**Итоговый индивидуальный проект по теме:**

Разработка системы хранения формул, визуализации их взаимосвязей и выводов на примере формул школьного курса математики

**Выполнил:**

Ануваров Руслан Маратович,

ученик 10 «А» класса

**Руководитель проекта:**

Мухайлова Резеда Фаридовна,

педагог дополнительного образования

**ПАСПОРТ ПРОЕКТА**

|  |  |
| --- | --- |
| **Название** | Разработка системы хранения формул, визуализации их взаимосвязей и выводов на примере формул школьного курса математики |
| **Автор** | Ануваров Руслан Маратович, 10 «А» класс |
| **Учебный предмет** | Информатика, математика |
| **Руководитель** | Мухайлова Резеда Фаридовна, педагог дополнительного образования |
| **Тип** | Информационный |
| **Продукт** | Приложение для классификации формул и демонстрации их взаимосвязей, выводов |
| **Актуальность** | Механическое запоминание формул мешает понять их логику и связь с реальными задачами. Приложение показывает, как формулы выводятся друг из друга, помогая в изучении математики и смежных дисциплин. Также оно систематизирует формулы в удобный справочник, делая подготовку к экзаменам более эффективной и увлекательной. |
| **Цель** | разработать приложение для отображения формул и работы с ними |
| **Задачи** | 1. Проанализировать инструментарий и возможности языка Python для реализации проекта. 2. Познакомится с базами данных как способом хранения информации, спроектировать базу данных для хранения формул, их выводов и взаимосвязей. 3. Познакомиться с различными форматами представления формул и выбрать формат. 4. Разработка универсального приложения, доступного для всех пользователей. |
| **Результат** | Приложение «Формулатория» |
| **Этапы работы** | 1. Анализ проблемы и выбор инструментария. 2. Проектирование системы. (Схема базы данных, интерфейс) 3. Разработка программы. 4. Тестирование, отладка. |
| **Материально-технические** | ПК, программное обеспечение (Python 3.13.3, библиотеки Python) |
| **Информационные ресурсы** | Учебники, сайты с формулами и разделами математики |

Содержание

[Введение 4](#_Toc198086818)

[1. Теоретическая часть 6](#_Toc198086819)

[1.1. Анализ инструментария и возможностей языка Python для реализации проекта 6](#_Toc198086820)

[1.2. Способы хранения информации и формул 7](#_Toc198086821)

[1.3. Формат представления формул 9](#_Toc198086822)

[1.4. Разработка рабочей системы для пользователя 11](#_Toc198086823)

[2. Практическая часть 12](#_Toc198086824)

[2.1. Инструменты, которые использовались в написании программы 12](#_Toc198086825)

[2.2. Реализация проекта 13](#_Toc198086826)

[Заключение 14](#_Toc198086827)

[Список литературы 15](#_Toc198086828)

[Приложение 16](#_Toc198086829)

# Введение

Математику в школе мы изучаем с первого класса и до окончания школы. Школьный курс математики охватывает множество разделов, таких как: арифметика, алгебра, геометрия, тригонометрия, вероятность и статистика. Каждый из разделов содержит свои формулы, которые необходимо знать и понимать для правильного решения задач.

**Изучение математических формул** — это основа, которая помогает не просто запоминать правила, а учит видеть связи между разными разделами математики. Когда школьники работают с формулами, они развивают умение анализировать, выводить одни знания из других и применять их в реальных задачах. Однако часто обучение сводится к простому заучиванию, из-за чего даже те, кто помнит все формулы, могут не понимать, как они связаны между собой.

Этот пробел в понимании и стал основой для моего проекта. Так родилась идея приложения, которое не просто хранит формулы, но и показывает, как они выводятся друг из друг.

**Актуальность.** Изучение математики часто сводится к механическому запоминанию формул, из-за чего ученики теряют понимание их взаимосвязей и логики. Проект направлен на то, чтобы изменить этот подход. Вместо заучивания школьники смогут видеть, как формулы выводятся друг из друга, как они связаны между собой и с реальными задачами. Это углубляет знания по математике, а также в дальнейшем помогает в изучении смежных дисциплин: физики, химии и другие.

Приложение поможет сделать изучение математики более увлекательным, а классификация формул и их выводы могут служить полезным справочным материалом, и это также может помочь в подготовке к экзаменам и контрольным работам.

Понимая, что сами формулы имеют сложную математическую структуру, их сложно отображать в приложении и хранить в базе данных, я задался вопросом: «можно ли создать такое приложение?»

**Цель проекта**: разработать приложение для отображения формул и работы с ними.

**Задачи проекта:**

1. Проанализировать инструментарий и возможности языка Python для реализации проекта.
2. Познакомится с базами данных как способом хранения информации, спроектировать базу данных для хранения формул, их выводов и взаимосвязей.
3. Познакомиться с различными форматами представления формул и выбрать формат.
4. Разработка универсального приложения, доступного для всех пользователей.

# 1. Теоретическая часть

## 1.1. Анализ инструментария и возможностей языка Python для реализации проекта

Python — это высокоуровневый язык программирования, который предлагает множество возможностей для реализации различных проектов благодаря своей простоте, читаемости и множеству библиотек.  Вот некоторые ключевые возможности Python, которые могут быть полезны при реализации проекта:

1. имеет простой и понятный синтаксис, что облегчает обучение и ускоряет процесс разработки:
2. включает множество встроенных модулей и пакетов, которые позволяют выполнять различные задачи;
3. поддерживает объектно-ориентированное и функциональное программирование. Это позволяет разработчикам выбирать наиболее подходящий подход для их проекта;
4. работает на различных операционных системах, включая Windows, macOS и Linux, что делает его универсальным инструментом для разработки;
5. предоставляет удобные библиотеки для работы с базами данных (например: SQLite), что позволяет легко интегрироваться с системами хранения;
6. предлагает множество инструментов и библиотек для разработки графических пользовательских интерфейсов (GUI);
7. Python имеет активное сообщество, которое предлагает множество ресурсов, таких как документация, учебные пособия и форумы, что упрощает поиск решений и помощь в процессе разработки.

## 1.2. Базы данных как способ хранения информации

Базы данных (БД) принципиально отличаются от других способов хранения информации. В отличие от других способов хранения информации, таких как текстовые файлы, электронные таблицы или простые системы хранения, базы данных обеспечивают более высокий уровень организации, управления и безопасности данных.

Базы данных организуют данные в **структурированном** формате, что позволяет легко управлять, извлекать и обновлять информацию.

Базы данных используют системы управления базами данных (СУБД), которые предоставляют инструменты для создания, чтения, обновления и удаления данных (CRUD). Это позволяет пользователям эффективно управлять данными и выполнять сложные запросы.

В 1970 году Эдгар Кодд предложил реляционную модель, которая стала основой для современных систем управления базами данных. В этой модели данные хранятся в таблицах с определёнными полями и типами данных, что обеспечивает целостность и согласованность. Базы данных позволяют устанавливать связи между различными наборами данных (например, с помощью внешних ключей), что упрощает выполнение сложных запросов и анализ данных. Такая модель упростила управление данными и сделало их более доступными для пользователей.

В 1974 году была разработана структура языка SQL (Structured Query Language), который стал стандартом для работы с реляционными базами данных. SQL позволяет получать доступ к данным из приложений, используя удобный и стандартизованный синтаксис ( строка теста).

Базы данных могут быть спроектированы для обработки больших объемов данных и поддерживать высокую производительность при увеличении нагрузки.

Кроме того, базы данных предлагают встроенные механизмы для резервного копирования и восстановления данных, что помогает защитить информацию от потери.

Также базы данных обеспечивают многопользовательских доступ, это когда много пользователей нуждаются в доступе к одной и той же информации.

Рассмотрим несложную реляционную бесплатную СУБД SQLite - это удобный и эффективный способ управления данными в приложениях.

SQLite работает на различных операционных системах, таких как Windows, macOS, Linux, Android и iOS (Кроссплатформенность).

Эта СУБД очень проста в установке и использовании. Она не требует отдельного сервера или сложной настройки. Все данные хранятся в одном файле на диске, что делает их легко переносимыми и управляемыми.

SQLite использует язык SQL (Structured Query Language) для выполнения запросов к данным, что делает его мощным инструментом для работы с базами данных.

## 1.3. Формат представления формул

Хранение и отображение формул достаточно сложная задача:

* Формулы могут иметь сложную математическую структуру, включая различные операции, переменные, функции, степени и корни. Эта сложность затрудняет их представление в текстовом формате и требует специальных форматов для корректного отображения дополнительных ресурсов и технологий для визуализации.
* При вводе формул пользователи могут допускать ошибки, которые могут быть трудно выявить и исправить. Необходимы механизмы для проверки правильности ввода и отображения формул.
* Формулы могут зависеть друг от друга, и изменение одной формулы может повлиять на другие. Это требует сложных систем управления зависимостями и обновления данных.

Рассмотрим, как работают с формулами некоторые знакомые нам приложения:

1) Калькулятор (стандартные- Windows). Пользователь вводит формулу через интерфейс калькулятора – набор определенных кнопок. Во-первых, набор этих кнопок очень ограничен, т.е. не любую формулу можно ввести. Во-вторых, пользователь не видит набранную формулу в привычном формате, а видит только последний операнд и затем результат.

2) Microsoft Word. Приложение Word позволяет вставлять и редактировать формулы с помощью мощного встроенного редактора. Функции этого редактора почти сопоставимы с самим текстовым редактором. Формулы можно вставлять, редактировать, использовать специальные математические символы (дроби, корни, степени, интегралы и другие специальные знаки). Можно использовать шаблоны и структуры для быстрого создания сложных выражений, таких как матрицы, системы уравнений и многочлены. Пользователи могут изменять стиль формул, включая размер шрифта, цвет и другие параметры форматирования.

Возможностей много, но если же вы захотите набрать статью с формулами, то, не освоив этот редактор быстро это не получится.

3) Microsoft Excel. Это мощный инструмент для работы с данными, который позволяет пользователям создавать и использовать формулы для выполнения различных расчетов и анализа.

В Microsoft Excel формулы могут отображаться как в виде результатов вычислений, так и в виде самих формул, в зависимости от того, как настроен вывод и предлагает множество встроенных функций для выполнения сложных расчетов. Таким образом в привычном формате математическую формулу в ячейке мы не увидим.

4) Отдельно можно выделить систему верстки **LaTeX**, которая широко используется в научной среде, особенно в таких областях, как математика, физика, инженерия и компьютерные науки, благодаря своей способности обрабатывать сложные формулы и создавать профессиональные документы.

Она основана на языке разметки TeX, разработанном Дональдом Кнутом. LaTeX обеспечивает высокое качество печати, особенно для математических формул и научных публикаций и отлично подходит для написания сложных математических формул в привычном нам виде.

Например: для отображения знаменитой формулы Эйнштейна достаточно написать \[ E=mc^2 \]

Вывод: не существует единого стандарта для представления математических формул, что приводит к несовместимости между различными системами и программами. Это затрудняет хранение и обработку формул.

## 1.4. Разработка рабочей системы для пользователя

1. Определение основной идеи и цели приложения
2. Проектирование: пользовательский интерфейс, технологии и инструменты (базы данных, языки программирования, библиотеки)
3. Разработка, кодирование
4. Тестирование
5. Пробная эксплуатацию.

# 2. Практическая часть

## 2.1. Инструменты, которые использовались в написании программы

Для разработки системы хранения и визуализации математических формул был выбран язык Python, который обладает широкими возможностями для решения поставленных задач. В проекте использовались следующие технологии:

Библиотека Tkinter применялась для создания удобного графического интерфейса пользователя. С ее помощью были реализованы основные элементы управления - кнопки, поля ввода, выпадающие списки, таблицы и другие.

Для работы с базами данных использовалась встроенная библиотека SQLite3. Она обеспечила хранение данных о формулах, их разделах и взаимосвязях. SQLite идеально подошла для проекта благодаря простоте использования, отсутствию необходимости в настройке сервера и высокой производительности при работе с небольшими объемами данных.

Библиотека Manim была задействована для создания визуализаций, демонстрирующих вывод сложных формул. Это позволило наглядно показать, как одни формулы следуют из других, а также красиво вывести саму формулу.

При разработке активно применялись принципы объектно-ориентированного программирования (ООП). Это помогло создать четкую структуру проекта, разделить логику на независимые модули и обеспечить легкую расширяемость системы.

Таким образом, выбранный инструментарий на основе Python полностью соответствует задачам проекта, предоставляя все необходимые средства для создания удобного и функционального приложения.

## 2.2. Реализация проекта

Реализация проекта началась с проектирования базы данных, которая должна была хранить формулы, их категории и взаимосвязи. Были созданы таблицы: formula\_category для хранения разделов формул, formula\_output для хранения выводов формул и formula\_link для установления связей между формулами. Это позволило организовать данные в структурированном виде и обеспечить их целостность.

Для работы с базой данных был разработан класс APIFormula, который предоставляет методы для выполнения CRUD-операций (создание, чтение, обновление, удаление). Класс использует SQLiteDB для подключения к базе данных и выполнения запросов. Методы select\_all, create\_branch, update\_branch и delete\_branch обеспечивают взаимодействие с таблицей formula\_category, позволяя управлять разделами формул. И также

Графический интерфейс был реализован с помощью библиотеки Tkinter. Были созданы три вкладки: BranchTab для управления разделами формул, FormulaTab для отображения формул и их свойств, и OutputTab для вывода результатов. Вкладка BranchTab включает Treeview для отображения списка разделов и элементы управления для добавления, изменения и удаления разделов. FormulaTab содержит Combobox для выбора раздела и Treeview для отображения формул выбранного раздела, вывод формулы, а также название, краткое описание.

Для удобства пользователя интерфейс был стилизован: шрифты, размеры элементов и их расположение были настроены для улучшения визуального восприятия. Обработка ошибок и проверка ввода были реализованы для предотвращения некорректных действий пользователя, таких как попытка создания дублирующего раздела или удаления без подтверждения.

Тестирование приложения проводилось на различных сценариях работы с разделами и формулами, что позволило выявить и исправить возможные ошибки. В результате было создано работоспособное приложение, которое соответствует поставленным задачам и предоставляет удобный инструмент для систематизации и изучения математических формул.

# Заключение

В результате работы над проектом было разработано приложение «Формулатория», которое позволяет систематизировать математические формулы, визуализировать их взаимосвязи и демонстрировать выводы. Использование языка Python, библиотеки Tkinter для интерфейса, SQLite для хранения данных и Manim для визуализации обеспечило создание функционального и наглядного инструмента. Приложение помогает не только запомнить формулы, но и понять их логику, делая изучение математики более осмысленным и доступным.

# Список литературы

1. [Как красиво писать формулы c LaTeX? / Хабр](https://habr.com/ru/articles/723564/)
2. [Внедряем формулы как в Эксель / Хабр](https://habr.com/ru/articles/889180/)
3. [С помощью Python создаём математические анимации, как на канале 3Blue1Brown / Хабр](https://habr.com/ru/companies/skillfactory/articles/556944/)
4. [Введение в базы данных / Хабр](https://habr.com/ru/articles/686816/)

# Приложение

**Файл api\_db.py:**

class APIFormula:

# это класс, разработанный для взаимодействия с базой данных SQLite в рамках проекта «Формулатория». Он служит промежуточным слоем (API) между графическим интерфейсом приложения (Tkinter) и базой данных.

**Файл connection.py:**

class SQLiteDB:

# Подключение к базе данных

**Файл main\_window.py:**

class TkinterWindow(tk.Tk):

# Класс для реализации интерфейса пользователя

TkinterWindow("Формулатория", (False, False), (1200, 1000))

**Файл tab\_frame.py:**

class BranchTab(ttk.Frame):

# класс для реализации вкладки Раздел

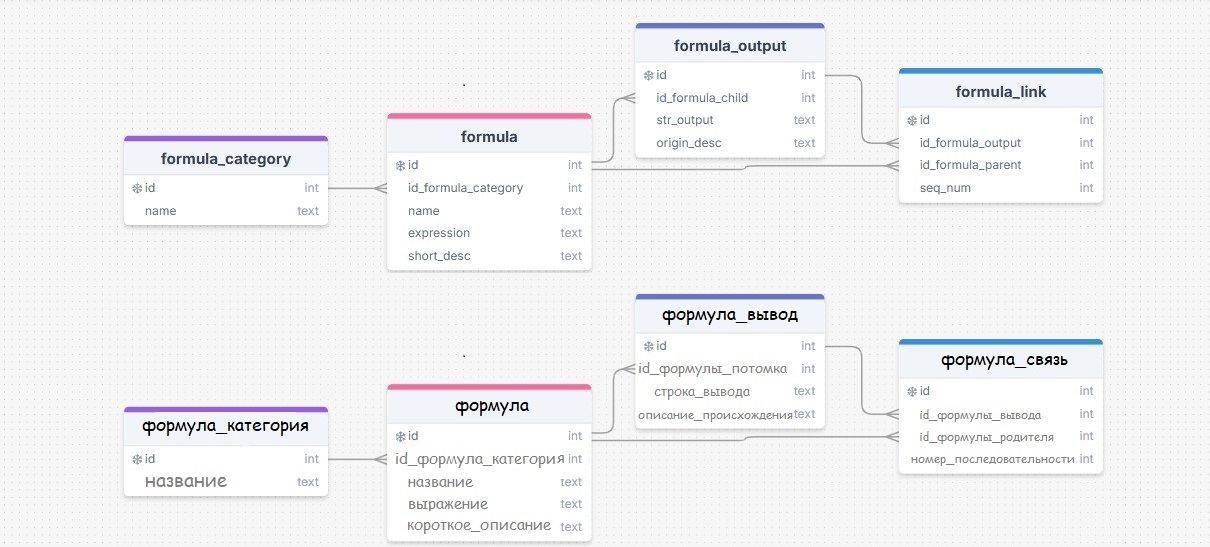
class FormulaTab(ttk.Frame):

# класс для реализации вкладки Формула

class OutputTab(ttk.Frame):

# класс для реализации вкладки Вывод

Структура базы данных:



Где можно найти приложение: [Lamwarld/Formulatory](https://github.com/Lamwarld/Formulatory) (github)