**МУНИЦИПАЛЬНОЕ АВТОНОМНОЕ ОБЩЕОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ГИМНАЗИЯ №16 ГОРОДА ТЮМЕНИ**

**ПРОЕКТНАЯ РАБОТА**

**РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ ТЕСТИРОВАНИЯ**

Автор: Сергеев Владислав Александрович,

Россия, Тюменская область,

г. Тюмень

МАОУ гимназия №16,

10 «Д» класс

Научный руководитель:

Евдокимова Мария Михайловна,

учитель информатики МАОУ гимназии №16

Тюмень, 2024 г.

# Содержание

[Введение 3](#_Toc159960433)

[1. Теория 4](#_Toc159960434)

[1.1 Классификация тестов 4](#_Toc159960435)

[1.2 Формы тестовых заданий 4](#_Toc159960436)

[1.3 Функции 5](#_Toc159960437)

[1.4 Достоинства и недостатки 6](#_Toc159960438)

[1.5 Средство разработки 8](#_Toc159960439)

[1.6 Возможности WPF 8](#_Toc159960440)

[1.7 Паттерн проектирования 10](#_Toc159960441)

[2. Практическая работа 10](#_Toc159960442)

[2. 1 Выбор среды разработки 10](#_Toc159960443)

[2.2 Шаги выполнения работы 10](#_Toc159960444)

[2.3 Описание некоторого функционала 11](#_Toc159960445)

[2.4 Итоговый результат 11](#_Toc159960446)

[Заключение 11](#_Toc159960447)

[Список литературы 11](#_Toc159960448)

[Приложения 12](#_Toc159960449)

# Введение

**Актуальность:** учителям очень часто нужно проводить тесты, поэтому такая вещь, как программа для проведения тестов, будет крайне полезна в образовательном процессе. Также программа позволит автоматизировать процесс тестирования и проводить его быстрее и экономнее, не тратя бумагу и силы учителя на проверку.

**Проблема:** может ли программа заменить обычные бумажные тесты? Возможно, ли написать свой тестировщик?

**Цель:** разработать программу-тестировщик при помощи Windows Presentation Foundation

**Задачи:**

1. Изучить систему тестирования.
2. Определить логику работы программы.
3. Разработать отдельный редактор тестов
4. Разработать проверяющую систему.
5. Отладить ошибки и провести тестирование программы.

**Предмет исследования:** Приложение-тестировщик на C#

**Методы:**

1. Анализ необходимой для написания программы литературы
2. Организация бизнес-логики приложения

# 1. Теория

**Программное обеспечение для тестирования (далее — программа-тестировщик) —** это система программ для создания и проведения компьютерного тестирования знаний, сбора и анализа результатов.

**Педагогический тест** — это инструмент оценивания обученности учащихся, состоящий из системы тестовых заданий, стандартизованной процедуры проведения, обработки и анализа результатов.

## 1.1 Классификация тестов

* по целям — информационные, диагностические, входные, обучающие, мотивационные, аттестационные;
* по процедуре создания — стандартизованные, не стандартизованные;
* по способу формирования заданий — детерминированные, стохастические, динамические;
* по технологии проведения — бумажные, в том числе бумажные с использованием оптического распознавания, компьютерные;
* по форме заданий — закрытого типа, открытого типа, установление соответствия, упорядочивание последовательности;
* по наличию обратной связи — традиционные и адаптивные

## 1.2 Формы тестовых заданий

* Задания с выбором одного правильного ответа

*При наборе текста слова отделяются друг от друга …*

*а) двоеточием;*

*б) запятой;*

*в) пробелом;*

*г) точкой.*

* Задания с выбором одного неправильного ответа

*Операция не имеет признака, по которому подобраны остальные операции, представленные в списке…*

*а) сохранение текста;*

*б) форматирование текста;*

*в) удаление фрагмента текста;*

*г) перемещение фрагмента теста;*

*д) копирование фрагмента текста.*

* Задания на установление соответствия

*Установите соответствие между командами и сочетанием клавиш.*

|  |  |
| --- | --- |
| *Команда* | *Сочетание клавиш* |
| *1. Вырезать фрагмент текста;* | *а) CTRL+X;* |
| *2. Копировать фрагмент текста;* | *б) CTRL+C;* |
| *3. Вставить фрагмент текста.* | *в) CTRL+V.* |

* Задания с выбором нескольких правильных ответов

*Использование слепого десятипальцевого метода ведет к …*

*а) снижению напряжения на пальцы;*

*б) уменьшению скорости печати;*

*в) уменьшению количества опечаток и ошибок;*

*г) быстрой утомляемости пальцев.*

* Упорядочивание последовательности

*Расположите в хронологическом порядке*

*1. Бородинская битва*

*2. Ледовое побоище*

*3. Куликовская битва*

* *Задания с открытым ответом*

Существует два способа освоения клавиатуры при печатании слепым десятипальцевым методом:

1. соло на клавиатуре ergosolo.ru \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. stamina\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 1.3 Функции

Тестирование в педагогике выполняет три основные взаимосвязанные функции: **диагностическую**, **обучающую** и **воспитательную**:

* **Диагностическая функция** заключается в выявлении уровня знаний, умений, 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.

## 1.4 Достоинства и недостатки

По сравнению с другими формами контроля знаний тестирование имеет свои преимущества и недостатки.

**Достоинства**

Тестирование является более качественным и объективным способом оценивания, его объективность достигается путём стандартизации процедуры проведения, проверки показателей качества заданий и тестов целиком.

* 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.
* 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.
* 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.
* 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.
* 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**

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.

* 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.
* Тест не позволяет проверять и оценивать высокие, продуктивные уровни знаний, связанные с творчеством, то есть вероятностные, абстрактные и методологические знания.
* 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.
* Обеспечение объективности и справедливости теста требует принятия специальных мер по обеспечению конфиденциальности тестовых заданий. При повторном применении теста желательно внесение в задания изменений.
* 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.

## 1.5 Средство разработки

Программа-тестировщик разработана при помощи **Windows Presentation Foundation** (далее — WPF) — система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая (презентационная) подсистема в составе .NET Framework, использующая язык XAML

XAML (англ. e**X**tensible **A**pplication **M**arkup **L**anguage) — расширяемый язык разметки для приложений — основанный на XML язык разметки для декларативного программирования приложений, разработанный Microsoft.

Для работы с WPF требуется любой .NET-совместимый язык. В этот список входит множество языков: C#, F#, C++, Ruby, Python, и многие другие. Для разработки был выбран язык C#.

## 1.6 Возможности WPF

* **Привязка данных**

Это гибкий механизм, который позволяет через расширения разметки XAML связывать различные данные (от значений свойств элементов управления до общедоступных свойств, реализующих поля базы данных через Entity Framework). Привязка данных представлена классом Binding, который в свою очередь унаследован от MarkupExtension, что позволяет использовать привязки не только в коде, но и в разметке:

**<StackPanel** Orientation="Horizontal"**>**

**<Slider** x:Name="slider" Width="200" Minimum="1" Maximum="100" Value="60"**/>**

**<TextBox** Text="{Binding ElementName=slider, Path=Value, Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}"**/>**

**</StackPanel>**

* **Стили**

Позволяют создавать стилевое оформление элементов и, как правило, используются только в разметке:

**<Button>**

**<Button.Style>**

**<Style** TargetType="Button"**>**

**<Setter** Property="FontSize" Value="20"**/>**

**<Setter** Property="Foreground" Value="LimeGreen"**/>**

**</Style>**

**</Button.Style>**

**</Button>**

* **Шаблоны элементов управления**

Позволяют менять графическое оформление элементов и представлены классом ControlTemplate. В отличие от стилей, можно менять не только графическое представление элемента, но и его структуру. При этом шаблон элемента управления задается через свойство Template.

Простой пример круглой кнопки:

**<Button** Content="Hey!" Background="LimeGreen" Foreground="White"**>**

**<Button.Template>**

**<ControlTemplate** TargetType="Button"**>**

**<Grid>**

**<Ellipse** Fill="{TemplateBinding Background}" Stroke="{TemplateBinding BorderBrush}" Stretch="Fill"**/>**

**<ContentPresenter** VerticalAlignment="Center" HorizontalAlignment="Center"**/>**

**</Grid>**

**</ControlTemplate>**

**</Button.Template>**

**</Button>**

## 1.7 Паттерн проектирования

В качестве паттерна проектирования (шаблона кода) был взят паттерн MVVM. **MVVM**— это паттерн разработки, позволяющий разделить приложение на три функциональные части **(Приложение №1)**:

* **Model** — основная логика программы (работа с данными, вычисления, запросы и так далее).
* **ViewModel** — модель представления, которая служит прослойкой между View и Model.
* **View** — вид или представление (пользовательский интерфейс).

# 2. Практическая работа

## 2. 1 Выбор среды разработки

Для разработки приложения было выбрана **IDE** (среда разработки) от **JetBrains — Rider**. Это кроссплатформенная интегрированная среда разработки программного обеспечения для платформы .NET, поддерживающий языки программирования C#, VB.NET и F#.

## 2.2 Шаги выполнения работы

1. Создание репозитория Git

Воспользовавшись системой управления версий Git и веб-сервисом GitHub, создан репозиторий «SchoolProjectTestSystem» **(Приложение №2 и №3)**.

1. Создание проекта

Перед тем, как начать разработку, необходимо создать проект в Rider. **(Приложение №4).**

1. Написание кода классов Model (модель)

Были написаны несколько моделей:

* + Модель варианта ответа в вопросе **(Приложение №5)**
  + Модель теста **(Приложение №6)**
  + Модель вопроса «Выбери 1 из 4 вариантов» **(Приложение №7)**

1. Написание кода класса ViewModel (модель представления)

* ViewModel редактора тестов **(Приложение №8)**
* ViewModel тестирующей системы **(Приложение №9)**

1. Написание разметки XAML (представления)

* Окно для редактора тестов **(Приложение №10)**
* Окно тестирующей системы **(Приложение №11)**

## 2.3 Описание некоторого функционала

1. Метод **OpenTest()** позволяет открывать тест из файла. **(Приложение №12)**
2. Методы **NextQuestion()** и **PreviousQuestion()** переключают вопросы. **(Приложение №13)**
3. Метод **GetTestResult()** подводит итоги теста и считает кол-во правильных ответов и оценку. **(Приложение №14)**
4. История изменений программы с помощью **системы управления версий Git (Приложение №15)**

## 2.4 Итоговый результат

* + Работа редактора тестов **(Приложение №16, №17, №18)**
  + Работа тестирующей системы **(Приложение №19, №20, №21, №22)**

# Заключение

Программа-тестировщик в будущем сможет заменить обычные тесты на бумаге. У меня получилось написать собственную программу-тестировщик. В будущем можно расширить функционал приложения, добавив другие виды вопросов и улучшив дизайн, тем самым увеличив полезность данной программы.

Тема раскрыта, актуальность доказана. проблема решена. цели и задачи были выполнены.

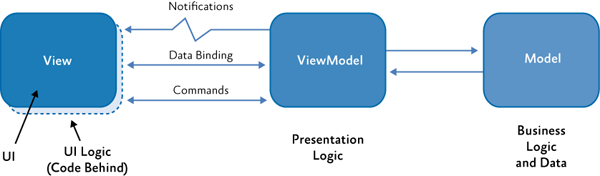
# Список литературы

1. <https://ru.wikipedia.org/wiki/Windows_Presentation_Foundation>
2. <https://ru.wikipedia.org/wiki/XAML>
3. <https://ru.wikipedia.org/wiki/Педагогическое_тестирование>
4. <https://design-hero.ru/articles/364300/>
5. <https://skillbox.ru/media/code/mvvm_proektirovanie_prilozheniy_dlya_windows/>

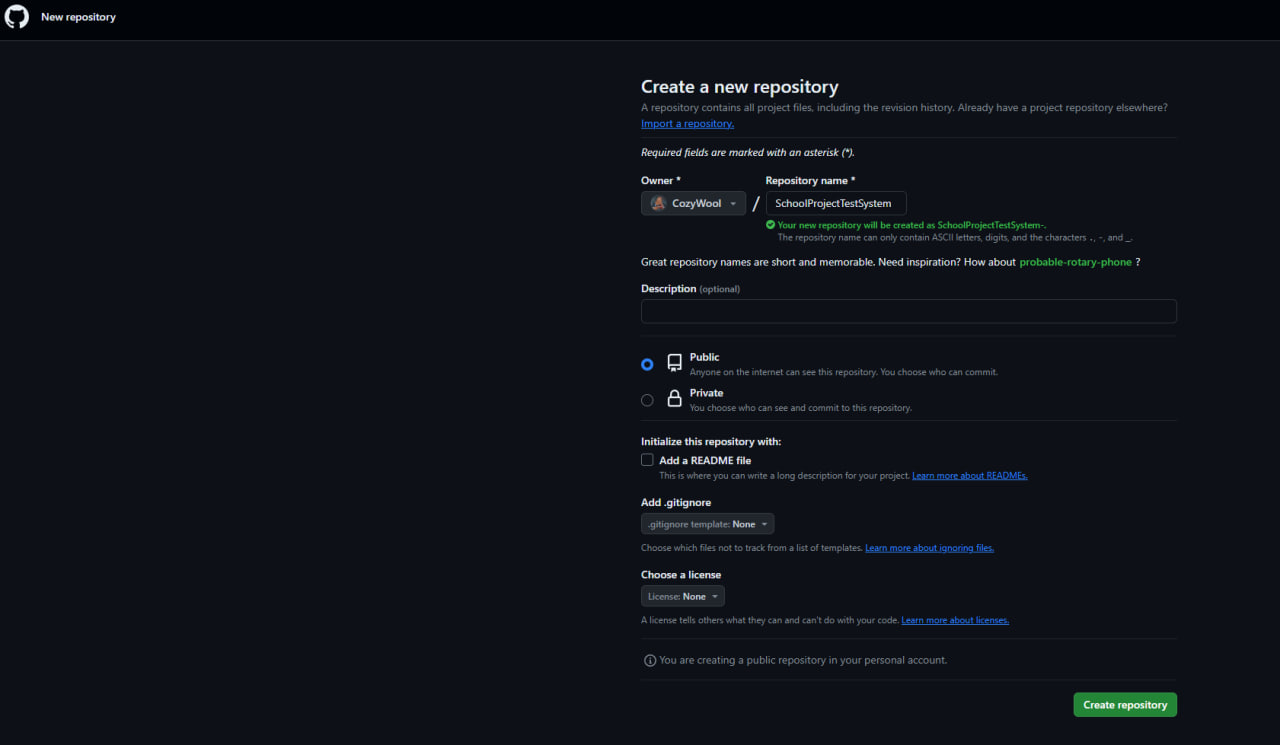
# 

# Приложения

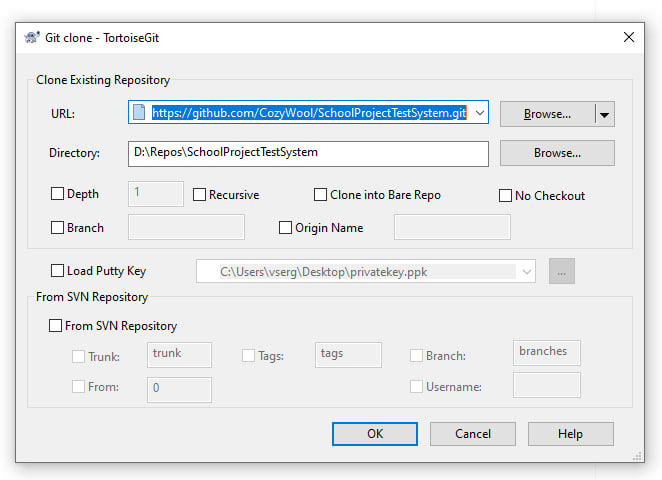
Приложение 1 — Диаграмма работы паттерна MVVM



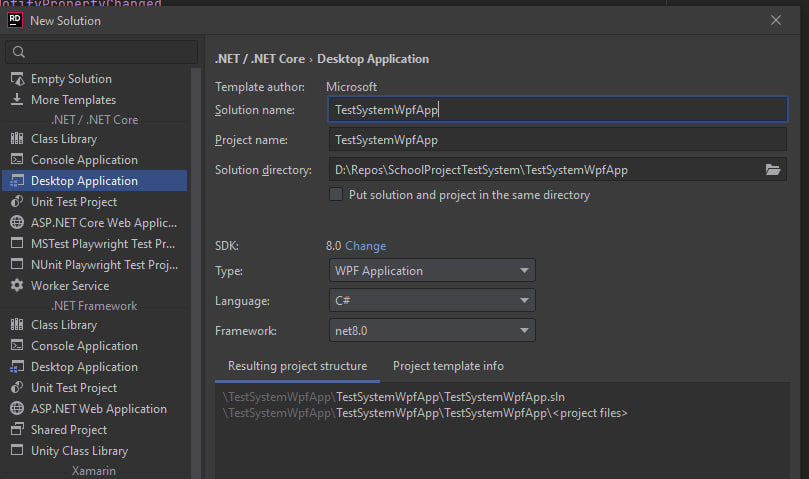
Приложение 2 — Создание репозитория на GitHub



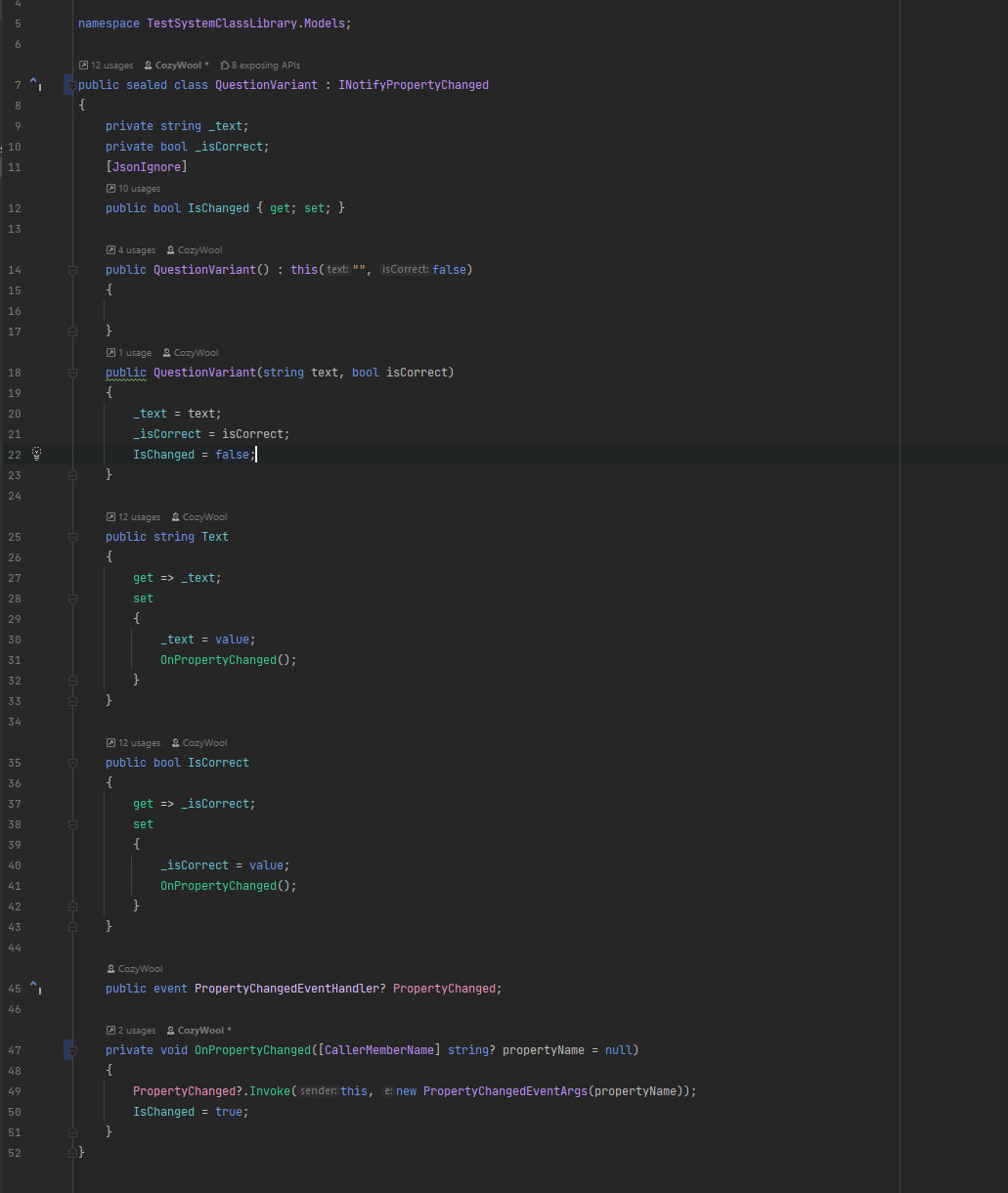
Приложение 3 — Клонирование репозитория с помощью TortoiseGit



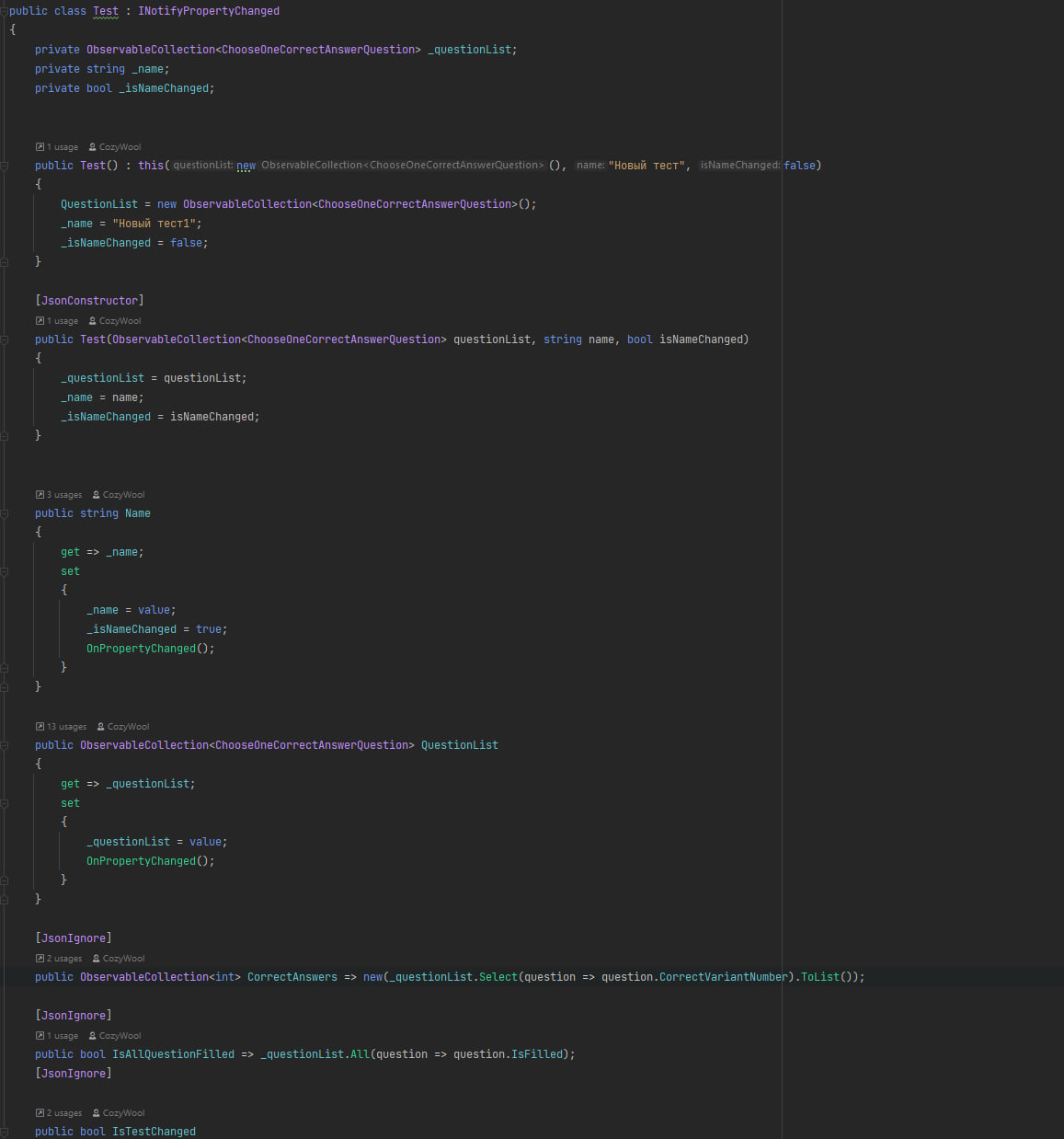
Приложение 4 — Создание проекта в Rider



Приложение 5 — Модель варианта ответа на вопрос



Приложение 6 — Модель теста



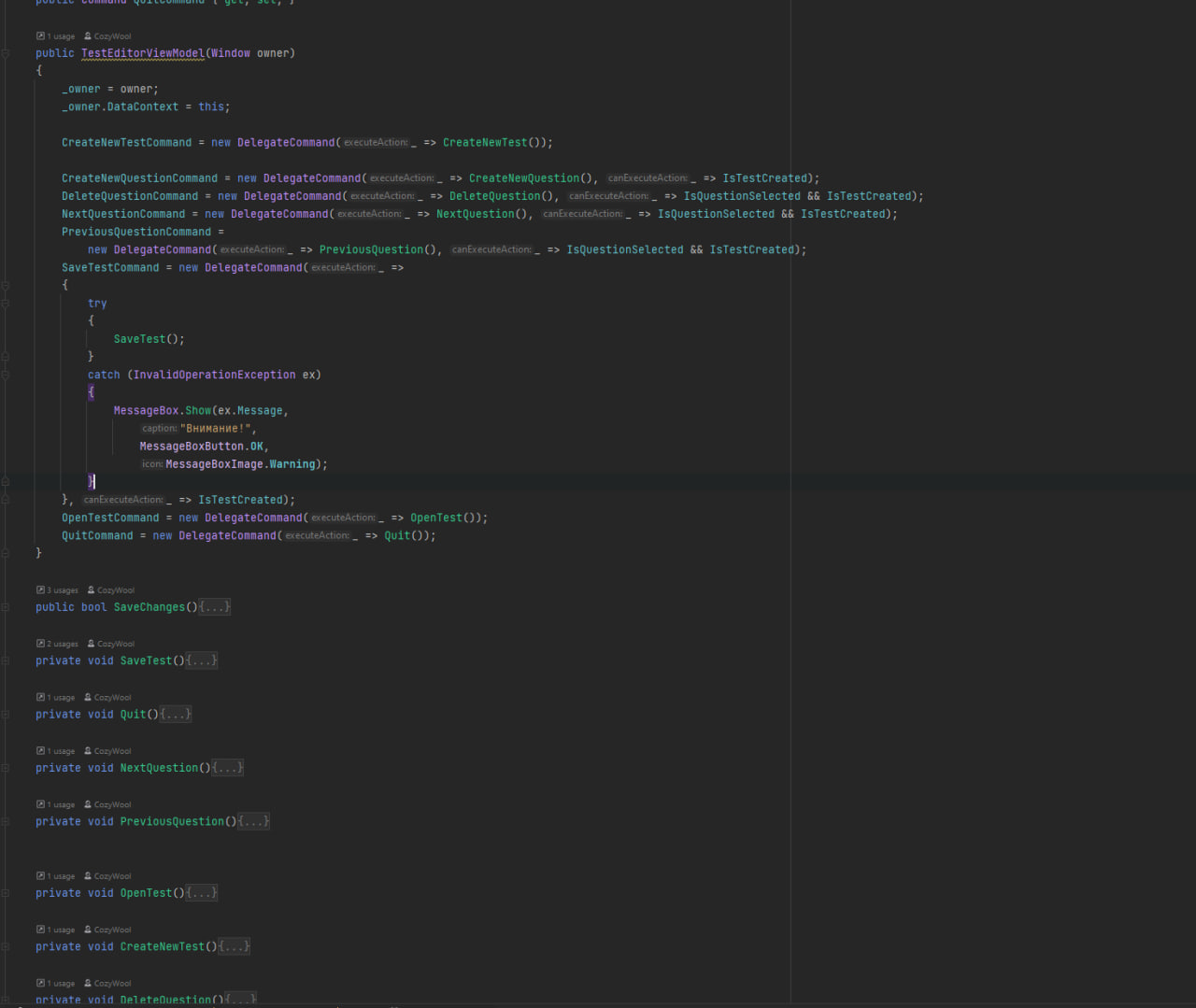
Приложение 7 — Модель вопроса «Выбери 1 из 4 вариантов»



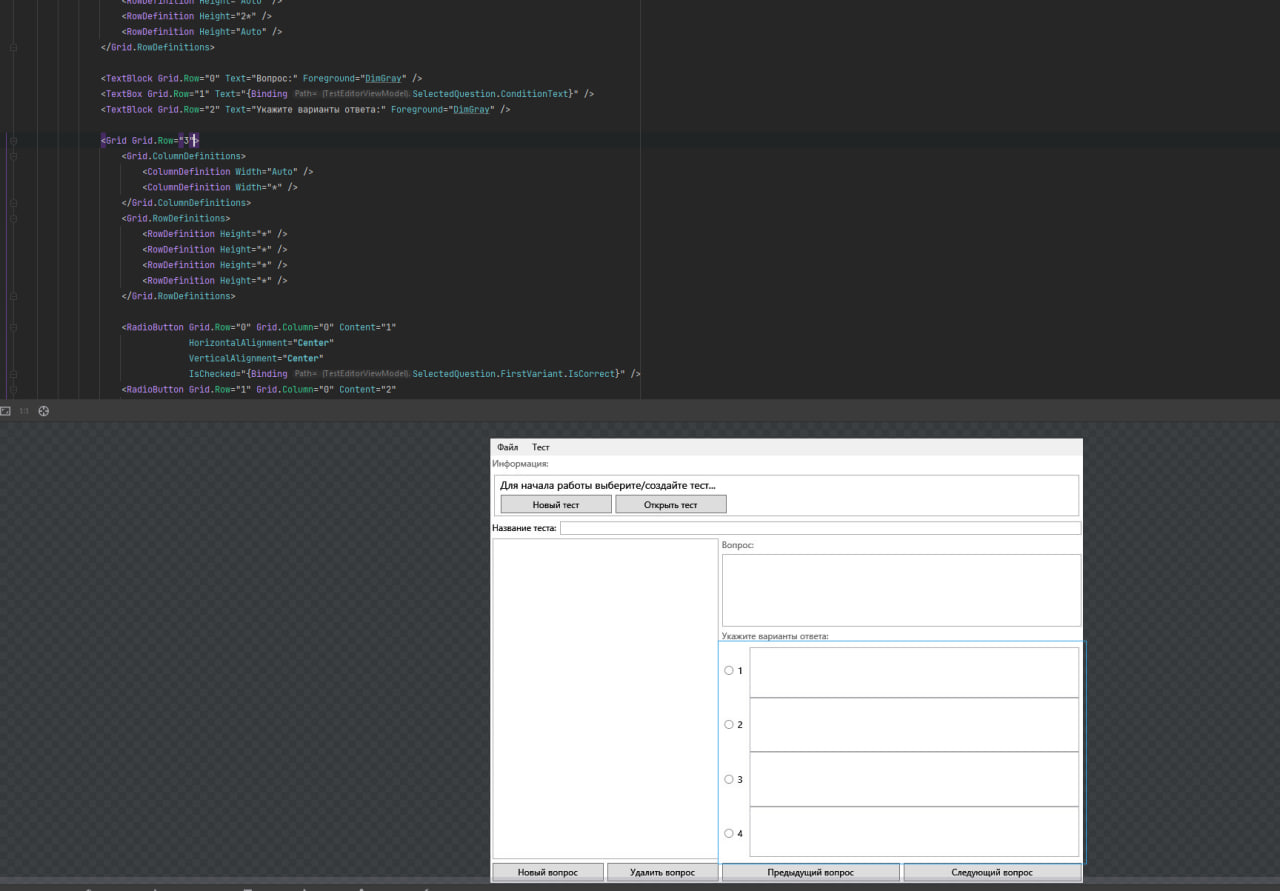
Приложение 8 — Модель представления тестирующей системы



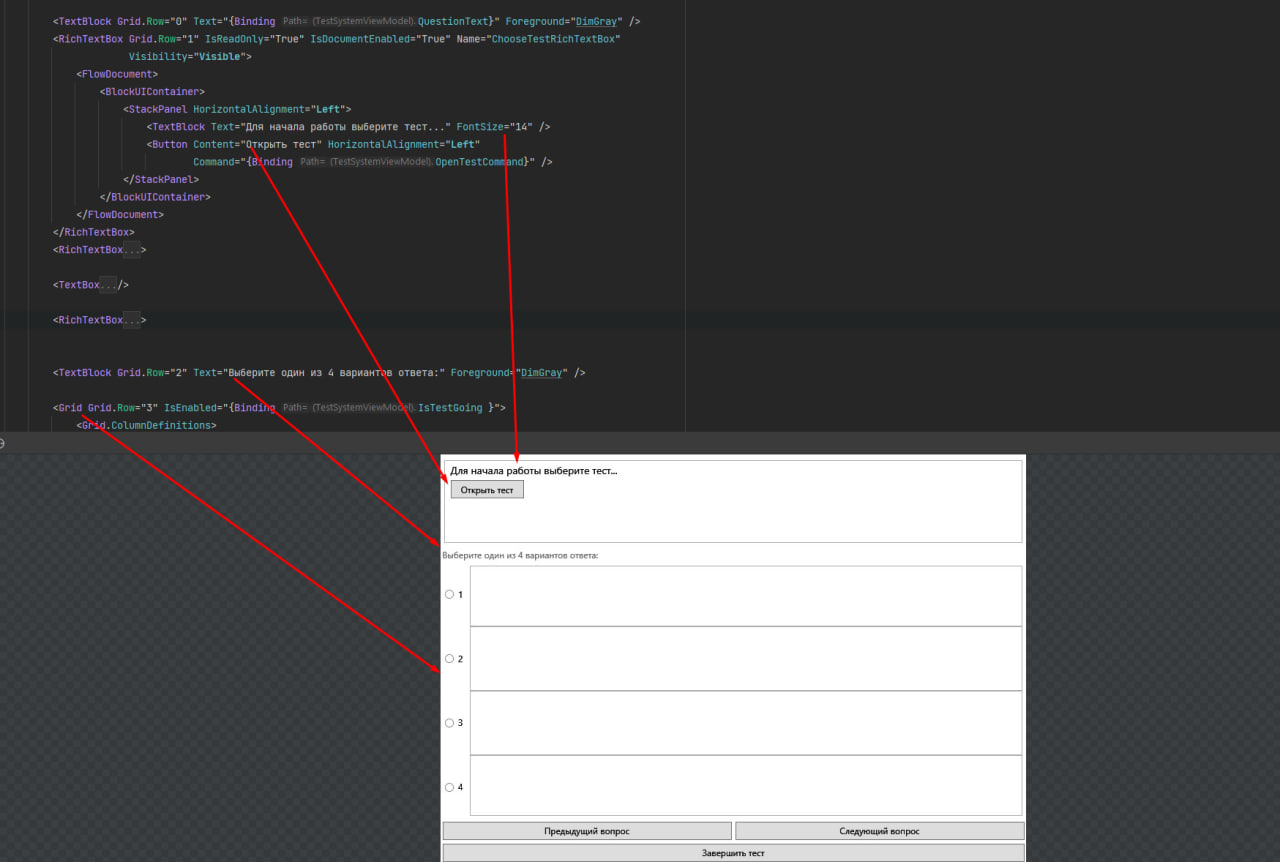
Приложение 9 — Модель представления редактора тестов



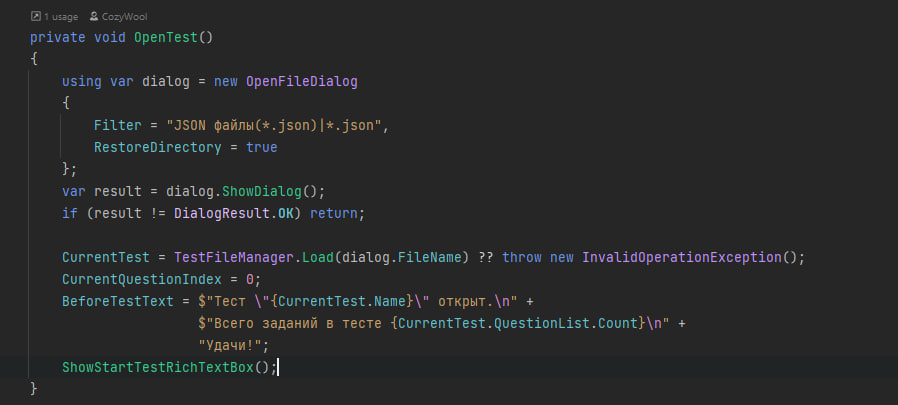
Приложение 10 — Представление редактора тестов



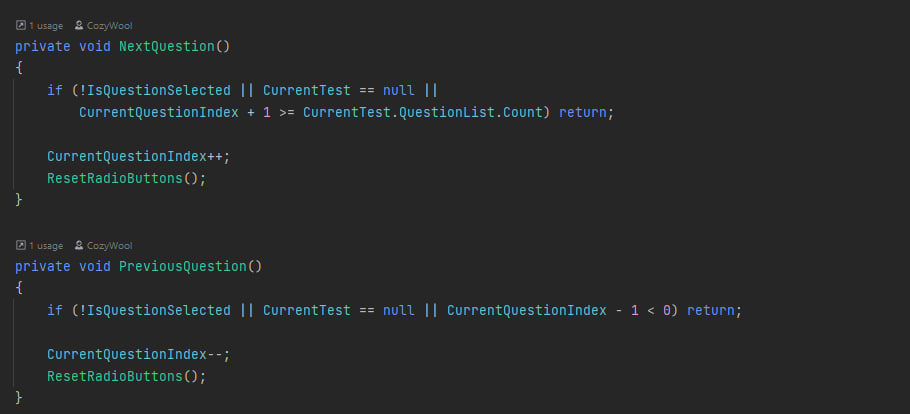
Приложение 11 — Представление тестирующей системы



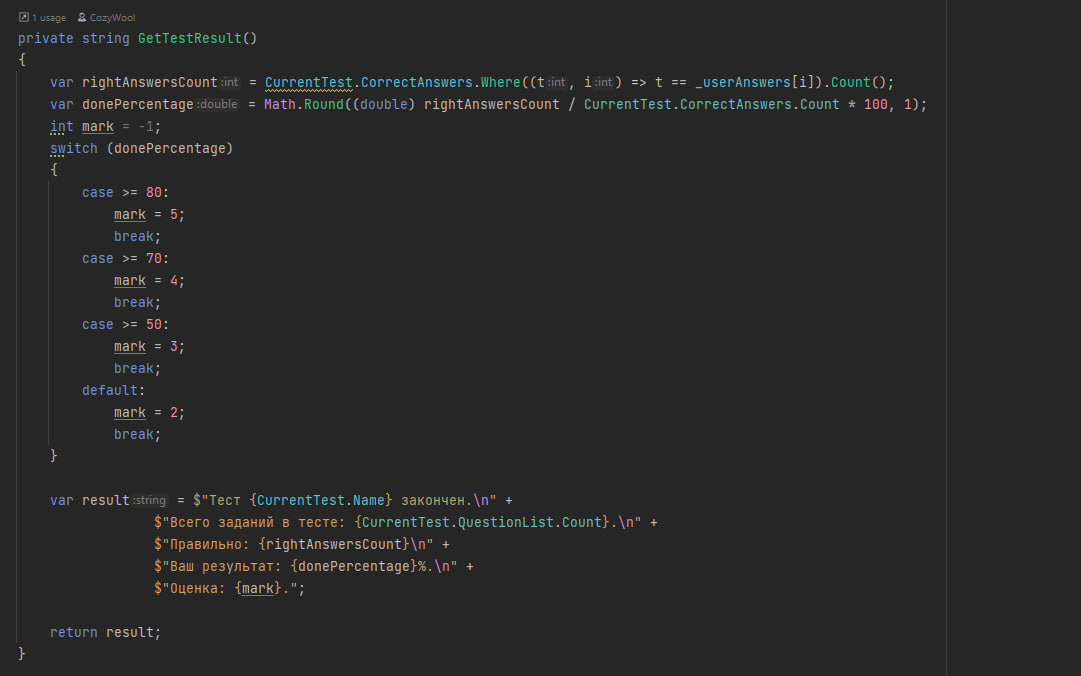
Приложение 12 — Метод OpenTest()



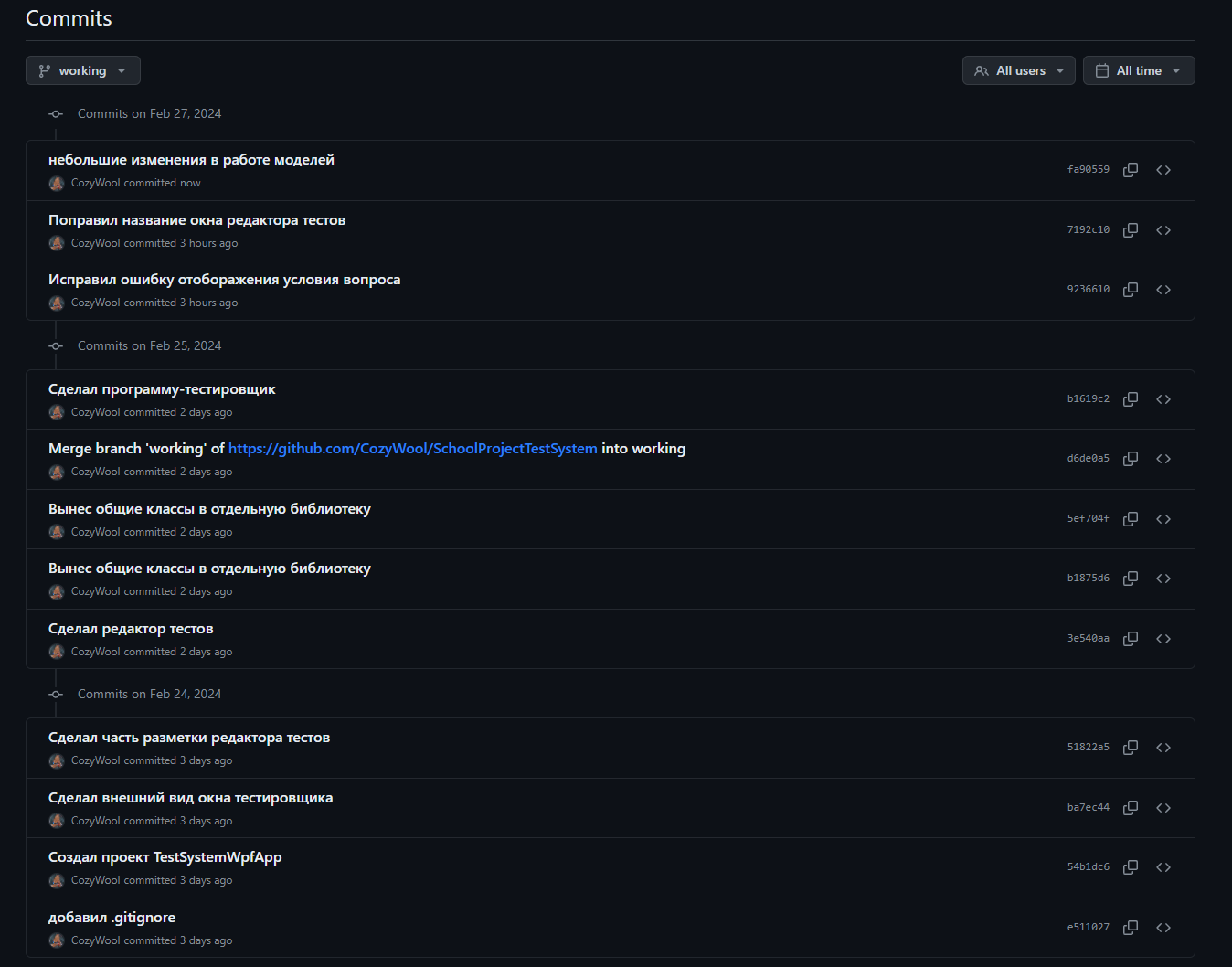
Приложение 13 — Методы NextQuestion() и PreviousQuestion()



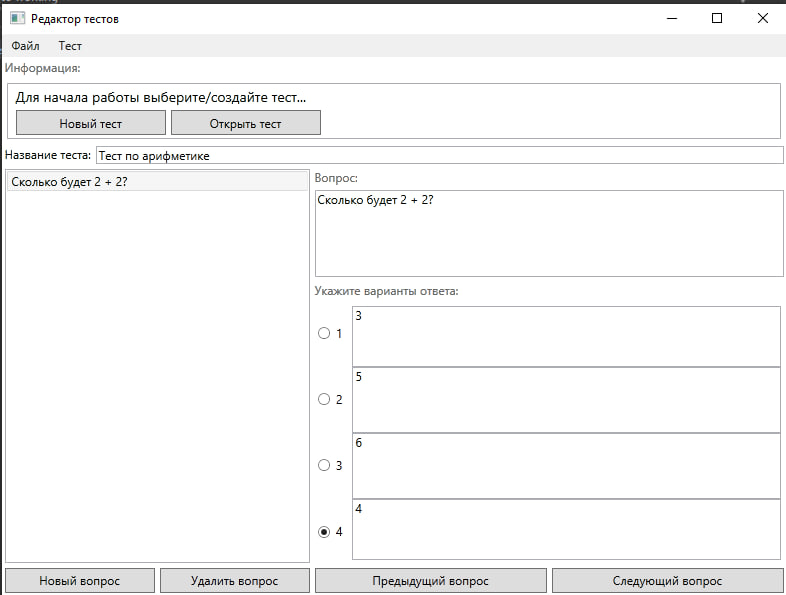
Приложение 14 — Метод GetTestResult()



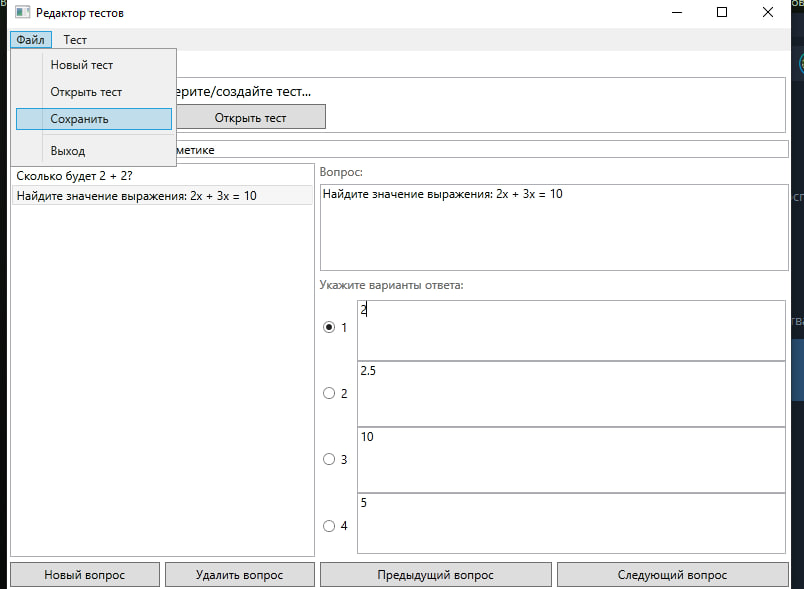
Приложение 15 — История изменения программы на GitHub



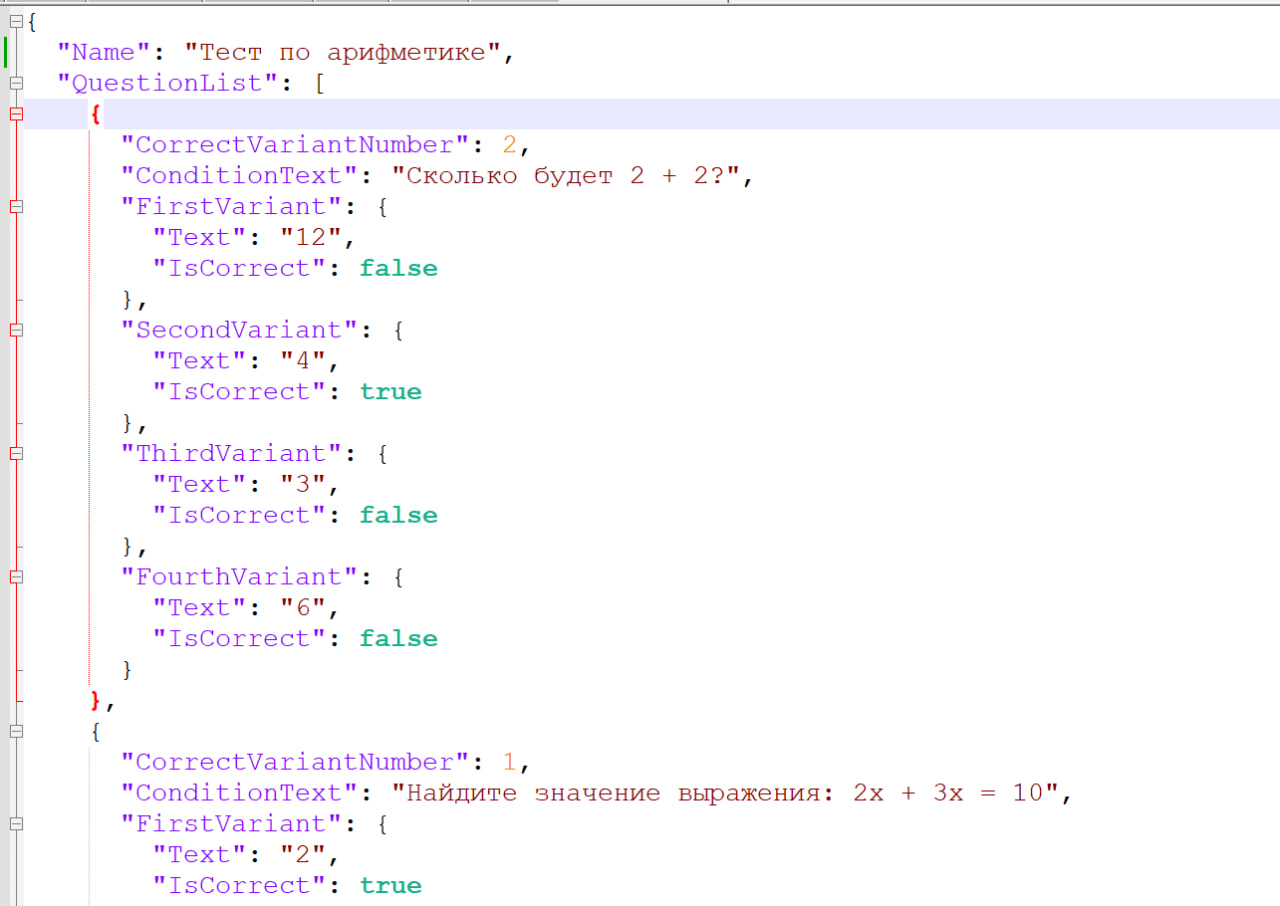
Приложение 16 — Создание вопроса в редакторе тестов



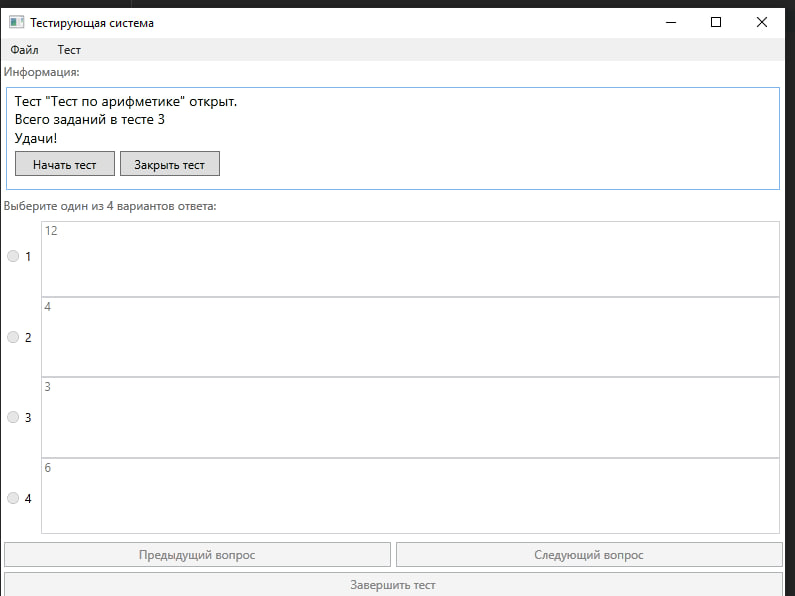
Приложение 17 — Сохранение теста в файл



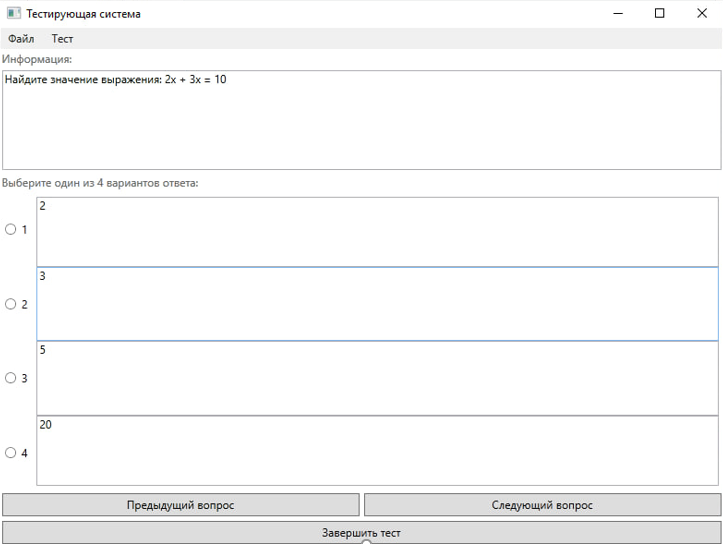
Приложение 18 — Вид теста в файле



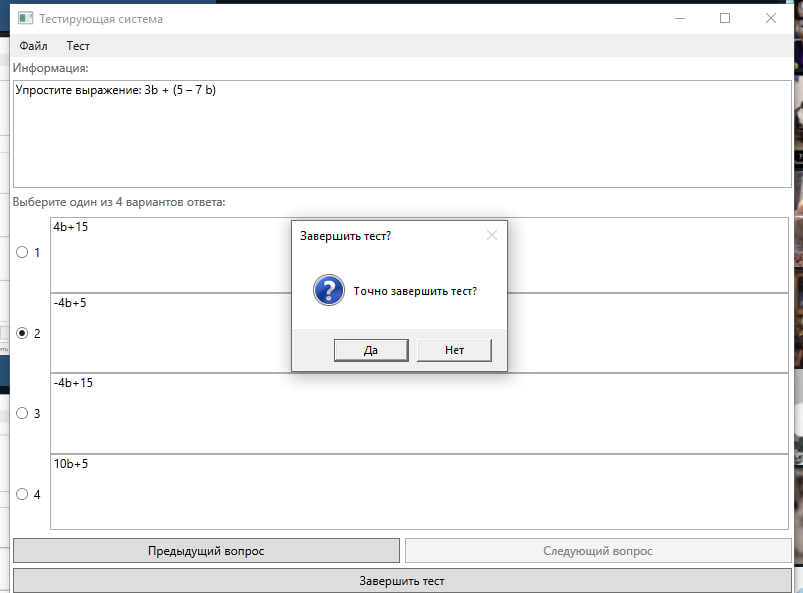
Приложение 19 — Открытие теста из файла



Приложение 20 — Интерфейс вопроса



Приложение 21 — Завершение теста



Приложение 22 — Результаты тестирования

