**Практическое задание**

*ОКЭИ 09.02.07 4025 23 П3*

Тема: «*Разработка динамических библиотек DLL*»

Выполнил: Яценко Алексей Александрович

**Содержание**

[Введение 3](#_Toc9402)

[1 Ход работы 4](#_Toc7603)

[1.1 Создание DLL библиотеки 4](#_Toc31437)

[1.2 Базовые арифметические операции 4](#_Toc5026)

[1.3 Дополнительные математические функции 5](#_Toc24618)

[1.4 Архитектурные особенности 6](#_Toc32557)

[2 Демонстрация работы проекта 7](#_Toc30527)

# Введение

Данная практическая работа посвящена разработке динамической библиотеки (DLL) на языке C#, инкапсулирующей функционал консольного калькулятора. Основная цель проекта — освоение принципов создания и использования динамически подключаемых библиотек, а также закрепление навыков модульного программирования.

Библиотека будет предоставлять основные арифметические операции (сложение, вычитание, умножение, деление), а также более сложные математические функции: возведение в степень, вычисление факториала и решение квадратных уравнений. Реализация будет выполнена в виде консольного приложения, которое использует разработанную DLL, что наглядно демонстрирует преимущества разделения кода на логические модули и их повторного использования.

# 1 Ход работы

# 1.1 Создание DLL библиотеки

Данный код представляет собой динамическую библиотеку (DLL), содержащую класс Calculator с набором статических методов для выполнения математических операций. Библиотека включает как базовые арифметические функции, так и сложные математические вычисления.

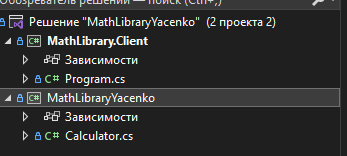


Рисунок 1 - Общая структура проекта в Solution Explorer

# 1.2 Базовые арифметические операции

Библиотека реализует стандартные арифметические операции:

* Сложение (Add) - возвращает сумму двух чисел;
* Вычитание (Substract) - возвращает разность между вторым и первым аргументом;
* Умножение (Multiply) - возвращает произведение двух чисел;
* Деление (Divide) - включает проверку деления на ноль.

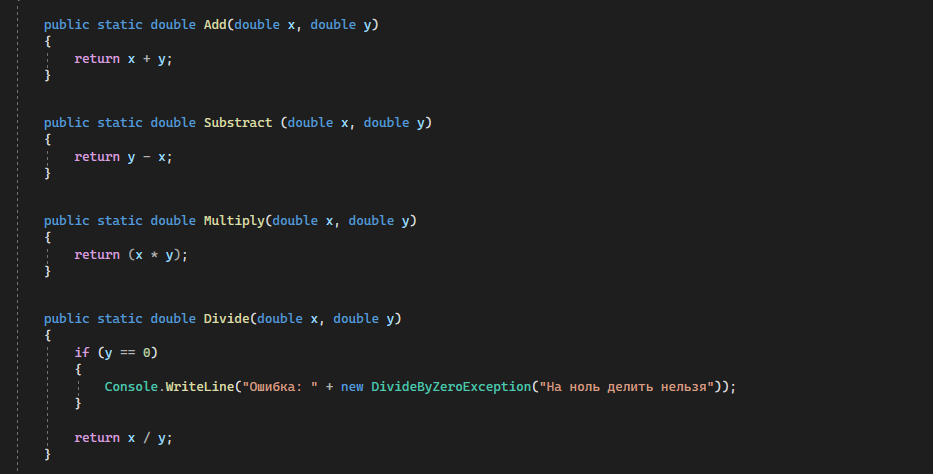


Рисунок 2 - Базовые операции

# 1.3 Дополнительные математические функции

Метод Power использует встроенную функцию Math.Pow для вычисления степени числа. Метод Factorial вычисляет факториал целого числа с использованием цикла.

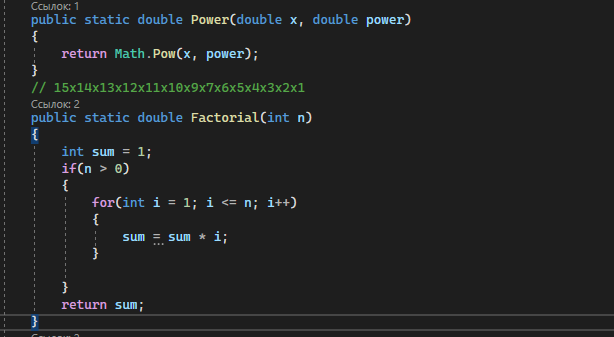


Рисунок 3 - Функции Power и Factorial.

Метод SolveQuadratic решает квадратные уравнения вида ax² + bx + c = 0:

* Вычисляет дискриминант;
* Обрабатывает особые случаи (когда a = 0);
* Возвращает корни через выходные параметры;
* Использует nullable типы для обозначения отсутствия корней.

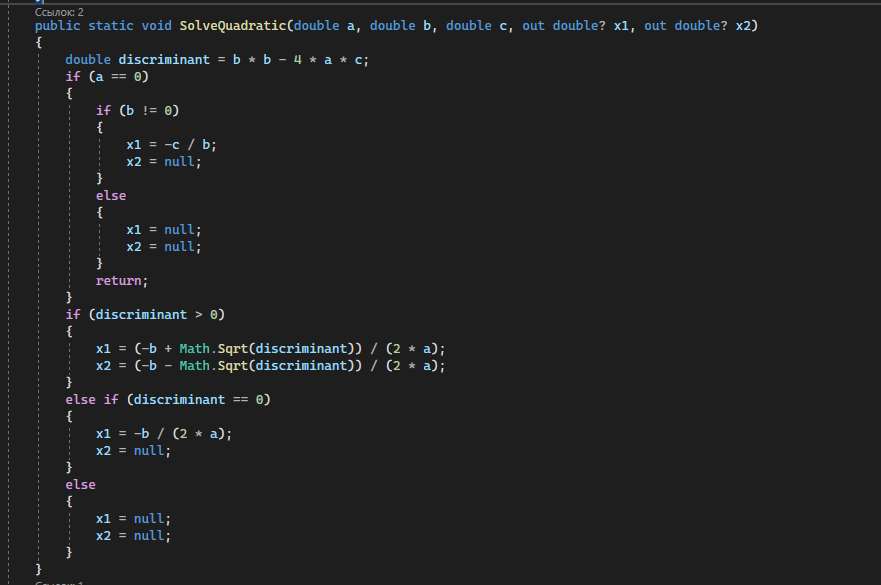


Рисунок 4 - Функция SolveQuadratic

Метод IsPrime определяет, является ли число простым:

* Использует оптимизации: проверка делимости на 2 и 3;
* Проверяет делители только до квадратного корня числа;
* Возвращает логическое значение.

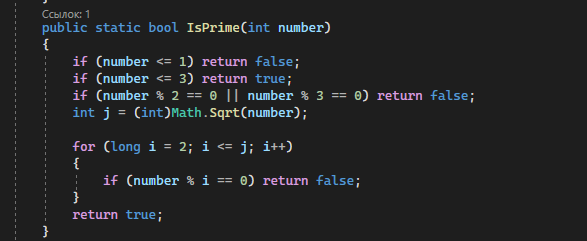


Рисунок 5 - Функция проверки простых чисел.

# 1.4 Архитектурные особенности

У проекта есть следующие архитектурные особенности:

* Все методы объявлены как static, что позволяет вызывать их без создания экземпляра класса;
* Использование nullable типов для возврата optional значений;
* Обработка исключительных ситуаций (деление на ноль);

# 2 Демонстрация работы проекта

На рисунках 6,7 представлен код Program.cs, который поочередно вызывает все методы и проверяет их работоспособность с выводом текстовых сообщений.

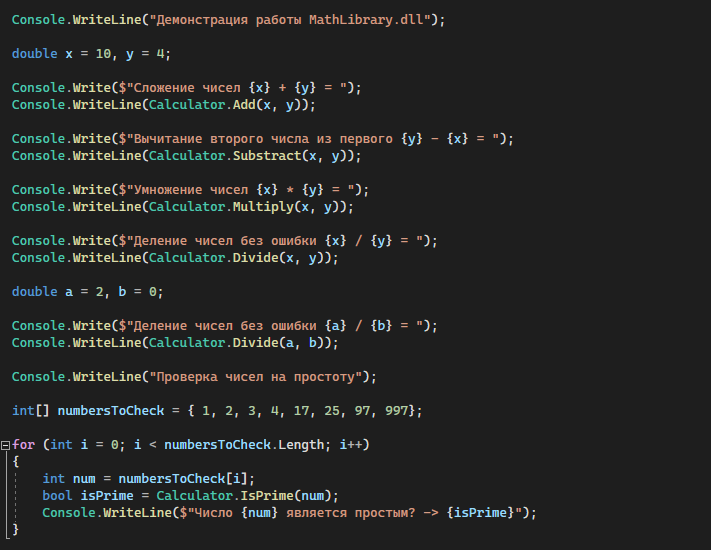


Рисунок 6 - Первая часть методов

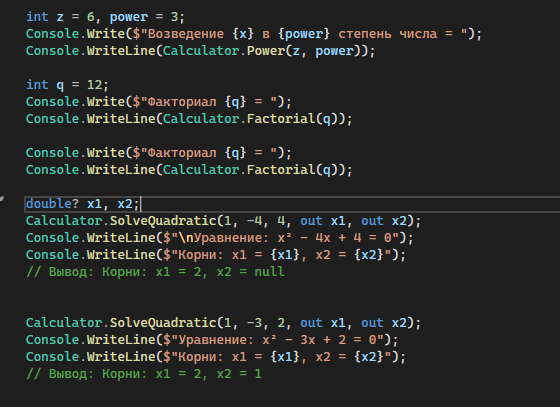


Рисунок 7 – Вторая часть методов

На рисунке 7 представлена демонстрация работы запущенного проекта со всеми пояснительными сообщениями.

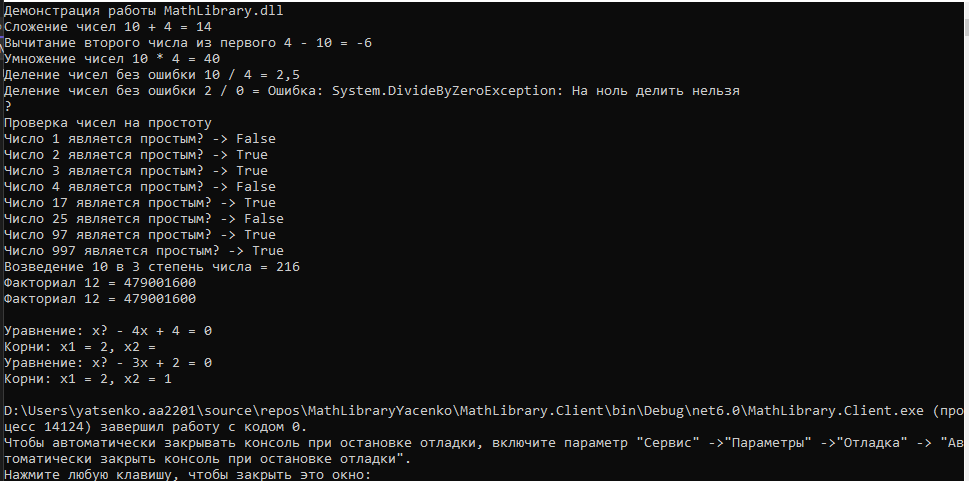


Рисунок 8 - Демонстрация работы проекта