МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированные технологии программирования и стандарты проектирования»

Тема «Приложение для прохождения тестов и решения математических задач»

Исполнитель

студент 2 курса 4 группы Костюкова Анна Олеговна

Руководитель работы ассистент Рауба А.А.

Курсовой проект защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель Пацей Н.В. \_

Минск 2020

Оглавление

[**Введение** 3](#_Toc40035012)

[1. Теоретическое введение 4](#_Toc40035013)

[1.1 Объектно-ориентированное программирование 4](#_Toc40035014)

[1.3 Архитектура платформы .NET Framework 5](#_Toc40035015)

[**2. Обзор прототипов** 5](#_Toc40035016)

# **Введение**

Цифровые технологии сильно изменили жизнь человека. Компьютер предоставляет большие возможности для работы с разного вида информацией.

В настоящее время существует большое количество программ, которые позволяют создавать и проходить тесты. Компьютер как средство для тестирования имеет свои достоинства и недостатки. Однако в объективности оценки ему отказать трудно. Видимо, именно этот фактор способствует расширению сферы применения компьютерного тестирования - оно успешно используется в школах, вузах, при поступлении на работу.

Сегодня разработчики предлагают разнообразные компьютерные инструменты для подготовки тестовых заданий, начиная от различных редакторов и программ для разработки презентаций и до использования языков программирования и возможностей сети Интернет. Но разработка качественного тестового инструментария - длительный, трудоемкий и дорогостоящий процесс.

Целью данного курсового проекта является разработка программного средства, позволяющего создавать и проходить тесты. В качестве интерфейса прикладного программирования был выбран обширный API-интерфейс - Windows Presentation Foundation (WPF), предназначенный для создания настольных программ с графически насыщенным пользовательским интерфейсом. В основе графической технологии WPF лежит мощная инфраструктура, основанная на DirectX. Это является одним из основных отличий WPF от более ранней технологии создания пользовательских интерфейсов - Windows Forms.

Для работы с WPF использовался объектно-ориентированный язык программирования с С-подобным синтаксисом - С#, разработанный для создания приложений на платформе Microsoft .NET Core. Кроме того, используемая в WPF модель разделения кода и дизайна предполагала работу с декларативным языком описания интерфейса - eXtensible Application Markup Language (XAML).

# 1. Теоретическое введение

## 1.1 Объектно-ориентированное программирование

Объектно-ориентированное программирование - это метод программирования, при использовании которого главными элементами программы являются понятия объектов и классов.

ООП возникло в результате развития идеологии процедурного программирования, где данные и подпрограммы (процедуры, функции) их обработки формально не связаны. В центре ООП находится понятие объекта.

Объект - это сущность, экземпляр класса, в которой можно посылать сообщения, и которая может на них реагировать, используя свои данные. Данные объекта скрыты от остальной программы. Сокрытие данных называется инкапсуляцией.

Наличие инкапсуляции достаточно для объектности языка программирования, но ещё не означает его объектной ориентированности - для этого требуется наличие наследования.

Но даже наличие инкапсуляции и наследования не делает язык программирования в полной мере объектным с точки зрения ООП. Основные преимущества ООП проявляются только в том случае, когда в языке программирования реализован полиморфизм; то есть возможность объектов с одинаковой спецификацией иметь различную реализацию.

Абстрагирование - это способ выделить набор значимых характеристик объекта, исключая из рассмотрения не значимые. Инкапсуляция - это свойство системы, позволяющее объединить данные и методы, работающие с ними в классе, и скрыть детали реализации от пользователя. Наследование - это свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью. Класс, от которого производится наследование, называется базовым, родительским или суперклассом. Новый класс - потомком, наследником или производным классом. Полиморфизм - это свойство системы использовать объекты с одинаковым интерфейсом без информации о типе и внутренней структуре объекта.

## 1.2 C#

C# - язык программирования, сочетающий объектно-ориентированные и контекстно-ориентированные концепции. Разработан в 1998-2001 годах группой инженеров под руководством Андерсa Хейлсбергa в компании Microsoft как основной язык разработки приложений для платформы Microsoft .NET. Компилятор с C# входит в стандартную установку самой .NET, поэтому программы на нём можно создавать и компилировать даже без инструментальных средств вроде Visual Studio.

C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет строгую статическую типизацию, поддерживает полиморфизм, перегрузку операторов, указатели на функции-члены классов, атрибуты, события, свойства, исключения, комментарии в формате XML. Переняв многое от своих предшественников - языков C++, Delphi, Modula и Smalltalk - С#, опираясь на практику их использования, исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем: так, C# не поддерживает множественное наследование классов (в отличие от C++).

## 1.3 [Архитектура платформы .NET Framework](javascript:void(0))

Появившись в 2002 году, платформа .NET прошла долгий путь и смогла завоевать сильные позиции как надёжное решение корпоративного уровня. Долго время .NET была мультиплатформенной средой, позволяющей создавать серверные и десктопные приложения, приложения для мобильных и встраиваемых устройств, приложения для развёртывания в облачной инфраструктуре. Но при этом не являлась кроссплатформенной и поддерживала исключительно операционные системы семейства Windows.

Новый .NET работает под Linux и Mac OS X. Именно возможность запуска платформы на ОС, отличных от Windows, и вызвала в свое время множество споров и обсуждений. Хотя, на самом деле, задолго до появления .NET Core уже существовали кроссплатформенные реализация .NET Framework.

.NET Core — это спроектированная практически с нуля платформа, изначально рассчитанная на работу с различными ОС. При этом большая часть кода которой писалась с тем, чтобы платформенно-специфичных зависимостей было как можно меньше.

По своей сути .NET Core — это практически полная перезагрузка стека .NET Framework. Из новой платформы по разным причинам был исключён ряд технологий. Следует понимать, что платформа .NET Core рассчитана в первую очередь на разработку для серверных и облачных решений. Для десктопных приложений лучше подходят классический .NET для Windows (с поддержкой WPF и Windows Forms) и Mono для Linux и Mac OS X (с поддержкой Windows Forms). Мобильные проекты можно создавать, используя Xamarin.

Таким образом, в .NET Core были исключены:

* ASP.NET WebForms;
* WCF;
* WPF;
* Windows Forms.

Зато инструменты для разработки консольных и веб-приложения получили новый этап развития. При разработке большинство необходимых компонент приложения могут загружаться как отдельные модули через пакетный менеджер NuGet. Это позволяет уменьшить количество избыточных зависимостей и общий размер готового продукта.

## 1.4 Entity Framework Core

Entity Framework Core (EF Core) представляет собой объектно-ориентированную, легковесную и расширяемую технологию от компании Microsoft для доступа к данным. EF Core является ORM-инструментом (object-relational mapping - отображения данных на реальные объекты). То есть EF Core позволяет работать базами данных, но представляет собой более высокий уровень абстракции: EF Core позволяет абстрагироваться от самой базы данных и ее таблиц и работать с данными независимо от типа хранилища. Если на физическом уровне мы оперируем таблицами, индексами, первичными и внешними ключами, но на концептуальном уровне, который нам предлагает Entity Framework, мы уже работаем с объектами.

Entity Framework Core поддерживает множество различных систем баз данных. Таким образом, мы можем через EF Core работать с любой СУБД, если для нее имеется нужный провайдер.

По умолчанию на данный момент Microsoft предоставляет ряд встроенных провайдеров: для работы с MS SQL Server, для SQLite, для PostgreSQL. Также имеются провайдеры от сторонних поставщиков, например, для MySQL.

Также стоит отметить, что EF Core предоставляет универсальный API для работы с данными. И если, к примеру, мы решим сменить целевую СУБД, то основные изменения в проекте будут касаться прежде всего конфигурации и настройки подключения к соответствующим провайдерам. А код, который непосредственно работает с данными, получает данные, добавляет их в БД и т.д., останется прежним.

Entity Framework Core многое унаследовал от своих предшественников, в частности, Entity Framework 6. В тоже время надо понимать, что EF Core - это не новая версия по отношению к EF 6, а совершенно иная технология, хотя в целом принципы работы у них будут совпадать. Поэтому в рамках EF Core используется своя система версий. Текущая версия - 3.0 была выпущена в сентябре 2019 года. И технология продолжает развиваться.

Как технология доступа к данным Entity Framework Core может использоваться на различных платформах стека .NET. Это и стандартные платформы типа Windows Forms, консольные приложения, WPF, UWP и ASP.NET Core. При этом кроссплатформенная природа EF Core позволяет задействовать ее не только на ОС Windows, но и на Linux и Mac OS X.

Центральной концепцией Entity Framework является понятие сущности или entity. Сущность определяет набор данных, которые связаны с определенным объектом. Поэтому данная технология предполагает работу не с таблицами, а с объектами и их коллекциями.

Любая сущность, как и любой объект из реального мира, обладает рядом свойств. Например, если сущность описывает человека, то мы можем выделить такие свойства, как имя, фамилия, рост, возраст. Свойства необязательно представляют простые данные типа int или string, но могут также представлять и более комплексные типы данных. И у каждой сущности может быть одно или несколько свойств, которые будут отличать эту сущность от других и будут уникально определять эту сущность. Подобные свойства называют ключами.

При этом сущности могут быть связаны ассоциативной связью один-ко-многим, один-ко-одному и многие-ко-многим, подобно тому, как в реальной базе данных происходит связь через внешние ключи.

Отличительной чертой Entity Framework Core, как технологии ORM, является использование запросов LINQ для выборки данных из БД. С помощью LINQ мы можем создавать различные запросы на выборку объектов, в том числе связанных различными ассоциативными связями. А Entity Framework при выполнении запроса транслирует выражения LINQ в выражения, понятные для конкретной СУБД (как правило, в выражения SQL).

При разработке приложения были использованы:

* интегрированная среда разработки Microsoft Visual Studio 2019;
* программная платформа .NET Core;
* язык программирования C#;
* расширяемый язык разметки XAML;
* технология WPF;
* Entity Framework Core.

**2. Обзор прототипов**

Программное средство *«Конструктор тестов»*

Конструктор тестов - программа для создания и проведения тестов. Приложение позволяет:

* создавать тесты из пяти различных типов вопросов;
* форматировать текст в вопросах;
* выводить тесты и результаты тестирования на печать;
* проверять орфографию;
* настраивать интерфейс приложения;
* использовать неограниченное количество тем и вопросов;
* задавать каждому вопросу индивидуальную оценку в баллах;
* искать вопросы в базе;
* включать в вопросы мультимедийные файлы (картинки, звук, видео).

Программа содержит три модуля: один предназначен для составления тестов и заполнения базы вопросов (рисунок 1.1); второй для проведения тестирования и третий - для оценки результатов тестирования.

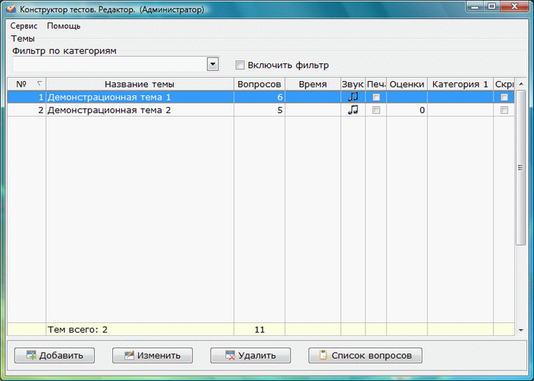


Рисунок 1.1 – Интерфейс для составления теста и заполнения базы вопросов

Тестируемый имеет возможность пропускать сложные вопросы. Преподаватель может приостанавливать тестирование, ограничивать его временными рамками, а также централизованно получать данные о прохождении тестов. Поддерживается разграничение прав доступа, резервное копирование и синхронизация базы. С сайта разработчиков программы можно загрузить множество вариантов готовых тестов на самую разнообразную тематику: ПДД, ЕГЭ, русский, французский и английский языки, экология, информатика, математика и др.

Программное средство «Универсальный тест»

Универсальный тест - приложение для создания тестов и проведения тестирования. Программа содержит в себе два модуля. Модуль под названием Tester позволяет проверять знания учащихся, студентов и персонала при помощи готовых тестов на самую разнообразную тематику - по русскому и английскому языку, информатике, ОБЖ, геометрии, физике, медицине, социологии, религиоведению, бухгалтерскому учету и т.д. База имеющихся тестов состоит из трех групп - тестов для школьников, тестов для ВУЗов, ПТУ и техникумов и тестов для проверки профессиональных знаний. Эти тесты составлялись пользователями предыдущих версий программы. Во время тестирования перед пользователем выводится серия вопросов с вариантами ответов (рисунок 1.2). Вопросы выбираются из базы в случайном порядке.

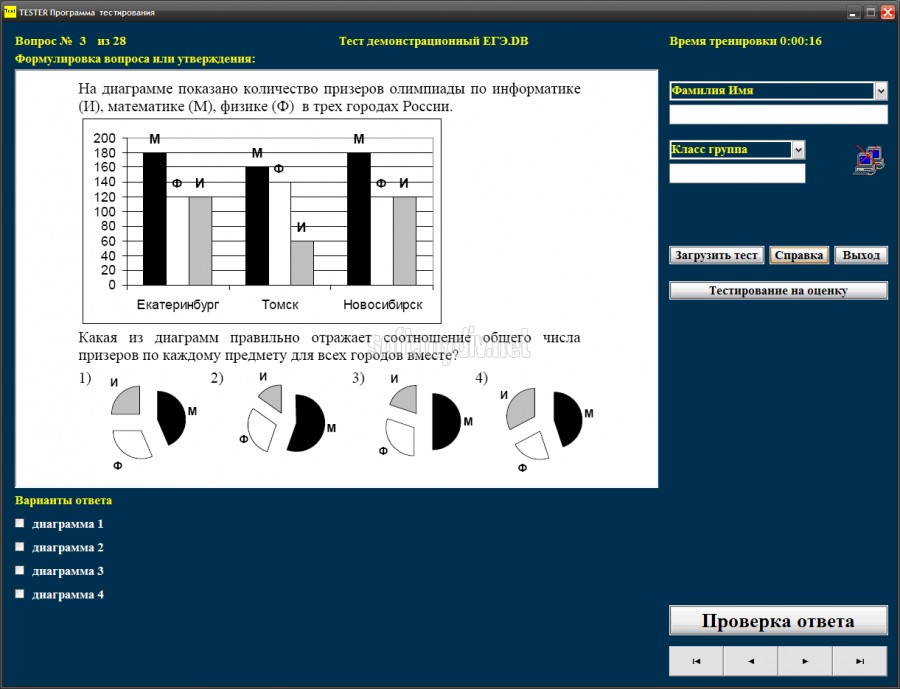


Рисунок 1.2 – Интерфейс для тестирования

При помощи модуля Test maker можно составлять и собственные тесты. В них можно использовать графику и тексты с форматированием. Доступ к модулю Test maker предоставляется либо на платной основе, либо в обмен на составленный пользователем собственный тест. Имеется возможность тестирования по локальной сети и сбора полученных результатов при помощи встроенного сервера.

Программное средство *«AnsTester»*

AnsTester - приложение для создания разнообразных тестов, основанных на системе начисления баллов. AnsTester изначально предназначался для проведения компьютерного тестирования в средней школе. Помимо тестирования, программа позволяет проводить различные опросы. В пакет входят несколько модулей - один предназначен для проведения тестирования (рисунок 1.3), второй - для просмотра результатов тестов и третий - для создания тестов.

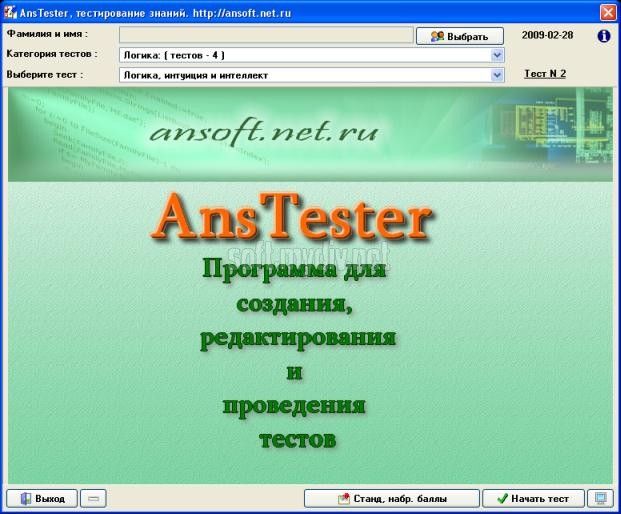


Рисунок 1.3 – Интерфейс для проведения тестирования