*Министерство образования и науки*

*Республики Казахстан*

*Петропавловский колледж*

*железнодорожного транспорта*

*СПЕЦИАЛЬНОСТЬ*

*«Вычислительная техника и программное обеспечение»*

*ДИПЛОМ*

*на тему: «Разработка тестовой оболочки на  
 Qt Framework»*

*Проверил: Bыполнил:  
Петрищев К.Р. Хайдаев Я.Е.  
 гр. ПО-21*

*2015*

СОДЕРЖАНИЕ

1. ВВЕДЕНИЕ 4
   1. АНАЛИТИЧЕСКАЯ ЧАСТЬ 7
      1. Постановка задачи 7
      2. Математическая реализация 8
   2. ПРАКТИЧЕСКАЯ ЧАСТЬ 15
      1. Описание программы 15
      2. Структура программы на языке С++ и UML-диаграмма 18
2. ЗАКЛЮЧЕНИЕ 24
3. СПИСОК ЛИТЕРАТУРЫ 25
4. ПРИЛОЖЕНИЕ А 26

ВВЕДЕНИЕ

В наши дни ведётся активное внедрение инновационных технологий в различные аспекты жизни такие, как здравоохранение, инженерия, сельскохозяйственная промышленность, телекоммуникации, служба массовой информации и другие аспекты жизни общества. Данная тенденция не обошла стороной и сферу образования в высших учебных заведениях вводятся технологии дистанционного обучения, создаются технологии интерактивного взаимодействия с учебным материалом, а также в обучении не обходим надлежащий контроль полученных знания. Основным способом проверки и оценки усвоенного материала является проведение экзаменов, в данной области, помимо устных методов, вводятся различные тестовые работы для автоматизации проверки правильности ответов, соответственно ускорения и упрощения процесса экзаменирования. В последние годы, в образовании всё чаще применяю компьютерные и сетевые технологии, в том числе и для организации тестирования учащихся.

Целью дынного проекта является разработка программного средства для централизованного и автоматизированного проведения экзаменов в тестовой форме, а также для подготовки к данным экзамена. Одним из важнейших элементов такого программного обеспечения является графический пользовательский интерфейс, так как экзамен – это зачастую волнительный и важный момент жизни, и в обучении, а визуальная составляющая напрямую влияет на психологическое состояние человека. Поэтому интерфейс должен быть максимально эргономичным и благоприятный для умственного сосредоточения: цветовая гамма не должна вызывать эмоционального возбуждения, а напротив успокаивать; форма элементов управления не должна вызывать отвращения и не преодолимого приступа тошноты.

В связи с последним из вышеуказанных аспектов, по мимо основной

разработки программного обеспечения, будет проведено исследование в области психологического восприятия человека, различных цветов и форм объектов, для создания наиболее подходящей графической составляющей приложения.

Эволюция технологии продолжает естественную эволюцию человечества. Освоение каменных орудий помогло сформироваться человеческому интеллекту. Металлические орудия повысили производительность физического труда. Машины механизировали физический труд. На этом пути развития информационная технология освобождает человека от рутинного умственного труда, усиливает его творческие возможности. Информационные технологии (ИТ) являются наиболее важной составляющей процесса использования информационных ресурсов общества. К настоящему времени ИТ прошли несколько эволюционных этапов, смена которых определялась главным образом техническим прогрессом, появлением новых технологических средств поиска и обработки данных. Последний по времени этап, часто называемый новым, характеризуется изменением направленности ИТ с развития технических средств на создания стратегического преимущества в бизнесе.

>>До недавнего времени информация не считалась важнейшим активом для компании. Процесс управления деятельностью организации в большой степени зависел от персонального воздействия первых лиц компаний без обширного процесса координации усилий менеджеров и анализа данных. Деловые решения принимались первыми лицами компаний чаще всего на основе опыта и интуиции, и лишь в небольшом числе случаев — на основе специально подготовленной информации, содержащей варианты решений и оценку вероятности их осуществимости. Лишь мощные компании могли позволить себе иметь аналитические центры, готовившие материал для принятия решений. Развитие вычислительной техники кардинально изменило окружающую среду бизнеса. Глобализация и интегрированное развитие индустриальных экономик значительно расширяет возможности бизнеса. Информационные технологии и информационные системы (ИТ/ИС) обеспечивают мобильный доступ и аналитическую мощь, которые удовлетворяют потребности в проведении торговли и руководстве предприятиями в масштабе стран и континентов. Это создает угрозы национальным и региональным фирмам: глобальная связь и системы управления доставляют потребителю информацию о предложениях, качестве и ценах и позволяют совершать сделки и заказы в течение 24 часов в сутки в любом месте, где есть доступ в сеть. Таким образом, мировой рынок становится открытым, ни одна из фирм не может чувствовать себя в безопасности. Чтобы стать эффективным участником этого рынка, компании нуждаются в мощной информационной поддержке и современных системах связи.

Существует несколько возможностей классификации развития ИТ с использованием компьютеров, которые определяются различными качественными признаками деления на этапы. Основной целью применения ИТ становится удовлетворение корпоративных и персональных информационных потребностей. Ниже приводится несколько таких классификаций.

Преимущества применения компьютерных технологий на современном этапе связаны с появлением персональных компьютеров. Изменился подход к созданию ИС — ориентация смещается в сторону индивидуального пользователя для поддержки принимаемых им решений. Пользователь заинтересован в проводимой разработке, налаживается контакт с разработчиком, возникает взаимопонимание обеих групп специалистов. На этом этапе используются как централизованная обработка данных, характерная для первого этапа, так и децентрализованная, базирующаяся на решении локальных задач и работе с локальными базами данных на рабочем месте пользователя.

Преимущества применения компьютерных технологий на современном этапе связаны с той ролью, которую они играют в бизнесе, и основаны на достижениях телекоммуникационных технологий и распределённой обработке информации. ИС имеют своей целью не просто увеличение эффективности обработки данных и помощь управленцу, а создание высокоэффективного производства. Применяемые ИТ должны помочь компании выстоять в конкурентной борьбе и получить преимущество.

Основными инструментальными технологическими средствами современного этапа развития ИТ стали "Internet/Intranet (новейшие)" технологии. Широко используются в различных областях науки, техники и бизнеса распределенные системы, глобальные, региональные и локальные компьютерные сети. Развивается электронная коммерция. В связи с переходом на микропроцессорную базу существенным изменением подвергаются технические средства связи, средства бытового, культурного и прочего назначений.

При традиционном подходе к организации, когда специализированные функции включаются в дело одна за другой, как в эстафете, высокая эффективность недостижима. Быстрота реагирования на внешние изменения требует постоянного сотрудничества между разными специализированными отделами и службами. Постоянно общаясь и обмениваясь информацией, они могут действовать быстро, согласованно и одновременно в самых разных направлениях. Информационные технологии исключительно полезны в случае такого координированного процесса. Применение ИТ позволяет радикально изменить стиль управления и сами бизнес-процессы и значительно улучшить основные показатели деятельности компании. Прежние правила ведения бизнеса стремительно устаревают. Компании, которые не в состоянии "увидеть" значимость этих изменений, рискуют сильно отстать.

Современное состояние ИТ можно охарактеризовать следующими положениями:

1. Наличие большого количества программно-аппаратных комплексов и платформ для эффективного управления и сопровождения производства, промышленно функционирующих баз данных и хранилищ знаний большого объема, содержащих информацию по всем направлениям деятельности общества;
2. Наличие технологий, обеспечивающих интерактивный доступ любого пользователя к информации и ресурсам — технической основой для этого служат корпоративные системы поиска информации (information retrieval systems — IRS), государственные и коммерческие системы связи, глобальные (global network systems), национальные (nns) и региональные (rns) информационно-вычислительные сети; международные соглашения, стандарты и протоколы обмена данными;
3. Расширение функциональных возможностей ит, обеспечивающих распределенную работу баз и хранилищ данных с данными разнообразной структуры и содержания, мультиобъектных документов, гиперсред; создание локальных и интегрированных проблемно-ориентированных ис различного назначения на основе мощных серверов и локально-вычислительных сетей;
4. Включение в ис специализированных интерфейсов пользователя для взаимодействия с экспертными системами (expert system — es), систем поддержки принятия решения (decision support system — dss), систем поддержки исполнения (executive support system — ess), систем машинного перевода (translating computer system — tcs) и другие технологии и средства.

<<

1. АНАЛИТИЧЕСКАЯ ЧАСТЬ.
   1. Анализ предметной области
      1. Общая характеристика объекта исследования

Тест (от слова англ. test — «испытание», «проверка»), тестирование — метод изучения глубинных процессов деятельности системы, посредством помещения системы в разные ситуации и отслеживание доступных наблюдению изменений в ней.

Педагогическое тестирование — это форма измерения знаний учащихся, основанная на применении педагогических тестов. Включает в себя подготовку качественных тестов, собственно проведение тестирования и последующую обработку результатов, которая даёт оценку обученности тестируемых.

* + 1. Организация предметной области

Тесты можно классифицировать по различным признакам:

* по целям — информационные, диагностические, обучающие, мотивационные, аттестационные;

1. по процедуре создания — стандартизованные, не стандартизованные;
2. по способу формирования заданий — детерминированные, стохастические, динамические;
3. по технологии проведения — бумажные, в том числе бумажные с использованием оптического распознавания, натурные, с использованием специальной аппаратуры, компьютерные;
4. по форме заданий — закрытого типа, открытого типа, установление соответствия, упорядочивание последовательности;
5. по наличию обратной связи — традиционные и адаптивные

Традиционный тест содержит список вопросов и различные варианты ответов. Каждый вопрос оценивается в определенное количество баллов. Результат традиционного теста зависит от количества вопросов, на которые был дан правильный ответ. По мнению, Аванесова В. С., традиционный тест — система заданий, предъявляемая в порядке увеличения сложности в одно и тоже время, с одинаковой системой оценивания для всех тестируемых.

Адаптивный тест – особый вид теста, в котором каждое последующее задание выбирается в зависимости от ответов на предыдущие задания. Последовательность заданий и их количество в таком виде теста определяется динамически. Самыми значимыми преимуществами компьютерного адаптивного тестирования перед традиционным являются:

1. возможность адаптации под уровень знаний тестируемого (не придется отвечать на слишком сложные или слишком простые вопросы);
2. экономия времени и сил за счет сокращения количества заданий (длина теста может быть уменьшена до 60 %) без потери уровня достоверности.

Существует несколько форм тестовых заданий, названия и примеры которых приведены далее:

Задания открытого типа – к вопросу не прилагаются варианты ответов, предлагается написать ответ, в зависимости от ситуации короткий или полный развёрнутый ответ.

Задания закрытого типа или с вариантами ответов, в свою очередь делятся на подтипы:

1. задания с выбором одного правильного ответа;

Пример:

Директивы препроцессора С++

1. #ifdef, #ifndef, #include, #define;
2. if(){}, else{}, switch(){};
3. for(){}, while(){};
4. break, continue.
5. задания с выбором одного неверного ответа;

Пример:

Вектор стандартной библиотеки С++ представляет собой

1. шаблон;
2. динамический массив;
3. контейнер;
4. метод;
5. задания на установление соответствия;

Пример:

Установите соответствие между командами и сочетанием клавиш

команда сочетание клавиш

1. Вырезать фрагмент текста; а) CTRL+V;

2. Копировать фрагмент текста; б) CTRL+C;

3. Вставить фрагмент текста. в) CTRL+X.

1. задания с выбором нескольких правильных ответов;

Пример:

Правильное написание имени файла

1. #fileReader.cpp;
2. std\filename.txt;
3. pragma.py;
4. config .
5. упорядочивание последовательности;

Пример:

Упорядочьте числа по возрастанию

1. ff0000
2. 0ff000
3. aac46f
4. ffffdd

Тестовое задание — составная часть педагогического теста, отвечающая требованиям технологичности, формы, содержания и, кроме того, статистическим требованиям:

1. известной трудности;
2. достаточной вариации тестовых баллов;
3. положительной корреляцией баллов задания с баллами по всему тесту

Типы заданий в тесте

1. Закрытые:
   1. задания альтернативных ответов;
   2. задания множественного выбора;
   3. задания на восстановление соответствия;
   4. задания на установление правильной последовательности.
2. Открытые:
   1. задания свободного изложения;
   2. задания – дополнения.

Педагогическое тестирование выполняет несколько функций в образовании, такие как:

Диагностическая функция заключается в выявлении уровня знаний, умений и навыков учащегося. Это основная и самая очевидная функция тестирования. По объективности, широте и скорости диагностирования, тестирование превосходит все остальные формы педагогического контроля.

Обучающая функция тестирования состоит в мотивировании учащегося к активизации работы по усвоению учебного материала. Для усиления обучающей функции тестирования могут быть использованы дополнительные меры стимулирования студентов, такие как: раздача преподавателем примерного перечня вопросов для самостоятельной подготовки, наличие в самом тесте наводящих вопросов и подсказок, совместный разбор результатов теста.

Воспитательная функция проявляется в периодичности и неизбежности тестового контроля. Это дисциплинирует, организует и направляет деятельность учащихся, помогает выявить и устранить пробелы в знаниях, формирует стремление развить свои способности.

Как и любые другие методы обучения и контроля знаний, тестирование обладает рядом преимуществ и недостатков.

Преимущества:

1. Тестирование является более качественным и объективным способом оценивания, его объективность достигается путем стандартизации процедуры проведения, проверки показателей качества заданий и тестов целиком;
2. Тестирование — более справедливый метод, оно ставит всех учащихся в равные условия, как в процессе контроля, так и в процессе оценки, практически исключая субъективизм преподавателя. По данным английской ассоциации NEAB, занимающейся итоговой аттестацией учащихся Великобритании, тестирование позволяет снизить количество апелляций более чем в три раза, сделать процедуру оценивания одинаковой для всех учащихся вне зависимости от места проживания, типа и вида образовательного учреждения, в котором занимаются учащиеся;
3. Тесты – это более объёмный инструмент, поскольку тестирование может включать в себя задания по всем темам курса, в то время как на устный экзамен обычно выносится 2-4 темы, а на письменный — 3-5. Это позволяет выявить знания учащегося по всему курсу, исключив элемент случайности при вытаскивании билета. При помощи тестирования можно установить уровень знаний учащегося по предмету в целом и по отдельным его разделам;
4. Тест – это более точный инструмент, так, например, шкала оценивания теста из 20 вопросов, состоит из 20 делений, в то время, как обычная шкала оценки знаний — только из четырёх;
5. Тестирование более эффективно с экономической точки зрения. Основные затраты при тестировании приходятся на разработку качественного инструментария, то есть имеют разовый характер. Затраты же на проведение теста значительно ниже, чем при письменном или устном контроле. Проведение тестирования и контроль результатов в группе из 30 человек занимает полтора два часа, устный или письменный экзамен — не менее четырёх часов;
6. Тестирование — это более мягкий инструмент, они ставят всех учащихся в равные условия, используя единую процедуру и единые критерии оценки, что приводит к снижению предэкзаменационных нервных напряжений.

Недостатки:

1. Разработка качественного тестового инструментария — длительный, трудоемкий и дорогостоящий процесс. Стандартные наборы тестов для большинства дисциплин ещё не разработаны, а разработанные обычно имеют очень низкое качество;
2. Данные, получаемые преподавателем в результате тестирования, хотя и включают в себя информацию о пробелах в знаниях по конкретным разделам, но не позволяют судить о причинах этих пробелов;
3. Тест не позволяет проверять и оценивать высокие, продуктивные уровни знаний, связанные с творчеством, то есть вероятностные, абстрактные и методологические знания;
4. Широта охвата тем в тестировании имеет и обратную сторону. Учащийся при тестировании, в отличие от устного или письменного экзамена, не имеет достаточно времени для сколько-нибудь глубокого анализа темы;
5. Обеспечение объективности и справедливости теста требует принятия специальных мер по обеспечению конфиденциальности тестовых заданий. При повторном применении теста желательно внесение в задания изменений;
6. В тестировании присутствует элемент случайности. Например, учащийся, не ответивший на простой вопрос, может дать правильный ответ на более сложный. Причиной этого может быть, как случайная ошибка в первом вопросе, так и угадывание ответа во втором. Это искажает результаты теста и приводит к необходимости учета вероятностной составляющей при их анализе.
   1. Технологии в образовании
      1. Применение информационных технологий в образовании

Информационными технологиями называют различные способы, механизмы и устройства обработки и передачи информации. Основное средство для этого – персональный компьютер, дополнительное – специальное программное обеспечение, возможность обмена информацией посредством сети Интернет и сопутствующее оборудование. Во многих учебных заведениях информационные технологии до сих пор считаются инновационными – то есть новыми, способными существенно изменить, оптимизировать учебный процесс. И хотя ежедневное использование компьютера уже давно стало нормой, но постоянное появление усовершенствованных программ значительно расширяет образовательные возможности.

Вот только некоторые процессы в обучении, которые значительно упрощают инновационные технологии:

1. Получение необходимой информации и повышение уровня знаний;
2. Систематизация информации, благодаря справочникам и электронным библиотекам;
3. Отработка различных навыков и умений, проведение удаленных лабораторных экспериментов;
4. Визуализация информации и ее демонстрация (например, на презентациях);
5. Проведение сложных расчетов и автоматизация рутинных операций;
6. Моделирование объектов и ситуаций с целью их изучения;
7. Обмен информацией между несколькими пользователями, находящимися на большом расстоянии друг от друга.

Нужно ли получить какую-то информацию, сделать расчеты по сложным формулам, проверить, как будет работать та или иная идея, обсудить с преподавателем и сокурсниками какую-то проблему, не выходя из дома, – все это можно сделать благодаря современным технологиям, что делает сам процесс получения знаний и обучения намного более эффективным.

Когда сегодня говорят об информационных технологиях в образовании, не редко подразумевают мультимедийные технологии, которые, по мнению российских и зарубежных исследователей, помогают более глубоко исследовать многие вопросы, при этом сокращают время на изучение материала. Мультимедиа представляет собой текстовую, видео, звуковую и фото-информацию, представленную в одном цифровом носителе, а также предполагающую возможность интерактивно взаимодействовать с ней. Проще говоря, мультимедиа позволяют вам одновременно работать с изображением, текстом и звуком, и при этом вам, как правило, отводится активная роль. Например, в обучающем курсе вы можете менять темп обучения или самостоятельно проверять, насколько вы хорошо освоили материал. Такой индивидуальный подход не только более успешно раскрывает способности учащегося, но и предполагает развития творческого начала.

В образовательном процессе мультимедиа используется и для проведения мультимедийных презентаций, и для создания обучающих курсов, и в дистанционном обучении.

* + 1. Автоматизация контроля знаний

Инновационные технологии можно применять в образовательных целях так же, как и во многих других областях жизни. Например, благодаря ним, есть возможность автоматизировать процесс проведения экзаменов и проверки правильности ответов, данных экзаменуемым. По мимо упрощения процесса экзаменирования, это так же способствует более объективной оценке знаний, так как проверка и оценивание производится вычислительной техникой и её алгоритмами, у которых не может быть предвзятого отношения к ученику. В программе для экзамена присутствуют только четко нормированные вопросы, без возможности задать какой-либо каверзный вопрос, заводящий в тупик, что снижает риск неудачи по желанию принимающего экзамен. Также снижается эмоциональная нагрузка на человека, что позволит собраться с мыслями и дать верный ответ. Таким образом автоматизированный срез знаний более выгоден как со стороны проверяемого, так и для учебных заведений.

Реализация автономного контроля знаний происходят средствами интерактивных программ, способных создавать варианты заданий на основе имеющегося набора вопросов и проводить срез знаний в форме тестовой работы. При этом могут быть заданы вопросы различного вида: с выбором верных ответов или же с написанием своего ответа.

* 1. Инструментальные средстава
     1. Язык программирования

В качестве языка программирования был выбран C++, так как это язык общего назначения, поддерживающий объектно-ориентированную парадигму программирования. С++ один из самых популярных языков программирования, применяемый в большом количестве целевых ниш. Является достаточно просты для изучения и хорошо читаемым синтаксисом, благодаря этому он применяет для обучения программированию, людей только в ходящих в данную отрасль программирования.

В любых аспектах жизни всегда присутствуют альтернативы и данным фактом не стоит пренебрегать, поэтому будет полезно взглянуть в сторону конкурентов, а так на влияние не посредственно C++ на индустрию в целом. В адрес данного языка с самого начала его существования было довольно много критики и попыток предложить альтернатив для прикладной и низкоуровневой отраслей программирования. Одним из первых достойных конкурентов стал язык Java, предложенный Sun Microsystems, его часто ошибочно считают прямым наследником C++. На самом деле в конкурент унаследовал лишь синтаксис, семантика была взята из языка Модула-2 и некоторые основы семантики C++. В то время как некоторые элементы семантики C(Си) наследованы из Lisp. Создатели Java для улучшения конъюнктуры выбрали синтаксис C++ и поощрили их сравнение в печати. Вскоре произошло общественное признание Java и C#, тогда были предприняты попытки совмещения безопасности и скорости разработки с эффективностью С++, что породило нового потомка язык D, подобные стремления адаптировать языки программирования к различным задачам с применением лучших и наиболее подходящих возможностей более ранних универсальных средств привело к появлению огромного множества конкурентно способных языков программирования.

Java и C++ это прямые конкуренты, унаследовавшие синтаксис C(Си), часто подвергающиеся сравнению как языков, не беря во внимание рад различий на всех уровнях, как в сфере применимости, так и в семантике и многом другом. Сравнение C++ и Java второе по частоте после C и C++.

Сравнивают их по следующим критериям:

1. Целевая ниша

Java применяется для очень ограниченного спектра промышленности: безопасный язык с низким порогом вхождения для разработки прикладных пользовательских приложений широкого рынка с высокими показателями переносимости – и с этой задачей справляется. С++ претендует на «универсальную применимость» во всех задачах для всех категорий программистов, но не удовлетворяет в полной мере требованиям ни одной из заявленных сфер применимости.

1. Исполнение программы

Java имеет формальную семантику, ориентированную на интерпретацию, но код Java компилируется в промежуточный код, который непосредственно перед запуском программы компилируется в машинный. C++ имеет естественную семантику, ориентированную на компиляцию, так что уже на аппаратной Java-машине был бы крайне неэффективен и ограничен по возможностям. Одно это определяет разницу в сферах применения языков: Java нецелесообразно использовать в низкоуровневом программировании; С++ — в разработке интернет-приложений. Механизм исполнения Java делает программы полностью портируемыми, по принципу «написано один раз — работает везде (write once — run everywhere)», хотя это не было первостепенной целью разработчиков. Стандартное окружение и среда исполнения позволяют выполнять программы на Java на любой аппаратной платформе и в любой ОС без каких-либо изменений, при условии существования на данной ОС и платформе среды исполнения. Усилия по переносу программ минимальны, и могут быть сведены к нулю соблюдением определённых рекомендаций при разработке. Ценой переносимости в данном случае становятся определённые накладные расходы (например, размер среды исполнения Java превышает даже их размеры у всех функциональных языков).

1. Парадигма программирования

С++ в значительно меньшей степени, чем Java, отвечает фундаментальному понятию «всё – объект». В так как он является наследником С в нем безо всяких проблем осуществляется как функциональное программирование, так и объектно-ориентированное. Более того даже приложение, написанное в парадигме ООП требует обязательной, самостоятельной функции main. Объявление глобальных переменных не составляет труда. В то время как в Java любая функция или переменная обязана быть членом класса, для объявления глобальных функций требуются поместить в фиктивные классы, назначив свойство статичности.

1. Объектная модель

Объектная модель сравниваемых языков наследуется из Симулы (в Java — через промежуточную ступень — язык Modula-2), и фундаментально отличается от применяемой в потомках Smalltalk (Objective-C, Python, Ruby). И между С++ и Java есть существенное отличие. В последнем все >>todo >>методы (функции-члены классов) [\\термины](file:///\\термины)<<. Есть >>todo>>синтаксический сахар \\термины<< для определения абстрактных классов: использование ключевого слова interface делает все методы класса чистыми виртуальными — такие классы называются в Java «интерфейсами». Множественное наследование допустимо только для них, но не для обычных классов, что улучшает дисциплину программирования — на этапе реализации нет возможности нарушить структуру проекта, построенную на этапе архитектурного проектирования (в C++ это делается легко).

1. Операторы

В Java отсутствует оператор безусловного перехода, его заменяет механизм именных, который выглядит как метки C++, дающий возможность безусловного перехода на следующую итерацию цикла или из всех вложенных блогов за пределы именного блока. Хотя, остальные конструкции являются простыми потомками Algol: алгоритмический порядок вычислений, присвоение, ветвление, циклы, аргументы в объявлении и вызове функции, инфиксные арифметические операции и прочее.

* + 1. Среда программирования

//\todo VisualStudio

* + 1. Библиотека классов и функций

//\todo Qt Framework

* 1. Постановка задачи
     1. Основные черты программного продукта

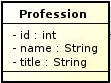
В рамках дипломного проекта будет разработано программное средство, позволяющее в автоматическом режиме проводить экзаменационные, тестовые работы для учащихся школ или учреждений профессионального образования. Способное создавать уникальные варианты тестов для каждого учащегося для снижения возможности списывания. Данная программная система имеет в составе арсенал различных видов тестовых заданий.

* + 1. Теоретическая реализация

Приложение, которое будет разработано должно создавать различные варианты теста из существующего набора вопросов, значит упомянутая коллекция вопросов должна где-то храниться. В качестве хранилища вопросов будет применяться база данных. Экзамены могут проводиться для различных учебных звеньев, классов подготовки и по различным профессиям (если речь идет о высших уровнях обучения), а также и по предметам. Все эти данные должны бить структурированы, с такой целью в базе данных должны присутствовать таблицы для сортировки вопросов по каждому критерию. Критерии отбора вопросов будут следующие: профессия (радиоинженер или путеец; школьник, если экзамен для девятиклассников, к примеру), предмет (математика, инженерная графика), тема (для периодического контроля знаний во время обучения). Таким образов, необходимы:

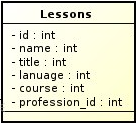
1. Таблица профессий Professions;
2. Таблица предметов Lessons;
3. Таблица тем по предметам Themes;
4. Таблица непосредственно вопросов Questions.

Данные таблиц будут связаны при помощи уникальных идентификаторов, которые присутствуют в каждой таблице. Кроме идентификаторов, разумеется, должны присутствовать и не посредственные данные. Для профессии – это название и подпись, которую будет видеть пользователь программы, схема таблицы приведена на следующем рисунке.



**Рисунок 1 –** схема таблицы профессии

В составе таблица предметов присутствуют следующие поля: идентификатор, название, подпись (три упомянутых поля будут присутствовать во всех последующих таблицах), язык (на котором преподается предмет), курс (обучения экзаменуемого), идентификатор профессии (к которой принадлежит предмет).



**Рисунок 2 –** схема таблицы предметов

Следующий элемент – таблица тем содержит: кроме общих полей, уровень сложности, идентификатор предмета, которому принадлежит тема.



**Рисунок 3 –** схема таблицы тем

Табель вопросов будет содержать текст самого вопроса, тип задания, рекомендуемое время на выполнение задания, идентификатор темы, информация о чувствительности к регистру, информация о чувствительности к пробелам в тексте. Для каждого вопроса необходимы и ответы, они будут храниться в соответствующем каталоге с полями: идентификатор, текст ответа, информация о правильности ответа, идентификатор привязки вопроса.

//TODO: SQL USING>> и образуются полем «счётчик», путем инкрементирования целого числа при добавлении записи в таблицу. Поле счетчик может иметь целочисленные значения большие или равное нулю, каждое значение должно быть уникально для этого поля. Последнее достигается путем добавления инкрементированного значения последней существующей записи в новую запись, при этом при удалении записей номера не пересчитываются согласно порядковым номерам, так как последнее повлечет нарушение структуры базы данных.

1. ПРАКТИЧЕСКАЯ ЧАСТЬ
   1. Описание программы

Первый элемент программы – генератор случайных чисел, он будет основан на совокупности нескольких источников псевдослучайных величин таких, как время, стандартный генератор случайных чисел и два вида алгоритмических.

Одним из алгоритмических методов генерирования случайных чисел будет – метод срединных произведений. В качестве первого множителя станет число, полученное путем сложения количества миллисекунд, полученных при двух запросах к системному времени. Второй множитель – число, сгенерированное стандартным генератором в диапазоне от 1111 до 9999. Количество шагов генерирования, также будет зависеть от количества миллисекунд.

Вторым алгоритмом станет метод срединных квадратов, в качестве начального числа, возводимого в квадрат, станет – сгенерированное стандартным ГСЧ, количество шагов генерации также будет зависеть от значения миллисекунд.

Далее все вышеперечисленные способы получения псевдослучайных чисел будет объединены, следующим образом: берется сумма системных миллисекунд, число, сгенерированное стандартным генератором, затем много кратно повторяется, следующие действия:

1. В генератор методом срединных произведений подаются выше полученные числа;
2. Генерируется число и подается в генератор срединных квадратов;
3. Последнее, в качестве второго операнда снова подаётся в генератор срединных произведений.

Последним действием полеченное четырехзначное число разделить на 10000. Таким образом можно получить величины, максимально распределенные в диапазоне от 0 до 1. Для получения случайных чисел в заданном диапазоне, сгенерированное число необходимо умножить на разность, максимального и минимального чисел диапазона, а затем, полученное произведение сложить, с максимальным числом диапазона.

Следующим звеном программы является математическая модель работы с фигурами. Любая фигура состоит из точек или вершин и сторон или ребер. Точки имеют лишь координаты на плоскости, но ребра, так как они являются линиями обладают коэффициентами углового и вертикального смещения. Коэффициенты должны вычисляться на этапе инициализации объекта ребро. Для вычисления данных коэффициентов следует вспомнить простое уравнение прямой

где

x, y – координаты точки;

k – коэффициент углового смещения;

b – коэффициент вертикального смещения

Множитель k для отрезка прямой вычисляется отношением разностей координат конечной и начальной точки, разность y1 и y0 делится на разность x1 и x0.

Вычисление b происходит путем подстановки известных координат любой из точек, коэффициента k и преобразования уравнения прямой, следующим образом:

Таким образом получается сторона фигуры со всеми необходимыми параметрами для вычисления столкновений, её и луча исходящего из случайной точки.

Для выявления столкновения луча и прямой есть два подхода. Первый – вычислить все точки, принадлежащие прямой, а так все точки, принадлежащие лучу на данном отрезке координат проверить каждую точку луча и прямей на совпадение координат. Второй способ – зная координаты случайной точки и коэффициенты прямой, путем подстановки в уравнение прямой можно узнать координаты пересечения, проверяемого отрезка и ребра фигуры. Так как необходимо проверять лучи, направленные вверх и вправо – расположение, проверяемой точки с расположением точки пересечения следует сравнить, следующим образом:

1. Если координата X точки, испускающей луч вправо меньше ординаты места пересечения линий – луч имеет пересечение, иначе пересечения нет.
2. Если координата Y точки, испускающей луч вверх меньше абсциссы места пересечения отрезков – пересечение луча и ребра существует, иначе пересечения с вертикальным лучом нет.

Как можно заметить второй метод более рационален по потребности в ресурсах памяти и времени выполнения, с минимальным количеством вычислений и проверок.

Для решения задачи по нахождению площадей фигур необходимо проверять точки на принадлежность фигуре. Выше описанный метод позволит проверить каждый луч, каждой случайной точки на пересечение, с каждым ребром фигуры. При наличии нечетного количества пересечений у обоих лучей точки – последняя считается, принадлежащей к фигуре.

Важной частью приложения является система ввода вывода. Здесь будет присутствовать как ввод параметров из командной строки, как и файловый ввод свойств многоугольника, а также вывод результата в виде текстового файла или строки в окне команд. Для указания пути к входному файлу будет обрабатываться параметр «-i» после, которого должен быть написан полный путь. Если координаты фигуры должны вводиться не из файла, а из командной строки вводится «-p» после чего пары координат, вершин. Чтобы указать файл для вывода обрабатывается аргумент «-o» вместе с полным путём, если же путь вывода не будет указан, результат вычислений отобразится в окне команд.

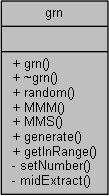
Неотъемлемой частью является модель метода Монте-Карло она будет непосредственно проводить статистический анализ: принимая параметры фигур, генерируя случайные точки при помощи генератора случайных чисел; подсчитывая количество точек, попавших в фигуру контрольную и испытуемую; вычислять соотношение числа точек в интересующем многоугольнике и контролирующем.

Само приложение будет проводить многократные испытания для последующего усреднения показаний, что даст наиболее точный результат.

* 1. Структура программы на языке С++ и UML-диаграмма

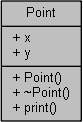
При создании программного решения была выбрана объектно-ориентированная парадигма программирования. Поэтому, все ключевые звенья были выделены в классы, а вычислительные процессы в методы данных классов. Так же применяется раздельное компилирование, то есть абстрактное объявление полей и методов класса находится в файле заголовка, а реализация – в файле исходного кода.

Один из основных элементов модели вычислений методом Монте-Карло является генератор случайных чисел. Данный класс был назван grn – generator of random numbers. Объекты данного класса являются статическими, это означает, что они не имеют полей, в них присутствуют лишь методы приватной и публичной степени защищённости. Схему данного класса можно увидеть на рисунке 5



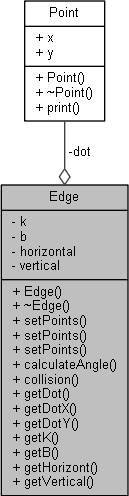
**Рисунок 5 –** схема класса grn

Модель обработки фигур состаит из нескильких классов: Point, Edge, Polyangle. Класс описыввающий точку содержит 2 открытых поля и конструктор класса, без каких-лиюбо других методов. Схема приведена на рисунке 6



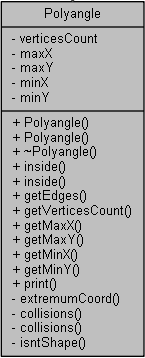
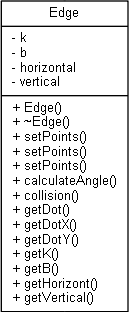
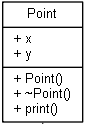
**Рисунок 6 –** схема класса Point

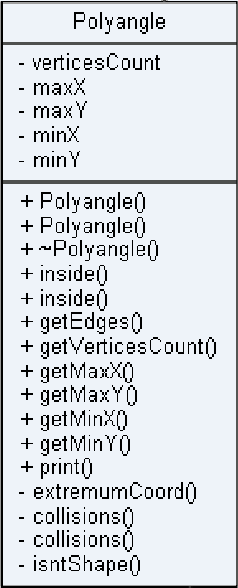
Класс представляющий ребро или сторону фигуры содержит в себе поле из указателя на объекты класса Point, а также поля, содержащие коэффициенты, присущие прямым, и переменные показывающие является ли линия вертикальной или горизонтальной. Для инициализации класса предусмотрены методы по вводу значений в поля – точки прямой, а также – вычисляющие коэффициенты прямой. Обеспечение доступа к полям, тоже происходит посредством публичных методов. Схема класса – рисунок 7.



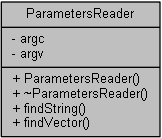
**Рисунок 7 –** схема класса Edge

Наибольшим звеном при работе с многоугольниками – является фигура – класс Polyangle. Он содержит в себе указатель на стороны фигур, которых может быть минимум три и до бесконечности. Среди полей присутствуют: количество вершин многоугольника, координат экстремума фигуры. Координаты фигуры инициализируются в конструкторе класса, который способен принимать указатель на ребра фигуры или указатель на массив вершин. Экстремум вычисляется приватным методом, сразу после инициализации всех рёбер. Существует функция объекта для проверки принадлежности точки к данной фигуре, которая принимает объект точка или координаты точки как целочисленные переменные. Если при инициализации фигуры были введены параметры прямой линии, функция isntShape() выбросит исключение, и программа напечатает сообщение о данной неполадке, для этого внутри данного класса присутствует внутренний класс – isLine. Схема класса приведена ни рисунке 8.



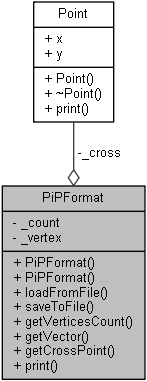
**Рисунок 8 –** схема класса Polyangle

Следующей важной частью – является система ввода команд и файлов и вывода файлов и сообщений. Её представляют классы ParameterReader и PiPFormat. Первый представляет набор методов для поиска среди параметров, нужной команды и возвращения, идущего после команды вектора чисел или строки. Вектор ищется функцией findeVector() и возвращиет std::vector<int>, если вектор значений пуст, функция вернёт исключение. Строка же находится при помощи функции findeString() и возвращает std::string, если строка после, указанного параметра пуста, функция вернёт исключение. Конструктор класса принимает количество параметров и строку, содержащую их.



**Рисунок 9 –** схема класса ParameterReader

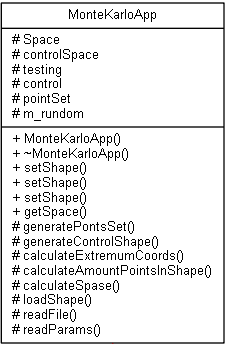
Второй элемент системы ввода вывода предназначен для работы с файлами. Он способен извлекать вектор чисел из файла и записывать в файл сформатированную строку при помощи методов, отведенных на это, в случае не удачного срабатывания функций предусмотрены исключения то момент, когда читаемый файл не найден, или путь вывода не доступен по каким-либо причинам.



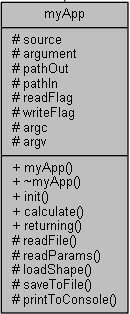
**Рисунок 10 –** схема класса PiPFormat

Главными частями приложения являются классы MonteKarloApp и производный от него – myApp, они выполняют все основные операции, а именно:

1. Инициализируют приложение;
2. Принимают параметры запуска и командной строки;
3. Создают фигуры – контрольную и испытуемую;
4. Генерируют случайные точки для вычислений;
5. И производят эти самые вычисления.

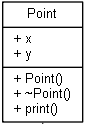
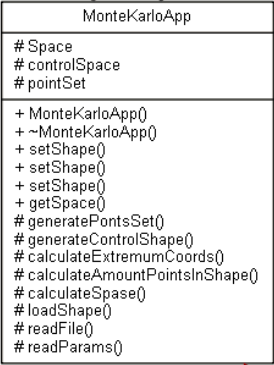


**Рисунок 11 –** схема класса MonteKarloApp



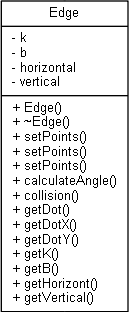
**Рисунок 12 –** схема класса myApp

Таким образом выполнена структура модели вычислений методом Монте-Карло и структура приложения по вычислению площадей фигур, созданной на основе первой. Некоторые ключевые моменты программного кода приведены в приложении А.

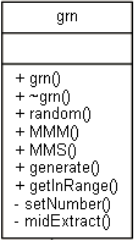


-dot

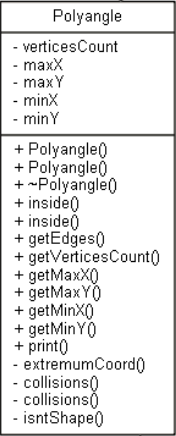
-\_cross



-edge

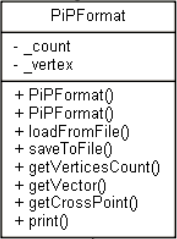


#m\_rundom

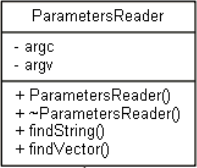


#testing

#coritrol



#source

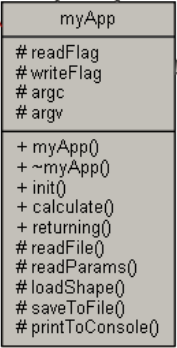


#argument



#pathOut

#pathln



**Рисунок 13 –** UML-диаграмма приложения

ЗАКЛЮЧЕНИЕ

В процессе работы над курсовым проектом был изучен принцип действия статистического метода анализа данных в определённом диапазоне. Разработан алгоритм, компьютерной программы. При разработке алгоритма была построена UML-диаграмма классов, необходимых, для реализации алгоритма компьютерной программы. Для программирования, в качестве инструментальных средств, выбран язык программирования С++. Была проведена работа по отладке программы, выявлению, исправлению ошибок и неточностей. В итоге получена реализация программы, позволяющей строить модель решения статистической задачи с применением метода Монте-Карло.

СПИСОК ЛИТЕРАТУРЫ

1. Н. Ш. Кремер. «Теория вероятностей и математическая статистика.» Москва, «Юнити», 2006;
2. Б. В. Гнеденко. «Теория вероятностей и математическая статистика.», Москва, «Наука», 1970;
3. С.М. Ермаков, Михайлов Г.А. «Статистическое моделирование.», Москва, «Наука», 1983;
4. Г. Крамер. «Математические методы статистики.», Москва, «Мир», 1975;
5. Р. Лафоре. «Объектно-ориентированное программирование в С++», Москва, «Питер», 2004;

ПРИЛОЖЕНИЕ A

grn.cpp

int grn::MMS(int n) // set dddd, sqr(dddd) = ddmmmmdd, mmmm is middle of square

{

int fNumber,//first number dddd

midSquare = 0; //mid of square

fNumber = n == 0 ? setNumber() : n;//dddd

int end = clockInThisSecond();

for (int i = 0; i < end; i++)

{

midSquare = midExtract(fNumber\*fNumber);

fNumber = midSquare == 0 ? midExtract(random()) : midSquare;

status(i, end, "drn::MMS");

}

return midSquare;

}

int grn::MMM(int n) // set dddd, dddd\*cccc = ddmmmmdd, mmmm is middle of multiplication

{

int fNumber,//first number dddd

sNumber,//second number cccc

midMultipl = 0; //mid of multiplication

fNumber = n | setNumber();//dddd

sNumber = random();//cccc

int end = clockInThisSecond();

for (int i = 0; i < end; i++)// for mutcher range

{

midMultipl = midExtract(fNumber \* sNumber);

fNumber = midMultipl;

if (midMultipl == 0)

{

fNumber = setNumber();

i--;

}

status(i, end, "drn::MMM");

}

return midMultipl;

}

float grn::generate()

{

float rNumber = 0.0;

intmidSquare = random(),

midMultipl = MMS();

int end = clockInThisSecond();

for (int i = 0; i < end ; i++)

{

midSquare = MMS(midMultipl);

midMultipl = MMM(midSquare);

status(i, end, "drn::generate");

}

rNumber = midMultipl;

rNumber /= 10000;

return rNumber;

}

edge.cpp

bool Edge::collision(Point p)

{

if (horizontal)

{

if (

p.x <= maxNumber(dot[0].x, dot[1].x) &&

p.x >= minNumber(dot[0].x, dot[1].x)

)

{

int y0 = k \* p.x + b;

return y0 >= p.y ? true : false;

}

else return false;

}

else if (vertical)

{

if (

p.y <= maxNumber(dot[0].y, dot[1].y) &&

p.y >= minNumber(dot[0].y, dot[1].y)

)

{

return dot[0].x >= p.x ? true : false;

}

else return false;

}

else

{

int x0,

y0;

if (

(

p.y <= maxNumber(dot[0].y, dot[1].y) &&

p.y >= minNumber(dot[0].y, dot[1].y)

)

||

(

p.x <= maxNumber(dot[0].x, dot[1].x) &&

p.x >= minNumber(dot[0].x, dot[1].x)

)

)

{

x0 = (p.y - b) / k;

y0 = k \* p.x + b;

return x0 >= p.x || y0 >= p.y ? true : false;

}

else return false;

}

return false;

}

polyangle.cpp

int Polyangle::collisions(Point p)

{

int collisions\_count = 0;

for (int i = 0; i < verticesCount / 2; i++)

{

if (edge[i].collision(p))

{

collisions\_count++;

}

status(i, verticesCount/2, "polyangle::collisions");

}

return collisions\_count;

}

bool Polyangle::inside(Point p)

{

// isntShape();

return collisions(p) % 2 == 0 ? false : true;

}

ParameterReader.cpp

string ParametersReader::findString(string opt)

{

std::string str;

if (argc > 1)

for (int i = 0; i < argc; i++)

{

string arg = string(argv[i]);

if (arg == opt)

{

if (

!(string(argv[i + 1])[0] == '-') &&

i + 1 <= argc

)

{

str = string(argv[i + 1]);

}

else throw Empty();

}

}

else

{

throw Empty();

}

if (str.empty())

throw Empty();

return " ";

}

std::vector<int> ParametersReader::findVector(string opt)

{

std::vector<int> v;

int sizeInt = sizeof(int)\*8,

\_sizeInt = -1 \* sizeInt;

if (argc > 1)

for (int i = 0; i < argc; i++)

{

string arg = string(argv[i]);

if (arg == opt)

{

if (

atoi(argv[i + 1]) >= \_sizeInt &&

atoi(argv[i + 1]) <= sizeInt

)

{

while (

(i + 1 >= argc) ? false :

(

atoi(argv[i + 1]) >= \_sizeInt

&& atoi(argv[i + 1]) <= sizeInt

)

)

{

v.push\_back(atoi(argv[i + 1]));

i++;

}

if (!v.empty())

return v;

else throw Empty();

}

else throw Empty();

}

}

else

{

throw Empty();

}

return v;

}