Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерного проектирования

Кафедра инженерной психологии и эргономики

Дисциплина: Основы алгоритмизации и программирования

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему

**КОНСТРУКТОР ЭКЗАМЕНАЦИОННЫХ БИЛЕТОВ**

БГУИР КР 6 - 05 - 06 12 01 029 ПЗ

Выполнил: студент группы 310901Усов А. М.

Проверил: Кабариха В. А.

Минск 2024

Содержание

[**ВВЕДЕНИЕ 3**](#_Toc161594816)

[**1. ТРЕБОВАНИЯ К ПРОГРАММЕ 5**](#_Toc161594817)

[**1.1 Описание предметной области 5**](#_Toc161594818)

[**1.2 Системные требования 5**](#_Toc161594819)

[**1.3 Функциональные требования 6**](#_Toc161594820)

[**2. Конструирование программы 7**](#_Toc161594821)

[**2.1 Описание модулей программы 7**](#_Toc161594822)

[**2.2 Выбор способа организации данных 8**](#_Toc161594823)

[**2.3 Разработка перечня пользовательских функций программы 9**](#_Toc161594824)

ВВЕДЕНИЕ

Традиционно подготовка экзаменационных билетов была крайне трудоемким и длительным процессом для преподавателей. Им приходилось вручную составлять сотни вопросов по пройденному материалу, распечатывать их, компоновать в билеты и следить за актуальностью вопросов из года в год. Этот процесс занимал колоссальное количество времени и требовал серьезных усилий.

С развитием компьютерных технологий появились первые программы, позволявшие автоматизировать отдельные этапы создания экзаменационных материалов. Преподаватели могли вводить вопросы в электронные базы данных, формировать из них случайные наборы для билетов. Это несколько ускоряло процесс, но по-прежнему требовало много ручного труда.

Современные тенденции заключаются в разработке специализированных приложений-конструкторов для максимальной автоматизации процесса на всех этапах – от создания шаблонов билетов и банка вопросов до формирования комплектов билетов и их печати. Такие решения способны существенно экономить время преподавателей, снижать вероятность ошибок, обеспечивать актуальность материалов.

Однако существующие программные продукты часто обладают ограниченным функционалом, плохо интегрируются с другими системами учебного заведения, имеют неудобный и запутанный интерфейс. Зачастую они решают лишь локальные задачи автоматизации, в то время как комплексный подход отсутствует.

Цель данной курсовой работы – создание программного обеспечения "Конструктор экзаменационных билетов" для автоматизации процесса подготовки экзаменационных материалов в учебных заведениях.

Для достижения поставленной цели были сформулированы следующие задачи:

* провести анализ требований к экзаменационным билетам в различных учебных заведениях;
* изучить существующие программные решения, выявить их достоинства и недостатки;
* спроектировать архитектуру приложения с учетом требований масштабируемости и расширяемости;
* реализовать функционал по созданию билетов и формированию банка вопросов;
* обеспечить возможность генерации комплектов билетов по заданным правилам;
* предусмотреть различные варианты вывода подготовленных билетов (печать, электронный формат).

Таким образом, выполнение данной курсовой работы позволит создать современное и эффективное программное решение, способное значительно упростить процесс подготовки экзаменационных материалов в учебных заведениях. Это обеспечит экономию времени преподавателей, повысит качество и актуальность экзаменационных билетов, а также облегчит процесс проверки знаний студентов.

1. ТРЕБОВАНИЯ К ПРОГРАММЕ

1.1 Описание предметной области

Учебные компьютерные программы, такие как «Конструктор экзаменационных билетов», позволяют моделировать процесс создания и проверки билетов, являясь эффективным инструментом для обучения. В то же время, такая программа позволяет визуализировать абстрактные объекты и процессы, которые в реальном мире не существуют, а являются моделями реальных объектов и явлений в области образования.

Процесс представления предметной области задачи в виде набора объектов, обменивающихся сообщениями, называется объектной декомпозицией. В рамках этого процесса основные элементы задачи делятся на классы. Для каждого класса определяются его свойства и поведение, а также взаимодействие с другими классами.

В разрабатываемой системе главными объектами являются экзаменационные билеты и информация, содержащаяся в них. Пользователь запускает программу, выбирает предмет, тему и уровень сложности билетов, после чего программа генерирует экзаменационные билеты для дальнейшего анализа и использования.

Язык программирования: C++;

Парадигма программирования: Процедурное программирование;

Средство разработки: Microsoft Visual Studio;

Способ хранения данных: Файловая система, текстовые файлы.

1.2 Системные требования

Для курсового проекта «Конструктор экзаменационных билетов» была выбрана среда разработки Microsoft Visual Studio. Ее преимуществами являлись популярность, обширная документация, удобный интерфейс, поддержка различных языков программирования и большое количество доступных плагинов и дополнений. Особенно удобной оказалась возможность создания и работы с текстовыми файлами непосредственно в процессе написания кода. Кроме того, в Microsoft Visual Studio уже имелось множество полезных библиотек, готовых к подключению в проекте.

В качестве языка программирования использовался C++, хорошо знакомый разработчику. Этот язык обладает такими преимуществами, как высокая скорость работы и богатая функциональность. C++ был спроектирован таким образом, чтобы предоставить программисту максимальный контроль над всеми аспектами структуры и порядка исполнения программы.

Курсовая работа заключалась в создании консольного приложения в рамках процедурной парадигмы. Для организации данных были выбраны классы, а для их хранения - файлы.

1.3 Функциональные требования

При создании программного средства для автоматизации создания билетов к экзамену определяются функциональные требования, которые необходимо внедрить в систему. Такие как:

* возможность регистрации новых пользователей, авторизации уже зарегистрированных пользователей и авторизации в качестве гостя;
* прохождение выбранного теста;
* возможность просмотра рейтинга пользователей, прошедших определенный тест;
* процесс экспорта созданного теста на печать;
* генерация билетов;
* создание и редактирование вопросов и ответов;
* процесс автоматической оценки после прохождения теста.
* возможность сортировать тесты по сложности;
* возможность фильтровать тесты по разным параметрам.

Гость должен иметь возможность пройти любой тест из списка публичных тестов.

Пользователь должен иметь возможность проходить публичные тесты, а также создавать свои собственные тесты с возможностью из редактирования.

Администратор должен иметь возможность делать все выше перечисленное, а также редактирование любых тестов а также рейтинга тестов.

Эти функциональные требования обеспечат эффективное управление процессом создания, редактирования и экспорта экзаменационных билетов, повысят удобство пользования сервисом как для персонала, так и для пользователей.

1.4 Описание основных нефункциональных требований

При выполнении курсовой работы «Конструктор экзаменационных билетов» для создания безопасного и надежного программного обеспечения были поставлены следующие нефункциональные требования:

* все хранимые данные должны быть представлены в виде шифра по определенному алгоритму;
* должна существовать документация для пользователя, которая научит его пользоваться программным обеспечением;
* должна существовать возможность экспорта готовых билетов в электронный формат или отправка на печать;
* должен быть реализовать интуитивно понятный интерфейс в консоли, реализованный на вводе данных пользователем;
* сохранение всех данных пользователей, тестов и рейтинга в файлы на жёстком диске

Реализация данных нефункциональных требований позволит достичь при создании программного обеспечения «Конструктор экзаменационных билетов» ………

2. Конструирование программы

2.1 Описание модулей программы

При проектировании программы конструктор экзаменационных билетов, модульная структура помогает разделить функциональность на независимые блоки с четкими задачами.

Таблица 2.1 – Описание основных модулей программного средства конструктор экзаменационных билетов

|  |  |  |
| --- | --- | --- |
| Название модуля | Описание | Функционал |
| Модуль регистрации | Обеспечивает процесс регистрации и авторизации пользователей. | * регистрация новых читателей; * авторизация уже зарегистрированных пользователей; * авторизация как гость. |
| Модуль управления тестами | Отвечает за управления тестами. | * создание тестов; * редактирование тестов; * поиск тестов. |
| Модуль работы с билетами | Отвечает за прохождение тестов и выставления оценки за прохождение. | * Прохождение теста; * Просмотр теста; * Выставление оценки. |
| Модуль экспорта | Отвечает за отправление билетов на печать. | * Печать билетов; * Сохранение в электронном формате. |
| Модуль администрирования | Обеспечивает процесс управления пользователями. | * Управление пользователями. |
| Модуль для работы с файлами | Обеспечивает взаимодействие программы с файлами | * Запись в файлы; * Чтение файлов. |

Каждый из этих модулей играет важную роль в обеспечении возможностей программного средства для эффективного создания экзаменационных билетов.

2.2 Выбор способа организации данных

Хранение данных в программе для создания экзаменационных билетов происходит в виде файлов. Хранение данных таким способом обеспечивает сохранность информации между сеансами работы системы. Он гарантирует доступность данных даже после перезапуска программы и обеспечивает эффективное управление информацией о пользователях, тестах и рейтинге.

Для каждой категории данных, таких как тесты, пользователи и операции (экспорт), будет выделен отдельный файл или группа файлов. Каждый файл будет содержать структурированную информацию, организованную в определенном формате для удобства чтения и записи данных.

Данные будут храниться в текстовом или бинарном формате в файлах. Для простоты и удобства обработки, данные могут быть структурированы в формате CSV, JSON. Каждая строка файла может представлять отдельную запись, а разделители или иерархическая структура могут использоваться для организации данных.

Модуль управления данными будет отвечать за чтение и запись данных в соответствующие файлы. При запуске приложения, модуль будет загружать данные из файлов в оперативную память для работы с ними. Обновленные данные по пользователям, тестам и рейтингу будут сохраняться обратно в файлы для сохранности информации между сеансами работы приложения.

Такой подход к хранению данных в файлах обеспечивает удобство работы с информацией, сохранность данных при перезапуске приложения и простоту обработки информации в программе для обслуживания читателей в библиотеке.

……..

При разработке программного обеспечения «Конструктов экзаменационных билетов» для удобного хранения и быстрого доступа была выбрана файловая структура хранения данных, где некоторые важные поля (например пароль пользователя) зашифрованы обеспечения безопасности пользователей.

Для реализации такого способа хранения данных было разработана следующая структура файлов:

1 Усов Пароль 1

2 Кабариха Пароль1 2

* файл всех пользователей, хранящий ID, имя, логин, пароль, уровень доступа и количество созданных тестов;
* Файл со всеми тестами с пометкой «публичный», такие тесты могут проходить незарегистрированные пользователи;
* Набор файлов, где каждый файл имеет в названии ID пользователя (для удобного поиска). Каждый такой файл хранит данные всех созданных этим пользователем тестов;
* Файл пользовательской документации, для удобного представления документации на выводе и упрощенного редактирования был создан этот файл;
* Файл рейтинга, это файл, хранящий в себя данные каждого пройденного теста, количество прохождений, а также имя пользователя, прошедшего тест и результат прохождения.

Для более удобной для разработчика структуры, файлы хранящие данные тестов находятся в отдельно созданной папке (кроме файла рейтингов), файлы, которые хранят информацию про пользователей также находятся в отдельной папке. Для файла документации была также выделена отдельная папка.

……… описать файлы и классы

2.3 Разработка перечня пользовательских функций программы

Необходимый перечень функций, которые будут задействованы в реализации консольной программы на C++ конструктор экзаменационных билетов представлен в таблице 2.2.

Таблица 2.2 – Перечень основных пользовательских функций

|  |  |  |  |
| --- | --- | --- | --- |
| №п/п | Функция | Аргументы | Описание |
| 1 | GetChoice() | Ничего не принимает. | Возвращает тест, который пользователь выбрал из перечня. |
| 2 | CreateExamBilets() | * Int num\_questions (количество вопросов в билете); * Int num\_bilets (Количество билетов). | Возвращает Сгенерированные экзаменационные билеты. |
| 3 | EditTest() | * Test myTest (ранее созданный тест). | Редактирует выбранный тест и возвращает его. |
| 4 | Register() | * string username (Имя пользователя); * string password (Пароль). | Регистрирует нового пользователя, сохраняя данные в файл. |
| 5 | Login() | * string username (Имя пользователя); * string password (Пароль). | Проверяет введенные данные пользователя и разрешает доступ к определенным функциям программы. |
| 6 | Encrypt() | * -string password (Пароль). | Хранит пароли в зашифрованном виде. |
| 7 | CreateTest() | Ничего не принимает. | Отвечает за создание теста и записи его в файл. |
| 8 | TryTest() | * Test test (ранее созданный тест). | Реализует прохождение теста и возвращает оценку за него. |
| 9 | ViewTest() | * Test test (ранее созданный тест). | Показывает тест с ответами. |
| 10 | EditUser() | * User user (зарегистрированный пользователь). | Отвечает за редактирования данных пользователя и его прав доступа (только для администратора). |
| 11 | EditRating() | * Test test (опубликованный тест). | Отвечает за редактирование рейтинга определенного теста (только для создателя теста). |
| 12 | Sort() | * Test \* tests (массив тестов); * string rule (правило сортировки). | Сортирует сходной массив тестов по определенному переданному правилу (переменной). |