**ВВЕДЕНИЕ**

Компьютерные справочно-правовые системы появились ещё во второй половине 60-х годов прошлого века по мере развития современных технологий и систем телекоммуникаций. Первой электронной картотекой для компьютерного поиска информации стала бельгийская система, заработавшая в 1967 году. Система была создана совместными усилиями университетов Бельгии и Бельгийского союза адвокатов и нотариусов. Разработка первой справочной правовой системы началась в том же 1967 году. В отличие от электронного каталога американская система была полнотекстовой, то есть позволяла не только находить необходимые документы среди сотен тысяч других, но и работать с самими текстами, а также предоставляла дополнительную информацию к ним, более того, стал возможен поиск по контексту и датам.

Справочная система предназначена для получения пользователем максимально точной ([релевантной](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D0%B5%D0%B2%D0%B0%D0%BD%D1%82%D0%BD%D0%BE%D1%81%D1%82%D1%8C)) информации по интересующей его/её (и ограниченной базой статей) теме. Обычно выбор статьи происходит по иерархии разделов справки. Справочные системы часто комбинируются с [поисковыми](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B8%D1%81%D0%BA%D0%BE%D0%B2%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0), где выборка [релевантных](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D0%B5%D0%B2%D0%B0%D0%BD%D1%82%D0%BD%D0%BE%D1%81%D1%82%D1%8C) статей определяется по заданным ключевым словам или (при [полнотекстовом поиске](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%BD%D0%BE%D1%82%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D0%BF%D0%BE%D0%B8%D1%81%D0%BA)) частью предложения.

Другой тип справочных — это вопросно-ответные системы, дающие один краткий ответ на поставленный вопрос. Примером может служить электронный телефонный справочник. Также существуют телефонные справочно-сервисные службы, сотрудники которых дают абонентам ответы на различные вопросы и оказывают различные услуги.

Все вышесказанное обуславливает актуальность выбранной мною темы: «Информационно-справочная система».

Темой курсового проекта является информационно-справочная система программа «Тригонометрия». Информационно-справочная система представляет собой справочник содержащий параграфы и задания к ним. Программа позволяет быстро переключаться между параграфами. Искать нужную нам информацию, а также способствует быстрому и постепенному изучению материала. А также программа имеет функцию вывода параграфа на печать.

**1 ПОСТАНОВКА ЗАДАЧИ**

**1.1 Исследование предметной области**

Тригонометрия — раздел математики, в котором изучаются тригонометрические функции и их использование в геометрии . Данный термин впервые появился в 1595 г. как название книги немецкого математика сама наука ещё в глубокой древности использовалась для расчётов в астрономии, архитектуре и геодезии.

Тригонометрические вычисления применяются практически во всех областях [геометрии](https://ru.wikipedia.org/wiki/%D0%93%D0%B5%D0%BE%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%8F), [физики](https://ru.wikipedia.org/wiki/%D0%A4%D0%B8%D0%B7%D0%B8%D0%BA%D0%B0) и [инженерного дела](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D0%B6%D0%B5%D0%BD%D0%B5%D1%80%D0%BD%D0%BE%D0%B5_%D0%B4%D0%B5%D0%BB%D0%BE).

Существует множество областей, в которых применяются тригонометрия и [тригонометрические функции](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B8%D0%B3%D0%BE%D0%BD%D0%BE%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B5_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8). Например, метод [триангуляции](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B8%D0%B0%D0%BD%D0%B3%D1%83%D0%BB%D1%8F%D1%86%D0%B8%D1%8F_(%D0%B3%D0%B5%D0%BE%D0%B4%D0%B5%D0%B7%D0%B8%D1%8F)) используется в [астрономии](https://ru.wikipedia.org/wiki/%D0%90%D1%81%D1%82%D1%80%D0%BE%D0%BD%D0%BE%D0%BC) для измерения расстояния до ближайших звезд, в [географии](https://ru.wikipedia.org/wiki/%D0%93%D0%B5%D0%BE%D0%B3%D1%80%D0%B0%D1%84) для измерения расстояний между объектами, а также в [спутниковых навигационных системах](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D1%83%D1%82%D0%BD%D0%B8%D0%BA%D0%BE%D0%B2%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BD%D0%B0%D0%B2%D0%B8%D0%B3%D0%B0%D1%86%D0%B8%D0%B8). [Синус](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%83%D1%81) и [косинус](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D1%81%D0%B8%D0%BD%D1%83%D1%81) имеют фундаментальное значение для теории [периодических функций](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B8%D0%BE%D0%B4%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F), например при описании звуковых и световых волн.

Информационно-справочная система «Тригонометрия» будет содержать в себе 3 таблицы, каждая из таблиц будет использоваться как индивидуальная характеристика, программа будет содержать информацию разделах тригонометрии, которые будут выведены в отдельную таблицу.

Для создания курсового проекта был произведен поиск информации о справочных системах и тригонометрии в целом. Изучив информацию о предметной области принялся за создание курсового проекта, использовав полученные знания о тригонометрии и ее разделах. Информационно-справочная система «Тригонометрия» представляет собой программу, с помощью которой пользователь может изучить теоретическое понятие о тригонометрии, а также проверить свои знания о тригонометрии выполняя задания к параграфам.

Основной задачей информационно-справочной системы функции является проверка уровня знаний и умений учеников, состояния их умственного развития, в изучении степени уровня усвоившегося материала, выявления навыков применения полученных знаний на практике.

Обучающая функция заключается в улучшении знаний и умений, их систематизации. В процессе обучения ученики вспоминают и закрепляют знания по пройденному материалу. При новых обстоятельствах они применяют знания и умения ранее изученного. С помощью развивающей функции появляется возможность стимулировать познавательную активность учащихся, осуществлять развитие творческих способностей.

**1.2 Входная и выходная информация**

Входная информация это информация поступающая в автоматизированную систему в виде документов, сообщений, данных, сигналов, необходимая для выполнения функций автоматизированной системы.

Выходная информация это информация, получаемая в результате выполнения функций автоматизированной системы и выдаваемая на объект ее деятельности, пользователю или в другие системы.

В отдельную таблицу с программе выводятся параграф и задание к нему, что отображается на центральной форме программы.

**2 ВЫЧИСЛИТЕЛЬНАЯ СИСТЕМА**

**2.1 Выбор программных средств и его обоснование**

Для создания базы данных в программе была использована программа «Open Server».

Программа «Open Server» содержит в себе все компоненты для работы с базой данных. Для создания базы данных был использован MySQL – Server который был уже встроен в программу «Open Server». MySQL является решением для малых и средних приложений. Входит в состав серверов WAMP, AppServ, LAMP и в портативные сборки серверов Денвер, XAMPP.

Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы. Гибкость СУБД MySQL обеспечивается поддержкой большого количества типов таблиц. Для удобной работы с MySQL использовали панель управления «PhpMyAdmin».

Для создания обучающе-контролирующей программы использовали программу «Delphi 10.1 Berlin». Delphi 10.1 Berlin - самая полная среда разработки на Object Pascal и библиотека компонентов для разработки кроссплатформенных нативных приложений с поддержкой облачных сервисов и применением IoT. Она предоставляет мощные компоненты VCL для Windows 10 и обеспечивает разработку на FMX для Windows, Mac и мобильных устройств. Delphi 10.1 Berlin - это выбор для корпоративно-ориентированной разработки. Посмотрите на увеличенный объем памяти для крупных проектов, расширенную поддержку нескольких мониторов, улучшенный инспектор объектов и многое другое.

Delphi 10.1 Berlin обеспечивает 5-кратно увеличенную скорость разработки и развертывания на нескольких настольных, мобильных, облачных средах и платформах баз данных, включая 32-разрядные и 64-битные ОС Windows 10. Самый быстрый способ создавать и обновлять интенсивно работающие с данными, распределенные и сильно взаимодействующие приложения с развитым и визуально привлекательным пользовательским интерфейсом.

**2.2 Операционная система**

Windows 8.1 – операционная система семейства [Windows NT](https://ru.wikipedia.org/wiki/Windows_NT) производства корпорации [Microsoft](https://ru.wikipedia.org/wiki/Microsoft), следующая по времени выхода за [Windows 8](https://ru.wikipedia.org/wiki/Windows_8) и перед [Windows 10](https://ru.wikipedia.org/wiki/Windows_10). Предназначена для [рабочих станций](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B1%D0%BE%D1%87%D0%B0%D1%8F_%D1%81%D1%82%D0%B0%D0%BD%D1%86%D0%B8%D1%8F), [персональных компьютеров](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%81%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80) и портативных устройств; версия, предназначенная для решения серверных задач — Windows Server 2012 R2. По сравнению с Windows 8 имеет ряд обновлений и изменений в работе с графическим интерфейсом. Windows 8.1, так же, как и Windows 8, ориентирована на [сенсорные](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D0%BD%D1%81%D0%BE%D1%80%D0%BD%D1%8B%D0%B9_%D1%8D%D0%BA%D1%80%D0%B0%D0%BD) ПК, но не исключает возможности использования на классических [ПК](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%81%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80).

Windows 8.1 является наиболее безопасной ОС, из всех когда-либо созданных Microsoft, особенно с ее впервые встроенной антивирусной защитой. Безопасность было трудно поддерживать в Windows XP, но абсолютно не нужно рассматривать в Windows 8.1.

Windows 8.1 (ранее известная как «Windows Blue») является комплексным обновлением для Windows 8. В новой версии Microsoft не отступается от интерфейса Modern UI (правильное название интерфейса, ранее называемого Metro). Windows 8.1 добавляет в пользовательский интерфейс множество настроек, наряду с некоторыми новыми и обновлёнными приложениям – и всё для того, чтобы пользователи подольше использовали новую среду Windows.

Системные требования:

1. процессор: 1 ГГц или выше с поддержкой PAE, NX и SSE2;
2. озу: 1 ГБ (для 32-разрядной системы) или 2 ГБ (для 64-разрядной системы);
3. свободное место на жестком диске: 16 ГБ (для 32-разрядной системы) или 20 ГБ (для 64-разрядной системы);
4. графическая плата: Microsoft DirectX 9 с драйвером WDDM;
5. разрешение экрана 1024 х 768 пикселей;
6. доступ в Интернет (может взиматься плата);
7. для работы некоторых функций необходима учетная запись Майкрософт.

В Windows 8.1 x64 среди системных требований появилась необходимость поддержки процессором инструкции CMPXCHG16B (CompareExchange128), что повлекло невозможность установки этой версии или обновления до неё на значительном количестве старых ПК (например, с весьма распространёнными процессорами Intel Core 2 ранних степингов на 65нм техпроцессе), на которых Windows 8 устанавливалась успешно. Это требование не касается Windows 8.1 x86.

Дополнительные требования для использования некоторых функций ОС:

1. для сенсорного ввода требуется планшет или монитор с поддержкой мультисенсорной технологии;
2. для доступа к магазину Windows, а также скачивания, запуска и прикрепления приложений требуется активное подключение к интернету и разрешение экрана не менее 1024 x 768 пикселей;
3. для работы некоторых функций требуется учетная запись Майкрософт;
4. требуется доступ к интернету (поставщиком услуг интернета может взиматься плата);
5. для защищенной загрузки необходимо встроенное ПО, которое поддерживает UEFI v2.3.1 Errata B и имеет в базе сигнатур UEFI сертификат центра сертификации Microsoft Windows;
6. для оптимальной работы некоторых игр и программ может потребоваться видеоадаптер, совместимый с DirectX 10 или более поздними версиями;
7. для просмотра DVD-дисков требуется отдельное программное обеспечение;
8. для функции BitLocker To Go требуется USB-устройство флэш-памяти (только для Windows 8.1 Профессиональная);
9. для функции BitLocker требуется доверенный платформенный модуль (TPM) 1.2 или USB-устройство флэш-памяти (только для Windows 8.1 Профессиональная);
10. для клиента Hyper-V требуется 64-разрядная система с поддержкой преобразования адресов второго уровня (SLAT) и дополнительные 2 ГБ ОЗУ (только для Windows 8.1 Профессиональная);
11. для просмотра и записи телепередач в Windows Media Center требуется ТВ-тюнер (только для программы обновления до Windows 8.1 Профессиональная и программы Media Center для Windows 8.1);
12. для использования стандарта Miracast необходим видеоадаптер с поддержкой архитектуры WDDM 1.3 и адаптер Wi-Fi с поддержкой технологии Wi-Fi Direct;
13. для печати с помощью технологии Wi-Fi Direct необходим адаптер Wi-Fi с поддержкой Wi-Fi Direct и устройство, поддерживающее эту технологию;
14. бесплатный контент интернет-телевидения зависит от географического расположения. Некоторые виды контента могут предполагать дополнительную плату (только для программы обновления до Windows 8.1 Профессиональная и программы Media Center для Windows 8.1);
15. для установки 64 - разрядной операционной системы на 64 - разрядном компьютере ваш процессор должен поддерживать инструкции CMPXCHG16b, PrefetchW и LAHF/SAHF;
16. технология InstantGo работает только на компьютерах, поддерживающих режим ожидания с подключением.

Windows 8.1 поставляется с новой версией DirectX 11.2. Это обновление нельзя назвать крупным, однако ряд важных нововведений, которые смогут задействовать создатели игр и мультимедийных приложений в нём, конечно, есть.

Windows 8.1 приблизит новое поколение ОС от Microsoft к пользователю классических компьютеров. Плиточные приложения появляются на панели задач, у них есть выезжающий заголовок, не исключено, что к финальной версии их вообще станет возможно запускать в окнах. Этим компания окончательно вернет «главенство» рабочего стола над стартовым экраном. Последний, кстати, получил контекстные меню, ориентированные на обычную мышку, а заодно, наконец-то, предложил быстрый и наглядный способ перезагрузки и выключения компьютера.

**2.3 Система программирования**

Информационно-справочная система была разработана в среде «RAD Studio Delphi 10.1 Berlin». RAD Studio Delphi 10.1 Berlin – самая полная среда разработки на Object Pascal и библиотека компонентов для разработки кроссплатформенных нативных приложений с поддержкой облачных сервисов и применением IoT. Она предоставляет мощные компоненты VCL для Windows 10 и обеспечивает разработку на FMX для Windows, Mac и мобильных устройств.

Delphi 10.1 Berlin – это выбор для корпоративно-ориентированной разработки. Посмотрите на увеличенный объем памяти для крупных проектов, расширенную поддержку нескольких мониторов, улучшенный инспектор объектов и многое другое. Delphi обеспечивает 5-кратно увеличенную скорость разработки и развертывания на нескольких настольных, мобильных, облачных средах и платформах баз данных, включая 32-разрядные и 64-битные ОС Windows 10.

RAD Studio **Delphi 10.1 Berlin** - самый быстрый способ создавать и обновлять интенсивно работающие с данными, распределенные и сильно взаимодействующие приложения с развитым и визуально привлекательным пользовательским интерфейсом для Windows 10, Mac, мобильных устройств, IoT и других платформ. Вы сможете с легкостью обновлять VCL- и FMX-приложения для Windows 10, используя новые элементы управления и VCL-стили для Windows, а также компоненты служб универсальной платформы Windows. В редакцию Delphi Architect встроено средство визуального моделирования данных ER/Studio (редакция Developer), позволяющее разработчикам наиболее эффективно использовать корпоративные структуры и базы данных.

Основные новые возможности RAD Studio Delphi 10.1 Berlin:

* изменился механизм установки RAD Studio (и до установки);
* поддержка Android 6.0 (API Level 23);
* улучшения в Android Service;
* улучшения в FireDAC для DB2, MongoDB, Firebird, Oracle, PostgreSQL, SQLite;
* новый компонент TAddressBook, который позволяет получить доступ к адресной книге на платформе Android и iOS;
* улучшен Style Designer в FireMonkey;
* улучшен механизм предварительного просмотра интерфейса для различных устройств;
* улучшен механизм поддержки IoT (Интернет вещей);
* новый элемент ListView;
* улучшен IFMXExtendedClipboardService для работы с буфером обмена;
* улучшен редактор списка изображений в компоненте FMX.TImageList;
* поддержка High DPI для Windows-приложений;
* поддержка EMS сервера Apache;
* новые возможности EMS;
* улучшен BeaconFence для работы с маячками;
* улучшена производительность с Bluetooth LE, Beacon;
* поддержка асинхронных HTTP запросов;
* улучшен класс TStrings, появились новые методы;
* улучшен TMemIniFile;
* улучшена производительность с Generics.

Таким образом RAD Studio **Delphi 10.1 Berlin** - самый быстрый способ создавать и обновлять интенсивно работающие с данными, распределенные и сильно взаимодействующие приложения с развитым и визуально привлекательным пользовательским интерфейсом для Windows 10, Mac, мобильных устройств, IoT и других платформ.

**3 ПРОЕКТИРОВАНИЕ ПРОГРАММ**

**3.1 Описание задачи в ключевых абстракциях предметной области**

Объектно-ориентированное программирование представляет собой способ программирования структурного типа, который позволяет создавать модульные программы с предоставлением данных, на определенном уровне абстракции. Оно является естественной эволюцией более ранних нововведений в разработке языков программирования.

Объектно-ориентированное программирование является более структурным, чем все предыдущие разработки, касающиеся структурного программирования. Оно также является более модульным и более абстрактным, чем предыдущие попытки абстрагирования данных и переноса деталей программирования на внутренний уровень.

Основным понятием объектно-ориентированного программирования является понятие объекта. Объект – это высший уровень абстракции данных, который объединяет в себе данные и методы. Данными, то есть полями объекта могут быть любые структурные данные, в том числе и другие объекты.

Инкапсуляция – это механизм, который объединяет данные и код, манипулирующий этими данными, а также защищает и то, и другое от внешнего вмешательства или неправильного использования. В объектно-ориентированном программировании код и данные могут быть объединены вместе; в этом случае говорят, что создаётся так называемый «чёрный ящик». Когда коды и данные объединяются таким способом, создаётся объект. Другими словами, объект – это то, что поддерживает инкапсуляцию.

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

Наследование является важным, поскольку оно позволяет поддерживать концепцию иерархии классов. Применение иерархии классов делает управляемыми большие потоки информации. Без использования иерархии классов, для каждого объекта пришлось бы задать все характеристики, которые бы исчерпывающи его определяли. Однако при использовании наследования можно описать объект путём определения того общего класса, к которому он относится, с теми специальными чертами, которые делают объект уникальным. Наследование играет важную роль в объектно-ориентированном программировании.

Полиморфизм – это свойство, которое позволяет одно и то же имя использовать для решения двух или более схожих, но технически разных задач. Целью полиморфизма, применительно к объектно-ориентированному программированию, является использование одного имени для задания общих для класса действий. Выполнение каждого конкретного действия будет определяться типом данных. В более общем смысле, концепцией полиморфизма является идея один интерфейс для группы близких по смыслу действий.

Преимуществом полиморфизма является то, что он помогает снижать сложность программ, разрешая использование того же интерфейса для задания единого класса действий. Выбор же конкретного действия, в зависимости от ситуации, возлагается на компилятор. Программисту, не нужно делать этот выбор самому. Нужно только помнить и использовать общий интерфейс.

UML является языком широкого профиля, это – открытый стандарт, использующий графические обозначения для создания абстрактной модели системы, называемой UML- моделью. UML был создан для определения, визуализации, проектирования, и документирования, в основном программных систем. UML не является языком программирования, но на основании UML- моделей возможна генерация кода.

Язык UML одновременно является простым и мощным средством моделирования, которой может быть эффективно использован для построения концептуальных, логических и графических моделей сложных систем самого различного целевого назначения. Этот язык вобрал в себя наилучшее качества методов программной инженерии, которое с успехом использовались на протяжении последних лет при моделировании больших и сложных систем.

Язык UML основан на некотором числе базовых понятий, которые могут быть изучены и применены большинством программистов и разработчиков, знакомых с методами объектно-ориентированного анализа и проектирования. При этом базовые понятия могут комбинироваться и расширяться таким образом, что специалисты объектного моделирования получают возможность самостоятельно разрабатывать модели больших и сложных систем в самых различных областях приложений.

Конструктивное использование языка UML основывается на понимании общих принципов моделирования сложных систем и особенностей процесса, объектно-ориентированного проектирования в частности. Выбор выразительных средств для построения моделей сложных систем предопределяет те задачи, которые могут быть решены с использованием данных моделей.

При этом одним из основных принципов построения моделей сложных систем является принцип абстрагирования, который предписывает включать в модель только те аспекты проектируемой системы, которые имеют непосредственное отношение к выполнению системой своих функций или своего целевого предназначения. При этом все второстепенные детали опускаются, чтобы чрезмерно не усложнять процесс анализа и исследования полученной модели.

Другим принципом построения моделей сложных систем является принцип многомодельности. Этот принцип представляет собой утверждение о том, что никакая единственная модель не может с достаточной степенью адекватности описывать различные аспекты сложной системы. Применительно к методологии объектно-ориентированного проектирования это означает, что достаточно полная модель сложной системы допускает некоторое число взаимосвязанных представлений (views), каждое из которых адекватно отражает некоторый аспект поведения или структуры системы.

При этом наиболее общими представлениями сложной системы принято считать статическое и динамическое представления, которые в свою очередь могут подразделяться на другие более частные представления.) феномен сложной системы как раз и состоит в том, что никакое ее единственное представление не является достаточным для адекватного выражения всех особенностей моделируемой системы.

Язык [UML](http://www.znannya.org/?view=concept:1709) предназначен для решения следующих задач:

Предоставить в распоряжение пользователей легко воспринимаемый и выразительный язык визуального моделирования, специально предназначенный для разработки и документирования моделей сложных систем самого различного целевого назначения. Речь идет о том, что важным фактором дальнейшего развития и повсеместного использования методологии ООАП является интуитивная ясность и понятность основных конструкций соответствующего языка моделирования. Язык [UML](http://www.znannya.org/?view=concept:1709) включает в себя не только абстрактные конструкции для представления метамоделей систем, но и целый ряд конкретных понятий, имеющих вполне определенную семантику. Это позволяет языку UML одновременно достичь не только универсальности представления моделей для самых различных приложений, но и возможности описания достаточно тонких деталей реализации этих моделей применительно к конкретным системам;

Снабдить исходные понятия языка [UML](http://www.znannya.org/?view=concept:1709) возможностью расширения и специализации для более точного представления моделей систем в конкретной предметной области. Хотя язык UML является формальным языком - спецификаций, формальность его описания отличается от синтаксиса как традиционных формально-логических языков, так и известных языков программирования. Разработчики из [OMG](http://www.znannya.org/?view=concept:1776) предполагают, что язык [UML](http://www.znannya.org/?view=concept:1709) как никакой другой может быть приспособлен для конкретных предметных областей. Это становится возможным по той причине, что в самом описании языка UML заложен механизм расширения базовых понятий, который является самостоятельным элементом языка и имеет собственное описание в форме правил расширения;

Описание языка UML должно поддерживать такую спецификацию моделей, которая не зависит от конкретных языков программирования и инструментальных средств проектирования программных систем. Речь идет о том, что ни одна из конструкций языка UML не должна зависеть от особенностей ее реализации в известных языках программирования. То есть, хотя отдельные понятия языка UML и связаны с последними очень тесно, их жесткая интерпретация в форме каких бы то ни было конструкций программирования не может быть признана корректной. Другими словами, разработчики из OMG считают необходимым свойством языка UML его контекстно-программную независимость;

Описание языка UML должно включать в себя семантический базис для понимания общих особенностей ООАП. Говоря об этой особенности, имеют в виду самодостаточность языка UML для понимания не только его базовых конструкций, но что не менее важно - понимания общих принципов ООАП. В этой связи необходимо отметить, что поскольку язык UML не является языком программирования, а служит средством для решения задач объектно-ориентированного моделирования систем, описание языка UML должно по возможности включать в себя все необходимые понятия для ООАП. Без этого свойства язык UML может оказаться бесполезным и невостребованным большинством пользователей, которые не знакомы с проблематикой ООАП сложных систем;

Поощрять развитие рынка объектных инструментальных средств. Более 800 ведущих производителей программных и аппаратных средств, усилия которых сосредоточены в рамках [OMG](http://www.znannya.org/?view=concept:1776), видят перспективы развития современных информационных технологий и основу своего коммерческого успеха в широком продвижении на рынок инструментальных средств, поддерживающих объектные технологии. Говоря же об объектных технологиях, разработчики из OMG имеют в виду, прежде всего, совокупность технологических решений CORBA и [UML](http://www.znannya.org/?view=concept:1709). С этой точки зрения языку UML отводится роль базового средства для описания и документирования различных объектных [компонентов](http://www.znannya.org/?view=concept:1735) CORBA;

Способствовать распространению объектных технологий и соответствующих понятий ООАП. Эта задача тесно связана с предыдущей и имеет с ней много общего. Если исключить из рассмотрения рекламные заявления разработчиков об исключительной гибкости и мощности языка [UML](http://www.znannya.org/?view=concept:1709), а попытаться составить объективную картину возможностей этого языка, то можно прийти к следующему заключению. Следует признать, что усилия достаточно большой группы разработчиков были направлены на интеграцию в рамках языка UML многих известных техник визуального моделирования, которые успешно зарекомендовали себя на практике. Хотя это привело к усложнению языка UML по сравнению с известными нотациями структурного системного анализа, платой за сложность являются действительно высокая гибкость и изобразительные возможности уже первых версий языка UML. В свою очередь, использование языка UML для решения всевозможных практических задач будет только способствовать его дальнейшему совершенствованию, а значит и дальнейшему развитию объектных технологий и практики ООАП;

Интегрировать в себя новейшие и наилучшие достижения практики ООАП. Язык [UML](http://www.znannya.org/?view=concept:1709) непрерывно совершенствуется разработчиками, и основой этой работы является его дальнейшая интеграция с современными модельными технологиями. При этом различные методы системного моделирования получают свое прикладное осмысление в рамках ООАП. В последующем эти методы могут быть включены в состав языка UML в форме дополнительных базовых понятий, наиболее адекватно и полно отражающие наилучшие достижения практики ООАП.

Использование UML не ограничивается моделированием программного обеспечения. Его также используют для моделирования бизнес-процессов, системного проектирования и отображения организационных структур.

**3.2 Описание интерфейсов классов**

Для описания интерфейса классов опишем основные процедуры, реализованные в проекте:

* + 1. procedure TForm10.Timer1Timer(Sender: TObject) – плавное загрузка формы;
    2. procedure FormCreate(Sender: TObject) – при создании формы выгружаются изображения для плавного перехода;
    3. procedure TForm1.Image1Click (Sender: TObject) – переключение параграфа назад;
    4. procedure TForm1.Image2Click(Sender: TObject) – переключение параграфа вперёд;
    5. procedure TForm1.Image3Click(Sender: TObject) – открытие формы для добавления параграфа и задания к нему;
    6. TForm1.Image4Click(Sender: TObject) – переход на форму изменения содержимого параграфа и задания к нему;
    7. procedure TForm1.Image5Click(Sender: TObject) – переход на форму удаления содержимого параграфа и задания к нему;
    8. procedure TForm2.Image1Click(Sender: TObject) – кнопка по нажатию на которую добавляется параграф и задание к нему ;
    9. procedure TForm2.Image2Click(Sender: TObject) – кнопка по нажатию на которую осуществляется переход обратно к родительской форме;
    10. procedure TForm2.Memo1Click(Sender: TObject) – по нажатию на первое поле редактирования оно очищается;
    11. procedure TForm2.Memo2Click(Sender: TObject) – по нажатию на второе поле редактирования оно очищается;
    12. procedure TForm3.Image2Click(Sender: TObject) – кнопка по нажатию на которую осуществляется переход обратно к родительской форме;
    13. procedure TForm3.Image1Click(Sender: TObject) – кнопка по нажатию на которую удаляется параграф и задание к нему;
    14. procedure TForm4.Image1Click(Sender: TObject) – кнопка по нажатию на которую осуществляется переход обратно к родительской форме;
    15. procedure TForm4.Image2Click(Sender: TObject) – кнопка по нажатию на которую изменяется параграф и задание к нему;

**4 ОПИСАНИЕ ПРОГРАММЫ**

Программное средство функционирует в операционной среде Windows 7 и более новых версиях Windows. Общий размер всех используемых файлов составляет более 40 Мб.

Исходный текст программы находится в файлах со следующим разрешением:

1. PAS – это текстовый файл, содержащий описание связанным с формой модулем (процедур и функций программной логики функционирования окна приложения в синтаксисе Object Pascal). Для каждой формы, включаемой в проект, создается отдельный модуль;
2. DFM – это файл формы содержит значения свойств формы и ее компонентов. Файл формы связывает графическое представление формы с обработчиками сообщений и используется методом TApplication.CreateForm для начального создания формы;
3. RES – это двоичный файл, содержащий все необходимые для проекта ресурсы, такие как, например, пиктограммы, графические изображения, курсоры мыши или строки. Этот файл создает и модифицирует Delphi. Он не должен изменяться или создаваться пользователем;
4. DCU – это двоичный файл модуля. Имеет одноимённое название с модулем;
5. DPR – это файл, который содержит главный программный блок. Файл проекта подключает все используемые программные модули и содержит операторы для запуска приложения. Этот файл среда Delphi создает и контролирует сама;
6. IDENTCACHE – это кэшированные привязки к файлам;
7. DPROJ.LOCAL – это исходный код проекта (xml);
8. EXE – это откомпилированная программа.

**5 ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ**

**5.1 Цель испытаний**

Тестирование программного обеспечения это процесс выявления ошибок в программном обеспечении (ПО). Существующие на сегодняшний день методы тестирования ПО не позволяют однозначно и полностью устранить все дефекты и ошибки и установить корректность функционирования анализируемой программы особенно в закрытых частных программах. Поэтому все существующие методы тестирования действуют в рамках формального процесса проверки исследуемого или разрабатываемого ПО.

Это оценка разрабатываемого программного обеспечения/продукта, чтобы проверить его возможности, способности и соответствие ожидаемым результатам. Существуют различные типы методов, используемые в области тестирования и обеспечения качества о них и пойдет речь в данной статье. Тестирование программного обеспечения является неотъемлемой частью цикла разработки программного обеспечения.

Такой процесс формальной проверки или верификации может доказать, что дефекты отсутствуют, с точки зрения используемого метода. (То есть нет никакой возможности точно установить или гарантировать отсутствие дефектов в программном продукте с учётом человеческого фактора, присутствующего на всех этапах жизненного цикла ПО).

Существует множество подходов к решению задачи тестирования и верификации ПО, но эффективное тестирование сложных программных продуктов – это процесс в высшей степени творческий, не сводящийся к следованию строгим и чётким процедурам или созданию таковых.

Тестирование проводится с целью обеспечить качество разрабатываемого программного продукта Основным параметром качества программы является надёжность. Надёжность определяется как вероятность его работы без отказов в течении определённого периода времени, рассчитанная с учётом стоимости для пользователя каждого отказа. Отказ программного обеспечения - это проявление ошибки в нём. Отсюда тестирование ПО – это процесс выполнения программы с целью обнаружения в ней ошибок.

Тестирование программного обеспечения – это попытка определить, выполняет ли программа то, что от неё ожидают. Качество программных средств можно определить, как совокупную характеристику исследуемого программного обеспечения с учётом следующих составляющих:

1. [надёжность](http://dic.academic.ru/dic.nsf/ruwiki/7191);
2. с[опровождаемость](http://dic.academic.ru/dic.nsf/ruwiki/139790);
3. практичность;
4. эффективность;
5. мобильность;
6. функциональность.

Так как, тестирование проводится с целью обеспечить качество разрабатываемого программного продукта. Основным параметром качества программы является надёжность. Данная программа полностью протестирована и она соответствует не только одному параметру надежности, а всем необходимым характеристикам.

**5.2 Функциональное тестирование**

При запуске программы появилось окно, где написано название темы и кто разработал программу, как показано на рисунке 1.

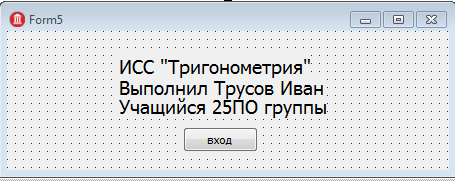


Рисунок 1 – Заставка

После загрузки заставки открылось окно просмотра содержимого справочника, которое показано на рисунке 2.

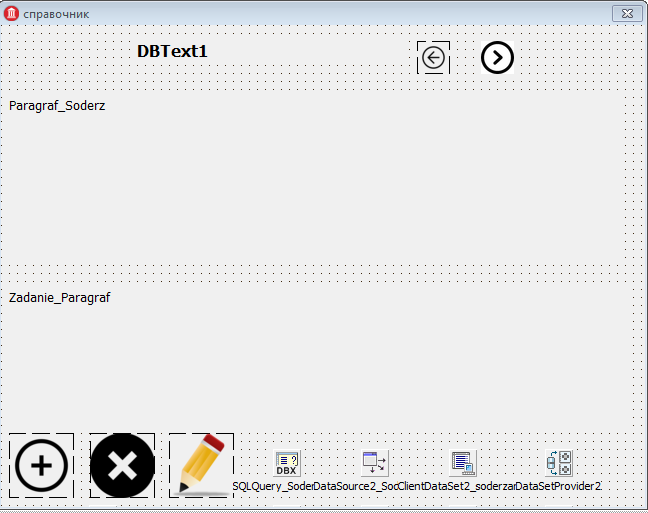


Рисунок 2 – Справочник

Для редактирования содержимого парграфа, выбрали на форме кнопку «Изменить», в данном окне можно изменять данные, как показано на рисунке 3

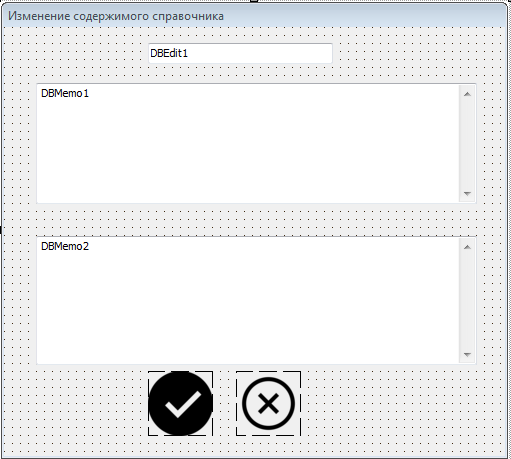
. 

Рисунок 3–Функция изменения данных

Далее рассмотрели функцию добавления содержимого. Для добавления использовали ввод новых данных в поля ввода, в которые ввели содержимое текста параграфа, имя параграфа и задание к нему. Для добавления параграфа нажали на кнопку «Добавить», как показано на рисунке 4.

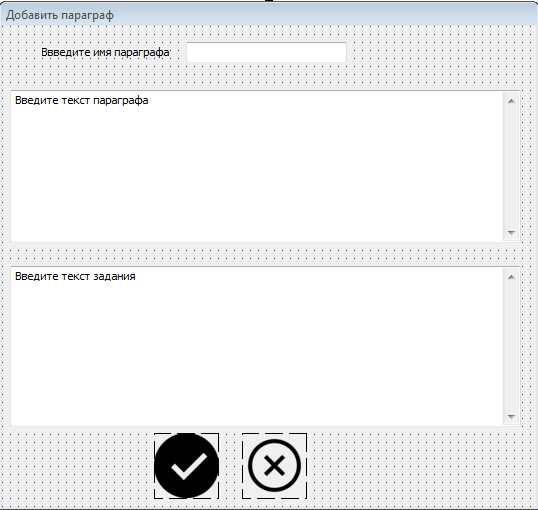


Рисунок 4 – Добавление данных.

Для удаления параграфа выбрали нужный параграф и нажали на кнопку удаления, после чего увидели окно, в котором отобразился номер удаляемого параграфа и кнопка для его удаления, как показано на рисунке 5.

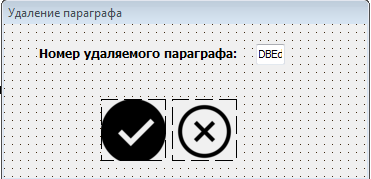


Рисунок 5 –Удаление данных

Для выхода из программы нажали на кнопку закрытия окна приложения, как показано на рисунке 6.



Рисунок 6–Выход

**5.3 Полное редактирование**

Рассмотрели полное тестирование на примере функции редактирования. Для редактирования содержимого параграфа, выбрали на форме кнопку «Изменить», как показано на рисунке 7.

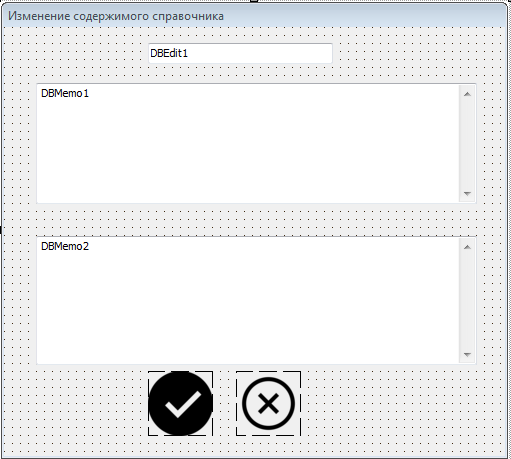


Рисунок 7 – Редактирование базы данных

Если не заполнили одно из полей, произошла ошибка, как показано на рисунке 8.

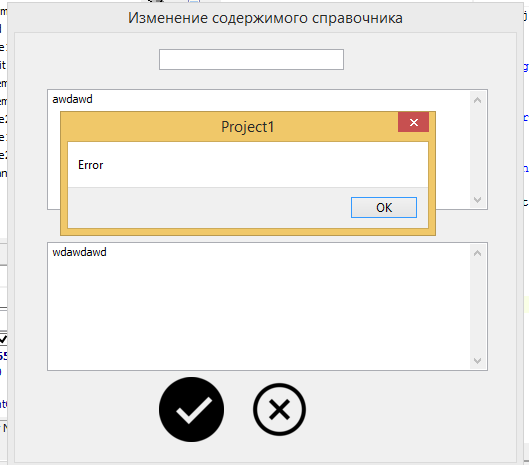


Рисунок 8- Сообщение об ошибке

После нажатия кнопки «ОК» имя параграфа и его содержимое изменилось, как показано на рисунке 9-10.

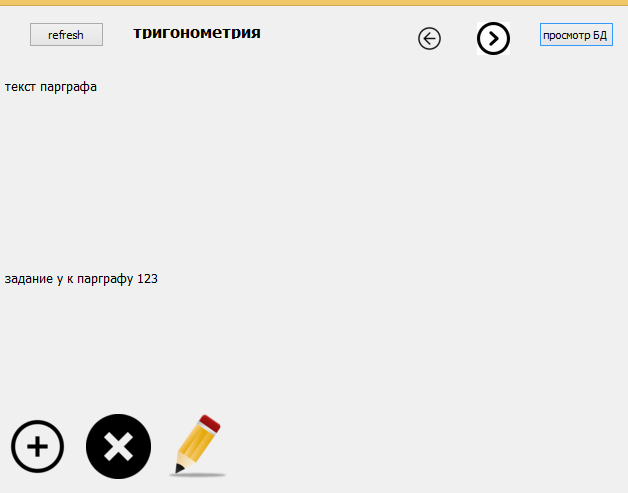


Рисунок 9 – Текст параграфа до изменения

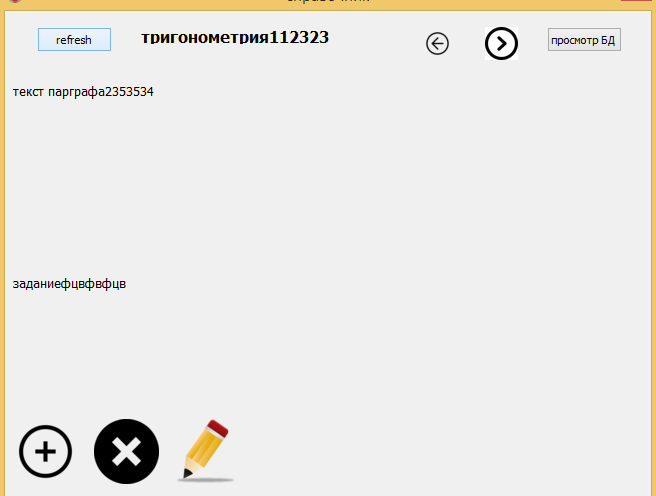


Рисунок 10 – Текст параграфа после изменения

Для изменения названия параграфа ввели новое имя в поле «Название параграфа» имя параграфа изменилось с «тригонометрия» на «тригонометрия(изменён)», как показано на рисунке 11-12.

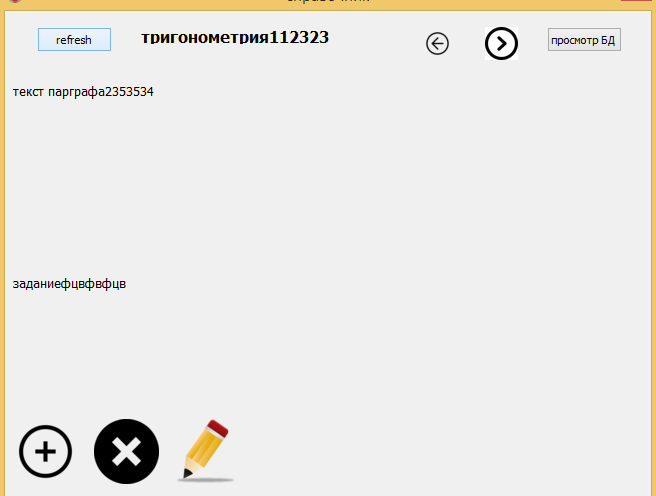


Рисунок 11 – Название параграфа до изменения

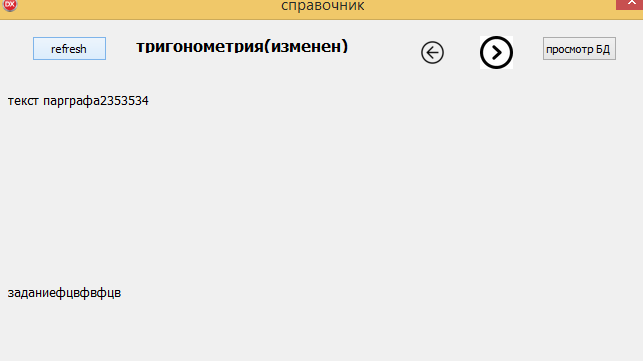


Рисунок 12 – Название параграфа после изменения

**6 ОПИСАНИЕ ПРИМЕНЕНИЯ**

Основной целью данной программы является обнаружение достигнутых успехов учеников, в указании правильных путей совершенствования, получения более углубленных знаний и умений, для того, чтобы появлялись условия для будущего включения учащихся в активную творческую деятельность, умениями и навыками, предусмотренных программой по предмету. Данная цель формирует у учащихся выработку таких качеств личности, как чувство ответственности за проделанную работу, реализация и проявление новых идей. При воплощении в жизнь всех вышеперечисленных целей данной программы мы можем говорить о том, что она выполняет следующие функции:

1. обучающую;
2. воспитывающую;
3. диагностическую;
4. развивающую.

Поэтому пользователь данной программы может использовать её для изучения нового материала, освоения новые технологий и поиска новой информации для всестороннего развития.

**6.1 Назначение и условия применения программы**

Назначение программы заключается в обучении и информировании учащихся в каком-либо учебном заведении о тригонометрии, данная программа может быть использована на сайте, в учебном процессе. Основной задачей данной программы является проверка уровня знаний и умений пользователя, состояние их умственного развития, в изучении степени уровня усвоившегося материала, выявления навыков применения полученных знаний на практике.

**6.2 Инсталляция**

Для установки программы требуется скопировать базу данных на используемый сервер. Далее необходимо установить среду разработки RAD Studio.После чего в ней необходимо открыть сохраненный проект. Далее установить настройки в соответствии с требования разработчика ПО. Затем нужно создать исполняемый файл программы(EXE) и запустить его. Если возникли неполадки в процессе установки ПО настоятельно рекомендуем проверить все вышесказанное еще раз или обратиться к разработчику ПО.

**6.3 Выполнение программ**

Информационно-справочная система представляет собой программу, с помощью которой пользователь может просматривать, добавлять, удалять разделы справочника. Находить нужные по характеристикам данные и многое другое.

С помощью проверки ученики могут выделить основное, главное в пройденном материале, закрепить полученные знания и навыки, в дальнейшем не допуская ошибок при решении.

**ЗАКЛЮЧЕНИЕ**

Темой курсового проекта является информационно-справочная система «Тригонометрия».

Информационно-справочная система собой программу, с помощью которой пользователь может изучить теоретическое понятие о тригонометрии, а также проверить свои знания о тригонометрии решив задания приложенные к параграфу.

Курсовой проект был создан с помощью программы «Open Server». Программа «Open Server» содержит в себе все компоненты для работы с базой данных. Для создания базы данных был использован MySQL – Server который был уже встроен в программу «Open Server». Для удобной работы с MySQL использовали панель управления «PhpMyAdmin», а также использовали «Delphi 10.1 Berlin». Delphi 10.1 Berlin - самая полная среда разработки на Object Pascal и библиотека компонентов для разработки кроссплатформенных нативных приложений с поддержкой облачных сервисов и применением IoT.

Программа была установлена на операционной системе Windows 8.1. Windows 8.1 добавляет в пользовательский интерфейс множество настроек, наряду с некоторыми новыми и обновлёнными приложениям – и всё для того, чтобы пользователи подольше использовали новую среду Windows.

Достоинства этой программы в том, она проста в использовании, может использоваться во многих операционных системах начиная с Windows XP. Недостатки данной программы состоят в том, что данная программа не может использоваться в операционных системах ниже Windows XP, занимает большой объём памяти.

Данная программа может использоваться в учебных заведениях, сайтах, контролях знаний.

Программа создана для увеличения уровня знаний и умений учеников, состояния их умственного развития, в изучении степени уровня усвоившегося материала, выявления навыков применения полученных знаний на практике.

Используются ИСС повсеместно начиная от ОС ЭВМ заканчивая государственными структурами и научно-исследовательскими институтами.

**ЛИТЕРАТУРА**

1. Андреев, А.Г. Microsoft Windows 2000 Professional / А.Г. Андреев. – Санкт-Петербург: БХВ – Петербург,2004. – 325 с.
2. Большакова, Е.С. Информатика общества / Е.С. Большакова. – Санкт-Петербург: Питер, 2003. – 654 с.
3. Волков, В.В. Понятный самоучитель работы в Microsoft Windows / В.В. Волков. – Москва: Вильямс, 2006. – 435 с.
4. Культин, Н.Б. Самоучитель C++ Builder / Н.Б Культин. – Санкт-Петербург: БХВ-Петербург, 2004. – 780 с.
5. Орлов, С.А. Технология разработки программного обеспечения / С.А. Орлов. – Санкт-Петербург: Питер, 2003. – 320 с.
6. Пахомов, Б.И. C/C++ и Borland C++ Builder для начинающих / Б.И. Пахомов. – Санкт-Петербург: Питер, 2005. – 605 с.
7. Фигурнов, В.Э. IBMPC для пользователя. Краткий курс / В.Э. Фигурнов. – Москва: Инфра – М, 2003. – 360 с.
8. Фаулер, М. UML в кратком изложении. Применение стандартного языка объектного моделирования / М.Фаулер, К. Скотт. – Москва: Мир, 2000. – 200 с.
9. Фленов, М.Е. Искусство программирования игр на C++ / М.Е. Фленов. – Санкт-Петербург: Питер, 2006. – 290 с.

10) Язык UML. Руководство пользователя [Электронный ресурс] / Электронные, текстовые и графические данные – Москва, 2002. – 300 с.

**ПРИЛОЖЕНИЯ**

**А) (текст программы)**

**Текст программы**

unit Unit1;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Data.DB, Data.DBXMySQL, Data.FMTBcd,

Datasnap.Provider, Datasnap.DBClient, Data.SqlExpr, Vcl.Grids, Vcl.DBGrids,

Vcl.StdCtrls, Vcl.DBCtrls, Vcl.Imaging.pngimage, Vcl.ExtCtrls;

type

TForm1 = class(TForm)

SQLConnection1: TSQLConnection;

SQLQuery1: TSQLQuery;

DataSource1: TDataSource;

ClientDataSet1: TClientDataSet;

DataSetProvider1: TDataSetProvider;

Zadanie\_Paragraf: TDBText;

Image1: TImage;

Image2: TImage;

Image3: TImage;

Image4: TImage;

Image5: TImage;

DBText1: TDBText;

SQLQuery\_Soderzanie: TSQLQuery;

DataSource2\_Soderzanie: TDataSource;

ClientDataSet2\_soderzanie: TClientDataSet;

DataSetProvider2: TDataSetProvider;

SQLQuery2\_Zadanie: TSQLQuery;

DataSource2\_Zadanie: TDataSource;

ClientDataSet2\_Zadanie: TClientDataSet;

DataSetProvider3\_Zadanie: TDataSetProvider;

SQLQuery22: TSQLQuery;

DataSource22: TDataSource;

ClientDataSet22: TClientDataSet;

DataSetProvider22: TDataSetProvider;

Paragraf\_Soderz: TDBText;

procedure FormCreate(Sender: TObject);

procedure Image2Click(Sender: TObject);

procedure Image1Click(Sender: TObject);

procedure Image3Click(Sender: TObject);

procedure Image5Click(Sender: TObject);

procedure Image4Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

i,n:integer;

implementation

{$R \*.dfm}

uses Unit2, Unit3, Unit4;

procedure TForm1.FormCreate(Sender: TObject);

begin

i:=0;

end;

procedure TForm1.Image1Click(Sender: TObject);

begin

dec(i);

SQLQuery1.Close;

SQLQuery1.SQL.Clear;

SQLQuery1.SQL.Add('select name from paragraf where id\_paragraf='+Inttostr(i));

SQLQuery1.Open;

ClientDataSet1.Active := false;

ClientDataSet1.Active := true;

SQLQuery\_Soderzanie.Close;

SQLQuery\_Soderzanie.SQL.Clear;

SQLQuery\_Soderzanie.SQL.Add('select soderzanie from soderzanie where id\_soderzanie='+Inttostr(i));

SQLQuery\_Soderzanie.Open;

ClientDataSet2\_soderzanie.Active := false;

ClientDataSet2\_soderzanie.Active := true;

SQLQuery2\_Zadanie.Close;

SQLQuery2\_Zadanie.SQL.Clear;

SQLQuery2\_Zadanie.SQL.Add('select задача from zadanie where id\_zadanie='+Inttostr(i));

SQLQuery2\_Zadanie.Open;

ClientDataSet2\_Zadanie.Active := false;

ClientDataSet2\_Zadanie.Active := true;

SQLQuery22.Close;

SQLQuery22.SQL.Clear;

SQLQuery22.SQL.Add('select id\_paragraf from paragraf where id\_paragraf='+Inttostr(i));

SQLQuery22.Open;

ClientDataSet22.Active := false;

ClientDataSet22.Active := true;

end;

procedure TForm1.Image2Click(Sender: TObject);

begin

inc(i);

SQLQuery1.Close;

SQLQuery1.SQL.Clear;

SQLQuery1.SQL.Add('select name from paragraf where id\_paragraf='+Inttostr(i));

SQLQuery1.Open;

ClientDataSet1.Active := false;

ClientDataSet1.Active := true;

SQLQuery\_Soderzanie.Close;

SQLQuery\_Soderzanie.SQL.Clear;

SQLQuery\_Soderzanie.SQL.Add('select soderzanie from soderzanie where id\_soderzanie='+Inttostr(i));

SQLQuery\_Soderzanie.Open;

ClientDataSet2\_soderzanie.Active := false;

ClientDataSet2\_soderzanie.Active := true;

SQLQuery2\_Zadanie.Close;

SQLQuery2\_Zadanie.SQL.Clear;

SQLQuery2\_Zadanie.SQL.Add('select задача from zadanie where id\_zadanie='+Inttostr(i));

SQLQuery2\_Zadanie.Open;

ClientDataSet2\_Zadanie.Active := false;

ClientDataSet2\_Zadanie.Active := true;

SQLQuery22.Close;

SQLQuery22.SQL.Clear;

SQLQuery22.SQL.Add('select id\_paragraf from paragraf where id\_paragraf='+Inttostr(i));

SQLQuery22.Open;

ClientDataSet22.Active := false;

ClientDataSet22.Active := true;

end;

procedure TForm1.Image3Click(Sender: TObject);

begin

form2.show;

form1.Hide;

end;

procedure TForm1.Image4Click(Sender: TObject);

begin

form4.show;

form1.hide;

end;

procedure TForm1.Image5Click(Sender: TObject);

begin

form3.Show;

form1.Hide;

end;

end.

unit Unit2;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Imaging.pngimage,

Vcl.ExtCtrls, Data.DBXMySQL, Data.FMTBcd, Datasnap.Provider,

Datasnap.DBClient, Data.DB, Data.SqlExpr;

type

TForm2 = class(TForm)

Memo1: TMemo;

Memo2: TMemo;

Edit1: TEdit;

Label1: TLabel;

Image1: TImage;

SQLConnection1: TSQLConnection;

SQLQuery1: TSQLQuery;

DataSource1: TDataSource;

ClientDataSet1: TClientDataSet;

DataSetProvider1: TDataSetProvider;

SQLQuery2: TSQLQuery;

DataSource2: TDataSource;

ClientDataSet2: TClientDataSet;

DataSetProvider2: TDataSetProvider;

Image2: TImage;

SQLQuery3: TSQLQuery;

DataSource3: TDataSource;

ClientDataSet3: TClientDataSet;

DataSetProvider3: TDataSetProvider;

SQLQuery4: TSQLQuery;

DataSource4: TDataSource;

ClientDataSet4: TClientDataSet;

DataSetProvider4: TDataSetProvider;

procedure Image1Click(Sender: TObject);

procedure Image2Click(Sender: TObject);

procedure Memo1Click(Sender: TObject);

procedure Memo2Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form2: TForm2;

i,k,BufHolidayID:integer;

BufHolidayStr:string;

implementation

{$R \*.dfm}

uses Unit1;

procedure TForm2.Image1Click(Sender: TObject);

begin

BufHolidayID := 0;

BufHolidayStr := '';

ClientDataSet1.Active := false;

SQLQuery1.Close;

SQLQuery1.SQL.Clear;

SQLQuery1.SQL.Add('SELECT MAX(id\_paragraf) AS id\_paragraf FROM paragraf');

SQLQuery1.Open;

ClientDataSet1.Active := true;

BufHolidayStr := IntToStr(SQLQuery1.FieldValues['id\_paragraf']);

BufHolidayID := StrToInt(BufHolidayStr) + 1;

SQLQuery2.SQL.Clear;

SQLQuery2.SQL.Add('INSERT INTO paragraf,soderzanie,zadanie(id\_paragraf,name,id\_soderzanie,soderzanie,id\_zadanie,задача) VALUES(');

SQLQuery2.SQL.Add(IntToStr(BufHolidayID) + ', "' + Edit1.Text+ IntToStr(BufHolidayID) + ', "' + memo1.Text +IntToStr(BufHolidayID) + ', "' + memo2.Text +'"where soderzanie.id\_zadanie=paragraf.id\_paragraf and zadanie.id\_zadanie=paragraf.id\_paragraf"');

showmessage('Параграф добавлен.');

form1.Show;

form2.Hide;

end;

procedure TForm2.Image2Click(Sender: TObject);

begin

form1.Show;

form2.Hide;

end;

procedure TForm2.Memo1Click(Sender: TObject);

begin

memo1.Lines.Clear;

end;

procedure TForm2.Memo2Click(Sender: TObject);

begin

memo2.Lines.Clear;

end;

end.

unit Unit4;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Imaging.pngimage, Vcl.ExtCtrls,

Vcl.StdCtrls, Vcl.DBCtrls, Vcl.Mask;

type

TForm4 = class(TForm)

Image1: TImage;

DBEdit1: TDBEdit;

DBMemo1: TDBMemo;

DBMemo2: TDBMemo;

Image2: TImage;

procedure Image1Click(Sender: TObject);

procedure Image2Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form4: TForm4;

i,k:integer;

implementation

{$R \*.dfm}

uses Unit1, Unit3;

procedure TForm4.Image1Click(Sender: TObject);

begin

form1.show;

form4.hide;

end;

procedure TForm4.Image2Click(Sender: TObject);

begin

form1.SQLQuery1.SQL.Clear;

form1.SQLQuery1.SQL.Add('update paragraf,soderzanie,zadanie set paragraf.name="' + dbEdit1.Text + '",soderzanie.soderzanie = "' + dbmemo1.Text + '", zadanie.задача = "' + dbmemo2.Text +'"where id\_paragraf="'+form3.dbedit1.text+'"');

form1.SQLQuery1.ExecSQL;

showmessage('Параграф изменен.');

form1.show;

form4.hide;

end;

end.