстрока не найдена, возвращается 0.

Функции преобразования имеют следующие назначения:

Val(Строка)

Эта функция преобразует Строку в число.

Str(Число)

Эта функция преобразует Число любого типа в строку.

Порядок выполнения работы:

Используя функции обработки строк, форму и код приложения (подсчитывать число символов в фамилии, имени, отчестве и выводить в первое отдельное поле первые буквы (инициалы), а во второе отдельное поле – фамилию, имя, отчество в виде одной строки, используя конкатенацию. Примерный вид формы:

Задания для самостоятельной работы

1. Подсчитать, сколько раз каждый символ русского алфавита встречается в заданной строке. Прописные и строчные символы считать одинаковыми.
2. Для заданной строки символов строчные буквы латинского алфавита преобразовать в прописные, а прописные – в строчные.
3. Вывести на экран заданную строку символов в обратном порядке.
4. Для заданного предложения вывести каждое слово в отдельную ячейку. Подсчитать количество слов в предложении.
5. Удалить из произвольного текстового выражения все пробелы. Подсчитать количество пробелов.
6. Для заданных фамилии, имени и отчества студента вывести на экран только фамилию и инициалы.
7. Написать заданное слово вразрядку (буквы отделены друг от друга пробелом). Определить количество букв в слове.
8. Ввести с клавиатуры 2 строки: фамилия, имя, отчество и номер группы. Получить строку вида: ФИО – студент группы 101xxx. Определить длину полученной строки.
9. Ввести с клавиатуры 2 строки. Определить, входит ли вторая строка в состав первой. Если да, то с какой позиции.
10. Преобразовать заданное число в строку. Сформировать строку вида: xxx рублей. Определить длину полученной строки.
11. Определить, какая из двух заданных строк длиннее. Результаты вывести в виде: первая строка (текст) длиннее второй (текст). Определить длину результирующей строки.
12. Подсчитать количество гласных и согласных букв в заданном слове.

Лабораторная работа № 15
ПРОГРАММИРОВАНИЕ ВЕТВЛЕНИЙ

Цель работы. Ознакомиться с условным оператором IF, оператором перехода Select Case.

Теоретическая часть

При решении большинства задач часто приходится выбирать, по какому из нескольких путей нужно идти к решению. Для реализации условия выбора в языке существует вид выражений – условные выражения.

Простое условие – это два выражения, между которыми помещается знак сравнения. Выражениями могут выступать числа, числовые переменные, функции, арифметические выражения, строки. Операции сравнения и их знаки приведены в таблице:

Операция	Описание операции
>	Больше чем
>=	Больше или равно
<	Меньше чем
<=	Меньше или равно
=	Равно
< >	Не равно

Простое условие, в зависимости от того, выполняется оно или нет, имеет значение True или False – Истина или Ложь. Примеры простых условий и их значений приведены в таблице:

2.9990<2.9991	имеет значение True
3.14<=3.14	имеет значение True
-Y^2>Abs(Y)	имеет значение False
"abc"="abc"	имеет значение True
"-abc"="abc"	имеет значение False

Сложное условие – это последовательность простых условий или других выражений, заключенных в круглые скобки, которые соединены между

собой знаками логических операций: AND – логического умножения, OR – логического сложения, NOT – логического отрицания. Каждое условное выражение вычисляется, а результатом является одно из двух значений: True или False – Истина или Ложь.

Правила вычисления значений логических выражений нужно знать так же, как таблицу умножения.

A	B	A AND B
True	True	True
True	False	False
False	True	False
False	False	False

A	B	A OR B
True	True	True
True	False	True
False	True	True
False	False	False

A	NOT A
True False	False True

Простые и сложные условия являются элементами условного оператора, позволяющего в программном коде выполнять ветвления. Условный оператор имеет две формы: однострочную и многострочную.

Синтаксис однострочной формы:

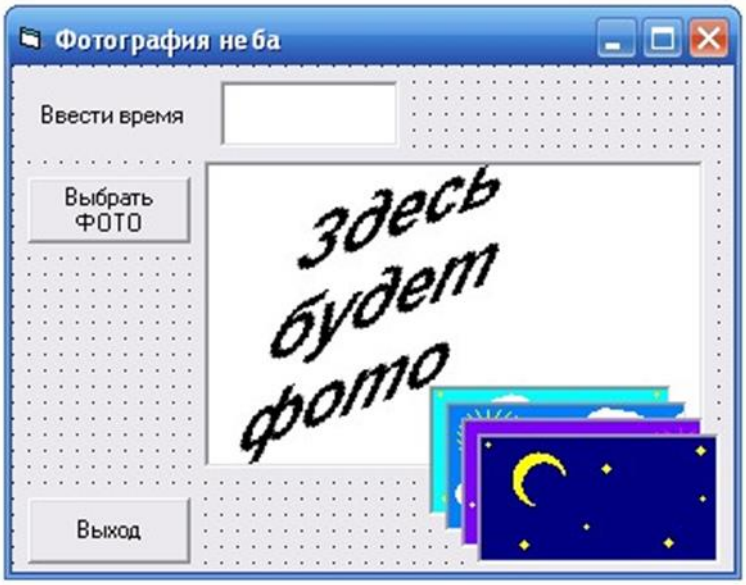
If Условное Выражение Then Оператор 1 [Else Оператор2]

Синтаксис многострочной формы:

If Условное Выражение Then Группа операторов
[Else Группа операторов] End If

Порядок выполнения работы:

1. В проекте будут использованы объекты: Label, Text, CommandBatton, PictureBox, Image:



В этой программе в текстовое окно вносится число от 0 до 24, а в графическом окне высвечивается фотография утреннего неба, если вводится число из интервала [5 ... 8], дневного неба, если вводится число из интервала [8 ... 18], вечернего или ночного неба, если вводится число из интервалов соответственно [18 ... 21] и [21 ... 5].

2. Код проекта:

```
Private Sub CommandButton1_Click() 'Выбрать ФОТО
If ((Val(TextBox1.Text) > 5) And (Val(TextBox1.Text)<=8))
Then Image1.Picture =Picture1(0).Picture
If ((Val(TextBox1.Text) > 8) And (Val(TextBox1.Text) <= 18))
Then Image1.Picture = Picture1(1).Picture
If ((Val(TextBox1.Text) > 18) And (Val(TextBox1.Text) <= 21))
Then Image1.Picture=Picture1 (2).Picture
If (((Val(TextBox1.Text) > 21) And (Val(TextBox1.Text) <= 24)) Or ((Val(TextBox1.Text) > 0) And (Val(TextBox1.Text) <= 5)))
Then Image1.Picture = Picture1(3).Picture
End Sub
Private Sub CommandButton2_Click() 'Выход
End
End Sub
```

3. Для альтернативы, или выбора варианта, существует оператор перехода Select Case. Пример использования оператора перехода Select Case и двух новых инструментов в наборе Tollbox - OptionButton (Кнопка-

переключатель) и Frame (Рамка) показан в приложении, форма которого представлена на рисунке:

В приложении в зависимости от выбора кнопки переключателя выводится соответствующий рисунок.

Код программы:

```
Dim AA As Byte   'описание глобальной переменной
Private Sub F_Case() 'функция выбора варианта рисунка по значению AA
Select Case AA
Case 1
Image1.Picture = Picture1(0).Picture
Case 2
Image1.Picture = Picture1(1).Picture
Case 3
Image1.Picture = Picture1(2).Picture
Case 4
Image1.Picture = Picture1(3).Picture
End Select
End Sub
Private Sub Option_1 Click()
AA= 1
F_Case
End Sub
Private Sub Option2_Click()
AA = 2
F_Case
End Sub
Private Sub Option3_Click()
AA = 3
F_Case
```

```
End Sub
Private Sub Option4_Click()
AA = 4
F_Case
End Sub
Private Sub Command5_Click() 'кнопка "Выход"
End
End Sub
```

Задания для самостоятельной работы

Каждое задание состоит из двух частей: а) и б).

а) Применение операторов Goto и If

1. Найти корни квадратного уравнения $ax^2 + bx + c = 0$.
2. Найти наибольшее и наименьшее из трех чисел A, B, C.
3. Ввести с клавиатуры число, месяц, год, день недели. Вывести на экран дату и день недели для следующего дня.
4. С клавиатуры ввести два числа. Разделить большее на меньшее.
5. По названию месяца определить количество дней в нем.
6. По введенному с клавиатуры году определить, является ли он високосным.
7. Определить принадлежность человека определенному знаку зодиака по дате его рождения.
8. Компьютер выдает на экран 5 вопросов и по 2 варианта ответов к каждому. Правильный ответ оценивается в 1 балл. Оценить уровень знаний тестируемого.
9. Узнайте какой день недели был 100 дней назад.
10. Найдите количество дней в текущем месяце. Программа должна работать независимо от месяца, в котором она запущена.

б) Применение оператора Select Case

1. Для заданного числа из диапазона от 1 до 10 выдать его словесное (символьное) представление.
2. По номеру месяца выдать его название и количество дней в нем.
3. Ввести два числа и знак арифметической операции между ними ("+", "*", "/"). Вычислить значение арифметического выражения согласно введенному варианту.
4. Ввести с клавиатуры первые две цифры штрих-кода товара. По введенному значению определить страну-производителя.
5. Организовать телефонный справочник известных аварийных и справочных служб.
6. По номеру группы определить количество студентов в ней и год поступления.
7. По номеру группы определить название специальности и курс.

8. По порядковому номеру выдать на экран фамилию студента вашей группы.

9. Вывести на экран список цветов (5-7 элементов). Выдать на экран свою фамилию выбранным цветом. Выбор цвета реализовать по его порядковому номеру или названию.

10. По номеру дня недели или его названию выдать на экран расписание занятий.

Лабораторная работа № 16
ПРОГРАММИРОВАНИЕ ПОВТОРЕНИЙ

Цель работы. Ознакомиться с операторами повторения.

Теоретическая часть

Повторение – это выполнение одного или нескольких операторов программы более одного раза. За счет повторений сокращается размер программного кода. Реализуются повторения многострочными операторами цикла двух видов: со счетчиком и с условием.

Цикл со счетчиком

Синтаксис оператора повторения для цикла со счетчиком:

For Имя=3начение1 To Значение2 [Step Значение3]
Повторяющиеся операторы (операторы цикла)
Next [Имя]

где Имя – имя переменной, которую называют счетчиком (индексом цикла);
Значение1 – начальное значение счетчика;
Значение2 – конечное значение счетчика;
Значение3 – величина, на которую изменяются значения счетчика при одном повторении.

Если нет противоречий в значениях, то операторы цикла выполняются при начальном значении счетчика. Далее значение счетчика меняется на величину шага и выполняется проверка со Значением2. Если значение счетчика меньше, то начинается повторное выполнение операторов цикла с новым значением счетчика, если больше, то цикл завершается и начинают отрабатываться следующие операторы программы.

Пример блока простейшей программы, использующей цикл со счетчиком:

```
Private Sub CommandButton1_Click()
For i = 1 To 10 Step 2
Cells(1, i) = i
Next i
End Sub
```

Задания для самостоятельной работы

1. Вывести на экран таблицу умножения для введенного с клавиатуры числа n в виде:

n×1=n
n×2=2n
…

$n \times 10 = 10n$

2. Для заданного целого числа вывести на экран список чисел, на которые оно делится без остатка.

3. Найти сумму четных цифр числа.

4. Найти сумму первой и последней цифр числа.

5. Найти сумму целых чисел от N до M

6. Даны натуральные числа от 20 до 50. Напечатать те из них, которые делятся на 3, но не делятся на 5.

7. Напишите программу, где пользователь вводит любое целое положительное число. Программа суммирует все числа от 1 до введенного пользователем числа.

8. Составьте программу, которая вычисляет произведение чисел от 1 до N. Значение N вводится с клавиатуры.

9. 1 кг творога стоит 12 руб. Вывести на экран таблицу стоимости творога массой 100 г, 200 г, …, 900 г.

10. 1 маркер стоит 3 руб. 50 коп. Вывести на экран таблицу стоимости 2, 3,…10 маркеров.

Лабораторная работа № 17
МАССИВЫ

Цель работы. Изучить понятие "массив". Научиться использовать массивы в приложениях VBasic.

Теоретическая часть

Дальнейшим развитием понятия "переменная" является понятие "массив".

Массив – это объединение переменных одного типа. У них одно имя, а отличаются они друг от друга своим номером – значением так называемого индекса. У переменной массива могут быть два, три или даже больше индексов – это многомерные массивы. Организация данных в виде массивов экономит место и упрощает алгоритмы.

Массивы могут состоять не только из чисел, но и из строк, объектов.

Пример нахождения минимального значения в массиве размерностью 10*12, полученного случайным образом:

```
Private Sub CommandButton1_Click()
Dim d123(1 To 10, 1 To 12) As Integer
For i = 1 To 10
For k = 1 To 12
d123(i, k) = Fix(Rnd * 90 + 10)
Cells(i, k) = d123(i, k)
Next k
Next i
Min = d123(1, 1)
For i = 1 To 10
For k = 1 To 12
If d123(i, k) < Min Then Min = d123(i, k)
Next k
Next i
Cells(12, 5) = "Минимальное значение"
Cells(13, 5) = Min
End Sub
```

Задания для самостоятельной работы

Каждое задание состоит из двух частей: а) и б).

а) Одномерные массивы

1. Подсчитать количество положительных, отрицательных, нулевых значений, количество целых и дробных чисел, хранящихся в одномерном массиве из 10 элементов.

2. Найти наибольший и наименьший элементы одномерного массива из 10 элементов.

3. Подсчитать отдельно сумму четных и нечетных элементов одномерного массива из 10 элементов.

4. Вычислить сумму и произведение элементов одномерного массива из 10 элементов. Разделить первую величину на вторую.

5. Дан массив целых чисел A. Четные значения элементов массива записать в массив B, нечетные – в массив C. Подсчитать количество тех и других.

6. 8. Два одномерных массива X и У (из 5 элементов каждый) объединить в один – Z. Четные элементы массива Z состоят из элементов массива X, нечетные – из элементов массива У.

9. В одномерном массиве хранятся значения температуры воздуха в течение суток с интервалом в один час. Определить средние температуры дня и ночи, среднесуточную температуру.

10. Поменять местами четные и нечетные элементы одномерного массива из 10 элементов.

12. Сформировать массив, содержащий элементы исходного массива в обратном порядке.

б) Многомерные массивы

1. Перемножить два массива A и B размерностью n×m и m×k. Размерности матриц и значения их элементов ввести с клавиатуры. Результат вывести на экран в виде матрицы n×k.

2. Сложить два массива A и B размерностью n×m. Размерности массивов и значения их элементов ввести с клавиатуры. Результат вывести на экран в виде матрицы n×m.

3. Транспонировать массив A размерностью n×m. Размерность массива и значения ее элементов ввести с клавиатуры. Результат вывести на экран в виде матрицы m×n.

4. Вычислить определитель массива A размерности 3×3.

5. Подсчитать сумму элементов каждой строки двумерного массива размерностью n×m.

6. Подсчитать среднее арифметическое каждого столбца двумерного массива размерностью n×m.

7. Найти наибольший элемент каждой строки и наименьший элемент каждого столбца двумерного массива размерностью n×m.

8. Дана матрица n×m. Вывести на экран дисплея элементы той строки, сумма элементов которой максимальна.

9. Дана матрица n×m. Вывести на экран дисплея элементы той строки, сумма элементов которой минимальна.

10. Дана матрица n×m. Вывести на экран дисплея элементы столбца, сумма элементов которой максимальна.

11. Дана матрица n×m. Вывести на экран дисплея элементы столбца, сумма элементов которой минимальна.

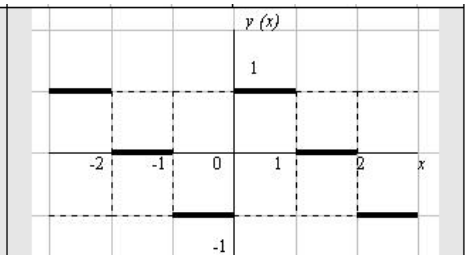
12. Подсчитать произведение элементов каждой строки двумерного массива размерностью n×m.

КОНТРОЛЬ ЗНАНИЙ

Экзаменационные вопросы

1. Логистические информационные системы. Основные понятия, виды и принципы построения логистических информационных систем.
2. Классификация и использование информационных систем.
3. Жизненный цикл программного обеспечения.
4. Процесс разработки прикладного программного обеспечения. Основные этапы создания прикладного программного обеспечения.
5. Пользовательский интерфейс приложения (Application Programming Interface) и его проектирование.
6. Графический пользовательский интерфейс (Graphical User Interface). Элементы графического интерфейса. Основные элементы управления.
7. Проектирование графического интерфейса пользователя. Формы пользователя. Ввод и вывод данных при помощи текстового окна.
8. Проектирование графического интерфейса пользователя. Ввод и вывод данных с помощью системных функций. Процесс разработки приложения с диалоговой формой.
9. Проектирование графического интерфейса пользователя. Процесс разработки приложения с диалоговой формой.
10. Формы. Элементы управления. События и свойства форм и элементов управления.
11. Формы. Создание элементов управления и меню. Управление свойствами элементов управления.
12. Работа с формами (добавление, установление свойств и обработка события).
13. Организация взаимодействия между формами. Обзор графических возможностей.
14. Пользовательское меню. Улучшение интерфейса программы и повышение удобства пользования приложением.
15. Обработка двумерных массивов в VBA.
16. Обработка одномерных массивов в VBA.
17. Работа с символами и строками в VBA.
18. Инструкции принятия решения в VBA.
19. Типы данных в VBA.
20. Структура редактора VBA. Интерфейс приложения (VBA в Excel).
21. Алгоритмы в логистике. Понятие, свойства и типы алгоритмов.
22. Организация циклов в VBA.
23. Элементы языка VBA.
24. Многооконные приложения в VBA.

25. Записать программу, которая на ввод значения аргумента выдает значение функции, заданной графиком.



26. Вывести на экран таблицу квадратов целых чисел от 1 до 10.

27. Вычислить факториал натурального числа N.

28. Дан массив целых чисел из *n* элементов, заполненный случайным образом числами из промежутка [-10, 10]. а) Найти сумму элементов, имеющих нечетное значение. б) Вывести индексы тех элементов, значения которых больше заданного числа *A*. в) Определить, есть ли в данном массиве положительные элементы, кратные заданному числу *K*.

29. Дан массив целых чисел из *n* элементов, заполненный случайным образом числами из промежутка [-15,15]. а) Найти произведение элементов, имеющих четное значение. б) Вывести индексы тех элементов, значения которых по модулю меньше заданного числа *A*. в) Определить, есть ли в данном массиве положительные элементы, делящиеся на заданное число *k* с остатком 2.

30. Дан массив целых чисел из *n* элементов, заполненный случайным образом числами из промежутка [-20,20]. а) Найти удвоенную сумму положительных элементов. б) Вывести индексы тех элементов, значения которых больше значения предыдущего элемента (начиная со второго). в) Определить, есть ли две пары соседних элементов с одинаковыми знаками.

31. Дан массив целых чисел из *n* элементов, заполненный случайным образом числами из промежутка [-100,100]. а) Найти сумму положительных элементов, значения которых меньше 10. б) Вывести индексы тех элементов, значения которых кратны 3 и 5. в) Определить, есть ли пара соседних элементов с суммой, равной заданному числу.

32. Дан одномерный массив из N элементов. а) Определить количество элементов, значения которых меньше заданного числа M. б) Заменить элементы с четными индексами, значением максимального элемента. в) Найти среднее арифметическое максимальных и минимальных элементов массива и элементов с нечетными номерами индексов.

33. Дан текст, заканчивающийся точкой. Посчитать сколько в нем слов.

34. Дан текст, среди символов которого есть «двоеточие». Определить сколько символов ему предшествует.

35. Дана строка символов, среди которых есть одна открывающаяся и одна закрывающаяся скобка. Вывести на экран все символы, расположенные внутри этих скобок.

36. Дано натуральное число. Приписать к нему такое же число.

37. Даны 3 действительных числа. Возвести в квадрат те из них, значения которых отрицательны, и в куб – положительны.

38. Для введенного с клавиатуры символа определить, является ли этот символ буквой русского или латинского алфавита, прописной или строчной буквой.

39. Найти количество отрицательных чисел среди a, b, c и абсолютное значение суммы этих чисел.

40. Найти количество различных цифр данного натурального числа.

41. Найти сумму всех чётных натуральных чисел от 1 до 100.

42. Напечатать числа a, b и c в порядке возрастания.

43. Определить плату за электроэнергию, если известны: старое и новое показания счётчика, стоимость одного кВт/ч электроэнергии, количества просроченных дней уплаты и размер пени за 1 день просрочки.

44. Определить произведение цифр натурального числа N.

45. Определить сколько раз первая цифра встречается в данном числе.

46. Определить сумму первой и предпоследней цифр данного четырехзначного натурального числа N.

47. Определить сумму цифр натурального числа N.

48. Определить частное от деления первой и последней цифр натурального числа N.

49. Определить является ли введенное число простым (простое число – делится только на 1 и на само себя).

50. Поменять порядок следования цифр в натуральном числе N на обратный.

51. Проверить, одинаковое ли число открывающихся и закрывающихся скобок в данной строке.

52. Произвести суммирование натуральных чисел *1, 2, 3,...,* пока их сумма *s* не станет равной или превысит величину *h*. Вывести на экран последнее слагаемое и значение суммы.

53. Создать приложение, которое для заданного числа из диапазона от 1 до 10 выдать его словесное (символьное) представление (с помощью оператора Select Case).

54. Создать приложение, которое на ввод времени суток выводит соответствующее пожелание доброго утра, доброго дня, доброго вечера и спокойной ночи.

Задание для выполнения расчетно-графической или контрольной работы

3 семестр

ССЫЛКА на pdf файл

4 семестр

Задание

Тема: «Создание приложения "***" с помощью Visual Basic»

Студент выбирает одно из предложенных приложений согласно номера в зачетной книжке (две последние цифры) или предлагает свой вариант (согласовать).

СТАРОСТА зафиксирует выбранный вариант приложения и список предоставит для согласования:

1. «Кредитный калькулятор»
2. «Будильник-таймер»
3. «Расчёт расстояния между городами Республики Беларусь (основные города)»
4. «Справочник»
5. «Путеводитель по любому городу»
6. «Ежедневник»
7. «Автокаталог»
8. «Расчёт стоимости автоперевозки»
9. «Расчет стоимости поездки на такси»
10. «Расчёт стоимости растаможки автомобиля»
11. «Депозитный калькулятор»
12. «Каталог услуг транспортно-логистической компании»
13. «Каталог транспортно-логистических компаний (Республика Беларусь)»
14. «База клиентов»
15. «Заполнение бланков (ТТН)»
16. «Статистика платежей»
17. «Контроль расходов»
18. «Формирование, учет и управление потребностями в грузоперевозке, возникающими на основании заявок на перевозку грузов конечными потребителями услуги (или заказов покупателей в случае, если отправной точкой бизнес-процесса рассматривать заказ на покупку товара)»
19. «Формирование, учет и управление потребностями в грузоперевозке, возникающими на основании заказов на поставку (например, в случаях “самовывоза”)»
20. «Формирование, учет и управление потребностями в грузоперевозке, возникающими на основании заявок на внутренние грузоперемещения (например, между собственными складами)»

21. «Создание, регистрация и контролирование исполнения заданий на грузоперевозку, которые могут создаваться как вручную, так и автоматически по заданным алгоритмам на основании сформированной потребности в перевозке»
22. «Создание общих рейсов для осуществления перевозок, указанных в различных нескольких заданиях или заявках»
23. «Контролирование исполнения рейсов с трекингом прохождения маршрута»
24. «Управление внешними и собственными ресурсами: создание/регистрация и контролирование исполнения заданий на выделение транспорта»
25. «Визуализация процесса и информации на электронных web-картах»
26. «Получение аналитики и статистических отчетов оценки KPI (для анализа выполнения бизнес-процесса грузоперевозок и анализа накопленной big data по перевозкам)»
27. «Оптимизация доставки товаров по городу»
28. «Автоматизированная система управления товарными запасами»
29. «Метод ABC»
30. «Метод XYZ»

Примерное содержание выполненной работы:
1. Описание практического применения приложения «***»
2. Процесс создания приложения «***»
3. Листинг программы
4. Возможности модернизации приложения
Список использованных источников

Содержание библиографического списка определяет студент, исходя из цели и задач выполнения работы. Библиографический список включает библиографические записи цитируемых, упоминаемых и изученных автором работы документов. В библиографический список включаются библиографические записи на все документы, независимо от их носителя (печатные материалы: книги, статьи из журналов, сборников, главы из книг; электронные документы, в том числе Интернет-ресурсы; аудиовизуальные, архивные документы).

Документы, включенные в список, представляются в виде библиографических записей, которые следует составлять в соответствии с требованиями государственных стандартов:

1. ГОСТ 7.1-2003 «Библиографическая запись. Библиографическое описание. Общие требования и правила составления».
2. ГОСТ 7.80-2000 «Библиографическая запись. Заголовок. Общие требования и правила составления».
3. ГОСТ 7.82-2001 «Библиографическая запись. Библиографическое описание электронных ресурсов. Общие требования и правила составления».

4. ГОСТ 7.12-93 «Библиографическая запись. Сокращение слов на русском языке. Общие требования и правила».

5. СТБ 7.12-2001 «Библиографическая запись. Сокращение слов и словосочетаний на белорусском языке».

6. ГОСТ 7.11-2004 (ИСО 832:1994) «Библиографическая запись. Сокращение слов и словосочетаний на иностранных европейских языках».

Пример оформления Интернет-ресурса:

1. Национальный правовой Интернет-портал Республики Беларусь [Электронный ресурс]. – Режим доступа: http://www.pravo.by. – Дата доступа: 22.11.2021.

Краткие требования к оформлению работы:

1. Размер бумаги А4, ориентация бумаги – книжная.
2. Размеры полей:
   левое – 3 см; правое – 1 см; верхнее – 2,5 см; нижнее – 2 см.
3. Рекомендуемый шрифт – Times New Roman.

Размер шрифта - 14 пт. Для выделения заголовков – полужирный.

4. Установить межстрочный интервал – 18 пт по всей работе.
5. Абзацный отступ в тексте - 1,25 см.
6. Выравнивание основного текста работы – по ширине.
7. Выравнивание заголовков, подзаголовков – по центру.
8. Каждый новый раздел с новой страницы.
9. Заголовки Содержание и Список использованных источников не нумеровать, записывать симметрично тексту (по центру).
10. В тексте работы обязательны ссылки на литературу или другие источники информации, по образцу: [1].
11. Номера страниц (арабские цифры без точки) располагаются в правом верхнем углу страницы.
12. Работа сшивается по левому краю.

ВСПОМОГАТЕЛЬНЫЙ РАЗДЕЛ
Выдержки из учебной программы

Согласно учебным планам на изучение учебной дисциплины отведено:

- для очной формы получения высшего образования всего 320 ч., из них аудиторных - 154 часа;
- для заочной формы получения высшего образования всего 320 ч., из них аудиторных - 38 часов.

Распределение аудиторных часов по курсам, семестрам и видам занятий приведено ниже.

Таблица 1.

Очная форма получения высшего образования					
Курс	Семестр	Лекции, ч.	Лабораторные занятия, ч.	Практические занятия, ч.	Форма текущей аттестации
2	3	34	52		зачет
2	4	34	34		экзамен

Таблица 2.

Заочная форма получения высшего образования					
Курс	Семестр	Лекции, ч.	Лабораторные занятия, ч.	Практические занятия, ч.	Форма текущей аттестации
2	3	8	12		зачет
2	4	8	10		экзамен

СОДЕРЖАНИЕ УЧЕБНОГО МАТЕРИАЛА

Раздел I. Предметная область, место и роль информационных систем в логистике

Тема 1.1. Информационная логистика

Понятие информационного потока. Роль информационного потока в логистической системе. Управление информационным потоком.

Логистическая информация: понятие, принципы, основы формирования, формы представления. Классификация информационных ресурсов в логистике; виды и средства управления, методы анализа и моделирования логистических информационных потоков.

Обзор и перспективы использования информационных технологий и систем в логистике.

Тема 1.2. Информационная инфраструктура логистики

Логистические информационные системы. Основные понятия, виды и принципы построения логистических информационных систем. Информационные системы управления логистической компанией. Рынок

пакетов программ планирования и управления производством; сравнительный анализ, функциональные возможности, достоинства и недостатки информационных систем в транспортной и складской логистике; перспективы развития логистических информационных систем.

Виды информационных систем в логистике. Понятие электронного обмена данными EDI, изучение систем JIT, MRP I, MRP II, ERP. Управление складом WMS и управление транспортировкой TMS.

Интеграция между информационными логистическими системами.

Раздел II. Корпоративные информационные системы (КИС) в логистике

Тема 2.1. Базы данных как основной элемент КИС

Понятие и классификация КИС. Классификация и эволюция корпоративных информационных систем. Способы формирования и внедрение КИС.

Основные концепции организации данных в КИС. Понятие и классификация СУБД. Иерархические, сетевые, реляционные и объектно-реляционные базы данных.

Основные понятия реляционной модели данных, проектирование реляционных СУБД. СУБД с централизованной архитектурой, архитектурой файл-сервер, клиент-сервер и трехуровневой архитектурой (тонкий клиент-сервер приложений -сервер базы данных).

Базы данных оперативной обработки транзакций (OLTP) и системы делового анализа (OLAP). Распределенные базы данных. Обзор существующих СУБД.

Тема 2.2. Проектирование реляционных баз данных

Типовая организация современной СУБД. Ранние подходы к организации СУБД. Общие понятия реляционного подхода к организации баз данных.

Базисные средства манипулирования реляционными данными Проектирование реляционных баз данных с использованием нормализации. Внутренняя организация реляционных СУБД. Структуры внешней памяти, методы организации индексов. Управление транзакциями, сериализация транзакций. Журнализация изменений БД.

Элементы языка SQL. Функции и основные возможности языка SQL. Выборка данных с использованием предложения SELECT. Вложенные подзапросы.

Манипулирование данными. Архитектура клиент-сервер (InterBase, MySQL, Oracle). Основные особенности архитектуры клиент-сервер. Описание данных на основе SQL. Триггеры и хранимые процедуры. Работа с BLOB и функции, определенные пользователем. Транзакции. Механизм транзакций.

Разработка приложений для работы с БД. Описание интерфейса среды (Delphi, C++ Builder, FoxPro) и ее компонентов для работы с клиент-

серверной БД. Компоненты доступа к данным и визуальные компоненты. Технология InterBaseExpress (IBX). Технология dbExpress. Технология доступа к данным ADO.

Тема 2.3. Разработка приложений для работы с базой данных

Система безопасности СУБД: установка пароля на базу данных, защита структуры базы данных в MDE-файлах, шифрование и дешифрование базы данных.

Защита на уровне пользователей: модель разграничения доступа, файл рабочей группы.

Макросы. Использование программ VBA в модулях и формах. Объектные модели DAO и ADO.

Тема 2.4. Введение в язык SQL

Язык SQL - назначение, особенности, преимущества, существующие стандарты.

Запросы в SQL: простые и многотабличные запросы, внутренние и внешние объединения, итоговые запросы, запросы с группировкой, подчиненные запросы. Представления в SQL. Псевдонимы и индексы таблиц.

Целостность данных: правила каскадного удаления и обновления данных, ограничения на значение столбцов, домены и утверждения в SQL.

Изменение данных при помощи SQL. Понятие транзакции, обработка транзакций. Работа в многопользовательском режиме и блокировка.

Раздел III. Автоматизация проектирования прикладных информационных систем в логистике

Тема 3.1. Проектный подход к разработке информационных систем

Система автоматизированного проектирования: понятие, структура и преимущества. Виды проектирования. Автоматизированное проектирование. Разработка технического задания. Системы автоматизированного проектирования с поддержкой функций внедрения и связывания объектов OLE (Object Linking and Embedding). Интерфейс программного продукта.

Тема 3.2. Разработка прикладных программных систем

Процесс разработки прикладного программного обеспечения. Основные этапы создания прикладного программного обеспечения. Алгоритмизация. Технологии программирования: структурное, модульное, объектно-ориентированное, императивное, функциональное, параллельное программирование.

Тема 3.3. Проектирование интерфейса информационных систем в логистике

Пользовательский интерфейс приложения (Application Programming Interface) и его проектирование. Состояние времени выполнения и дизайна. Графический пользовательский интерфейс (Graphical User Interface). Элементы графического интерфейса. Основные элементы управления. Обзор современных информационных систем логистического управления.

Тема 3.4. Основы технологии разработки программных средств

Объектно-ориентированное программирование (ООП). Концепции ООП. Методологии моделирования предметной области. Моделирование взаимодействия между объектами.

Проектирование графического интерфейса пользователя. Формы пользователя. Ввод и вывод данных при помощи текстового окна. Ввод и вывод данных с помощью системных функций. Процесс разработки приложения с диалоговой формой.

Тема 3.5. Конструирование прикладных информационных систем в логистике

Структура программы. Объявление переменных и их инициализация. Константы. Время жизни и область видимости переменных. Выражения. Правила построения выражений. Операции и их приоритеты. Разработка линейных программ.

Операторы. Простые и составные операторы. Пустой оператор. Условный и безусловный переход. Операторы выбора. Операторы цикла. Операторы передачи управления. Циклы, управляемые счетчиком. Циклы, выполняющиеся или завершающиеся по условию. Разработка программ с использованием операторов выбора, цикла и передачи управления.

Символы и строки в прикладной информационной системе. Строки постоянной и переменной длины. Разработка программ с использованием символов и строк.

Статические и динамические массивы. Одномерные, многомерные и ступенчатые массивы. Способы описания и создания. Разработка программ с использованием массивов.

Тема 3.6. Приложения с графическим пользовательским интерфейсом

Создание приложений с графическим интерфейсом пользователя (GUI). SDI- и MDI-приложения. Создание родительских и дочерних форм. Взаимодействие между формами. Формы. Элементы управления. События и свойства форм и элементов управления. Создание элементов управления, меню, панелей инструментов, строки состояния. Работа с формами

(добавление, установление свойств и обработка события).

Организация взаимодействия между формами. Обзор графических возможностей.

Пользовательское меню. Улучшения интерфейса программы и повышение удобства пользования приложением.

УЧЕБНО-МЕТОДИЧЕСКАЯ КАРТА УЧЕБНОЙ ДИСЦИПЛИНЫ
очная форма получения высшего образования

Номер раздела, темы	Название раздела, темы	Количество аудиторных часов					Количество часов УСР	Форма контроля знаний
		Лекции	Практические занятия	Семинарские занятия	Лабораторные занятия	Иное		
1	2	3	4	5	6	7	8	9
	3 семестр							
1.	Предметная область, место и роль информационных систем в логистике							
1.1	Информационная логистика	2			2			
1.2	Информационная инфраструктура логистики	4			6			опрос
2.	Корпоративные информационные системы (КИС) в логистике							
2.1	Базы данных как основной элемент КИС	4			8			опрос
2.2	Проектирование реляционных баз данных	12			18			расчетно-графическая работа
2.3	Разработка приложений для работы с базой данных	4			10			опрос
2.4	Введение в язык SQL	8			8			
	Итого за семестр	34			52			зачет

1	2	3	4	5	6	7	8	9
	4 семестр							
3.	Автоматизация проектирования прикладных информационных систем в логистике							опрос
3.1	Проектный подход к разработке информационных систем	2			2			
3.2	Разработка прикладных программных систем	8			8			опрос
1	2	3	4	5	6	7	8	9
3.3	Проектирование интерфейса информационных систем в логистике	2			2			
3.4	Основы технологии разработки программных средств	12			12			расчетно-графическая работа
3.5	Конструирование прикладных информационных систем в логистике	6			4			опрос
3.6	Приложения с графическим пользовательским интерфейсом	4			6			
	Итого за семестр	34			34			экзамен
	Всего аудиторных часов	154						

УЧЕБНО-МЕТОДИЧЕСКАЯ КАРТА УЧЕБНОЙ ДИСЦИПЛИНЫ
заочная форма получения высшего образования[1]

Номер раздела, темы	Название раздела, темы	Количество аудиторных часов					Количество часов УСР	Форма контроля знаний
		Лекции	Практические занятия	Семинарские занятия	Лабораторные занятия	Иное		
	3 семестр							
2.	Корпоративные информационные системы (КИС) в логистике							
2.1	Базы данных как основной элемент КИС	1			2			
2.2	Проектирование реляционных баз данных	3			4			контрольная работа
2.3	Разработка приложений для работы с базой данных	3			4			опрос
2.4	Введение в язык SQL	1			2			
	Итого за семестр	8			12			зачет
	2 семестр							
3.	Автоматизация проектирования прикладных информационных систем в логистике							
3.2	Разработка прикладных программных систем	2			2			
3.4	Основы технологии разработки программных средств	4			4			опрос
3.5	Конструирование прикладных информационных систем в логистике	2			4			контрольная работа
	Итого за семестр	8			10			экзамен
	Всего аудиторных часов	38						

[1] Темы учебного материала, не указанные в Учебно-методической карте, отводятся на самостоятельное изучение студентом.

Рекомендуемая литература

Основная литература

1. Автоматизация проектирования БД. Разработка базы данных: практикум / Е.Г. Гриневич, И.Г. Орешко, Ю.Н. Силкович. - Минск: Печатный Дом "Вишневка", 2018. - 143 с.
2. Базы данных / Б. Я. Советов, В. В. Цехановский, В. Д. Чертовской. - 3-е изд., переработанное и дополненное. - Москва: Юрайт, 2018. - 418, [2] с.
3. Базы данных: [в 2 кн.] / В. П. Агальцов. - 2-е изд., переработанное. - Москва: Форум, Инфра-М, 2017.
4. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. Приемы объектно-ориентированного проектирования. Паттерны проектирования. — СПб: Питер, 2020. — 368 с.
5. Информационные технологии в логистике: учебно-методическое пособие для студентов дневной и заочной форм обучения / О.Ю. Кунцевич. - Минск: МИТСО, 2017. - 81 с.
6. Информационные технологии на транспорте: учебник для студентов высших учебных заведений, обучающихся по инженерно-техническим направлениям и специальностям / А.Э. Горев. - Москва : Юрайт, 2016. – 270 с.
7. Информационные технологии: учебное пособие / И. А. Коноплева, О. А. Хохлова, А. В. Денисов. - 2-е изд. - Москва: Проспект, 2017. - 327 с.
8. Компьютерные информационные технологии / Министерство образования Республики Беларусь, Учреждение образования "Полоцкий государственный университет". Ч. 3, кн. 2 : Технологии баз данных и знаний: в 2 кн. / С. Е. Рясова. — 2017. — 140 с.
9. Коннолли Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика: пер. с англ. / Т. Коннолли, К. Бегг. - М.: Вильямс, 2017. - 1439 с.
10. Лукин, В.Н. Введение в проектирование баз данных / В.Н. Лукин. - М.: Вузовская книга, 2015. - 144 с.
11. Основы информационных технологий: курс лекций / С.Н. Батан, Л.В. Батан, О. В. Малашук. - Могилев : МГУ, 2016. - 118, [1] с.
12. Стружкин, Н.П.Базыданных:проектирование.Практикум/Н.П. Стружкин, В. В. Годин. — Москва: Издательство Юрайт, 2019. — 291 с.
13. Технологии создания компьютерных баз данных: пособие / Н.Ф. Богданова. - Минск: ИВЦ Минфина, 2019. - 88 с.

Дополнительная литература

1. Гаврилов, М.В. Информатика и информационные технологии / М.В. Гаврилов, В.А. Климов; Рецензент Л.В. Кальянов, Н.М. Рыскин. - М.: Юрайт, 2013. - 378 с.

2. Базовые и прикладные информационные технологии: учебник для студентов высших учебных заведений, обучающихся по техническим специальностям / В. А. Гвоздева. - Москва : Форум, Инфра-М, 2014. - 382 с.
3. Гвоздева, В.А. Информатика, автоматизированные информационные технологии и системы: Учебник / В.А. Гвоздева. - М.: ИД ФОРУМ, НИЦ ИНФРА-М, 2013. - 544 с.
2. Голицына, О.Л. Информационные технологии: Учебник / О.Л. Голицына, Н.В. Максимов, Т.Л. Партыка, И.И. Попов. - М.: Форум, ИНФРА-М, 2013. - 608 с.
3. Исаев, Г.Н. Информационные технологии / Г.Н. Исаев. - М.: Омега-Л, 2013. - 464 с.
4. Киселев, Г.М. Информационные технологии в экономике и управлении (эффективная работа в MS Office 2007): Учебное пособие / Г.М. Киселев, Р.В. Бочкова, В.И. Сафонов. - М.: Дашков и К, 2013. - 272 с.
5. Логинов, В.Н. Информационные технологии управления: Учебное пособие / В.Н. Логинов. - М.: КноРус, 2013. - 240 с.
6. Максимов, Н.В. Современные информационные технологии: Учебное пособие / Н.В. Максимов, Т.Л. Партыка, И.И. Попов. - М.: Форум, 2013. - 512 с.
7. Румянцева, Е.Л. Информационные технологии: Учебное пособие / Е.Л. Румянцева, В.В. Слюсарь; Под ред. Л.Г. Гагарина. - М.: ИД ФОРУМ, НИЦ ИНФРА-М, 2013. - 256 с.
8. Синаторов, С.В. Информационные технологии.: Учебное пособие / С.В. Синаторов. - М.: Альфа-М, НИЦ ИНФРА-М, 2013. - 336 с.
9. Советов, Б.Я. Информационные технологии / Б.Я. Советов, В.В. Цехановский. - М.: Юрайт, 2013. - 263 с.
10. Федотова, Е.Л. Информационные технологии и системы: Учебное пособие / Е.Л. Федотова. - М.: ИД ФОРУМ, НИЦ ИНФРА-М, 2013. - 352 с.
11. Черников, Б.В. Информационные технологии управления: Учебник / Б.В. Черников. - М.: ИД ФОРУМ, НИЦ ИНФРА-М, 2013. - 368 с.