Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Курсовой проект по дисциплине:

«Программирование»

Пояснительная записка к курсовой работе

Тема работы:

«Система автоматизированного тестирования знаний»

Исполнитель

студентка гр. 653502 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сухоцкая А.В.

(подпись дата)

Руководитель

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Козуб В.Н.

(подпись дата)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка)

Минск

2017 год

Содержание

[Введение 3](#_Toc480311999)

[1. Qt Creator 4](#_Toc480312000)

[1.1 Описание IDE 4](#_Toc480312001)

[2. Разработка программы 5](#_Toc480312004)

[2.1 Создание проекта 5](#_Toc480312005)

[2.2 Проектный файл 6](#_Toc480312007)

[2.3 Заголовочный файл 6](#_Toc480312008)

[2.4 Главный файл 7](#_Toc480312009)

[2.5 Графический интерфейс 10](#_Toc480312009)

[2.6 Функционал программы 10](#_Toc480312016)

[2.6.1 Взаимодействие с пользователем 11](#_Toc480312019)

[2.6.2 Работа с файлом 13](#_Toc480312022)

[2.6.3 Подсчет ответов, вывод результатов 14](#_Toc480312024)

[Заключение 15](#_Toc480312025)

[Список использованных источников 16](#_Toc480312026)

## Введение

О необходимости существования подобной программы я начала задумываться еще будучи учащейся средней школы. Мне показалось, что система тестирования знаний учеников, существующая в моей школе, далека от совершенства. Во-первых, банальная проверка усвоения материала прошлого занятия занимала около половины урока (раздача и сбор заданий и бланков, пояснение заданий, оформление бланков). Во-вторых, мы, ученики, прибегали к различным хитростям: от элементарного списывания, вплоть до поиска листка с правильными вариантами ответа в личных вещах учителя. В-третьих, я предположила, что проверка бланков с ответами отнимает у преподавателя немало времени и сил, и, кроме того, человеческий фактор не исключает наличие неточностей при проверке.

Благодаря оснащению компьютерной техникой в средних общеобразовательных учебных заведений (к прошлому году почти все кабинеты моей школы в большей или меньшей степени были оборудованы компьютерами), появилась возможность, даже необходимость внедрения её непосредственно в учебный процесс. Выполнение тестов в программе, представленной в данной курсовой работе, решило бы все проблемы, отмеченные в предыдущем абзаце.

Приступив к разработке программы, я поняла, что не стоит ограничивать сферу ее применения учреждениями среднего образования, ведь подобная программа упростила бы процесс проверки знаний в ССУЗах и ВУЗах, а также в автошколах и прочих специализированных учебных заведениях.

Таким образом, целью данной курсовой работы стало создание программы – универсальной системы автоматизированного тестирования знаний.

Я решила использовать объектно-ориентированный язык С++, так как ООП является основой данной программы. Преимуществом этого языка так же стал опыт работы с ним на протяжении двух семестров.

Касаемо среды разработки, мой выбор пал на кроссплатформенную интегрированную среду Qt Creator. Данная IDE предоставляет возможности как для создания новых приложений, так и для изменения уже существующих. Помимо этого, работа с Qt Creator знакома мне благодаря предмету ОАиП, что также повлияло на мой выбор.

## 1. Qt Creator

## 1.1 Описание IDE

**Qt Creator** — это мощная и современная интегрированная среда разработки (IDE), ориентированная на Qt-проекты на C++, основным преимуществом которой является платформонезависимая реализация приложений.

Очевидно, что за платформонезависимостью будущее программной индустрии, ведь позволив приложению работать под разными ОС, заметно увеличится количество пользователей; значительно сократится время разработки; улучшится и качество приложения, так как появится возможность тестировать его на нескольких платформах, а ошибки будут исправляться централизованным путем в одном и том же исходном коде программы.

Помимо этого, Qt Creator является мощной системой визуального объектно-ориентированного программирования, которая позволяет решать множество задач, а в частности:

* Создавать законченные приложения самой различной направленности (от вычислительных и логических, до графических и мультимедиа) для большинства современных операционных систем.
* Быстро создавать (даже начинающим программистам) профессионально выглядящий оконный интерфейс для любых приложений.
* Не ограничивать возможности пользователя. Полная документация находится в открытом доступе, поэтому даже не обладая опытом работы в данной среде, легко узнавать и использовать новые компоненты и функции.

Qt Creator — чрезвычайно быстро развивающаяся система, так как ее создатели постоянно отслеживают все тенденции информационных технологий. Регулярно выпускаются новые версии IDE и библиотеки Qt. Бибилотека Qt – это более 500 классов, охватывающих большую часть функциональных возможностей операционных систем, предоставляющих разработчику мощные механизмы, расширяющие и упрощающие разработку приложений. Данная библиотека использовалась при создании таких известных программ, как интернет-пейджер Skype, предназначенный для голосовой связи Voice Over IP, звонков на мобильные телефоны и проведения видеоконференций через интернет; Adobe Photoshop Album для обработки растровых изображений; сетевая карта мира Google Earth, которая помогает рассматривать интересующие участки нашей планеты с высоты до 200 м.

В своей программе я также использовала данную библиотеку. Перейдем к рассмотрению её компонентов и описанию процесса создания программы.

## 2. Разработка программы

## 2.1 Создание проекта

Проект – это собрание файлов реализации, заголовочных файлов, ресурсов и файла описания проекта (pro-файл). Для проектов не было придумано нового формата файла специально для среды Qt Creator, поэтому в проекты можно загружать уже существующие pro-файлы.

Qt Creator предоставляет готовые шаблоны проектов, с помощью которых можно легко начать создание проекта. Шаблон – это «скелет», содержащий ресурсы, исходный текст и заголовочный файл.

Перейдем непосредственно к созданию проекта. Для этого в меню **Файл** необходимо выбрать команду **Новый проект или файл**, затем выбрать тип проекта – **Проект Qt Widget.** В открывшемся диалоговом окне необходимо выбрать пункт **GUI (**graphical user interface) **приложение Qt.** В меню слева отобразятся компоненты проекта, переименуем и рассмотрим их (рис. 1).

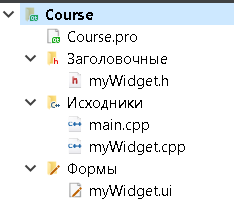
****

Рис. 1 Меню проекта

*Course.pro* – проектный файл;

*myWidget.h* – заголовочный файл (объявление класса и методов);

*main.cpp* – главный файл;

*myWidget.cpp* – файл, в котором содержатся все методы (функции, принадлежащие классу), объявленные в заголовочном файле; для лучшего понимания механизма работы приложения рассмотрим данный файл в последнюю очередь;

*myWidget.ui* – файл, содержащий графический интерфейс.

## 2.2 Проектный файл

Проектные файлы содержат всю необходимую информацию для построения приложения. О настройках конфигурации, именах файлов и путей, используемых для процесса постройки, сообщают переменные.

Списки значений в переменных HEADERS и SOURCES используется для указания заголовочных файлов и файлов ресурсов, находящихся в той же директории, что и проектный файл.

В переменной QT содержатся Qt-специфичные настройки конфигурации, а в переменной FORMS – список фалов форм, обрабатываемых uic.

Переменная TEMPLATE используется для определения типа проекта, который должен быть построен. Тип данного проекта – app – создает make-файл для постройки приложения. TARGET содержит имя данного проекта.

## 2.3 Заголовочный файл

## Работа программы невозможна без использования функций и слотов, содержащихся в стандартных классах Qt. Подключить их можно с помощью команды #include. В данной программе использованы функции из следующих классов:

* *QWidget* – базовый класс для всех объектов пользовательского интерфейса, обеспечивает взаимодействие с оконной системой.
* *QTextStream* – класс для работы с файлами – чтение, запись, обработка ошибок.
* *QFile* – обеспечивает удобный интерфейс для чтения и записи текста.

## Данная программа основана на принципах ООП. Объектно-ориентированное программирование — методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного класса, а классы образуют иерархию наследования.

Данный файл содержит конструктрор - специальный метод класса, который предназначен для инициализации элементов класса некоторыми начальными значениями.

## Объектная модель Qt подразумевает, что все построено на объектах. Фактически, класс QObject – основной, базовый класс. Любой класс, имеющий сигналы (методы, способные осуществлять пересылку сообщений; вырабатываются, когда происходят определенные события) и слоты (функции, возникающие в ответ на сигналы), должен быть унаследован от данного класса.

## В данной программе происходит множественное наследование: от класса QObject наследуется базовый класс QWidget, от него – созданный нами класс Widget. Далее определяются его объекты, методы и слоты. Рассмотрим их подробнее в пункте 2.6.

**2.4 Главный файл**

Данный файл является «стартовой площадкой» приложения. Я дала ему имя main.cpp, так как при развитии и совершенствовании проекта, он будет пополняться большим количеством новых файлов, но найти отправную точку всего проекта не составит труда.

Связываем данный файл с заголовочным, а также подключаем базовый класс **QApplication.**

Объект данного класса осуществляет контроль и управление приложением. Для его создания в конструктор класса передаются два аргумента. Первый аргумент представляет собой информацию о количестве аргументов в командной строке, с которой происходит обращение к программе, а второй – это указатель на массив символьных строк, содержащих аргументы, по одному в строке.

Данный объект должен быть создан до использования операций, связанных с пользовательским интерфейсом.

Затем создается объект ранее упомянутого класса **Widget**. Элементы управления Qt по умолчанию невидимы, поэтому для их отображения необходимо вызвать метод *show().*

Наконец, в последней строке программы приложение запускается вызовом *QApplication::exec()*. С его запуском приводится в действие цикл обработки событий, являющийся базовым для **QApplication**. Этот цикл передает получаемые от системы события на обработку соответствующим объектам. Он продолжается до тех пор, пока не закроется окно последнего элемента управления. По завершению работы приложения метод *QApplication::exec()* возвращает значение целого типа, содержащее код, информирующий о его завершении.

**2.5 Графический интерфейс**

Так как несложный процесс создания профессионально выглядящего графического интерфейса – одно из главных преимуществ Qt Creator, пора приступить к описанию создания интерфейса моего приложения.

Встроенная программа Qt Designer позволяет редактировать форму с помощью технологии drag & drop, не написав ни единой строчки программного кода. Чтобы изменить форму с помощью данной программы, достаточно дважды кликнуть по её названию (*myWidget.ui)* в меню слева.

В первую очередь я решила использовать надпись для создания стартовой формы. Она представлена виджетом **Label**, который расположен в списке виджетов слева. Необходимо выбрать его и перетянуть в форму. В результате этой операции на форме появится надпись *TextLabel* (рис. 2)

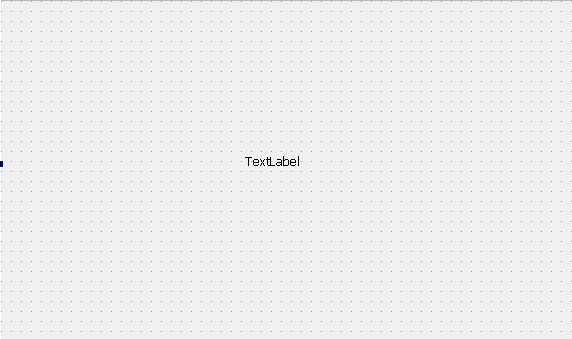


Рис. 2 Виджет **Label**

Это не тот текст, который я хотела отобразить, поэтому его нужно изменить. Для этого необходимо дважды щелкнуть по тексту надписи, после чего надпись перейдет в режим редактирования. Теперь можно набрать нужный текст «Добро пожаловать! Начать тест?»

Следующий компонент, который я использую в данной форме – виджет **Push Button** (рис. 3)

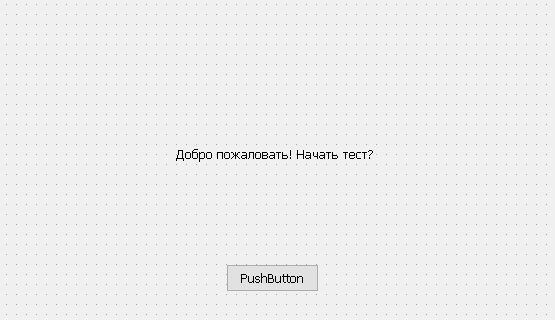


Рис. 3 Label и Push Button

Я решила изменить название объекта (поле *objectName* в характеристиках объекта) на *okButton*, так как оно лучше отображает его суть, а затем точно так же, как и для надписи, необходимо изменить текст на «Ок».

В результате наших действий, а также несложных преобразований шрифтов, стандартная форма преобразится и примет следующий вид (рис. 4)

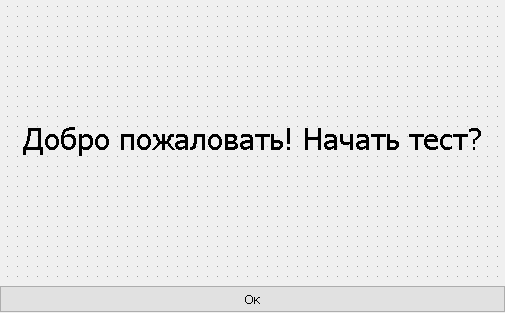


Рис. 4 Вид стартовой формы

Перейдём к созданию второй страницы формы – непосредственно каркаса для вопросов и вариантов ответов теста.

Текст вопроса я решила помещать в уже знакомый нам виджет **Label**. Перетаскиваю его из меню слева, в верхней части формы оказывается надпись **TextLabel**. В данной форме нет необходимости менять текст надписи, потому что в ходе работы программы он будет заменяться указанными строками из файла с вопросами теста. Однако, название компонента в графе *objectName* необходимо изменить, чтобы оно имело смысл. Данный компонент я назвала *questionLabel*.

Варианты ответов я решила разместить в виджете-переключателе **Radio Button.** Свое название данный виджет получил из-за сходства с радиоприемником, на панели которого может быть нажата только одна из кнопок. Нажатие на другую кнопку приводит к тому, что в группе автоматически отключается кнопка, нажатая до этого.

Переключатель может находиться в одном из двух состояний: включено (on) и выключено (off). Эти состояния пользователь может устанавливать с помощью мыши или клавиши <Пробел>, когда кнопка находится в фокусе.

Этот виджет предоставляет пользователю выбор из нескольких (в данном случае четырех) альтернатив, что делает невозможным выбор более одного варианта ответа, а также исключает возможность оставить вопрос не отвеченным (начиная со второго, для первого все же придется прописать исключение).

Расположив четыре кнопки под надписью **TextLabel**, меняю значения параметров *objectName* на их порядковые номера – first, second, third, fourth.

Последний компонент, который мне хотелось бы использовать в данной форме – знакомый нам виджет Push Button. По нажатию данной кнопки ответ, указанный пользователем (зажатая кнопка **Radio Button**), будет принят.

Разместится он в самом низу формы.

В результате добавления всех виджетов, форма примет следующий вид (рис. 5).

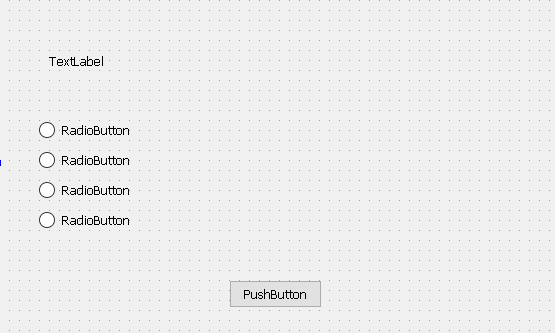


Рис. 5 Форма с виджетами

После того, как все виджеты добавлены, можно заняться их размещением. Мне хотелось бы, чтобы между вопросом, вариантами ответов и кнопкой было расстояние. В места, которые должны оставаться свободными, следует поместить заполнитель пространства (**Spacer**). Для этого в окне виджетов необходимо выбрать вертикальный заполнитель **Vertical Spacer** и поместить его в нужное место.

Изменив внешний вид формы, а также шрифты и тесты надписей, получим следующее (рис. 6).

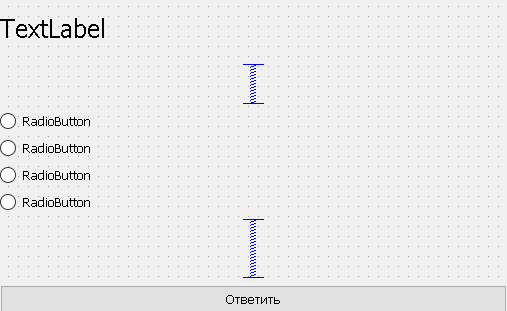


Рис. 6 Виджеты и разделители

На этом работа с графическим интерфейсом завершена, перейдем к основной части программы.

**2.6 Функционал программы**

* + 1. **Взаимодействие с пользователем**

Для меня было очевидно, что текст вопросов и вариантов ответов необходимо считывать и загружать из файла, а не прописывать непосредственно в программе.

Во-первых, это делает программу универсальной. Изменив задания и ответы, мы получим совершенно новый тест (другой вариант, предмет и т.д.).

Во-вторых, это позволяет далекому от программирования человеку (среднестатистическому школьному учителю, преподавателю университета) самостоятельно менять задания, не взаимодействуя с кодом программы. Единственный необходимый для этого навык – умение работать с Блокнотом.

Я создала специальный файл «Для преподавателя» со следующим содержанием:

«Добро пожаловать в систему автоматизированного тестирования знаний!

Данный файл - руководство по созданию тестовых заданий.

Для начала работы необходимо открыть файл Test.txt. В нем будет пример оформления тестовых заданий. В первую строку необходимо записать вопрос. Во 2-5 - четыре варианта ответов. Строка с правильным ответом должна начинаться с символа "+".

Следующий вопрос необходимо записать в новую строку, варианты ответов в следующие четыре строки.

Не пропускайте строки для разделения вопросов.

После того, как вы введете желаемое количество вопросов, просто сохраните файл.

Удачи в создании тестов!»

Рассмотрим пример оформления тестовых заданий на вопросе из демонстрационного теста по русской литературе (рис. 7).

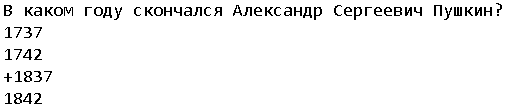


Рис. 7 Пример оформления вопроса

В силу того, что учащиеся всеми способами пытаются найти лазейку и заполучить правильные ответы, рекомендую поставить пароль на данный файл, чтобы ограничить доступ к нему.

Как же реализована работа с файлом? Рассмотрим в следующем пункте.

* + 1. **Работа с файлом**

Для начала необходимо инициализировать файл. Для этого я прописываю путь, по которому его можно найти. Если по каким-то причинам файл невозможно открыть, на экран с помощью виджета **Message Box** будет выведено критическое сообщение об этом (рис. 8).

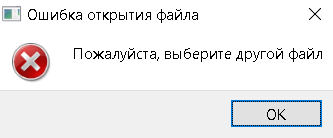


Рис. 8 Ошибка открытия файла

Познакомимся поближе с данным виджетом, так как он встретится нам еще не раз.

Диалоговое окно сообщения – это самое простое диалоговое окно, которое отображает текстовое сообщение и ожидает реакции со стороны пользователя. Его основное предназначение состоит в том, чтобы информировать пользователя об определенном событии.

Все окна, предоставляемы классом *QMessageBox* – модальные. Они могут содержать кнопки, заголовок и текст сообщения.

Класс *QMessageBox* предоставляет целую серию статических методов, с помощью которых можно создавать окна сообщений. Эти методы предоставляют поддержку сообщений трех уровней важности: информационного, предупреждающего и критического, которые выбираются в зависимости от обстоятельств. Уровень важности и текст надписи на кнопке необходимо прописывать самостоятельно (рис. 9).

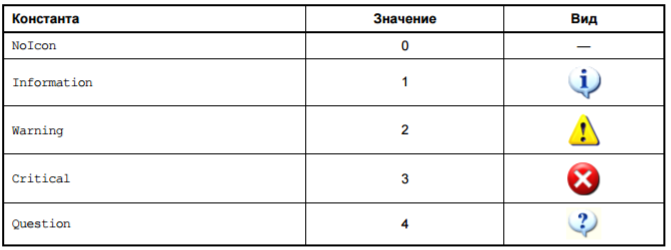


Рис. 9 Растровые изображения

После успешного открытия файла можно приступать к считыванию строк.

Текст вопроса сразу же вписывается в виджет **Label**, запускается счетчик общего количества вопросов.

С вариантами ответов немного сложнее – для начала мы проверяем первый (а для компьютера – нулевой) символ каждой из последующих четырех строк. Если для одной из них этот символ равен «+», то этот ответ помечается как правильный, символ «+» из начала строки удаляется, все вопросы записываются друг за другом в компоненты **Radio Button**. В результате заполнения виджетов строками из файла, форма принимает следующий вид (тот же самый вопрос из демонстрационного теста по русской литературе) (рис. 10).

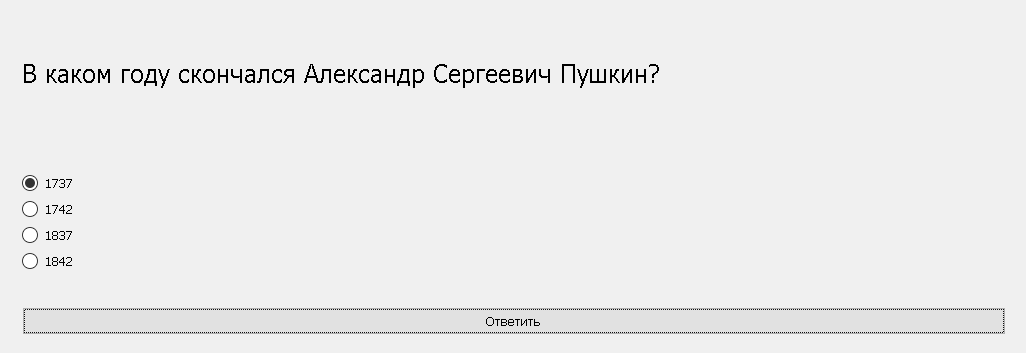


Рис. 10

Аналогичное считывание производится до конца файла. По окончании теста счетчик примет значение, равное общему количеству вопросов теста, а на экран выведется следующее информационное сообщение (рис. 11).

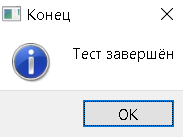


Рис. 11 Конец теста

**2.6.3 Подсчет ответов, вывод результата**

Когда пользователь начинает взаимодействовать с программой, запускаем счетчик правильных ответов, чтобы в конце вывести сообщение о соотношении правильных ответов и общего количества вопросов.

Очевидно, что на каждый из вопросов пользователь может либо выбрать правильный вариант ответа и увидеть информационное сообщение с подтверждением (рис. 12), либо ответить неверно и увидеть критическое сообщение об ошибке (рис. 13). Однако в первом вопросе пользователь может не выбрать ни один из вариантов ответа (в последующих такая возможность исключена – с предыдущего вопроса останется зажатой одна из кнопок Radio Button, и пользователь будет вынужден ответить на вопрос). На этот случай предусмотрен вывод критического сообщения о необходимости выбрать вариант ответа (рис. 14), переход к следующему вопросу будет возможен только при получении ответа от пользователя.

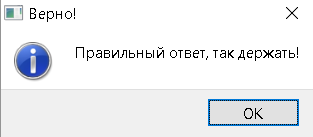


Рис. 11 Верный ответ

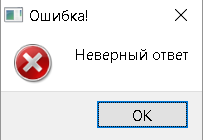


Рис. 12 Неверный ответ

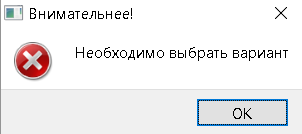


Рис. 13 Отсутствие ответа

После одного из данных сообщений форма будет обновлена, в виджеты будут помещены следующие вопросы и варинты ответов.

По окончании выполнения теста, на экран будет выведен результат – количество верных ответов и общее количество вопросов (рис. 14).

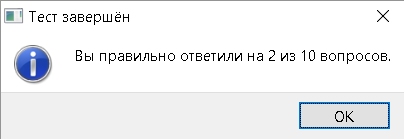


Рис. 14 Итоговый результат

На его основании учитель может выставить оценку ученику, а преподаватель автошколы принять решение, сдал ли учащийся экзамен по ПДД.

## Заключение

Целью данной курсовой работы я ставила разработку универсальной системы автоматизированного тестирования знаний, упрощающей работу преподавателя и усложняющей жизнь учащегося, простой в редактировании и в использовании.

Я считаю, что задача была выполнена, однако это лишь стартовый минимум. Программа должна совершенствоваться, направлений для ее развития бесконечное множество. Так, например, мне хотелось бы, чтобы вопросы содержали изображения (неотъемлемая часть теста ПДД), аудиофайлы (например, для людей со слабым зрением), видеофайлы (внесут разнообразие, увеличат интерес). Хотелось бы также, чтобы вопросы могли генерироваться в рандомном порядке, тогда учителю не пришлось бы создавать разные варианты одного теста (как принято в школе), данные варианты генерировались бы самостоятельно. Безусловно, стоит поработать и над дизайном приложения, благо, Qt Creator практически не ограничивает программиста в этом.

Программа должна также найти своих пользователей. Например, я собираюсь предложить ее учителям из своей школы. Это поможет выяснить, актуальна ли программа, действительно ли она облегчает и ускоряет работу, выявлены ли какие-то недостатки, есть ли пожелания. Возможно, она распространится за пределы моей школы – в другие учебные заведения города Мозыря.

А получение новых знаний и навыков в области программирования на последующих курсах поможет осуществить это.

## Список использованных источников

1. Макс Шлее "Qt 5.3. Профессиональное программирование на C++."
2. <https://ru.wikipedia.org/wiki/Qt>
3. <https://ru.wikipedia.org/wiki/Qt_Creator>
4. doc.crossplatform.ru/qt/
5. doc.qt.io/
6. <https://ru.wikipedia.org/wiki/Объектно-ориентированное_программирование>
7. https://msdn.microsoft.com