МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(институт)

Прикладная математика и информатика
(кафедра)

09.03.03 Прикладная информатика
(код и наименование направления подготовки, специальности)

Бизнес-информатика
(наименование профиля, специализации)

БАКАЛАВРСКАЯ РАБОТА

на тему Разработка инструментальной обучающей среды на Component Pascal

Студент А.В. Фурсиков _____
(И.О. Фамилия) (личная подпись)

Руководитель Е.В. Панюкова _____
(И.О. Фамилия) (личная подпись)

Допустить к защите

Заведующий кафедрой к.т.н., доцент, А.В. Очеповский _____
(ученая степень, звание, И.О. Фамилия) (личная подпись)

«_____» _____ 2019 г.

Тольятти 2019

Росдистант
ВЫСШЕЕ ОБРАЗОВАНИЕ ДИСТАНЦИОННО

АННОТАЦИЯ

Тема выпускной квалификационной работы «Разработка инструментальной обучающей среды на Component Pascal».

Целью данной работы является разработка инструментальной обучающей среды на Component Pascal.

Объектом исследования работы является процесс обучения программированию в Лицее города Троицка Новая Москва.

Предметом данной работы является автоматизация процесса обучения программированию через внедрение инструментальной обучающей среды на Component Pascal.

В первой главе была определена технология концептуального моделирования, построены диаграммы обучения школьников Компонентному Паскалю. А также поставлена цель разработки и выделены основные задачи, ставшие основанием для разработки многофункциональной инструментальной обучающей среды.

Во второй главе было описано логическое проектирование разрабатываемой обучающей системы, произведен выбор технологии логического моделирования, построены диаграммы вариантов использования, спроектирована логическая модель данных и выработаны требования к аппаратно-программному обеспечению разрабатываемой системы, приведены примеры ее интерфейсов.

В третьей главе выпускной квалификационной работы было проведено физическое проектирование автоматизированной обучающей системы и рассмотрены ее функциональные возможности.

В работе 60 страниц, 30 рисунков, 9 таблиц.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Одна из самых важных проблем общества – это обучение и воспитание нового поколения. Современные тенденции обучения направлены на увеличение количества часов информатики, как в школе, так и в высших учебных заведениях. Компьютерная грамотность является неотъемлемым требованием в большинстве вакансий, а работа программиста является одной из самых востребованных. Однако процесс обучения детей программированию не только неудобен и тяжел для большинства школьников, но и полностью оторван от его продолжения в высших учебных заведениях.

Одной из предпосылок развития Компонентного Паскаля и является желание его разработчиков сделать язык как можно более простым, но с учетом всей необходимой функциональности и уменьшить количество совершаемых пользователем ошибок. Безусловно, если максимально упростить язык и ограничить его функциональность, то область его применения может снизиться. Но сторонники Компонентного Паскаля выступают за вынесение части функций языка в отдельные библиотеки, которые пользователь будет подключать при необходимости. Так, например, изучивший в школе основной функционал Компонентного Паскаля и поступивший на обучение в университет ученик может уже в высшем учебном заведении изучить те библиотеки языка, которые нужны для его специальности. При этом будет поддерживаться целостность обучения не только на уровне школы.

В рамках данной работы проведена разработка обучающей среды для работы учащихся с программным средством для обучения навыкам. Программирования и моделирования различных процессов физических и математических задач.

Актуальность темы разработки обусловлена необходимостью развития у учащихся навыков моделирования в различных средах. Успешно освоив базовые основы программирования, они зачастую не могут адаптироваться к требованиям ВУЗов по ИТ-специальностям, что приводит к затруднениям в освоении материалов по специальностям – в области информационных

технологий, так и в областях, где необходимы навыки в области компьютерного моделирования.

Таким образом, для повышения эффективности усвоения учащимися навыков программирования, модулирования и алгоритмизации необходимо использование специализированных обучающих систем.

Цель работы: разработка интерактивной обучающей среды на Component Pascal.

Объект исследования: процесс работы с интерактивной обучающей средой Component Pascal.

Предмет исследования: разработка обучающей системы по программированию в среде Component Pascal.

Для достижения цели исследования поставим задачи работы:

- провести анализ технологий моделирования в среде Component Pascal;
- провести анализ бизнес-процессов организации обучения с использованием среды Component Pascal;
- установить список задач автоматизации разработки обучающей среды;
- обосновать средства разработки;
- провести разработку, опытную эксплуатацию интерактивной обучающей среды на Component Pascal;
- провести анализ результатов использования интерактивной обучающей среды на Component Pascal в учебном процессе.

Теоретические методы исследования, используемые в данной работе: анализ, обобщение, систематизация, методы моделирования бизнес-процессов при помощи case-средств, стандарты проектирования и моделирования, а также проводились наблюдения и описания.

В первой главе описано функциональное моделирование инструментальной обучающей среды, с подробным описанием требований к её аппаратно-программному обеспечению.

Во второй – проектирование и логическая модель программы.

В третьей главе проводится физическое проектирование автоматизированной обучающей системы, и демонстрируются ее функциональные возможности.

В работе 60 страниц, 30 рисунков, 9 таблиц.

Глава 1 Функциональное моделирование инструментальной обучающей среды

1.1 Постановка технического задания на разработку обучающей среды

Цель работы: разработка обучающей программы для изучения языка программирования Компонентный Паскаль и работы на нем.

Система должна состоять из 2 подсистем: подсистема обучения программированию, включающая тренажеры и обучающие средства мультимедиа, подсистема контроля результатов обучения.

Полное наименование системы: «Разработка многофункциональной инструментальной обучающей среды на Component Pascal».

Краткое наименование системы: «МИОС КП».

Наименование предприятия разработчика и заказчика (пользователя) системы: заказчик – Методическое объединение преподавателей информатики Лицея города Троицка и проект Информатика XXI.

Основание для проведения работ: договор разработки программного обеспечения.

Плановые сроки начала и окончания работы:

- дата начала: 10.03.2019;
- дата окончания: 10.04.2019.

Порядок оформления и предъявления заказчику результатов работ:

- Работы по созданию системы сдаются Разработчиком поэтапно в соответствии с календарным планом проекта.
- По окончании каждого из этапов работ Разработчик сдает Заказчику соответствующие отчетные документы этапа, которые определены договором № 1 от 05.03.2019.

Требования к разрабатываемой системе:

- Требования к системе в целом.

Количество специалистов, работающих с МИОС КП и уровень их квалификации определяется с учетом следующих требований:

– проектирование структуры, конфигурирование системы и реализация

программной системы проводится в целях сокращения количественного состава обслуживающих специалистов;

– для управления системой к администраторам предъявляются требования, связанные с безопасностью данных и разграничением доступа. Знание учебных методик администраторами не является обязательным;

– программно-аппаратная система должна обеспечивать бесперебойность работы системы без необходимости постоянного присутствия администраторов (задачи обслуживания базы должны выполняться автоматически с запуском по расписанию).

- Работа специалистов с МИОС КП должна производиться в рабочее время.

Максимальная работоспособность системы должна обеспечиваться на протяжении рабочего времени с установленными регламентированными перерывами: через 2 часа после начала рабочей смены и через 1.5-2.0 часа после обеденного перерыва продолжительностью 15 минут каждый или продолжительностью 10 минут через каждый час работы.

Таблица 1.1. – Требования к функциям, выполняемым системой

Задача	Требования к временному регламенту
Учет базовых знаний по программированию	Весь период работы учащихся с системой Component Pascal
Редактирование учебного материала	Весь период работы учащихся с системой Component Pascal
Реализация учебных моделей	Весь период работы учащихся с системой Component Pascal
Контроль самостоятельного процесса работы с компьютерными моделями	Весь период работы учащихся с системой Component Pascal
Анализ результатов обучения работы со средой Компонентного Паскаля	Весь период работы учащихся с системой Component Pascal

Определим используемые в рамках данной работы решения по видам обеспечения:

- Математическое обеспечение – это совокупность, включающая математические модели и алгоритмы, используемые при решении задач и обработке информации с использованием средств вычислительной техники.

Компонентами математического обеспечения являются [7]:

– технические документы (включая описание задач, алгоритмов их решения, используемый перечень экономико-математических моделей);

– средства математического обеспечения (средства по моделированию типовых задач управления, методов многокритериальной оптимизации, программные средства, содержание аппарат математической статистики и др.);

– методологию выбора математического обеспечения (набор методов определения типов задач, методов оценивания вычислительной сложности алгоритмов, методов оценивания степени достоверности результатов).

- Программное обеспечение.

Программные модули, составляющие структурную схему ИС, сгруппированы на несколько, слабосвязанных между собой программные сущности, представляющих собой пакеты. Каждый пакет отвечает за определенную функциональность системы (например, вход в систему, доступ к базе данных, формирование отчётов и т.п.) [12].

- Техническое обеспечение представляет комплекс технических средств, предназначенных для обработки данных в ИС. В состав комплекса входят электронные вычислительные машины, осуществляющие обработку экономической информации, средства сбора и регистрации информации, средства передачи данных по каналам связи, средства накопления и хранения данных и выдачи результатной информации, вспомогательное оборудование и организационная техника.

Структура входной и выходной информации, а также используемых справочников, соответствует требованиям нормативных документов. Это позволяет «общаться на одном языке» в рамках технологии интерактивного обучения, а также существенно облегчить актуализацию информационных ресурсов системы [13].

Формы интерфейса пользователя, используемые для представления входных/выходных документов, должны по возможности предоставлять информацию в той же последовательности и в том же расположении, что и их бумажные аналоги.

Все внешние компоненты технических средств, которые находятся под напряжением, должны обладать системой защиты от случайных прикосновений, а также занулением или защитным заземлением в соответствии с требованиями ГОСТ 12.1.030-81.

Система электроснабжения должна обеспечивать возможности защитного отключения в случае возникновения перегрузок и короткого замыкания в цепях нагрузки, иметь возможности аварийного ручного отключения.

Система обеспечения пожарной безопасности должна соответствовать стандартам защищенности электрооборудования.

Факторы, которые оказывают вредное воздействие на здоровье со стороны всех компонентов системы (включая инфракрасные, ультрафиолетовые, рентгеновские и электромагнитные излучения, вибрацию, шумы, электростатические поля, ультразвуковые волны строчной частоты и т.д.), по своей интенсивности должны соответствовать действующим стандартам (СанПиН 2.2.2./2.4.1340-03 от 03.06.2003 г.) [8].

В случае сбоев в работе технических средств восстановление штатного функционирования системы должно производиться в случаях [7]:

– проведения перезагрузки операционных систем;

– запуска системных исполняемых файлов.

При ошибках в работе аппаратных средств (кроме носителей данных и программ) восстановление функции системы возлагается на ОС.

При ошибках, связанных с программным обеспечением (ОС и драйверы устройств), восстановление работоспособности возлагается на ОС.

В случае ошибочных действий пользователей, ошибках в используемых форматах или недопустимых значениях входного потока данных, система должна выдавать соответствующие сообщения, с дальнейшим возвратом в

режим штатного функционирования.

Для обеспечения штатного режима работы системы необходимо обеспечить выполнение требований и соблюдение условий эксплуатации информационной системы МИОС КП и аппаратного комплекса системы, которые указываются в технической и эксплуатационной документации.

Обеспечение технической и физической защиты аппаратных компонентов системы, носителей информации, бесперебойного энергоснабжения, резервирования ресурсов, необходимо проведение текущего обслуживания посредством технических и организационных средств [14].

Для штатного функционирования МИОС КП необходимо обеспечение бесперебойного питания аппаратной части системы. При эксплуатации система должна быть обеспечена соответствующая стандартам хранения носителей и эксплуатации ПЭВМ температура и влажность воздуха.

Периодическое техническое обслуживание используемых технических средств должно проводиться в соответствии с требованиями технической документации изготовителей, но не реже одного раза в год.

Периодическое техническое обслуживание и тестирование технических средств должны включать в себя обслуживание и тестирование всех используемых средств, включая рабочие станции, серверы, кабельные системы и сетевое оборудование, устройства бесперебойного питания.

В процессе проведения периодического технического обслуживания должны проводиться внешний и внутренний осмотр и чистка технических средств, проверка контактных соединений, проверка параметров настроек работоспособности технических средств и тестирование их взаимодействия.

При вводе системы в опытную эксплуатацию должен быть разработан план выполнения резервного копирования программного обеспечения и обрабатываемой информации. Во время эксплуатации системы, персонал, ответственный за эксплуатацию системы должен выполнять разработанный план [18].

Размещение оборудования, технических средств должно соответствовать

требованиям техники безопасности, санитарным нормам и требованиям пожарной безопасности.

Все пользователи системы должны соблюдать правила эксплуатации электронной вычислительной техники.

Обеспечение защиты обработки данных об эффективности оказания образовательных услуг должно проводиться в соответствии с действующим законодательством Российской Федерации.

Обязательным требованием, с точки зрения законодательства, является обеспечение конфиденциальности персональной информации [17].

Информация не должна быть подвергнута любому виду распространения, поэтому система должна иметь защиту от несанкционированного доступа в виде защитных паролей. С системой должны работать только специалисты, указанные в техническом задании.

Программное обеспечение должно восстанавливать свое функционирование при корректном перезапуске аппаратных средств. Должна быть предусмотрена возможность организации автоматического и (или) ручного резервного копирования данных системы средствами системного и базового программного обеспечения, входящего в состав программно-технического комплекса Заказчика [19].

Для разработки конфигурации потребуется компьютер, удовлетворяющий следующим требованиям:

- операционная система: MS Windows 7/10;
- процессор тактовой частотой от 2ГГц);
- оперативная память 2гб и выше (рекомендуется 4гб);
- жесткий диск.
- операционная система: MS Windows 7 и выше;
- процессор тактовой частотой выше 2 ГГц;
- оперативная память 2гб и выше (рекомендуется 4гб);
- сервер баз данных:
- Microsoft SQL Server 2012.

МИОС КП должна реализовывать возможность дальнейшей модернизации как программного обеспечения, так комплекса технических средств.

По мере использования системы у заказчика могут возникнуть дополнительные требования к системе. Для выполнения требований заказчика ему потребуется обратиться к разработчику. После уяснения задачи между заказчиком и разработчиком заключается договор на модернизацию ПО.

По окончанию разработки информационной системы разработчик должен предоставить необходимые документы и инструкции для дальнейшей работы с системой.

1.2 Моделирование процесса обучения Component Pascal

Следует описать работу программы, ее функции, а также дать инструкции специалистам.

Экономические факторы. Внедрение тех или иных информационных технологий в деятельность предприятий должно быть оправдано экономически.

Так, в процессе проведения анализа предметной области (в нашем случае – обучающей системы) необходимо провести переподготовку педагогического состава. В образовании очень сложно оценить экономический эффект от внедрения новых педагогических технологий. Но на примере Лицея города Троицка (Новая Москва) можно отметить, что успешное выступление учащихся на олимпиадах по Информатике различного уровня позволило Лицею получить статус лучшей школы России и получение грантов, успешно использованных на новые современные компьютеры. А в долгосрочной перспективе это новые научные и производственные кадры, хорошо образованные и востребованные.

Анализ современных технологий обучения и программы изучения информатики в Лицее города Троицка позволил сформировать требования с моделируемой системе:

- Пользовательские факторы. Факторы данного типа предполагают возможность учета процессов, связанных с разработкой, внедрением и сопровождением системы.

Необходим учёт следующих факторов [15]:

– степень удобства пользовательского интерфейса в т.ч. по характеристикам быстродействия;

– порядка получения обновлений системы, как при обнаружении ошибок, так при изменении технологии работы специалистов;

– порядка организации службы технической поддержки (от необходимости наличия ИТ-специалистов на местах, технологий удаленного управления, необходимости периодического выезда разработчиков).

- Факторы соответствия существующей системной архитектуры. Внедряемые программные решения должны соответствовать особенностям архитектуры автоматизированной системы организации.

При внедрении программных решений необходим учёт следующих факторов [16]:

– степень соответствия аппаратным требованиям;

– степень соответствия сетевой архитектуре, отсутствие конфликтов при работе с ресурсами сети;

– соответствия требований к МИОС КП (при использовании имеющихся у клиента средств осуществляется разработчиком).

- Факторы соответствия требованиям защиты информации.

Требования информационной безопасности при внедрении прикладных программных решений должны учитываться в виде [11]:

- наличия системы разграничения доступа (например, на административный, гостевой и оперативный);

- наличия системы парольной защиты и регламентация использования паролей (сложность, периодичность смены и др.);

- возможность проведения экспертизы на наличие недокументированных возможностей;

- наличия требований к правам пользователя на уровне операционной системы (необходимо ли обеспечивать права администратора на рабочей станции при функционировании системы);
- наличия системы автоматического резервного копирования и восстановления базы данных.

В данной работе используется BlackBox Component Builder.

Компонентами данной системы являются: среда разработки, компилятор (exe и dll), среда поддержки исполнения (виртуальная машина), профилировщик, и готовые компоненты для возможности обращения к интерфейсам ОС (WinApi, WinOle, COM), набор инструментов для работы с базами данных, поддержка коммуникационных протоколов TCP/IP и т.д.

В системе имеется компонентный каркас, необходимый для реализации диалоговых систем, систем интерактивной графики, присутствуют возможности с составными документами, поддерживается технология OLE.

В установочных файлах присутствует большой набор нетривиальных примеров программ, имеется литература, посвященная работе в BlackBox. Система предназначена для комплексного информационно-аналитического обеспечения процессов, в части исполнения следующих процессов:

- обучение программированию в среде Компонентного Паскаля в интерактивном режиме;
- учет качества выполненных работ;
- формирование отчетности о результатах обучения. МИОС КП обеспечивает:

1. Поддержку технологий обучения навыкам программирования.
2. Обучение навыкам компьютерного моделирования.
3. Возможность получения аналитики по результатам обучения.
4. Обеспечение наглядности и структурности данных, удобства в работе с обучающей системой.

Новая система предназначена для интерактивного обучения навыкам программирования.

Система должна выдавать отчеты о результатах обучения программированию, обладать справочной системой по всем разделам обучения Component Pascal, иметь модуль проверки создаваемых программ и моделей.

Проведем анализ задач автоматизации и результаты представим в виде таблицы 1.2

Таблица 1.2 – Анализ задач автоматизации

Структурное подразделение	Наименование процесса	Возможность автоматизации	Решение об автоматизации в ходе проекта
Методическое объединение по информатике	Учет базовых знаний по программированию	Возможно	Будет автоматизировано
Методическое объединение по информатике	Редактирование учебного материала	Возможно	Будет автоматизировано
Методическое объединение по информатике	Реализация учебных моделей	Возможно	Будет автоматизировано
Методическое объединение по информатике	Контроль самостоятельного процесса работы с компьютерными моделями	Возможно	Будет автоматизировано
Методическое объединение по информатике	Анализ результатов обучения работы со средой Компонентного паскаля	Возможно	Будет автоматизировано

Все рассмотренные задачи автоматизации могут быть успешно реализованы при помощи Component Pascal.

1.3 Анализ бизнес-процессов использования Component Pascal в обучении школьников

Проведем анализ бизнес-процессов использования Component Pascal в обучении школьников. На рисунке 1.1 приведена контекстная диаграмма.

Как показано на рисунке 1.1, входящие информационные потоки в систему обучения в среде Component Pascal включают задачи программирования и задачи моделирования. Результирующий поток: данные о результатах освоения среды программирования. На рисунке 1.2 приведена диаграмма использования компонентного Паскаля в учебном процессе.

Как показано на рисунке 1.2, учебный процесс освоения навыков работы с компонентным паскалем включает:

– изучение синтаксиса языка;
– получение навыков программирования;
– решение прикладных задач с использованием Component Pascal;
– анализ качества полученных знаний.

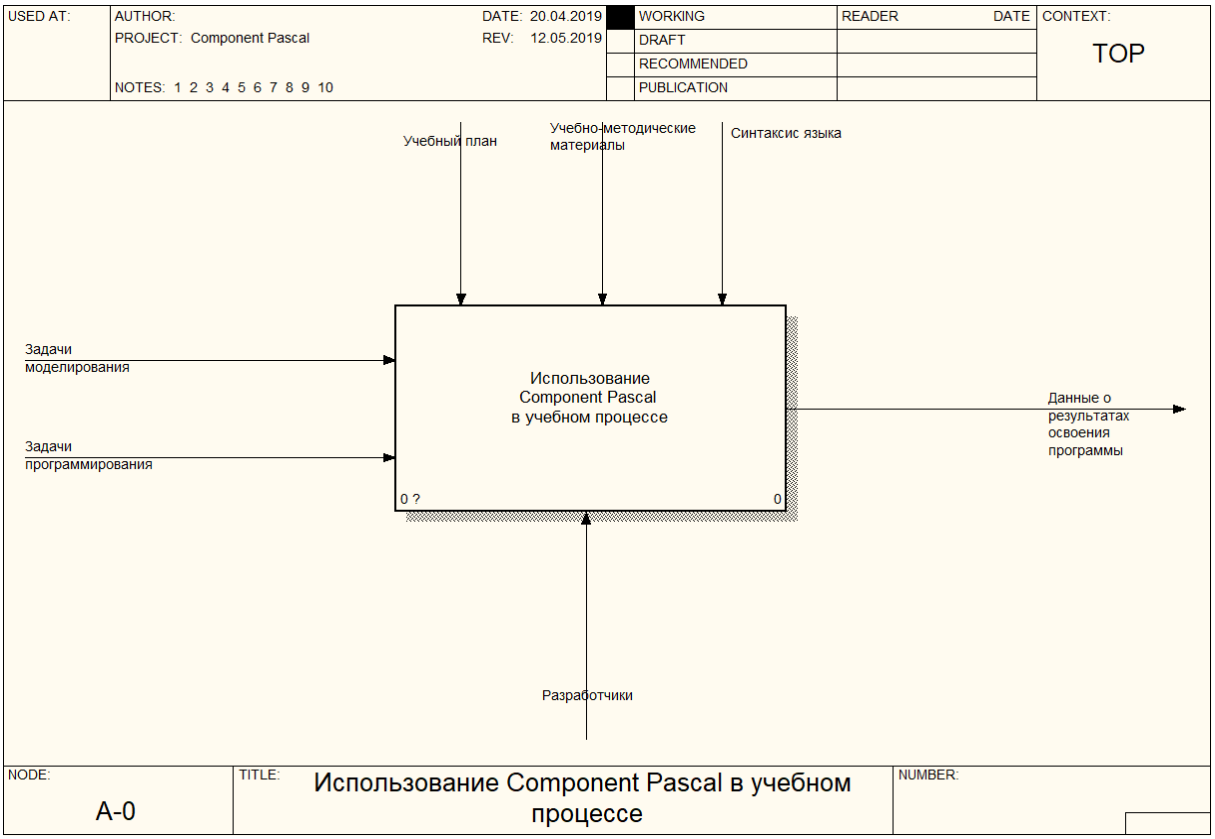


Рисунок 1.1. – Контекстная диаграмма «Как есть»

Как показано на рисунке 1.2, на первоначальном этапе при изучении Компонентного Паскаля проводится изучение синтаксиса, основных операторов, правил программирования.

Далее проводится обучение решению типовых задач по программированию.

После получения навыков программирования учащиеся решают задачи моделирования физических процессов.

Далее проводится оценка качества полученных знаний в различных формах (написание итогового проекта, экзамен и т.д.).

На рисунке 1.3 приведена диаграмма изучение синтаксиса языка, на рисунке 1.4 – диаграмма обучения программированию.

На рисунке 1.5 приведена диаграмма обучения навыкам моделирования.

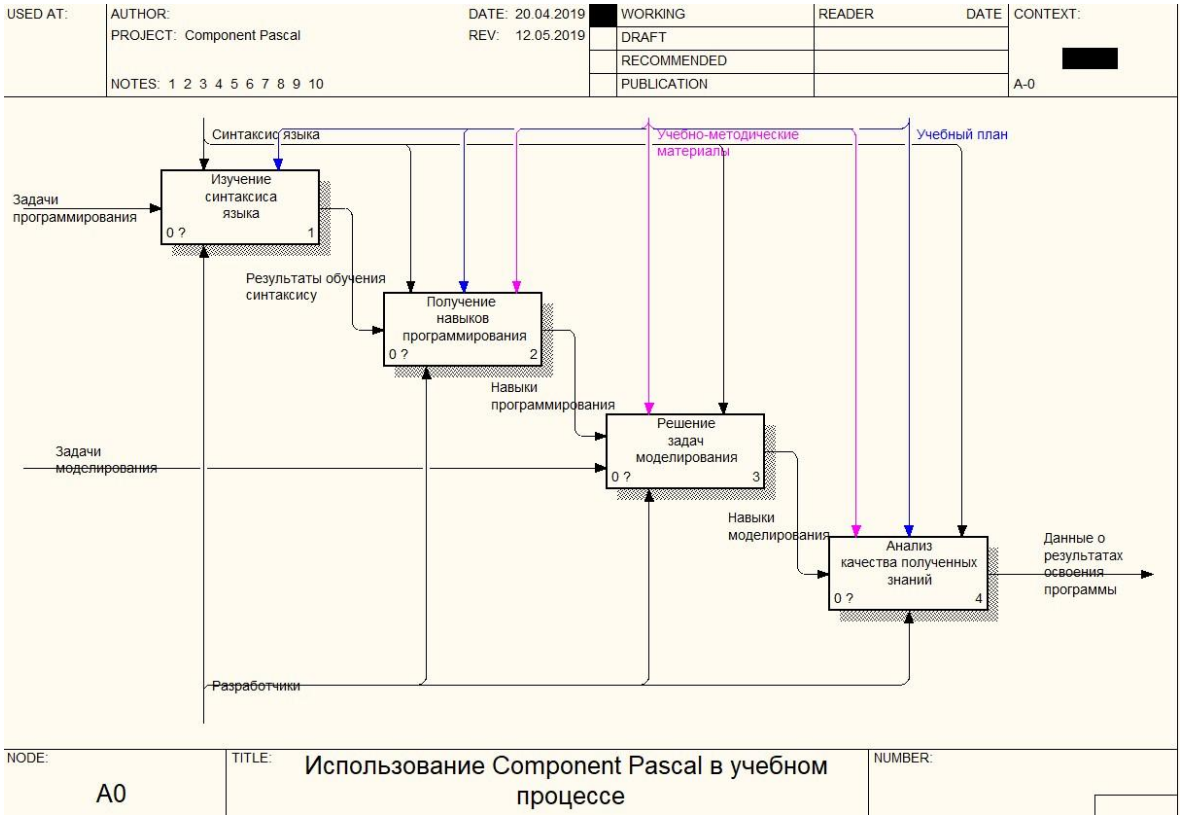


Рисунок 1.2 – Диаграмма декомпозиции использования компонентного Паскаля в учебном процессе

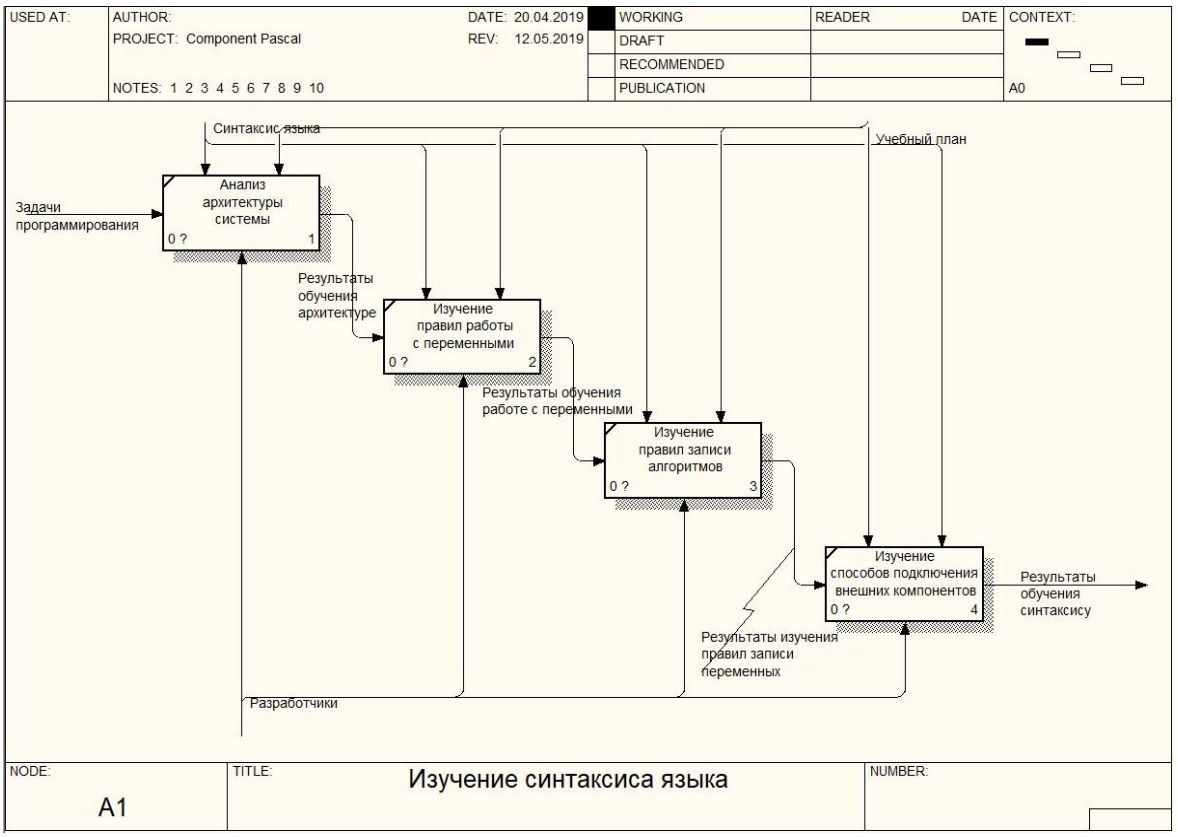


Рисунок 1.3 – Диаграмма декомпозиции процесса изучения синтаксиса языка

Как показано на рисунке 1.5, решение задач моделирования (включающих математические, экономические, естественнонаучные и другие задачи) с использованием Компонентного Паскаля предполагает анализ поставленной задачи, проведение ее формализации, выбор алгоритмов реализации с дальнейшим написанием программного кода и последующим тестированием результатов. Тестирование проводится путем анализа корректности полученной путем расчета информации, сравнение результатов решения и имеющихся теоретических данных по указанному вопросу.

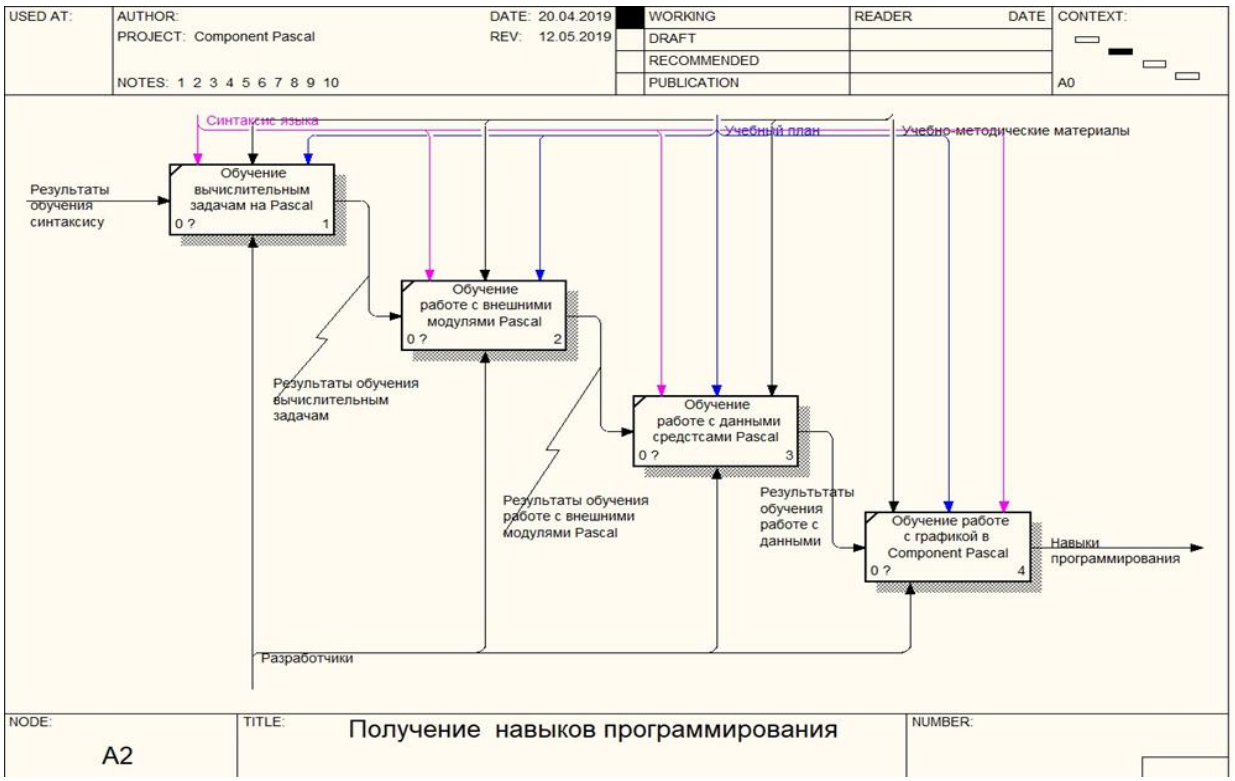


Рисунок 1.4 – Диаграмма декомпозиции получения навыков программирования

Для возможности решения прикладных задач и построения моделей необходимо иметь сформированные навыки программирования. Решение задач моделирования позволяет закрепить навыки работы со средами программирования с дальнейшим освоением более сложных задач в различных предметных областях.

Проверка навыков работы с Компонентным Паскалем предполагает многоаспектный анализ на знание приемов программирования, умение проводить построение моделей, решать прикладные задачи.

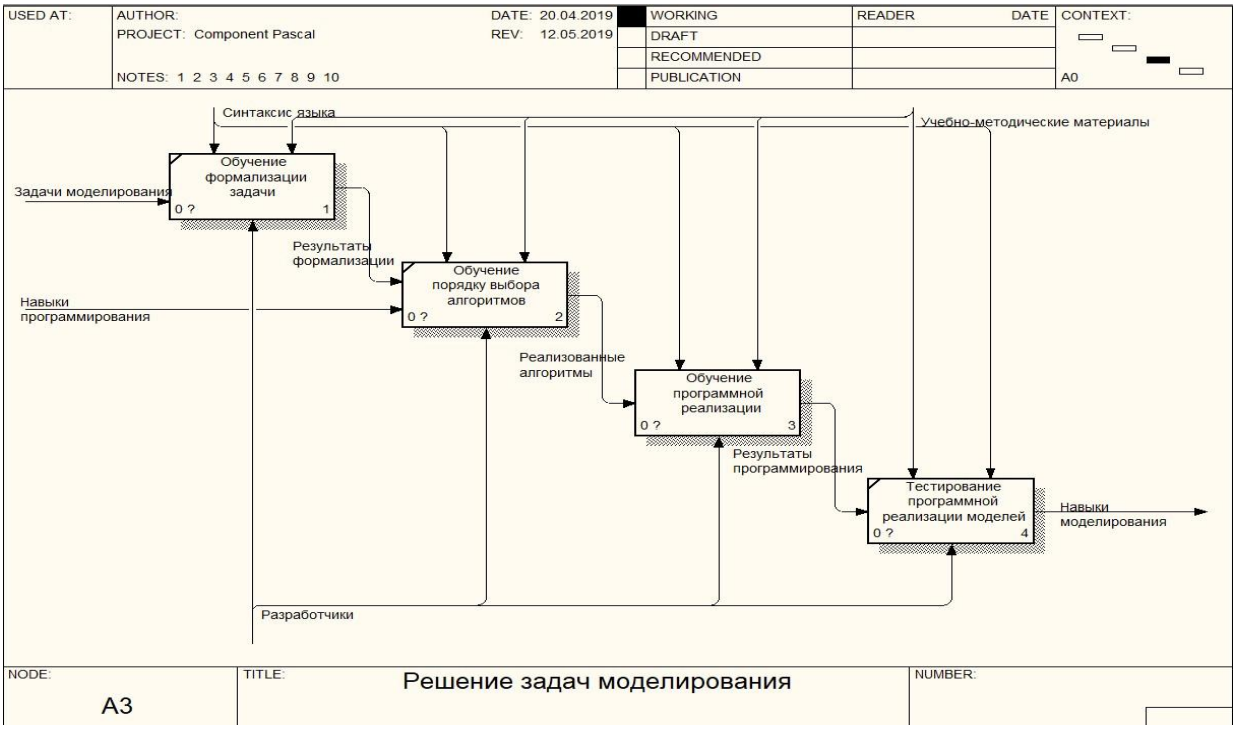


Рисунок 1.5 – Диаграмма декомпозиции процесса решения задач моделирования

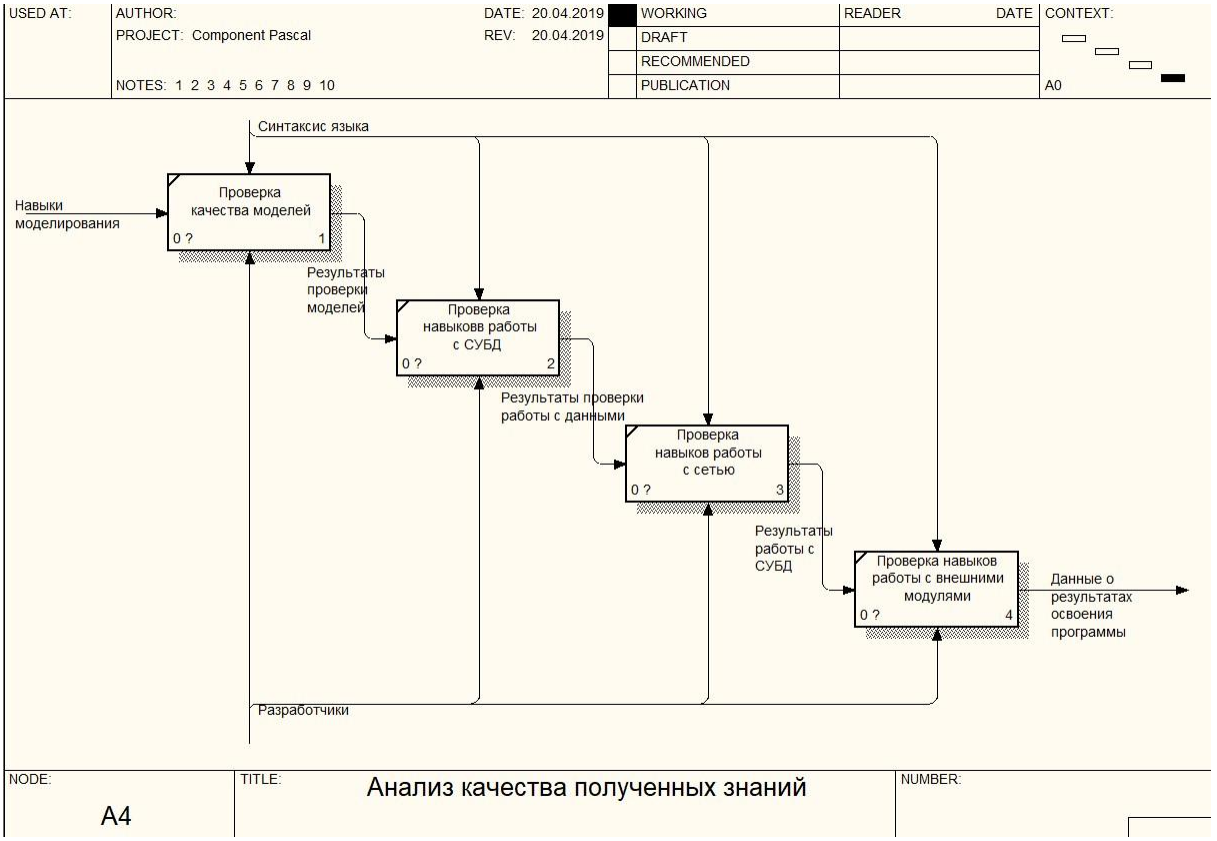


Рисунок 1.6 – Диаграмма анализа качества построенных моделей

Ожидаемый результат обучения работы с Компонентным Паскалем – получение учащимися набора знаний, позволяющего осваивать программы ВУЗов, без прохождения стадии первоначальной адаптации.

В силу сложности для понимания для школьников определенной тематики по разделам программирования и моделирования, для поддержки возможности освоения курса в интерактивном режиме необходимо использование обучающего средства. На рисунке 1.7 приведена контекстная диаграмма модернизированного бизнес-процесса.

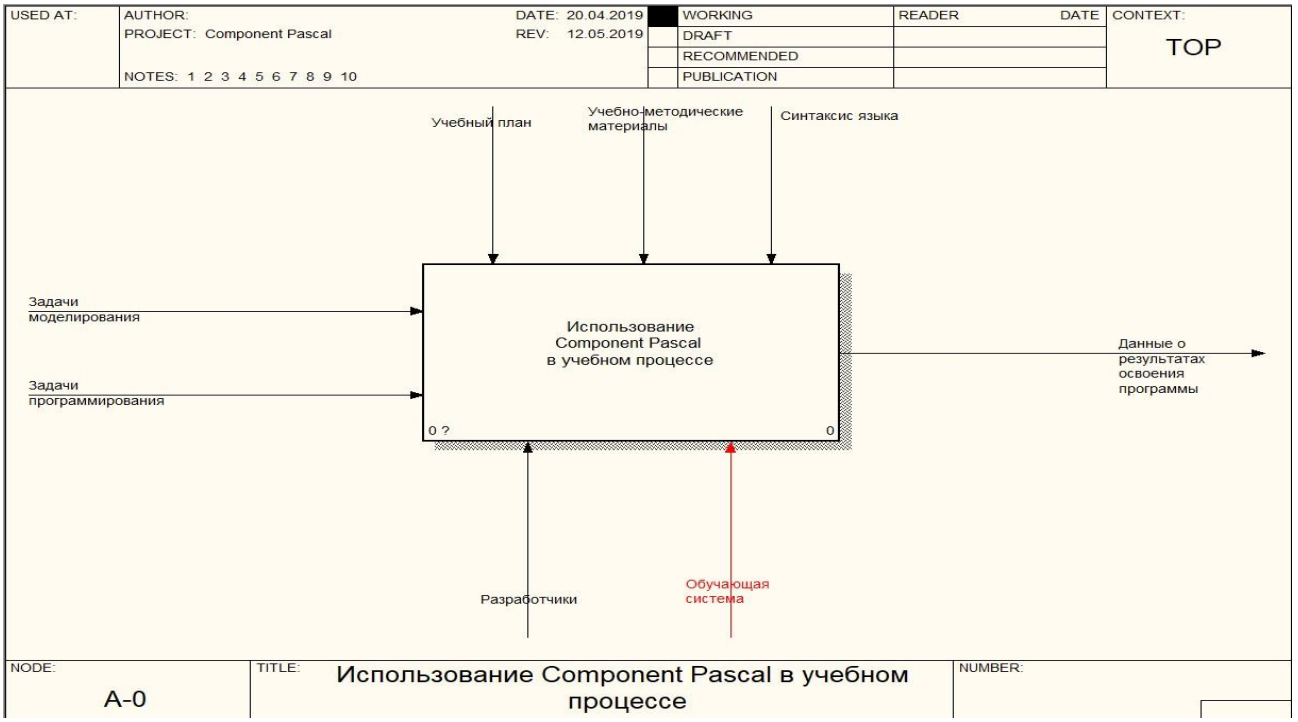


Рисунок 1.7 – Контекстная диаграмма «Как должно быть»

На рисунке 1.9 приведена ARIS-диаграмма обучения программированию с использованием среды компонентного Паскаля. На рисунке 1.10 приведена диаграмма потоков данных для сценария работы преподавателя. На рисунке 1.11 приведена диаграмма потоков данных для сценария работы учащихся с системой. Таким образом, планируется разработка или внедрение программного средства, решающего следующие задачи создания интерактивного учебного пособия для обучения программированию с использованием среды Компонентного Паскаля.

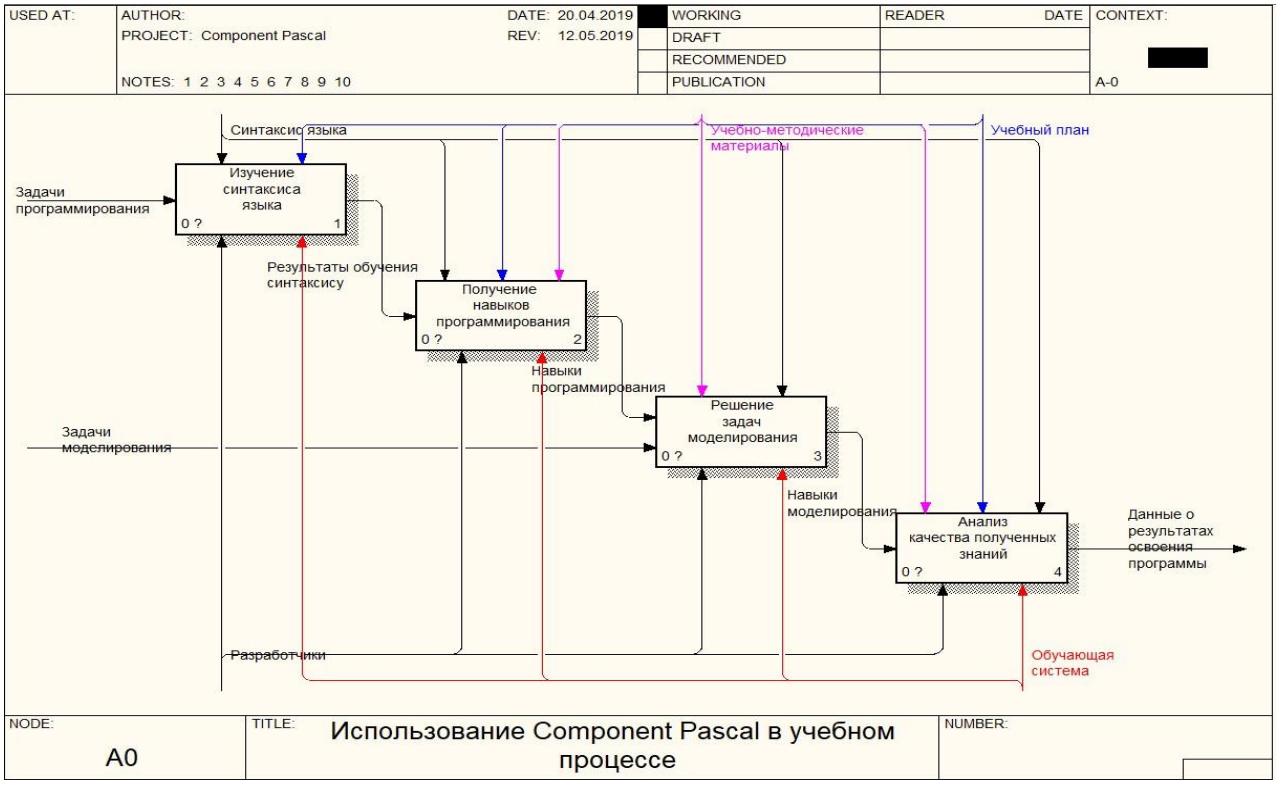


Рисунок 1.8 – Диаграмма декомпозиции модернизированного процесса

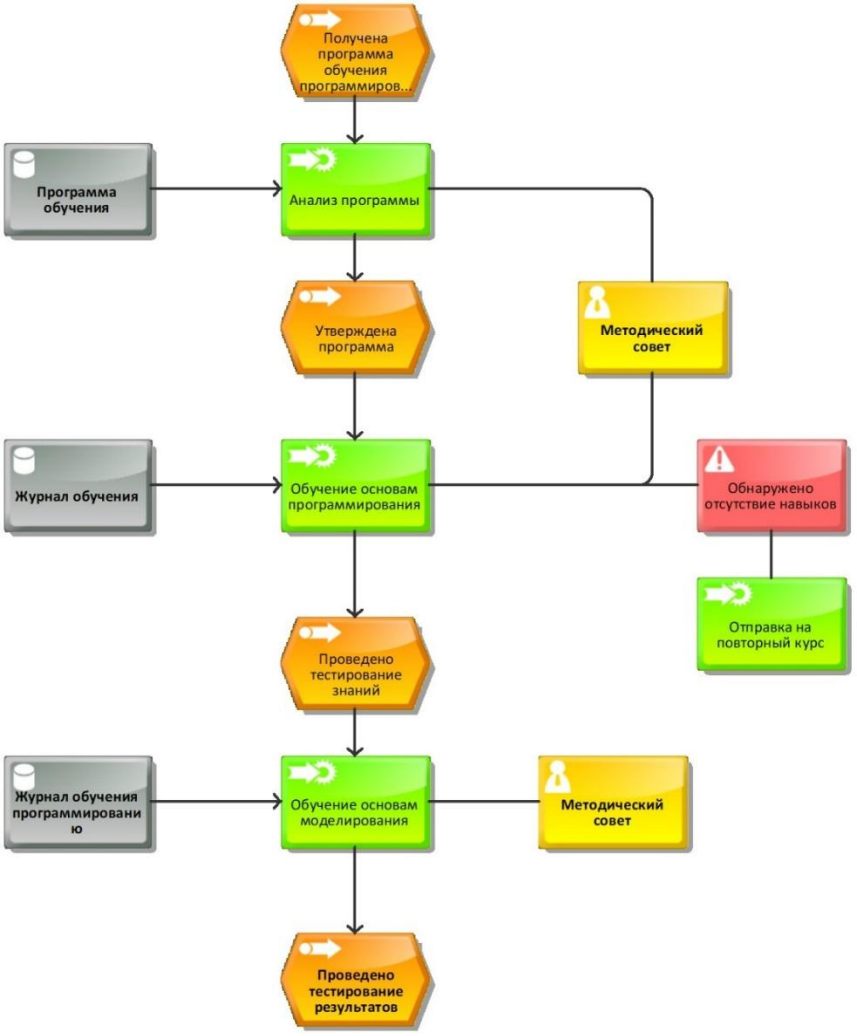


Рисунок 1.9 – ARIS-диаграмма обучения программированию

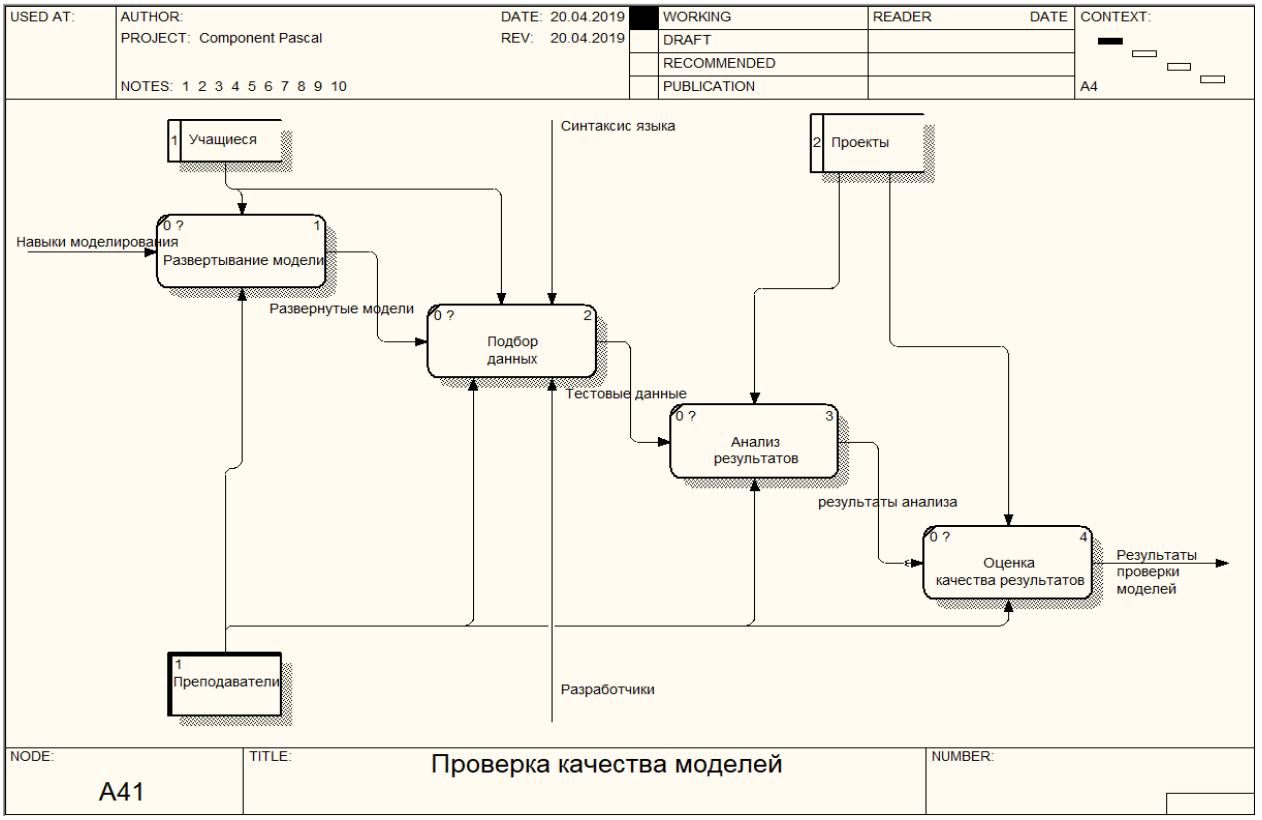


Рисунок 1.10 – Диаграмма потоков данных

Рисунок 1.11 – Диаграмма потоков данных для сценария работы учащихся

На основе полученных данных были выявлены области, где возможно внедрение информационных технологий. В частности, представляется целесообразным внесение следующих изменений в действующие бизнес процессы.

1.4 Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие

Этапы выполнения работ предоставлены в таблице 1.3.

Таблица 1.3 – этапы выполнения работ

Название этапа	Содержание работы
Анализ предметной области	Отчет
Подготовка технического задания	Техническое задание
Проектирование системы	Технический проект
Разработка системы	Руководство пользователя, описание программы

Порядок контроля и приемки. С целью дальнейшей корректной работы проекта осуществляется контроль системы заказчиком, проверяется работоспособность и необходимость тех или иных компонентов системы на всем этапе проектирования МИОС КП.

Приемка программы осуществляется при завершении всех работ над МИОС КП, система должна удовлетворять требованиям заказчика, иметь всё, что было указано в техническом задании, а также считаться пригодной для дальнейшего использования.

Список мероприятий [23]:

– подготовка информационной системы к развертыванию МИОС КП»;

– провести обучение педагогов информатики, ознакомление с документацией на программное обеспечение;

– подготовка приказов о вводе в промышленную эксплуатацию МИОС КП;

– назначение администраторов приложения и безопасности МИОС КП.

Программа должна быть создана на основе следующих документов:

- ГОСТ 19,105-78 ЕСПД, ГОСТ 19.201-78 ЕСПД, ГОСТ 19.402-78 ЕСПД, ГОСТ 19.404-79 ЕСПД, ГОСТ 19.504-79 ЕСПД, ГОСТ 19.505-79 ЕСПД, ГОСТ;

- 19.506-79 ЕСПД.

По результатам выполнения работ заказчику передаётся следующая документация:

- Настоящее техническое задание.
- Технический проект.
- Руководство пользователя.
- Описание программы.

Выводы по первой главе

В первой главе была определена технология концептуального моделирования, построены диаграммы обучения школьников Компонентному Паскалю. А также поставлена цель разработки и выделены основные задачи, ставшие основанием для разработки инструментальной обучающей среды.

Глава 2 Проектирование обучающей среды

2.1 Диаграмма вариантов использования

Диаграмма вариантов использования наглядно отображает взаимодействие между действующими лицами (актерами) и вариантами использования. В качестве актеров в данной предметной области выступают Преподаватель и Учащийся.

Рисунок 2.1 – Диаграмма вариантов использования

На рисунке 2.1 отражены варианты использования, включенные в систему.

Таблица 2.1 - Спецификация прецедента

Название актера	Название прецедента	Краткое описание прецедента
Преподаватель	Настройка доступа	Включить настройку доступа к информационной системе
Преподаватель	Работа с учебной программой	Работать, используя все возможности системы
Преподаватель	Выбор модулей	Выбор модулей, исходя из программы обучения
Преподаватель	Контроль знаний	Оценить знания учащегося
Учащийся	Обучение	Освоение операторов и возможностей языка
Учащийся	Программирование	Решение задач на языке Компонентный Паскаль

С помощью диаграммы использования выделяют основные роли актеров и разграничивают их права в разрабатываемой системе.

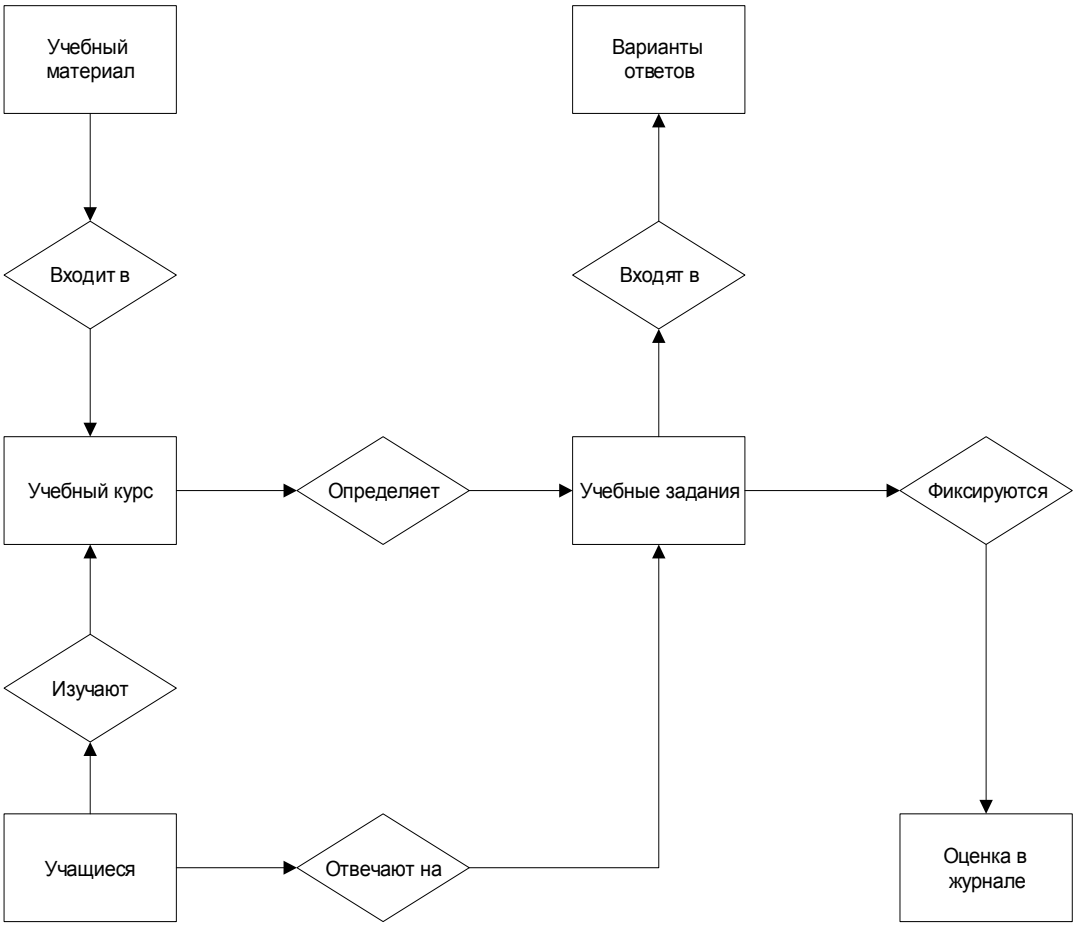


Рисунок 2.2 – Диаграмма «Сущность-Связь»

На основании диаграммы «сущность-связь» проведем проектирование логической модели данных.

Структура нормативно-справочной информации, используемой для разработки системы электронного тестирования, приведена в таблице 2.2.

Таблица 2.2 – Структура нормативно-справочной информации

№ п/п	Наименование кодирования множества объектов	Значность кода	Система кодирования	Вид классификатора
1	2	3	4	5
1	Код учащегося	XXXXXX	порядковая	локальный
2	Код занятия	XXX	порядковая	локальный
3	Код учебного материала Component Pascal	XXX	порядковая	локальный
4	Код оценки в журнале	XXX	порядковая	локальный

Описание систем классификации и кодирования.

Код студента. Длина кода XXXXXX, где XXXXXX – порядковый номер тестируемого в классификаторе учебного заведения.

Код занятия. Длина кода XXX, где XXX – порядковый номер занятия в классификаторе.

Код учебного материала. Длина кода XXX, где XXXX – порядковый номер учебного материала.

Код оценки в журнале Длина кода XXX, где XXXX – порядковый номер оценки.

Описание структуры логической модели данных приведено ниже.

Таблица 2.3 – Справочник «Учащиеся»

Наименование поля	Синоним	Тип данных	Размер поля
1	2	3	4
Код учащегося	Code_st	Числовой	Ц
ФИО	fio	Текстовый	4
Группа	grp	Текстовый	4

Таблица 2.4 – Справочник «Учебные дисциплины»

Наименование поля	Синоним	Тип данных	Размер поля
1	2	3	4
Код учебной дисциплины	Code_dsc	Числовой	Целое
Наименование	nam	Текстовый	40

Таблица 2.5 – Справочник «Учебные материалы»

Наименование поля	Синоним	Тип данных	Размер поля
1	2	3	4
Код учебного материала	Code_mat	Числовой	Целое
Наименование	nam	Текстовый	40
Путь к файлу учебного материала	path	Текстовый	255
Код учебной дисциплины	Code_dsc	Числовой	Целое

Таблица 2.6 – Журнал оценок

Наименование поля	Синоним	Тип данных	Размер поля
1	2	3	4
Код оценки	Code_oc	Числовой	Целое
Код занятия	Code_zn	Числовой	Целое
Код учебного курса	Code_uk	Числовой	Целое
Наименование	nam	Текстовый	100
Балл	ball	Числовой	Целое

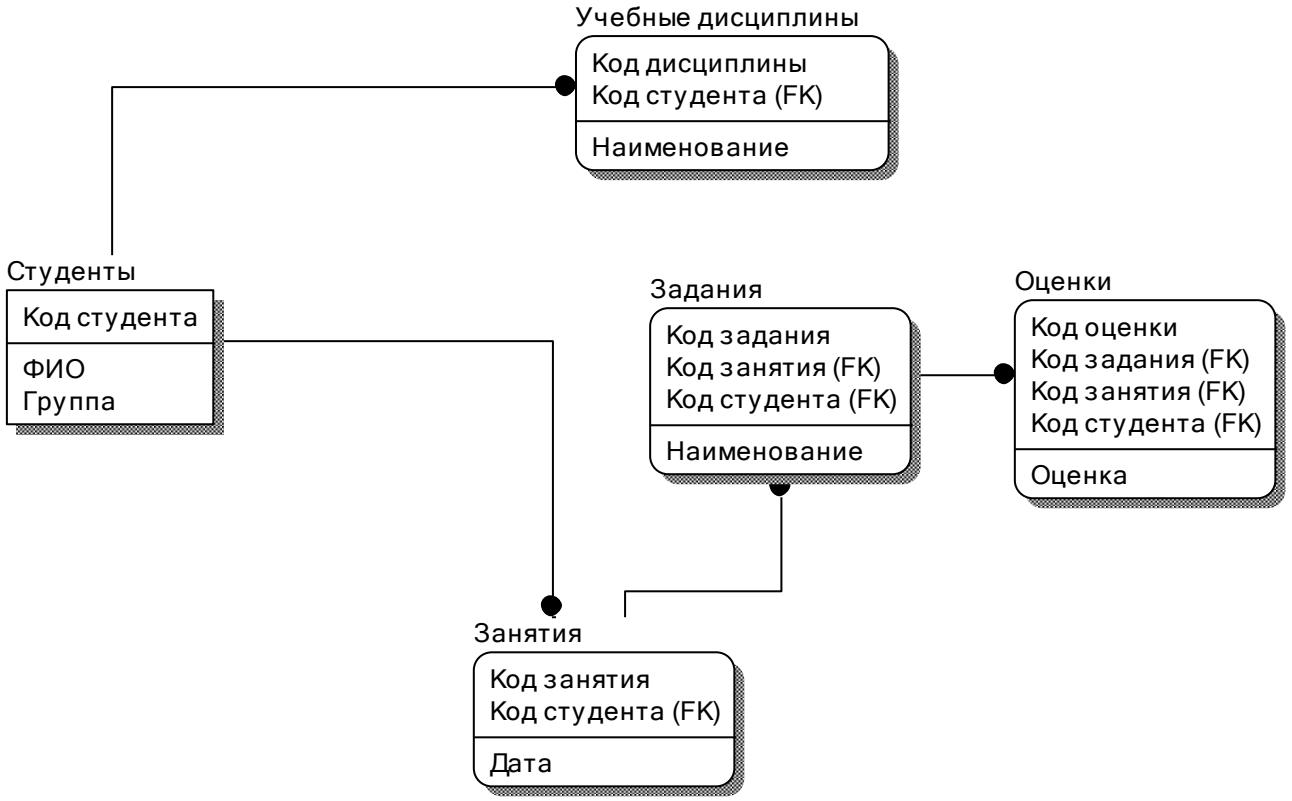


Рисунок 2.3 – Логическая модель данных

Информационная система включает объекты:

– Учащиеся (для учета данных об обучающихся);

– Занятия (для ведения журнала занятий);

– Учебные дисциплины (для ведения учета видов проводимых занятий в рамках курса обучения компонентному Паскалю);

– Задания (картотека учебных заданий);

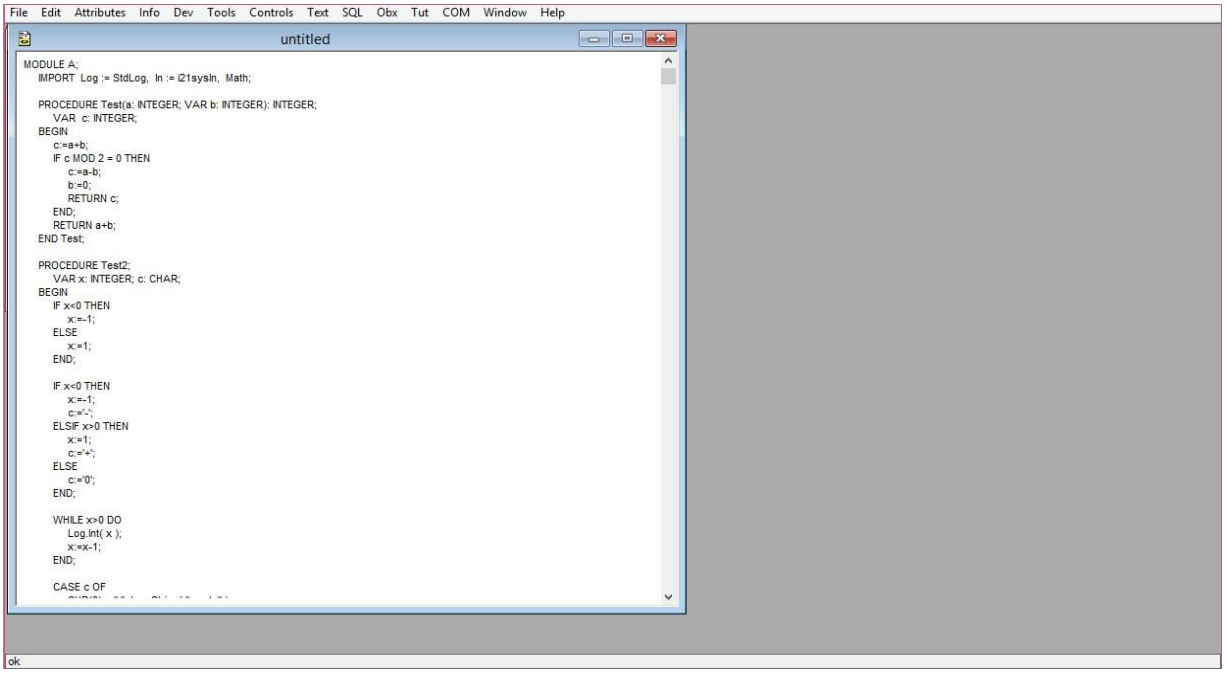
– Оценки (данные о результатах выполнения заданий).

Для дальнейшего проектирования системы необходимо сделать обоснование выбора среды разработки системы.

2.2 Выбор среды разработки

На рисунке 2.4 приведен режим среды разработки BlackBox, на рисунке 2.5 – пример работы с системой.

Подчеркнем отличия среды разработки BlackBox от других языков программирования и сред разработки. Создавая эту среду, разработчики



преследовали широкий набор целей и задач.

Рисунок 2.4 – Режим среды разработки BlackBox

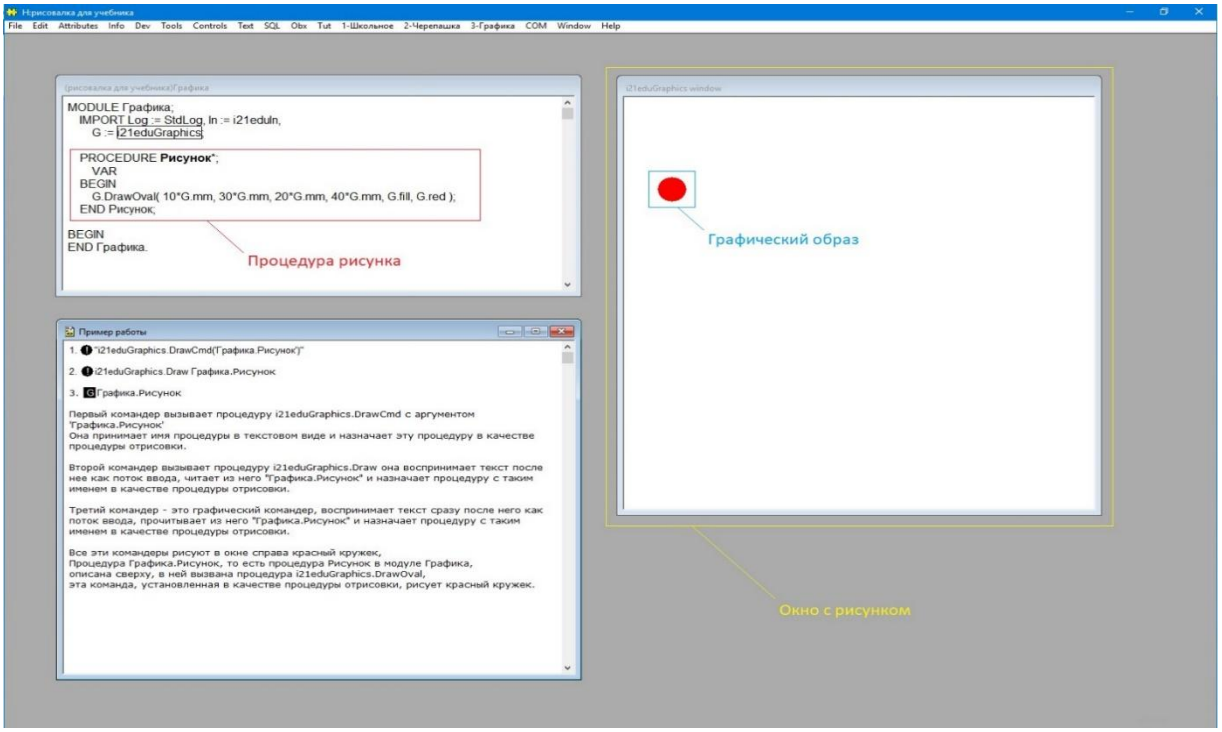


Рисунок 2.5 – Пример работы с системой

Как говорилось ранее, одной из основных черт проекта Оберон в целом была идея о максимально возможной простоте разрабатываемых технологий и продуктов. Оценив современные тенденции к развитию программирования и основные требования пользователей к разрабатываемым программам, разработчики проекта BlackBox пришли к выводу о том, что использование модальных окон при программировании является устаревшей и неудобной технологией, и не включили возможность создания модальных окон в реализацию своего продукта. Все диалоговые окна немодальные, за исключением очень малого числа встроенных в операционную систему, таких как стандартное окно открытия файлов. В общем случае, вы можете всегда оставить диалоговое окно и вернуться к нему позже.

Другой особенностью среды разработки является механизм обработки сообщений (то есть пользовательских событий: щелчков мыши, нажатий клавиш и т.д.). Система BlackBox скрывает цикл обработки событий от программистов прикладных программ. Он даже скрывает и платформо-зависимые черты, такие как окна и меню. Этого удалось добиться, сосредоточившись на абстракциях, которые представляют содержимое окна: так называемый тип View.

Современные тенденции развития программного обеспечения сводятся к все более широкому использованию клиент-серверных технологий. Безусловно, ни одна современная среда разработки программного обеспечения не может обойтись без поддержки этой возможности. Подсистема Sql BlackBox Component Framework обеспечивает простой интерфейс распределенных IT объектов, специализированный для доступа к реляционным базам данных (SQL-серверам). Более общие распределенные объекты, основанные на технологии DCOM (COM), могут быть доступны и реализованы при помощи компилятора Direct-To-COM для языка Компонентный Паскаль. Таким образом, выбор Компонентного Паскаля очевиден, он прост в освоении, но одновременно позволяет решать сложные задачи.

Выводы по второй главе

В данной главе было описано логическое проектирование разрабатываемой обучающей системы, произведен выбор технологии логического моделирования, построены диаграммы вариантов использования, спроектирована логическая модель данных и выработаны требования к аппаратно-программному обеспечению разрабатываемой системы, приведены примеры ее интерфейсов.

Глава 3 Проектирование и разработка инструментальной обучающей среды на компонентном паскале

3.1 Общие сведения о системе

Целью разработки Среды является ее внедрение в учебную деятельность школы. Для младших классов школы часто используются такие элементы обучающей системы Кумир как Робот или Черепашка. Они нужны для наиболее наглядной демонстрации основных принципов алгоритмизации.

Заметим, что возможности системы BlackBox позволяют добавить в нее модули, аналогичные модулю Черепашка и тогда обучение школьников с самого начала будет происходить в единой среде (на данный момент во многих школах практикуется переход от Школьного Алгоритмического Языка к Паскалю).

Задача данной работы – смоделировать и создать Среду, с которой может работать и учитель, и ученик одновременно.

В системе можно работать с графической и видео информацией, рисовать и делать небольшие мультфильмы.

Есть возможность добавлять новые курсы по различным предметам, использовать для построения графиков, при проведении лабораторных работ. При этом Среда должна быть достаточно простой для внедрения её в школьную систему образования.

Основными требованиями являются:

1. Обеспечить отображение диалогового окна рисунка, в котором будут размещаться все графические примитивы. Данное окно будет необходимым для использования графических примитивов, так как именно в нем будет происходить их отрисовка.
2. Обеспечить задание следующих свойств для окна рисунка:

a. Ширина;

b. Высота;

c. Имя процедуры для отрисовки изображения.

3. Обеспечить отображение в окне рисунка следующих графических переменных:

d. Прямая линия;

e. Овал;

f. Ломаная линия или кривая Безье;

g. Прямоугольник;

h. Текст.

4. Обеспечить возможность отображения видео в окне рисунка. Видео может находиться в одном из двух состояний: проигрываться или быть на паузе.

Разработанная система должна быть оформлена в виде отдельного модуля, подгружаемого к системе BlackBox, что позволит использовать ее в процессе обучения школьников Компонентному Паскалю.

3.2 Описание разработанной системы

Рассмотрим основные процедуры разработанной программы:

i21eduGraphicsCommanders – *Модуль обеспечивает работу графического коммандера.*

View* = POINTER TO LIMITED RECORD (Views.View) END – *Тип (Класс), который унаследован от Views.View и реализующий объект отрисовки для графических коммандеров.*

i21eduGraphics – *Модуль обеспечивает работу графического окна, на котором можно рисовать, используя команды этого модуля,* описать сущность программной компоненты "рисовалка" и ввести там понятие "Окно рисунка" и описать его пользовательские свойства

View* = POINTER TO RECORD (Views.View) END;

Класс, который унаследован от Views.View, следовательно реализует объект отрисовки.

Глобальные переменные

drawing: RECORD

name: ARRAY 512 OF CHAR;

proc: DrawingProc;

view: View;

prefW, prefH: INTEGER;

END;

*Переменная, которая хранит данные об **"окне рисунка"**:*

name – имя процедуры которая рисует изображение в текстовом виде.

proc – указатель на процедуру.

view – объект отрисовки, отображаемый в "окне рисунка".

prefW, prefH – ширина и высота "окна рисунка" по умолчанию.

frame: Views.Frame;

Объект доступа для отрисовки окна рисунка.

font: Fonts.Font;

Указатель на шрифт, используемый процедурами для отрисовки цепочек литер в окне рисунка.

Movie types

MovieAction = POINTER TO RECORD (Services.Action) END;

класс, унаследованный от Services.Action, обеспечивает выполнение некоторой процедуры через определенный промежуток времени.

movie: RECORD

state: INTEGER;

restartTicks: Ticks;

pauseTime: Seconds;

action: MovieAction;

END;

Переменная которая хранит данные для отрисовки видео в окне рисунка:

state – Состояние видео проигрывателя, принимает два значения: moviePaused (видео на паузе), movieRunning (видео проигрывается).

restartTicks, pauseTime – используются, чтобы корректно вычислить время воспроизведения, с учетом пауз видео.

action – указатель типа MovieAction, обеспечивает смену кадров в видео режиме, через равные промежутки времени вызывая перерисовку ***окна рисунка***

PROCEDURE **Empty***;

Процедура, рисующая "пустой" рисунок. Она используется по умолчанию, когда окно рисунка только создано.

PROCEDURE SetDefaultDrawing;

Устанавливает в качестве ***процедуры рисунка*** *Empty*

PROCEDURE UpdateDrawingProc;

На момент вызова UpdateDrawingProc задан command.name – имя текущей процедуры рисунка,

по нему определяет command.proc – указатель на эту процедуру.

Для чего пытается загрузить модуль содержащий эту процедуру, если он ещё не загружен.

PROCEDURE UpdateDrawingName;

на момент вызова UpdateDrawingName задан command.proc – указатель на процедуру;

по нему определяет command.name – имя рисующей процедуры.

View – тип (класс) объекта отрисовки, показываемого в окне рисунка.

PROCEDURE (v: View) **HandlePropMsg-** (VAR p: Views.PropMessage);

PROCEDURE (v: View) **Restore*** (f: Views.Frame; l, t, r, b: INTEGER);

PROCEDURE (v: View) **Externalize-** (VAR wr: Stores.Writer);

PROCEDURE (v: View) **Internalize-** (VAR rd: Stores.Reader);

PROCEDURE (v: View) **CopyFromSimpleView-** (source: Views.View);

PROCEDURE (v: View) **HandleCtrlMsg*** (f: Views.Frame;
VAR msg: Views.CtrlMessage;
VAR focus: Views.View);

the global singlet Drawing

PROCEDURE **InitSize*** (w, h: INTEGER);

устанавливает размеры окна рисунка

должно вызываться до его открытия

PROCEDURE ViewWindow (): Windows.Window;

Возвращает указатель на окно рисунка, если оно не открыто, то возвращает NIL

PROCEDURE OpenView;

Открывает окно рисунка, если они еще не открыто.

PROCEDURE CloseView;

Закрывает окно рисунка, если оно открыто.

PROCEDURE **DrawProc*** (proc: PROCEDURE);

*Устанавливает процедуру proc в качестве **процедуры рисунка**.*

PROCEDURE **DrawCmd*** (cmd: ARRAY OF CHAR);

*Устанавливает процедуру с именем cmd в качестве **процедуры рисунка**.*

PROCEDURE **Draw***;

***процедура рисунка** берется через коммандер; выбор потока ввода – как In.Open:*

i21eduGraphics.Draw ЗдесьНачинаетсяПотокВвода

Movie

PROCEDURE **Time*** (): Seconds;

Время воспроизведения в секундах, перестает меняться когда видео на паузе, возобновляется при продолжении просмотра,

*до включения видео режима возвращает ноль. **Процедура рисунка** может использовать процедуру Time в своем алгоритме, тогда рисунок будет зависеть от времени в режиме видео.*

PROCEDURE InitTime;

PROCEDURE StartTime;

PROCEDURE PauseTime;

PROCEDURE ContinueTime;

PROCEDURE SetupNextFrame;

Процедуры, используемые дальнейшими процедурами, выделены для удобства написания кода.

PROCEDURE (ma: MovieAction) Do;

Метод, предопределённый для Services.Action, реализован для обеспечения периодической отрисовки кадров в видео режиме.

PROCEDURE **StartMovie***;

Запускает видео, включая видео режим для окна рисунка.

PROCEDURE **ContinueMovie***;

Возобнавляет видео.

PROCEDURE **PauseMovie***;

Ставит видео на паузу.

PROCEDURE **StartMovieCmd*** (frameName: ARRAY OF CHAR);

Устанавливает процедуру с именем frameName в качестве процедуры рисунка и переводит окно рисунка в видео режим.

PROCEDURE **StartMovieProc*** (frameProc: PROCEDURE);

Устанавливает процедуру frameProc в качестве ***процедуры рисунка*** *и переводит* ***окно рисунка*** *в видео режим.*

цвет и шрифт

PROCEDURE **RGBColor*** (red, green, blue: INTEGER): Color;

Получает красную, зеленую и синюю компоненты и возвращает цвет

PROCEDURE **InvertedColor*** (c: Color): Color;

Получает цвет как аргумент и возвращает инвертированный цвет.

Следующие процедуры меняют значение переменной font, задавая параметры шрифта для процедур, отрисовывающих текст:

PROCEDURE **SetFont*** (typeface: ARRAY OF CHAR; size: INTEGER; weight: INTEGER);

Устанавливает гарнитуру, размер и толину шрифта.

PROCEDURE **SetTypeface*** (typeface: ARRAY OF CHAR);

Устанавливает гарнитуру шрифта.

PROCEDURE **SetSize*** (size: INTEGER);

Устанавливает размер шрифта.

PROCEDURE **SetBold***;

Устанавливает жирный шрифт.

PROCEDURE **SetItalic***;

Устанавливает курсив.

PROCEDURE **SetUnderline***;

Включает подчёркивание.

PROCEDURE **SetStrikeout***;

Включает зачеркивание.

PROCEDURE **SetNormal***;

Отменяет эффекты комманд SetBold, SetItalic, SetUnderline, SetStrikeout.

PS: После вызова SetBold и SetItalic вновь печатаемый текст будет жирным и наклоненным, действия этих комманд суммируются.

Примитивы рисования

Следующие процедуры предоставляют графические примитивы для рисования в окне рисунка, скрывая от пользователя соответствующий объект доступа.

PROCEDURE **DrawLine*** (x0, y0, x1, y1, s: INTEGER; col: Color);

Отрезок прямой

PROCEDURE **DrawOval*** (l, t, r, b: INTEGER; s: INTEGER; col: Color);

Овал

PROCEDURE **DrawPath*** (IN p: ARRAY OF Point; n: INTEGER; s: INTEGER; col: Color; path: INTEGER);

ломанная или кривая Безье.

PROCEDURE **DrawRect*** (l, t, r, b: INTEGER; s: INTEGER; col: Color);

Прямоугольник

Следующие процедуры используются для форматирования и отрисовки текстов в окне рисунка:

PROCEDURE **CharIndex*** (x, pos: INTEGER; IN s: ARRAY OF CHAR): INTEGER;

PROCEDURE **CharPos*** (x, index: INTEGER; IN s: ARRAY OF CHAR): INTEGER;

PROCEDURE **DrawString*** (x, y: INTEGER; col: Color; IN s: ARRAY OF CHAR

); *рисует строку из юникодных символов*

PROCEDURE **SCharIndex*** (x, pos: INTEGER; IN s: ARRAY OF SHORTCHAR): INTEGER;

PROCEDURE **SCharPos*** (x, index: INTEGER; IN s: ARRAY OF SHORTCHAR): INTEGER;

PROCEDURE **DrawSString*** (x, y: INTEGER; col: Color; IN s: ARRAY OF SHORTCHAR);

рисует строку из АСКИ символов

Инициализация модуля

PROCEDURE Init;

Вызывается при загрузке модуля i21eduGraphics и подготавливает его к работе.

Многофункциональная система состоит из двух подсистем: графики **i21eduGraphics** и коммандера **i21eduGraphicsCommanders.**

Подсистему графики можно условно назвать библиотекой процедур (процедуры описаны выше). Учащиеся с помощью коммандера выбирают необходимые им процедуры (операторы языка к данному моменту должны быть хорошо усвоены) и решают поставленную задачу: рисуют, используют при оформлении лабораторных работ по физике, разрабатывают модели, делают мультфильмы.

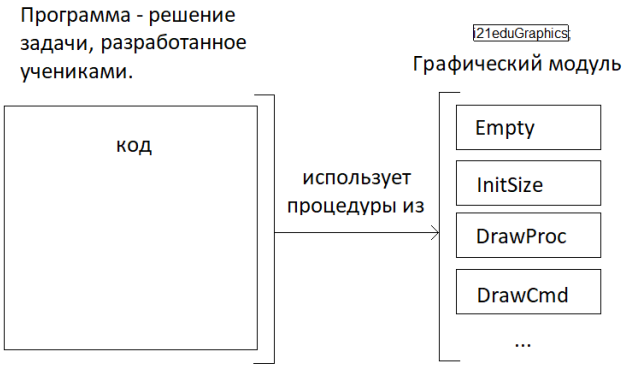


Рисунок 3.1 – схематичное изображение работы с i21eduGraphics.png

На рисунке 3.3 показан фрагмент окна системы.

На рисунке видно, что для работы с системой открывается специальные

окна, например, есть специальное окно Графика.

Разработанная многофункциональная среда может быть присоединена в качестве модуля к системе BlackBox и может быть использована для обучения детей основам программирования в школе. При этом все задания, предложенные школьникам для реализации на языке Паскаль в рамках текущего курса обучения в школе, могут быть легко модернизированы для работы в среде BlackBox.

Рисунок 3.2 – схематичное изображение работы, рисующее Красный кружок

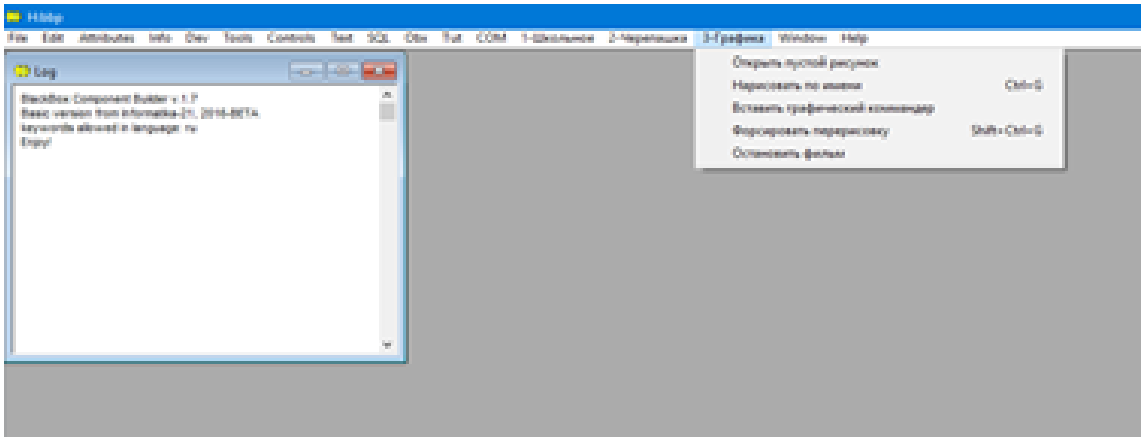


Рисунок 3.3 – Фрагмент окна системы

Детям школьного возраста часто тяжело дается работа с текстовой и численной информацией. При этом работа с графикой и видео более популярна среди них. То есть предположительно, задания по работе с графической и видео информацией интереснее для детей и развивают не только навыки

программирования, но и творческие способности. Модули графики и коммандера позволяют ученику решать задачу, не думая о том, как работать с классами View, Frame, Action, при этом являются «авторской обложкой» над стандартными графическими библиотеками Black Box.

Примеры работы с системой Учитель-Ученик - учитель создает ссылку на изучаемый материал.

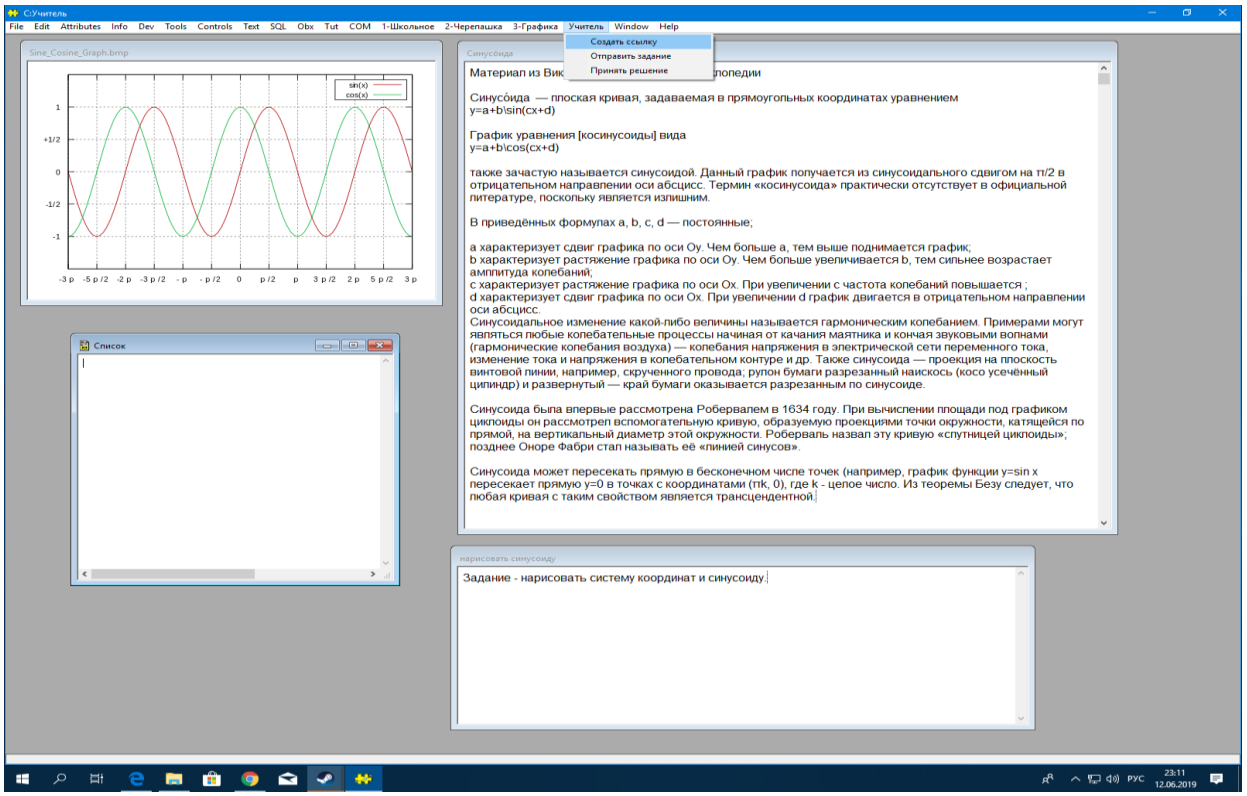


Рисунок 3.4 - Создание ссылки на изучаемый материал

Учитель отправляет задание ученику.

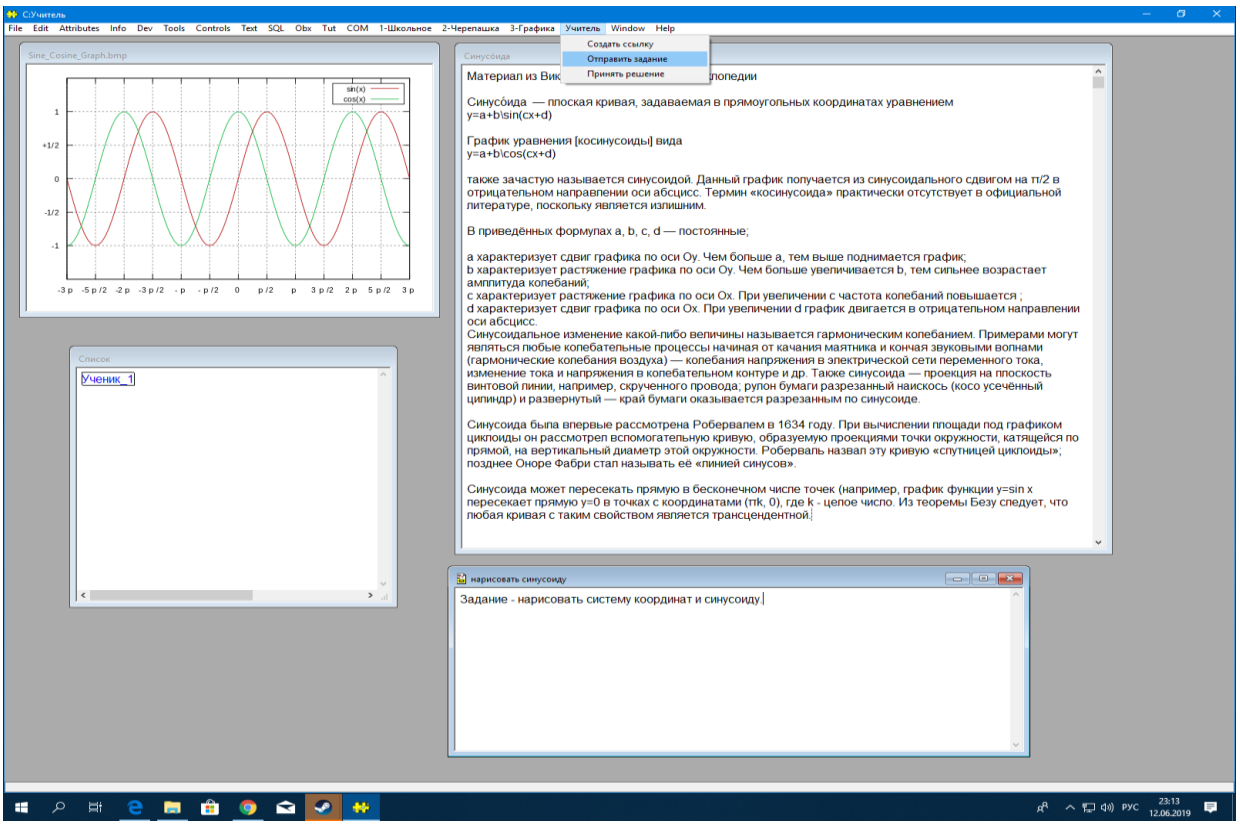


Рисунок 3.5 - Отправка задания ученику

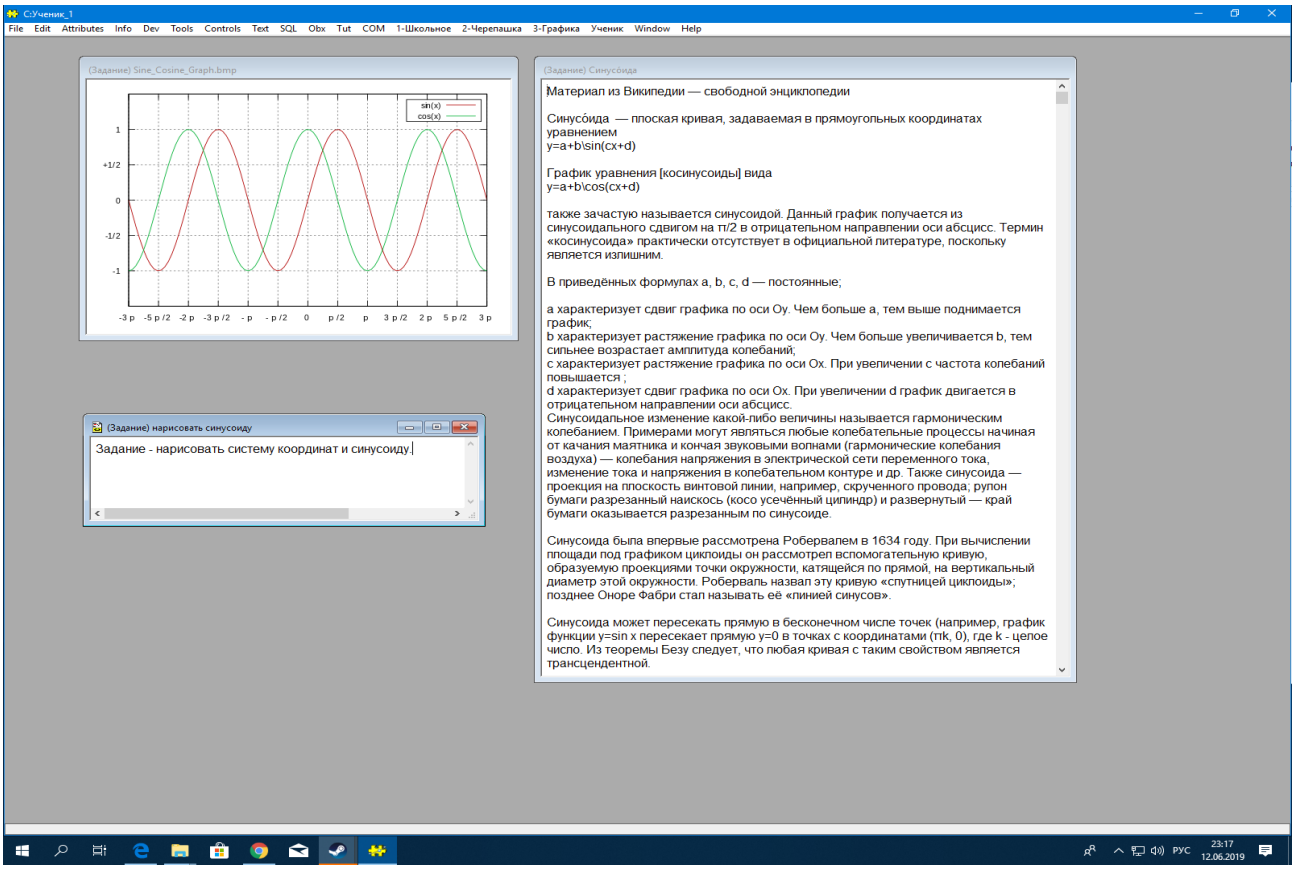


Рисунок 3.6 - Получение задания учеником

Ученик решает задание и отправляет на проверку.

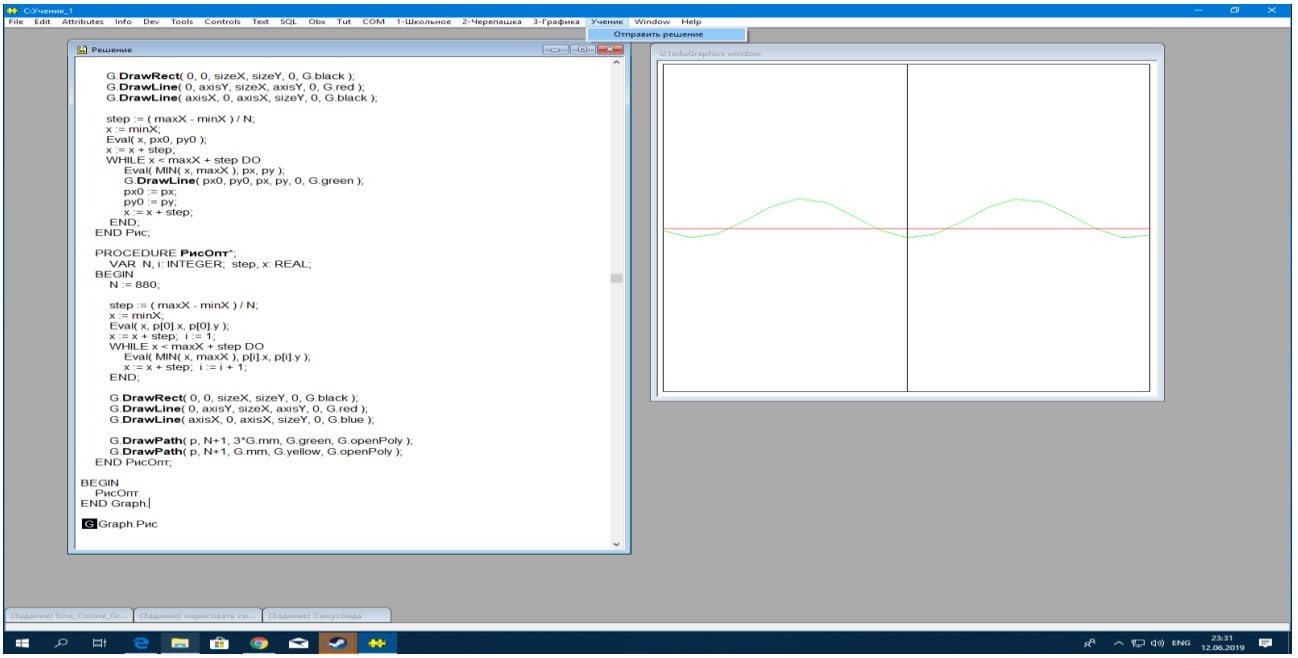


Рисунок 3.7 – Отправка задания на проверку

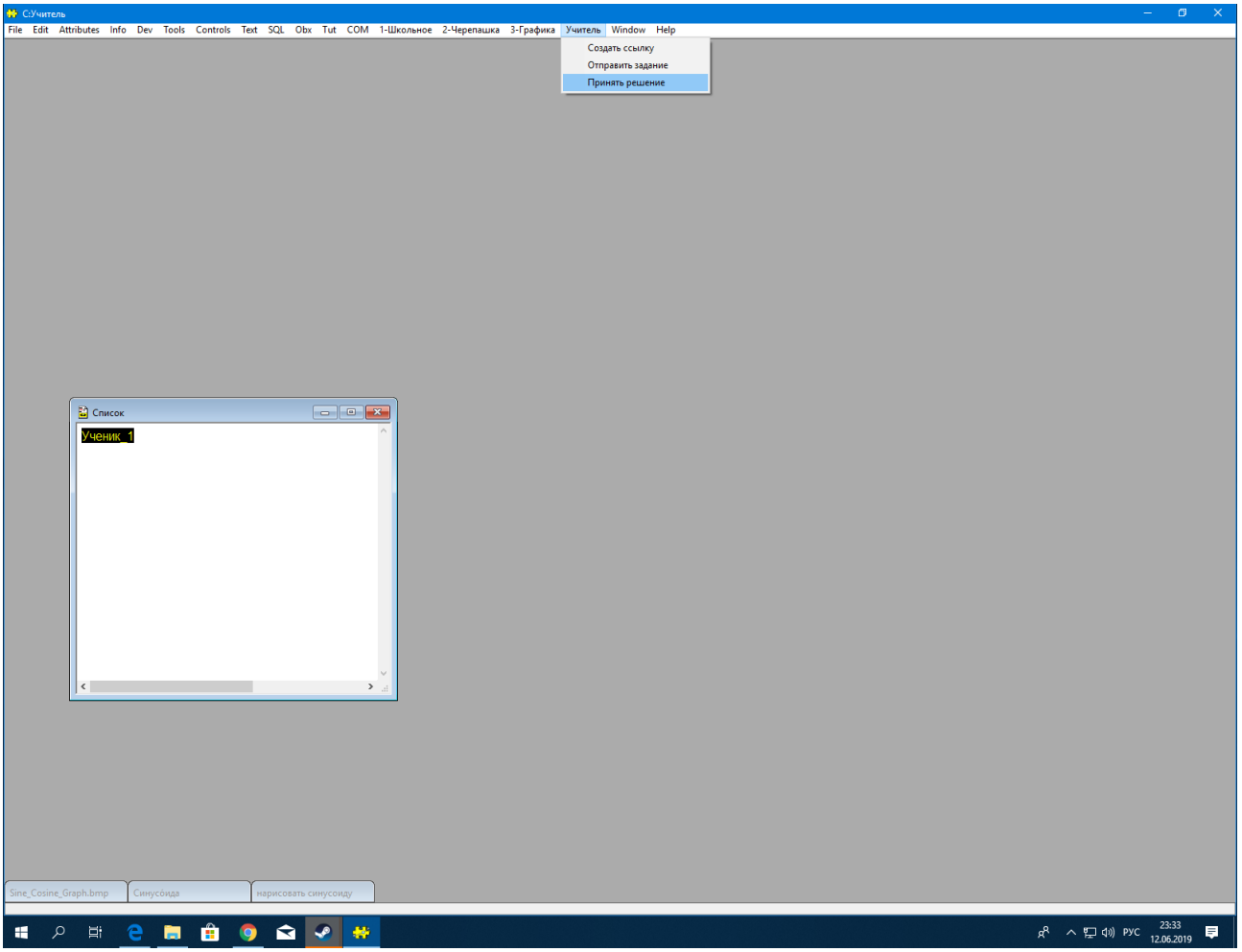


Рисунок 3.8 - Учитель принимает решение у конкретного ученика

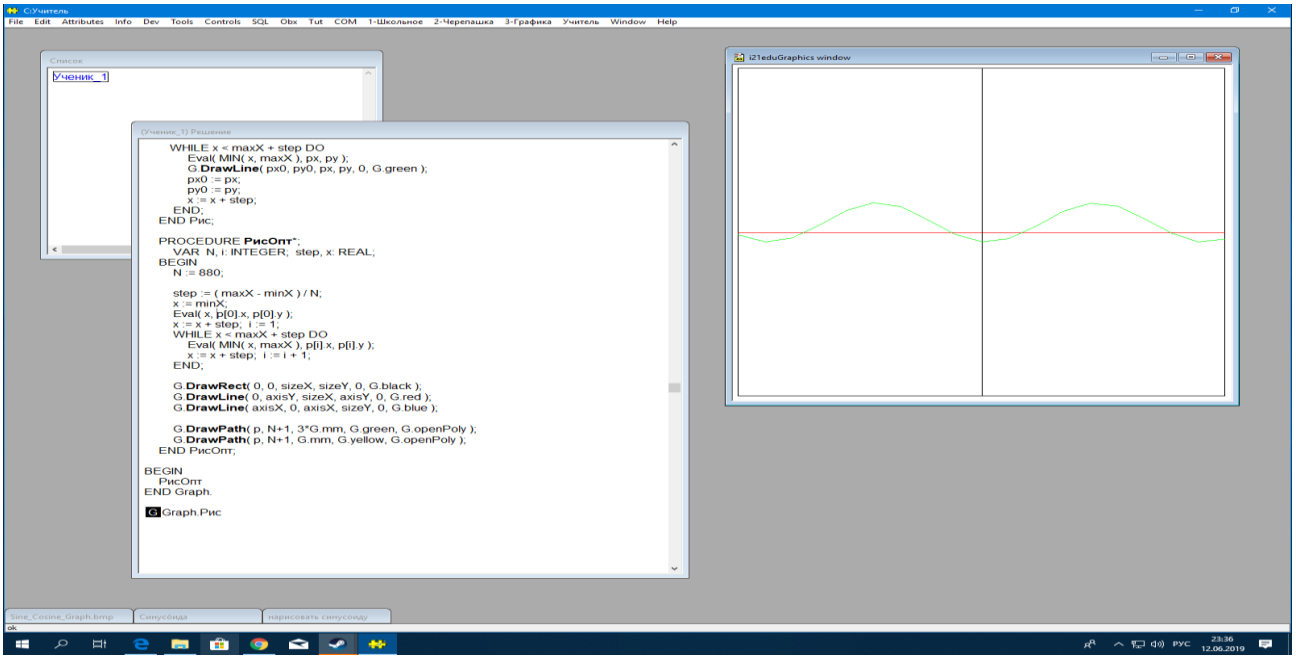


Рисунок 3.9 - Проверка решения учителем

Примеры задач, решенные учениками:

Задача расчета траектории полета:

```
MODULE BinomN122matmod;
(* здесь нужны комментарии в тексте задачника насчёт диалоговой формы и т.п.
добавить справку и кнопку в диалог*)
    IMPORT  Log := StdLog,  In := i21eduIn,  Math,
        G := i21eduGraphics, Dialog;

    CONST  g = 9.81;  mm = G.mm;  scale = G.mm;

    VAR
        начСкорость*: REAL;
        угол*: REAL;
        уголРад: REAL;
        параметры*: RECORD
            высота-,  дальность-,  время-: REAL
        END;

    PROCEDURE Round ( x: REAL ): INTEGER;
    BEGIN  RETURN  SHORT(ENTIER( Math.Round( x ) ))
    END Round;

    PROCEDURE Дальность ( v0, a: REAL ): REAL;
```

```
BEGIN  RETURN  Math.Power( v0, 2 ) * Math.Sin( 2*a ) / g;
END Дальность;

PROCEDURE Высота ( v0, a: REAL ): REAL;
BEGIN  RETURN  v0 * Math.Power( Math.Sin( 2*a ), 2 ) / 2 / g;
END Высота;

PROCEDURE Время ( v0, a: REAL ): REAL;
BEGIN  RETURN  2 * v0 * Math.Sin( a ) / g;
END Время;

PROCEDURE Рис*;
        VAR  t, T, x, y, x0, y0: REAL;  bkg, col: G.Color;
BEGIN
        bkg := G.grey6;
        col := G.black;
        G.DrawRect( 0, 0, 100*mm, 100*mm, G.fill, bkg );
        G.DrawLine( 10*mm, 40*mm, 60*mm, 40*mm, 0, col );
        G.DrawLine( 10*mm, 40*mm, 10*mm, 20*mm, 0, col );
        G.DrawString( 10*mm, 10*mm, col, 'ТРАЕКТОРИЯ ПОЛЕТА' );
        G.DrawString( 8*mm, 21*mm, col, 'Y' );
        G.DrawString( 58*mm, 42*mm, col, 'X' );

        t := 0;
        T :=  Время( начСкорость, уголРад );
        x0 := 0;
        y0 := 0;
        WHILE  t < T  DO
                y := начСкорость * Math.Sin( уголРад ) * t - g * t * t / 2;
                x := начСкорость * Math.Cos( уголРад ) * t;
                G.DrawLine(
                        10 * mm + Round( scale * x0 ),
                        40 * mm - Round( scale * y0 ),
                        10 * mm + Round( scale * x ),
                        40 * mm - Round( scale * y ),
                        0,
                        col
                );
                x0 := x;
                y0 := y;
                t := t + 0.025
        END
END Рис;

PROCEDURE Do*;
```

```
        VAR  ok: BOOLEAN;  s: CHAR;  r: INTEGER;
    BEGIN
        (* ввод:
            начальная скорость – веществ,
            угол – веществ,
            код параметра, который надо вычислить (1, 2, 3),
            надо ли рисовать траекторию (литера y/n
        *)

        In.Open( ok );  ASSERT( ok );
        In.Real( начСкорость, ok );  ASSERT( ok & ( начСкорость > 0 ));
        In.Real( угол, ok );  ASSERT( ok & ( 0 < угол ) & ( угол < 90 ) );
        уголРад := угол * Math.Pi()/180;

        In.Int( r, ok );  ASSERT( ok ); (* корректность значения проверяется
отдельно, чтобы выдать сообщение *)
        In.Char( s, ok );  ASSERT( ok ); (* пропустить пробел *)
        In.Char( s, ok );  ASSERT( ok );

        IF r = 1 THEN
            Log.Real( Дальность( начСкорость, уголРад ) )
        ELSIF r = 2 THEN
            Log.Real( Высота( начСкорость, уголРад ) )
        ELSIF r = 3 THEN
            Log.Real( Время( начСкорость, уголРад ))
        ELSE
            Log.String('Неверно набран код параметра');
            HALT(123);
        END;
        Log.Ln;

        IF  s='y'  THEN  Рис  END
    END Do;

    PROCEDURE Notifier* ( op, from, to: INTEGER );
        VAR
    BEGIN
        уголРад := угол*Math.Pi()/180;
        параметры.высота := Высота( начСкорость, уголРад );
        параметры.дальность := Дальность( начСкорость, уголРад );
        параметры.время := Время( начСкорость, уголРад );
        Dialog.Update( параметры )
    END Notifier;

END  BinomN122matmod.
```

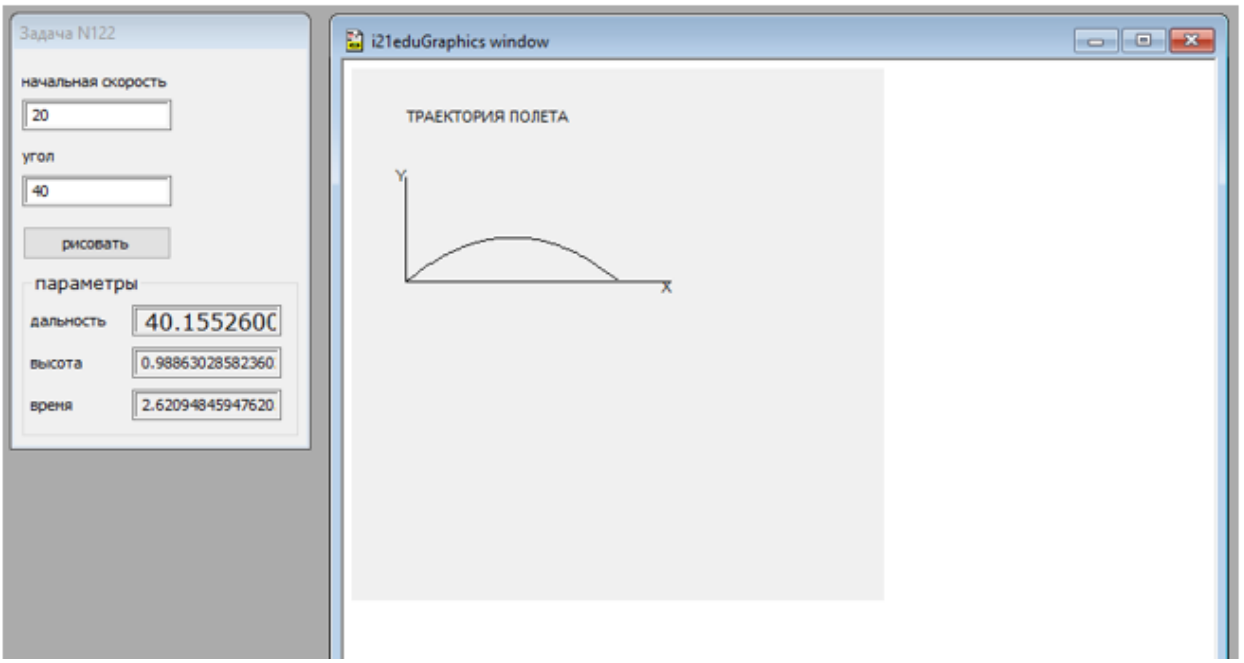


Рисунок 3.10 – Иллюстрация к задаче Расчет траектории полета

Задача нарисовать квадрат с градиентом

(Много разноцветных кружков нарисованных близко друг к другу создают иллюзию квадрата с градиентом).

```
MODULE BinomN124ellips;
(**  Набор эллипсов разного размера и цвета  **)
    IMPORT  G := i21eduGraphics;

    PROCEDURE ЦветныеКруги*;
        CONST  rad = 5 * G.mm;  d = G.mm DIV 2;
        VAR  col: G.Color;  x, y, l, t, r, b, i: INTEGER;
    BEGIN
        x := 20 * G.mm;
        y := 10 * G.mm;

        i := 0;
        t := y - rad;
        b := y + rad;
        WHILE i < 256 DO
            x := x + d;
            l := x - rad;
            r := x + rad;
            col := G.RGBColor( 255, i, 0 );
            G.DrawOval( l, t, r, b, G.fill, col );
            i := i+1
```

```
        END;

        i := 0;
        l := x - rad;
        r := x + rad;
        WHILE i < 256 DO
                y := y + d;
                t := y - rad;
                b := y + rad;
                col := G.RGBColor( 255 - i, 255, 0 );
                G.DrawOval( l, t, r, b, G.fill, col );
                i := i+1
        END;

        i := 0;
        t := y - rad;
        b := y + rad;
        WHILE i < 256 DO
                x := x - d;
                l := x - rad;
                r := x + rad;
                col := G.RGBColor( 0, 255 - i, i );
                G.DrawOval( l, t, r, b, G.fill, col );
                i := i+1
        END;

        i := 0;
        l := x - rad;
        r := x + rad;
        WHILE i < 256 DO
                y := y - d;
                t := y - rad;
                b := y + rad;
                col := G.RGBColor( i, 0, 255 - i );
                G.DrawOval( l, t, r, b, G.fill, col );
                i := i+1
        END;

    END ЦветныеКруги;

END BinomN124ellips.
```

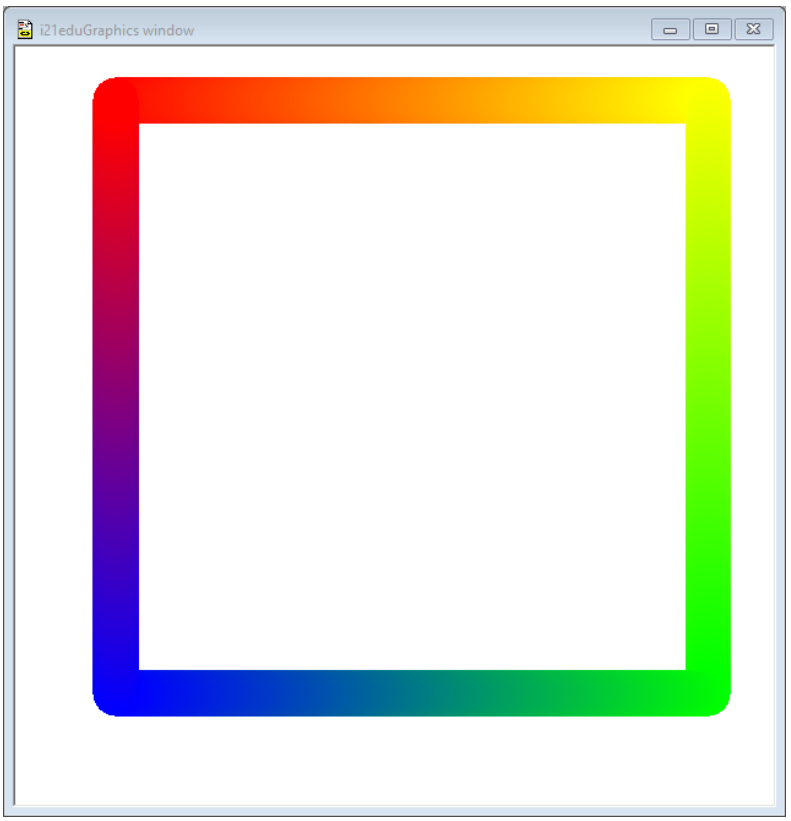


Рисунок 3.11. – Иллюстрация к задаче квадрат с градиентом

Демонстрация кривых Безье

```
MODULE BinomNbezier;
	IMPORT Math,  Lib := BinomLib,  G := i21eduGraphics;


	PROCEDURE DrawBox ( x, y: INTEGER );
		CONST  s = G.mm;
		VAR  l, t, r, b: INTEGER;
	BEGIN
		l := x - s;  t := y - s;
		r := x + s;  b := y + s;
		G.DrawRect( l, t, r, b, 0, G.red );
	END DrawBox;
	PROCEDURE DrawSlice* ( x0, y0, r, s: INTEGER;  a0, a1: REAL;  col:
G.Color );
		VAR  p: ARRAY 120 OF G.Point;  i, nSegm, N: INTEGER;  da, l, z:
REAL;
	BEGIN
		nSegm := MAX( 1, Lib.Round( ABS(a1 - a0) / 10 ) );
		N := 3*nSegm;
		z := Math.Pi() / 180;

		a0 := a0 * z;
```

```
		a1 := a1 * z;

		da := ( a1 - a0 ) / nSegm;
		l := da * r / 3;

		i := 0;
		WHILE i <= N - 3 DO
			p[i+0].x := x0 + Lib.Round( r * Math.Cos( a0 ) );
			p[i+0].y := y0 - Lib.Round( r * Math.Sin( a0 ) );
			(* DrawBox( p[ i ].x, p[ i ].y ); *)

			p[i+1].x := p[i+0].x - Lib.Round( l * Math.Sin( a0 ) );
			p[i+1].y := p[i+0].y - Lib.Round( l * Math.Cos( a0 ) );

			a1 := a0 + da;
			p[i+3].x := x0 + Lib.Round( r * Math.Cos( a1 ) );
			p[i+3].y := y0 - Lib.Round( r * Math.Sin( a1 ) );

			p[i+2].x := p[i+3].x + Lib.Round( l * Math.Sin( a1 ) );
			p[i+2].y := p[i+3].y + Lib.Round( l * Math.Cos( a1 ) );

			i := i + 3;  a0 := a1
		END;
		(* p[i+0] - это бывшая p[i+3] *)
		p[i+3].x := x0;
		p[i+3].y := y0;
		p[i+1] := p[i+3];
		p[i+2] := p[i+0];

		i := i + 3;
		p[i+1] := p[0];
		p[i+2] := p[i+0];

		G.DrawPath( p, N + 6, s, col, G.closedBezier );

(*
		G.DrawLine( x0, y0, p[ 0 ].x, p[ 0 ].y, 0, col );
		G.DrawLine( x0, y0, p[ N ].x, p[ N ].y, 0, col );
*)
	END DrawSlice;

	PROCEDURE TestDrawSlice*;
		VAR  x0, y0, r: INTEGER;
	BEGIN
		x0 := 50*G.mm;
```

```
        y0 := 50*G.mm;
        r := 40*G.mm;
        DrawSlice( x0, y0, r, G.fill, 180, 15, G.black );
        r := r + G.mm DIV 2;
        G.DrawOval( x0-r, y0-r, x0+r, y0+r, 0, G.red );
    END TestDrawSlice;

    PROCEDURE TestDrawSlice2*;
        VAR  x0, y0, r: INTEGER;
    BEGIN
        x0 := 50*G.mm;
        y0 := 50*G.mm;
        r := 40*G.mm;
        DrawSlice( x0, y0, r, r DIV 2, 180, 30, G.green );
        r := r + G.mm DIV 2;
        G.DrawOval( x0-r, y0-r, x0+r, y0+r, 0, G.red );
    END TestDrawSlice2;

BEGIN
END BinomNbezier.
```

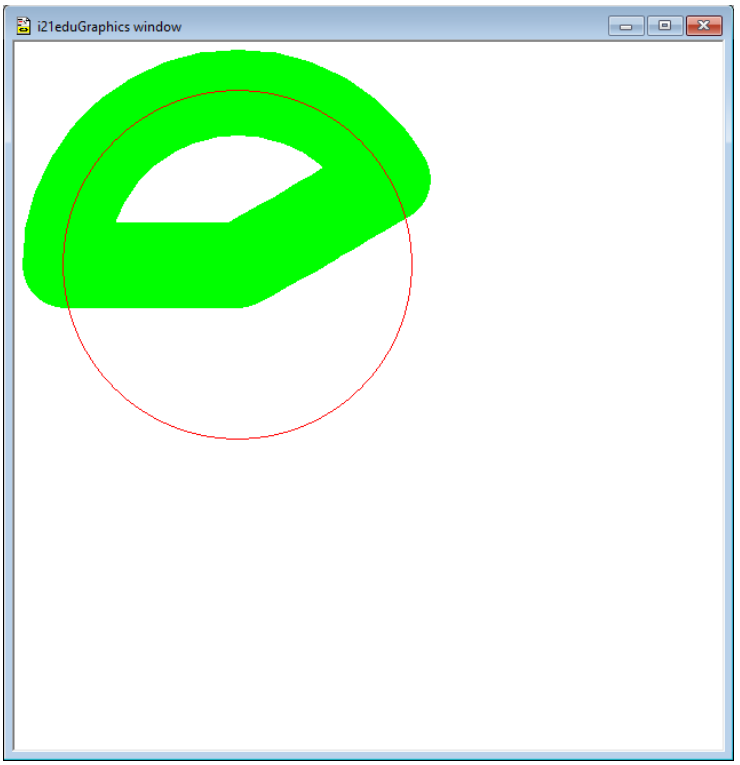


Рисунок 3.12 – Демонстрация кривых Безье

Построение графика функции, заданной формулой

```
MODULE BinomN125yxsinx_N127parab;
    IMPORT Log := StdLog, In := i21eduIn, Math,
        G := i21eduGraphics,  Lib := BinomLib;
```

```
CONST
        minX = -10;  maxX = 10;
        minY = -10;  maxY = 10;

        sizeX = G.defaultSize;
        sizeY = sizeX;

        scaleX =  sizeX / ( maxX - minX );
        scaleY =  sizeY / ( maxY - minY );

        axisX = SHORT( ENTIER( -minX * scaleX ) );
        axisY = SHORT( ENTIER( -minY * scaleY ) );

TYPE Function = PROCEDURE( x: REAL ): REAL;

VAR  p: ARRAY 1000 OF G.Point;

PROCEDURE Eval ( f: Function;  x: REAL;  OUT px, py: INTEGER );
        VAR  y: REAL;
BEGIN
        y := f( x );
        px := Lib.Round( scaleX * ( x - minX ) );
        py := Lib.Round( scaleY * ( y + maxY ) );
END Eval;

PROCEDURE F1 ( x: REAL ): REAL;
        CONST a=-1; k=1/2;
BEGIN
        RETURN  a*x*Math.Cos( k*x ) + x*x / 20;
END F1;

PROCEDURE F2 ( x: REAL ): REAL;
        CONST k=2/15;
BEGIN
        RETURN  k*x*x - 3
END F2;

PROCEDURE Рис*;
        VAR  f: Function;  N: INTEGER;
                step, x: REAL;  px0, py0, px, py: INTEGER;
BEGIN
        f := F1;
        N := 18;
```

```
            G.DrawRect( 0, 0, sizeX, sizeY, 0, G.black );
            G.DrawLine( 0, axisY, sizeX, axisY, 0, G.red );
            G.DrawLine( axisX, 0, axisX, sizeY, 0, G.black );

            step := ( maxX - minX ) / N;
            x := minX;
            Eval( f, x, px0, py0 );
            x := x + step;
            WHILE x < maxX + step DO
        Eval( f, MIN( x, maxX ), px, py );
        G.DrawLine( px0, py0, px, py, 0, G.green );
        px0 := px;
        py0 := py;
        x := x + step;
      END;
    END Рис;

    PROCEDURE РисОпт*;
      VAR  f: Function;  N, i: INTEGER;  step, x: REAL;
    BEGIN
      f := F1;
      N := 880;

      step := ( maxX - minX ) / N;
      x := minX;
      Eval( f, x, p[0].x, p[0].y );
      x := x + step;  i := 1;
      WHILE x < maxX + step DO
        Eval( f, MIN( x, maxX ), p[i].x, p[i].y );
        x := x + step;  i := i + 1;
      END;

      G.DrawRect( 0, 0, sizeX, sizeY, 0, G.black );
      G.DrawLine( 0, axisY, sizeX, axisY, 0, G.red );
      G.DrawLine( axisX, 0, axisX, sizeY, 0, G.blue );

      G.DrawPath( p, N+1, 3*G.mm, G.green, G.openPoly );
      G.DrawPath( p, N+1, G.mm, G.yellow, G.openPoly );
    END РисОпт;

BEGIN
  РисОпт
```

END BinomN125yxsinx_N127parab.

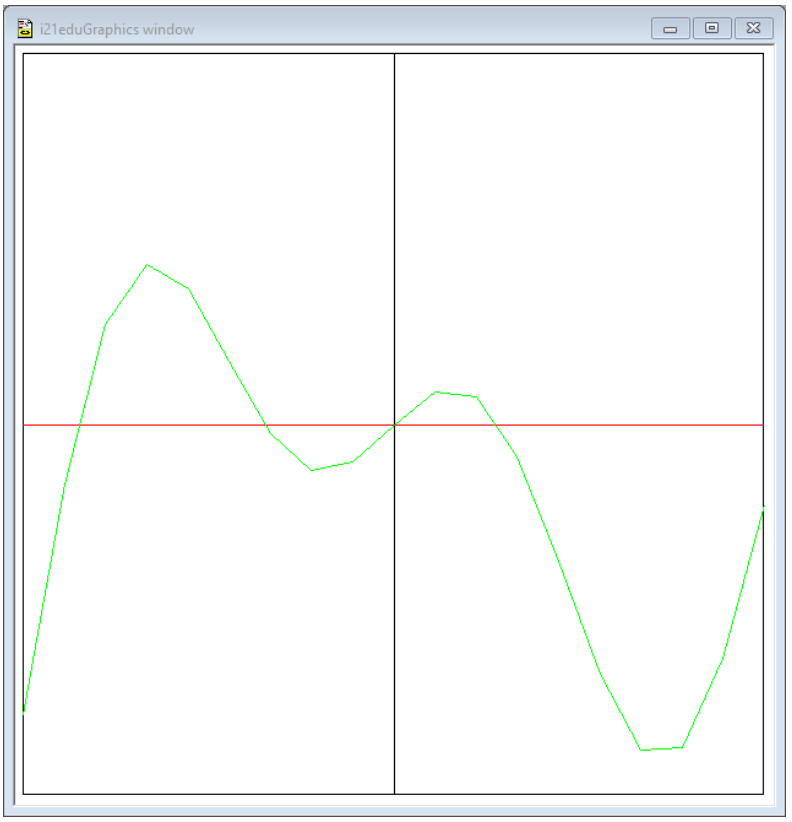


Рисунок 3.13 – График функции, заданный формулой

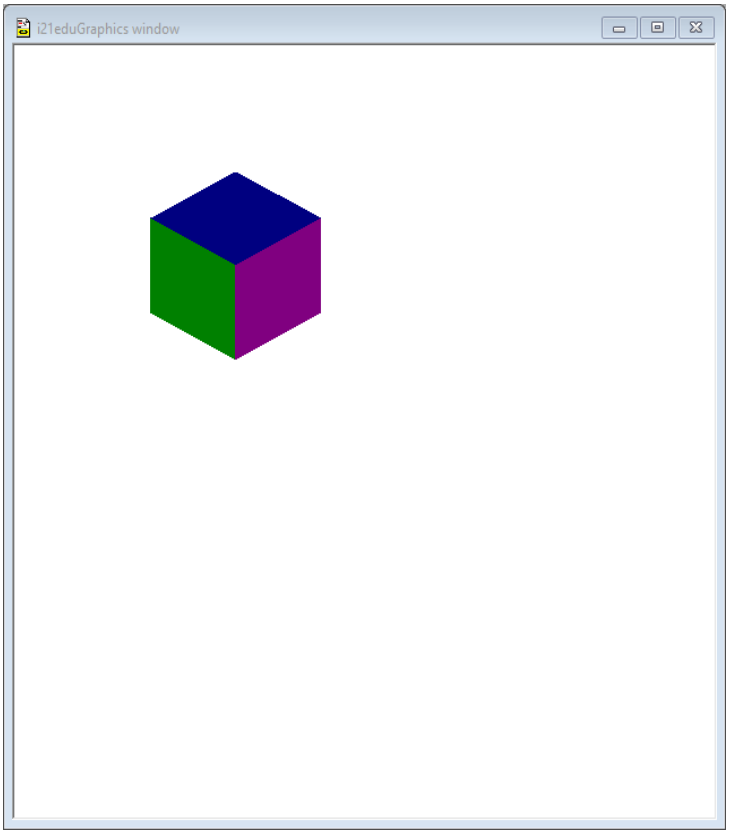


Рисунок 3.14 – Ортографическая проекция куба

Выводы по третьей главе

В третьей главе выпускной квалификационной работы было проведено физическое проектирование автоматизированной обучающей системы и рассмотрены ее функциональные возможности

ЗАКЛЮЧЕНИЕ

В рамках квалификационной выпускной работы была разработана многофункциональная инструментальная обучающая среда на Компонентном Паскале. В рамках данной цели были поставлены и выполнены следующие задачи:

1. Было произведено моделирование обучающей среды и анализ бизнес-процессов использования Component Pascal в обучении школьников.
2. Выполнено проектирование обучающей среды, выявлены преимущества Компонентного Паскаля.
3. Спроектирована и разработана инструментальная обучающая среда на Компонентном Паскале.

Разработанная среда позволяет использовать возможности отрисовки графических примитивов и видео в системе BlackBox. Она может быть внедрена в деятельность любой школы за счет того, что система BlackBox является бесплатной, основана на принципах Паскаля и является его логическим продолжением и поддерживает возможность подключения сторонних модулей.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

Научно-методическая литература

1. Акперов, И.Г. Информационные технологии в менеджменте: Учебник / И.Г. Акперов, А.В. Сметанин, И.А. Коноплева. - М.: НИЦ ИНФРА-М, 2013. – 400 с.

2. Барабас А. А. Применение информационной системы программно-технического комплекса обеспечения процедуры аттестации педагогических работников в рамках внутренней системы оценки качества образования: методические рекомендации /А.А. Барабас. - Челябинск : РЦОКИО, 2017. - 98 с.

3. Венделева, М.А. Информационные технологии в управлении: Учебное пособие для бакалавров / М.А. Венделева, Ю.В. Вертакова. - М.: Юрайт, 2013. - 462 с.

4. Виталий Потопахин. Современное программирование с нуля!. – ДМК Пресс, 2011. – 240 с. – 1000 экз. – ISBN 978-5-94074-665-2.

5. Виталий Потопахин. Современный самоучитель по алгоритмам. – ДМК Пресс, 2012. – 320 с. – 500 экз. – ISBN 978-5-94074-804-5. (старое издание Виталий Потопахин. Искусство алгоритмизации. – ДМК Пресс, 2011. – 320 с. – 1000 экз. – ISBN 978-5-94074-621-8.)

6. Голицына, О.Л. Разработка баз данных: Учебное пособие / О.Л. Голицына, Н.В. Максимов, И.И. Попов. - М.: Форум, 2012. - 400 с.

7. Грекул, В. И. Проектирование информационных систем/ В.И.Грекул, Денищенко Г. Н., Коровкина Н. Л. – М.: Интернет-университет информационных технологий – М.: ИНТУИТ.ру, 2013. с.135

8. Гринберг, А.С. Информационные технологии в управлении / А.С. Гринберг, Н.Н. Горбачев, А.С. Бондаренко.-М.: ЮНИТИ, 2013.-479 с.

9. Диго, С.М. Базы данных: проектирование и использование: [Учеб. для вузов по специальности "Прикладная информатика (по обл.)"] /С.М. Диго.-М.: Финансы и статистика, 2010.-591 с.

10. Зыков С.В. Лекция: Компонентное программирование в .NET / Введение в теорию программирования. Объектно-ориентированный подход. intuit.ru. Проверено 25 октября 2010. Архивировано 13 февраля 2012 года.

11. Ивасенко, А.Г. Информационные технологии в экономике и управлении: [учеб. пособие для вузов по специальностям "Прикладная информатика (по обл.)", "Менеджмент орг.", "Гос. и муницип. упр."] / А. Г. Ивасенко, А. Ю. Гридасов, В. А. Павленко.-М.: КноРус, 2011.-153 с.

12. Информационные системы и технологии в экономике и управлении: [учеб. для вузов по специальности "Прикладная информатика (по обл.)" и др. экон. специальностям] /[В. В. Трофимов и др.] ; под ред. В. В. Трофимова.-М.: Высш. образование, 2010.-480 с.

13. Информационные технологии: [учеб. для студентов вузов, обучающихся по специальности 080801 "Прикладная информатика" и др. экон. специальностям /В. В. Трофимов и др.] ; под ред. проф. В. В. Трофимова.-М.: Юрайт, 2009.-624 с.

14. Карпова, И.П. Проектирование баз данных/ И.П. Карпова. - СПб.: Питер, 2013. - 240 с.

15. Коноплева, И.А. Основы информатики/ И. А. Коноплева, О. А. Хохлова, А. В. Денисов.-М.: Проспект, 2010.-294 с.

16. Луенбергер, Д.Д. Основы информатики /Дэвид Дж. Луенбергер ; пер. с англ. Ю. Л. Цвирко под ред. д.т.н. К. К. Колина.-М.: Техносфера, 2008.-447 с.

17. Маклаков, С.В. CA Erwin Process Modeller r7. Использование Case-средства при разработке информационных систем/ С.В. Маклаков − М. : ДИАЛОГ-МЭФИ, 2009.

18. Максимов, Н.В. Современные информационные технологии: Учебное пособие / Н.В. Максимов, Т.Л. Партыка, И.И. Попов. - М.: Форум, 2013. - 512 с.

19. Никлаус Вирт. Алгоритмы и структуры данных. Новая версия для Оберона. – ДМК Пресс, 2010. – 272 с. – 1000 экз., примеры к книге (для BlackBox Component Builder), английский оригинал

20. Никлаус Вирт. Построение компиляторов. – ДМК Пресс, 2010. – 192 с. – 1000 экз., примеры и дополнительные материалы к книге плюс обновлённый английский оригинал на сайте Oberon Core

21. Пфистер К. «Компонентное программное обеспечение» (Pfister C. «Component Software»)

22. Руденко Ю.С. Современное общество как коммуникативная среда: междисциплинарные аспекты: монография / Ю. С. Руденко, д.п.н., проф.,

23. Трофимов, В.В. Информатика / В.В.Трофимов. - М.: Юрайт, 2013. - 910 с.

24. Шень, А.: Теоремы и задачи по программированию, Москва (1995); Birkhauser (2000)

25. Яковлева О. Н. Подготовка и аттестация научных и педагогических работников : монография / Яковлева О. Н., Мальчук О. И., Гнедова Н. П. - Москва : Буки Веди ФКУ НИИ ФСИН, 2018. - 96 с.

Электронные ресурсы

26. Проект Информатика-21, http://www.inr.ac.ru/~info21/

Литература на иностранном языке

27. Bardin, D., et al.: Project SANC (2003); e.g. http://arxiv.org/abs/hep-ph/0212209

28. C. Szyperski. Component Software – Beyond Object-Oriented Programming. Addison- Wesley, 1998 ISBN 978-0-201-17888-3

29. Gutknecht, J.: in Proc. of JMLC'1997. Lecture Notes in Computer Sciences 1024, Springer Verlag; Reali, P.: Active Oberon Language Report, http://bluebottle.ethz.ch/languagereport/

30. Oberon microsystems, Inc.: Component Pascal Language Report (2001); http://www.oberon.ch

31. Rails Cells: Компонентно-Ориентированное Программирование для Rails (англ.)

32. Tkachov, F.V., Manakova, G.I., Tatarchenko, A.F: How Much Better Than Vegas Can We Integrate in Many Dimensions? FERMILAB-CONF-95-213-T (1995)

33. Tkachov, F.V.: Algebraic Algorithms for Loop Calculations, http://arXiv.org/abs/hep-ph/9609429

34. Wirth, N., Mossenbock, H.: Oberon-2 Language Report (1992)

35. Wirth, N.: Computing Science Education: The Road not Taken, Opening Address at the ITiCSE Conference, Aarhus (2002)

36. Wirth, N.: The Programming Language Oberon, Software – Practice and Experience, 18 (1988) 671-690; Wirth, N., Gutknecht, J.: Project Oberon: the Design of an Operating System and Compiler, ACM Press (1992)