­­­Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное   
образовательное учреждение высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ

Сравнение рекурсивных, циклических алгоритмов

и обработка чисел на нечётность

Пояснительная записка к лабораторной работе по учебной дисциплине «Основы программирования: Алгоритмические языки и программирование»

по специальности 09.03.01 Информатика и вычислительная техника  
ПТИ.ЛР 4093 004.021ПЗ

|  |  |
| --- | --- |
|  | Руководитель  / И. Ю. Кулаков  «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2025 г. |
|  | Студент группы 4093  / М. М. Нематжанов  «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2025 г. |

Содержание

[Введение 2](#_Toc198653036)

[1 Постановка задачи 4](#_Toc198653037)

[1.1 Наименование задачи 4](#_Toc198653038)

[1.2 Словесное описание 4](#_Toc198653039)

[1.3 Внешние спецификации данных 5](#_Toc198653040)

[1.4 Внешние спецификации функции данных 5](#_Toc198653041)

[1.5 Математическая формулировка задачи 5](#_Toc198653042)

[1.6 Специфика интерфейса 6](#_Toc198653043)

[1.7 Внешние данные тестирования 7](#_Toc198653044)

[1.8 Пример работы программы 7](#_Toc198653045)

[2 Проектирование задачи 9](#_Toc198653046)

[2.1 Наименование программы 9](#_Toc198653047)

[2.2 Уточнение словесного описания 9](#_Toc198653048)

[2.3 Выбор метода решения поставленной задачи 10](#_Toc198653049)

[2.4 Уточнение глобальных данных программы и пользовательские типы 10](#_Toc198653050)

[2.5 Декомпозиция функции 10](#_Toc198653051)

[2.6 Алгоритмизация 11](#_Toc198653052)

[Заключение 15](#_Toc198653053)

[Приложение А 16](#_Toc198653054)

[Приложение Б 23](#_Toc198653055)

[Приложение В 27](#_Toc198653056)

# Введение

Задачей данной лабораторной работы является разработка программного приложения на C++/CLI с использованием Windows Forms для анализа и сравнения эффективности рекурсивных и циклических подходов к вычислению заданных математических функций (Формула 2 и Формула 9). Дополнительно, программа реализует рекурсивный алгоритм для решения задачи №2 из предложенного списка – вычисление суммы цифр натурального числа.

Программа предоставляет пользователю графический интерфейс для ввода исходного числа n, выбора интересующей задачи и метода вычисления (рекурсия или цикл для F2/F9). Для сравнительного анализа измеряется время выполнения каждого алгоритма с использованием библиотеки <chrono> и подсчитывается количество рекурсивных вызовов для рекурсивных функций.

Актуальность работы заключается в демонстрации различий в производительности и потреблении ресурсов между рекурсивными и итерационными алгоритмами на конкретных примерах, что важно для понимания принципов оптимизации программ и выбора подходящего метода решения задач в программировании.

Основные задачи, решенные в ходе работы

- анализ предоставленных математических формул (F2, F9) и задачи на вычисление суммы цифр;

- разработка рекурсивных и циклических алгоритмов для функций F2 и F9;

- реализация рекурсивного алгоритма для вычисления суммы цифр числа;

- интеграция механизма замера времени выполнения алгоритмов;

- реализация счетчика количества вызовов для рекурсивных функций;

- создание графического пользовательского интерфейса для взаимодействия с программой;

- реализация обработки возможных ошибок ввода пользователя;

- подготовка документации и базового набора тестов для проверки корректности работы программы.

1 Постановка задачи

1.1 Наименование задачи

Разработка программного приложения для сравнения производительно-сти рекурсивного и циклического методов вычисления математических функций F2 и F9, а также для рекурсивного вычисления суммы цифр натурального числа.

1.2 Словесное описание

Программа позволяет пользователю ввести натуральное число n. Для этого числа программа может выполнить следующие действия:

Вычислить значения функций F2 и F9 двумя способами: рекурсивно и циклически. При этом измеряется время выполнения каждого способа и количество вызовов для рекурсивных функций.

Вычислить сумму цифр введенного числа n с использованием рекурсивного алгоритма.

Программа предоставляет результаты вычислений, время работы и количество вызовов в удобном для пользователя виде через графический интерфейс.

1.2.1 Ввод

- целое положительное число n, вводимое пользователем в текстовое поле интерфейса.

1.2.2 Вывод

Для Задачи 1:

- результат вычисления функции F2 (рекурсивно/циклически);

- результат вычисления функции F9 (рекурсивно/циклически);

- время выполнения для каждого метода в микросекундах;

- количество рекурсивных вызовов для рекурсивных методов.

Для Задачи 2:

- сумма цифр введенного числа;

- сообщения об ошибках при некорректном вводе;

- отображение математических формул по запросу пользователя.

1.3 Внешние спецификации данных

Таблица объектов программы представлена в приложении A.1.

1.4 Внешние спецификации функции данных

Внешние спецификации функции данных включают:

- функция рекурсивного вычисления F9;

- функция циклического вычисления F9;

- функция рекурсивного вычисления F2;

- функция циклического вычисления F2;

- функция рекурсивного вычисления суммы цифр числа;

- функция измерения времени выполнения участков кода;

- функция подсчета количества рекурсивных вызовов;

- функции обработки событий графического интерфейса (нажатия кнопок, ввод текста);

- функции отображения информации пользователю (обновление текстовых меток).

1.5 Математическая формулировка задачи

Функция 9 (Вариант 9):

- F(n) = 1, при n = 1;

- F(n) = n + 2 \* F(n - 1), если n чётно;

- F(n) = 1 + 3 \* F(n - 2), если n > 1 и n нечётно;

Функция 2 (Вариант 2)

- F(n) = n, при n < 3;

- F(n) = 3 \* (n - 1) + F(n - 1) + 5, если n >= 3 и n чётно;

- F(n) = 3 \* (n + 1) + F(n - 2) - 2, если n >= 3 и n нечётно.

1.6 Специфика интерфейса

Программа имеет графический интерфейс пользователя, реализованный с помощью C++/CLI Windows Forms. Интерфейс состоит из двух основных форм:

А. Главная форма (MyForm):

- начальный экран с кнопками выбора задания ("Задание №1", "Задание №2");

- панель для Задания №1, содержащая:

- поле для ввода целого числа n;

- кнопка для запуска рекурсивного расчета обеих функций;

- кнопка для запуска циклического расчета обеих функций;

- кнопка для отображения математических формул Функции 9 и Функции 2;

- кнопка для перехода к Заданию №2;

- кнопка для возврата на главный экран;

- метка для отображения результатов расчетов, времени выполнения и количества вызовов;

- звуковое сопровождение нажатия кнопок.

Б. Форма Задания №2:

- кнопка для запуска расчета суммы цифр;

- кнопка для отображения формулы/описания расчета суммы цифр;

- кнопка для закрытия формы и возврата;

- метка для отображения результата расчета суммы цифр;

- звуковое сопровождение нажатия кнопок.

1.7 Внешние данные тестирования

Программа включает обработку следующих потенциальных ошибок:

A. Некорректный ввод: Если пользователь вводит нецелочисленное значение в поля ввода программа перехватывает исключение и выводит сообщение об ошибке ("Ошибка! Введите целое число.").

Б. Выход за пределы диапазона (Рекурсия): В рекурсивных функциях для Функции 9 и Функции 2 существует ограничение на максимальное значение n (в коде n <= 20000), чтобы избежать переполнения стека. При вводе большего значения выводится ошибка.

Выход за пределы диапазона (Цикл): В циклических функциях также установлено ограничение (в коде 0 <= n <= 10,000,000), вероятно, для предотвращения слишком длительного выполнения или потенциального переполнения типа int при расчетах. При выходе за диапазон выводится ошибка.

В. Отрицательные числа: Циклические и рекурсивные функции для F9/F2 ожидают n >= 1. В коде F9\_loop(0) возвращает 0, но проверка диапазона в интерфейсе не позволяет ввести 0 для цикла. Сумма цифр корректно обрабатывает отрицательные числа (суммируя цифры по модулю).

1.8 Пример работы программы

Сценарий 1: Расчет Функции 9 и 2 (Рекурсия):

А. запустить программу;

Б. нажать кнопку "Задание №1";

В. в поле ввода ввести число 5;

Г. нажать кнопку "Рекурсия";

Д. ожидаемый результат: В метке отобразится;

- результат Функции 9 для n=5;

- количество вызовов рекурсии для Функции 9;

- время выполнения расчета Функции 9;

- результат Функции 2 для n=5;

- количество вызовов рекурсии для Функции 2;

- время выполнения расчета Функции 2.

Сценарий 2: Расчет Суммы цифр:

А. запустить программу;

Б. нажать кнопку "Задание №2". Откроется вторая форма;

В. в поле ввода (textBox11) ввести число 2024;

Г. нажать кнопку "Расчитать";

Д. ожидаемый результат: В метке labelresalt отобразится текст "Сумма цифр: 8".

2 Проектирование задачи

2.1 Наименование программы

"Программа для анализа эффективности рекурсивных и циклических алгоритмов вычисления математических функций и расчета суммы цифр числа".

2.2 Уточнение словесного описания

Данное программное приложение предназначено для сравнительного анализа производительности двух подходов к решению вычислительных задач: рекурсивного и циклического. В качестве примеров задач используются две математические функции (Функция 9 и Функция 2), заданные рекуррентными соотношениями. Пользователь может ввести параметр n, и программа вычислит значение функций обоими методами, предоставив данные о времени выполнения и, для рекурсии, о количестве вызовов.

Дополнительно реализована вторая задача: вычисление суммы цифр заданного целого числа с использованием исключительно рекурсивного подхода.

Программа реализована на платформе .NET с использованием языка C++/CLI и библиотеки Windows Forms для создания графического интерфейса. Для измерения времени используется стандартная библиотека C++ <chrono>. Обработка ошибок ввода и контроль допустимых диапазонов входных данных обеспечивают стабильность работы приложения.

Приложение служит практическим инструментом для студентов, изучающих алгоритмы и структуры данных, позволяя наглядно увидеть разницу в эффективности рекурсии и итерации, понять концепцию глубины рекурсии и оценить временные затраты различных подходов.

2.3 Выбор метода решения поставленной задачи

Вычисление Функции 9 и Функции 2:

- рекурсивный метод: Реализован путем прямого перевода рекуррентных соотношений в код функций Для подсчета вызовов используется целочисленный параметр, передаваемый по ссылке. Этот метод демонстрирует классическое применение рекурсии, но может быть неэффективен по времени и потреблению памяти для больших n.

- циклический метод: Реализован с использованием динамического программирования. Создается массив dp для хранения уже вычисленных значений F(i). Расчет ведется последовательно от базовых случаев до требуемого n. Этот метод обычно более эффективен по времени и памяти для данных задач по сравнению с наивной рекурсией.

Вычисление суммы цифр:

- рекурсивный метод: Реализован функцией, которая выделяет последнюю цифру числа (n % 10) и рекурсивно вызывает себя для оставшейся части числа (n / 10). Базовый случай - n = 0. Этот метод элегантен и соответствует требованию задания использовать рекурсию.

- графический интерфейс: Выбрана платформа Windows Forms в C++/CLI для создания интерактивного оконного приложения, удобного для пользователя.

2.4 Уточнение глобальных данных программы и пользовательские типы

Таблица данных программы представлена в приложении А.3

2.5 Декомпозиция функции

Таблица функций программы представлена в приложении А.8

2.6 Алгоритмизация

Алгоритмы функций:

А. Formulas::F9\_recursion:

- увеличить counter на 1;

- если n == 1, вернуть 1;

- если n чётно (n % 2 == 0), вернуть n + 2 \* F9\_recursion(counter, n - 1);

- иначе (n нечётно), вернуть 1 + 3 \* F9\_recursion(counter, n - 2).

Б. Formulas::F9\_loop:

- если n == 1, вернуть 1;

- создать динамический массив dp размера n + 1;

- установить dp[1] = 1;

- для i от 2 до n;

- если i чётно (i % 2 == 0), установить dp[i] = i + 2 \* dp[i - 1];

- иначе (i нечётно), установить dp[i] = 1 + 3 \* dp[i - 2];

- сохранить dp[n] в переменную res;

- освободить память, выделенную под dp;

- вернуть res.

В. Formulas::F2\_recursion:

- увеличить counter на 1;

- если n < 3, вернуть n;

- если n чётно (n % 2 == 0), вернуть 3 \* (n - 1) + F2\_recursion(counter, n - 1) + 5;

- иначе (n нечётно), вернуть 3 \* (n + 1) + F2\_recursion(counter, n - 2) – 2;

Г. Formulas::F2\_loop:

- если n < 3, вернуть n;

- создать динамический массив dp размера n + 1;

- установить dp[1] = 1, dp[2] = 2;

- для i от 3 до n:

- если i чётно (i % 2 == 0), установить dp[i] = 3 \* (i - 1) + dp[i - 1] + 5;

- иначе (i нечётно), установить dp[i] = 3 \* (i + 1) + dp[i - 2] – 2;

- сохранить dp[n] в переменную res;

- освободить память, выделенную под dp;

- вернуть res.

Д. Formulas::SumDigits:

- если n == 0, вернуть 0;

- вернуть (n % 10) + SumDigits(n / 10). (Примечание: в коде C++ может потребоваться обработка отрицательных чисел, например, abs(n % 10) или обработка знака отдельно);

Алгоритм работы программы

А. Запуск:

- инициализация главного окна MyForm;

- отображение кнопок выбора задания ("Задание №1", "Задание №2") и метки заголовка;

Б. Нажатие "Задание №1":

- воспроизвести звук;

- сделать панель panelTask1 видимой;

- скрыть кнопки выбора задания (или оставить их).

В. Нажатие "Задание №2":

- воспроизвести звук;

- создать и отобразить второе окно dz2.

Г. Работа в panelTask1 (Задание №1):

Нажатие "Рекурсия":

- воспроизвести звук;

- считать текст из текстового поле;

- преобразовать текст в целое число;

- проверить n на допустимость (n <= 20000). Если неверно, вывести ошибку и выйти;

- инициализировать счетчики counterV9 = 0, counterV2 = 0;

- замерить время выполнения, сохранить результат timeV9;

- замерить время выполнения, сохранить результат timeV2;

- сформировать строку с результатами;

- вывести строку в labelResult.

Нажатие "Цикл"

- воспроизвести звук;

- считать текст из текстового поле;

- преобразовать текст в целое число n;

- проверить n на допустимость (<= 10 млн). Если неверно, вывести ошибку и выйти;

- замерить время выполнения, сохранить результат timeV9;

- замерить время выполнения, сохранить результат timeV2.

- сформировать строку с результатами.

- вывести строку в labelResult.

Нажатие "Формула":

- воспроизвести звук.

- вывести текст формул F9 и F2 в labelResult.

Нажатие "Домой" :

- воспроизвести звук.

- скрыть panelTask1.

- показать кнопки выбора задания (если были скрыты).

Работа в окне dz2 (Задание №2):

нажатие "Рассчитать" :

- воспроизвести звук.

- считать текст из текстового поле;

- преобразовать текст в целое число n;

- вызвать int sum;

- вывести "Сумма цифр: " + sum в labelresalt.

Нажатие "Формула":

- воспроизвести звук;

- вывести описание/пример расчета суммы цифр в labelresalt.

Нажатие "Домой":

- воспроизвести звук;

- закрыть окно dz2.

Заключение

В ходе выполнения лабораторной работы было разработано программное приложение на C++/CLI с использованием Windows Forms, предназначенное для сравнения производительности рекурсивных и циклических алгоритмов на примере двух математических функций, а также для вычисления суммы цифр числа с помощью рекурсии.

Реализованы алгоритмы для рекурсивного и циклического (итерационного, с применением динамического программирования) вычисления Функции 9 и Функции 2. Внедрены механизмы замера времени выполнения с использованием <chrono> и подсчета количества рекурсивных вызовов. Отдельно реализована рекурсивная функция для расчета суммы цифр числа.

Разработан графический пользовательский интерфейс, позволяющий пользователю вводить данные, выбирать задачи и методы расчета, просматривать формулы и получать результаты вычислений, включая время выполнения и количество вызовов. Реализована базовая обработка ошибок ввода.

Проведенное (или планируемое) тестирование подтверждает корректность работы реализованных алгоритмов и функциональность приложения в соответствии с поставленными задачами. Разработанное приложение может служить наглядным пособием для изучения различий между рекурсией и итерацией.

Приложение А

(Обязательное)

Таблица А.1 – Основные объекты программы, запрашиваемые у пользователя

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Объект программы | Имя этого объекта в программе | Тип данных | Диапазоны представления | Простая/  структура | Вход/  выход/  Константа |
| MyForm.h | txtInputN | TextBox | Текстовое поле для ввода n | Структура | Ввод |
| MyForm.h | n (из txtInputN) | int | 0 <= n <= 10,000,000 (цикл), 1 <= n <= 20,000 (рекурсия) | Простая | Ввод |
| MyForm.h | labelResult | Label | Поле для вывода результатов Задачи 1 | Структура | Выход |
| MyForm.h | resV9 (рекурсия) | int / long long | Зависит от n, возможны большие значения | Простая | Выход |
| MyForm.h | resV2 (рекурсия) | int / long long | Зависит от n, возможны большие значения | Простая | Выход |
| MyForm.h | counterV9 | int | counterV9 >= 0 | Простая | Выход |
| MyForm.h | counterV2 | int | counterV2 >= 0 | Простая | Выход |
| MyForm.h | timeV9 (рекурсия) | µs | timeV9 >= 0 | Простая | Выход |
| MyForm.h | timeV2 (рекурсия) | µs | timeV2 >= 0 | Простая | Выход |
| MyForm.h | resV9\_loop | int / long long | Зависит от n, возможны большие значения | Простая | Выход |
| MyForm.h | resV2\_loop | int / long long | Зависит от n, возможны большие значения | Простая | Выход |
| MyForm.h | timeV9 (цикл) | µs | timeV9 >= 0 | Простая | Выход |
| MyForm.h | timeV2 (цикл) | µs | timeV2 >= 0 | Простая | Выход |
| dz2.h | textBox11 | TextBox | Текстовое поле для ввода числа | Структура | Ввод |
| dz2.h | labelresalt | Label | Поле для вывода результата Задачи 2 | Структура | Выход |
| dz2.h | sum | int | sum >= 0 | Простая | Выход |

Таблица А.2 — Тестирование рекурсивных алгоритмов

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case ID/Priority | TC001 | | 1 |
| IDEA: Проверка работоспособности программы | | | |
| Revision History | | | |
| Created on 20/05/2025 | | Новый тест кейс | |
| ACTION | | | |
| 1) Открыть программу LAB1.exe;  2) Нажать кнопку “Задание 1”;  2) Ввести в поле “Ввести число для функции” число “7”;  3) Нажать кнопку “Формулы” и ознакомиться с данной формулой для обоих функций;  4) Нажать кнопку “Рекурсивный” или “Циклический”;  5) Проверить, что программа корректно выводит результат, время выполнения и количество вызовов под полем ввода числа;  10) Нажать кнопку “Задание 2”;  11) В поле ввести чисел “5 8 9 5”;  12) Проверить, что программа корректно суммирует числа в поле “Результата”;  13) Проверить обработку ошибок для обоих заданий:  - Ввести символы и убедиться, что программа выводит сообщение об ошибке;  - Проверить пустую строку и убедиться, что программа выводит сообщение об ошибке;  - Ввести слишком большое число и убедиться, что программа выводит сообщение об ошибке. | | | |

Таблица А.3 Уточнение глобальных данных программы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Объект программы | Имя этого объекта в программе | Характеристики | | | Использование в программе |
| Тип данных | Диапазоны представления | Простая/  Структура/  Константа |
| MyForm.h | resV9 (рекурсия) | long long | -9\*10^20<res1<9\*10^20 | Простая | Выход |
| MyForm.h | resV2 (рекурсия) | long long | -9\*10^20<res1<9\*10^20 | Простая | Выход |
| MyForm.h | resV9\_loop | long long | -9\*10^20<res1<9\*10^20 | Простая | Выход |
| MyForm.h | resV2\_loop | long long | -9\*10^20<res1<9\*10^20 | Простая | Выход |
| MyForm.h | counterV2 | int | >= 0 | Простая | Выход |
| MyForm.h | timeV9 (рекурсия) | long long (мкс) | >= 0 | Простая | Выход |
| MyForm.h | timeV2 (рекурсия) | long long (мкс) | >= 0 | Простая | Выход |
| MyForm.h, | timeV9 (цикл) | long long (мкс) | >= 0 | Простая | Выход |
| MyForm.h, | timeV2 (цикл) | long long (мкс) | >= 0 | Простая | Выход |
| dz2.h | sum | int | sum >= 0 | Простая | Выход |

Таблица A.4 – Unit-тестирование рекурсивной функции 2 математической формулы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Unit-Test ID | Вводные данные | | Результат | |
| Базовый случай | | | | |
| UT1 | n = 1 | | F1\_recursive = 1 | |
| UT2 | n = 2 | | F1\_recursive = 2 | |
| Проверка на чётность | | | | |
| UT3 | n = 4 | | F1\_recursive = 25 | |
| Проверка на нечётность | | | | |
| UT4 | n = 3 | | F1\_recursive = 11 | |
| Проверка на ноль | | | | |
| UT5 | | n = 0 | | F1\_recursive = 0 |

Таблица A.5 – Unit-тестирование циклической функции 2 математической формулы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Unit-Test ID | Вводные данные | | Результат | |
| Базовый случай | | | | |
| UT1 | n = 1 | | F1\_iterative= 1 | |
| UT2 | n = 2 | | F1\_iterative = 2 | |
| Проверка на чётность | | | | |
| UT3 | n = 4 | | F1\_iterative = 25 | |
| Проверка на нечётность | | | | |
| UT4 | n = 3 | | F1\_iterative = 11 | |
| Проверка на ноль | | | | |
| UT5 | | n = 0 | | F1\_iterative = 0 |

Таблица A.6 – Unit-тестирование рекурсивной функции 9 математической формулы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Unit-Test ID | Вводные данные | | Результат | |
| Базовый случай | | | | |
| UT1 | n = 1 | | F2\_recursive = 1 | |
| UT2 | n = 2 | | F2\_recursive = 4 | |
| Проверка на чётность | | | | |
| UT3 | n = 4 | | F2\_recursive = 12 | |
| Проверка на нечётность | | | | |
| UT4 | n = 3 | | F2\_recursive = 4 | |
| Проверка на ноль | | | | |
| UT5 | | n = 0 | | F2\_recursive = 0 |

Таблица A.7 – Unit-тестирование циклической функции 2 математической формулы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Unit-Test ID | Вводные данные | | Результат | |
| Базовый случай | | | | |
| UT1 | n = 1 | | F2\_iterative = 1 | |
| UT2 | n = 2 | | F2\_iterative = 4 | |
| Проверка на чётность | | | | |
| UT3 | n = 4 | | F2\_iterative = 12 | |
| Проверка на нечётность | | | | |
| UT4 | n = 3 | | F2\_iterative = 4 | |
| Проверка на ноль | | | | |
| UT5 | | n = 0 | | F2\_iterative = 0 |

Таблица А.8 – Функции программы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Имя | Параметры | | Процедура/ функция | Тестируется/не тестируется |
| Ввод: тип | Вывод: тип |
| Рекурсивная F9 | Formulas::F9\_recursion | n: int | int | Функция | Тестируется |
| Циклическая F9 | Formulas::F9\_loop | n: int | int | Функция | Тестируется |
| Циклическая F2 | Formulas::F2\_loop | n: int | int | Функция | Тестируется |
| Рекурсивная F2 | Formulas::F2\_recursion | n: int | int | Функция | Тестируется |
| Сумма цифр (рекурсия) | Formulas::SumDigits | n: int | int | Функция | Функция |

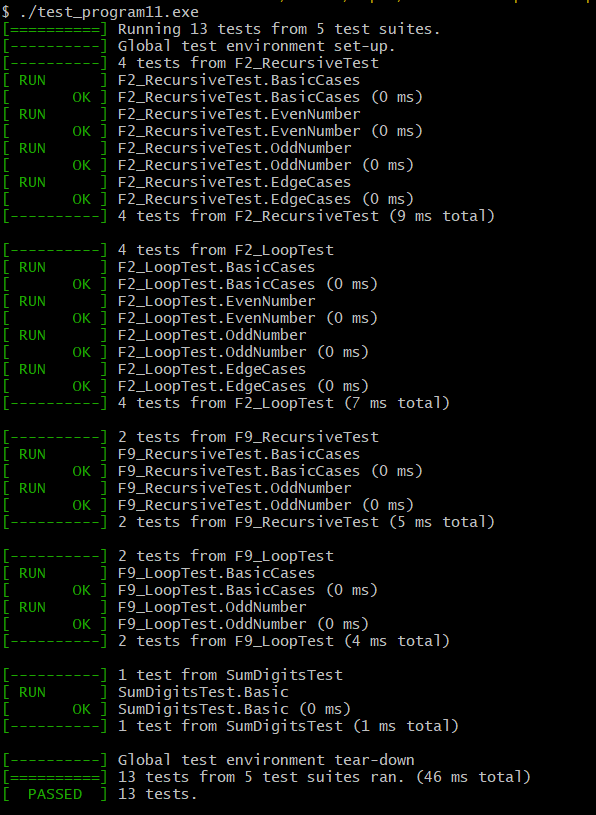


Рисунок А.1 – Обозреватель тестов

Приложение Б

(Обязательное)

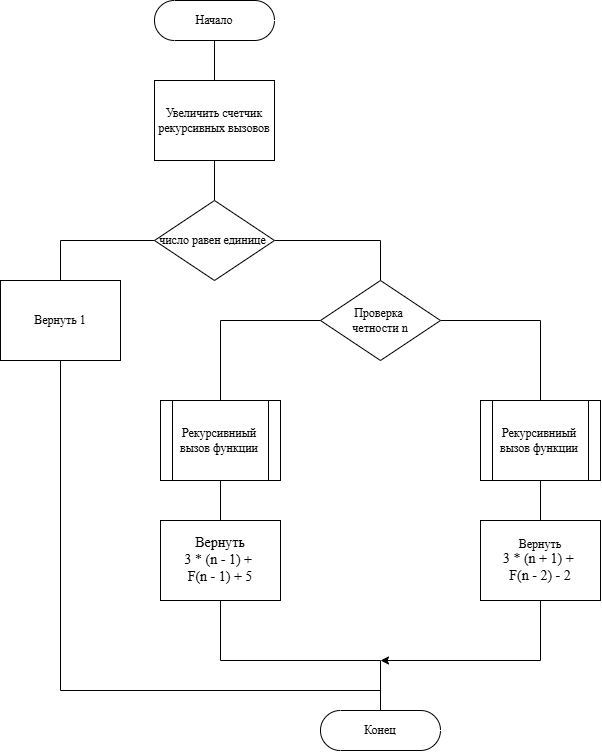


Рисунок Б.1 — Блок-схема для Formulas::F9\_loop класса Form1as

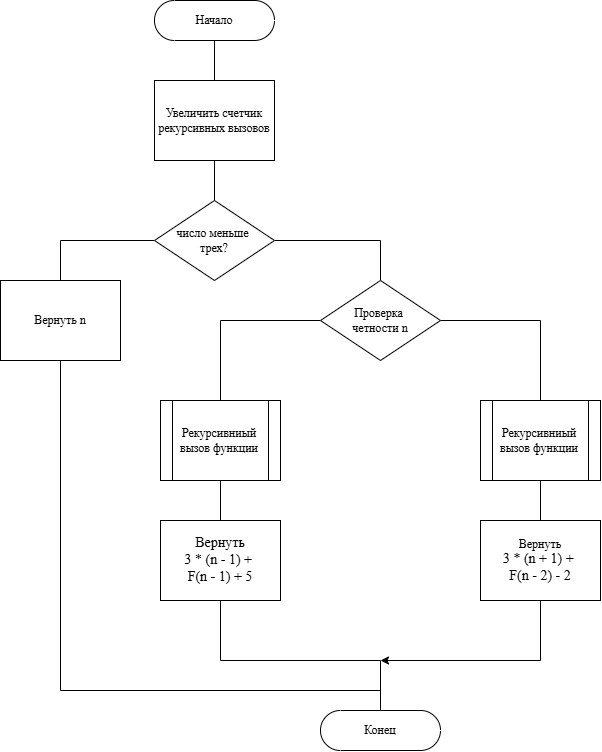


Рисунок Б.2 - Блок-схема для Formulas::F2\_loop класса Form1as

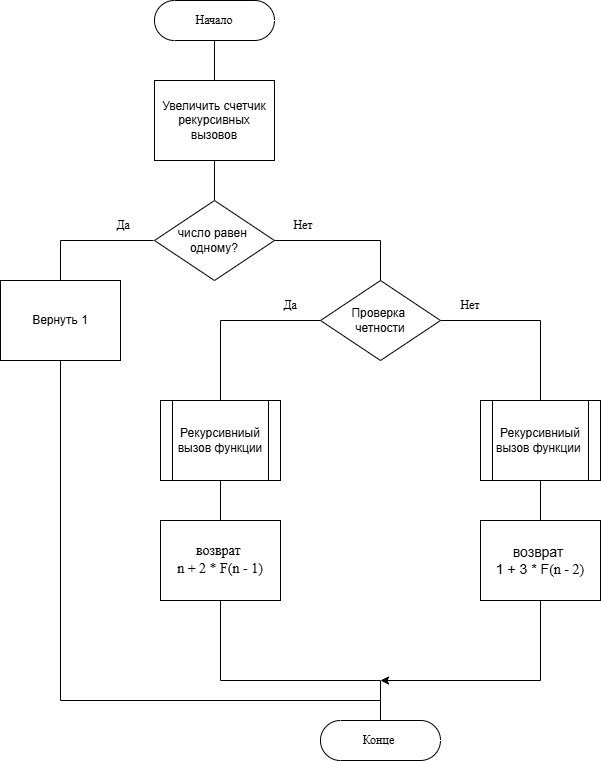


Рисунок Б.3 - Блок схема метода int Formulas::F9\_recursion класса Form1as

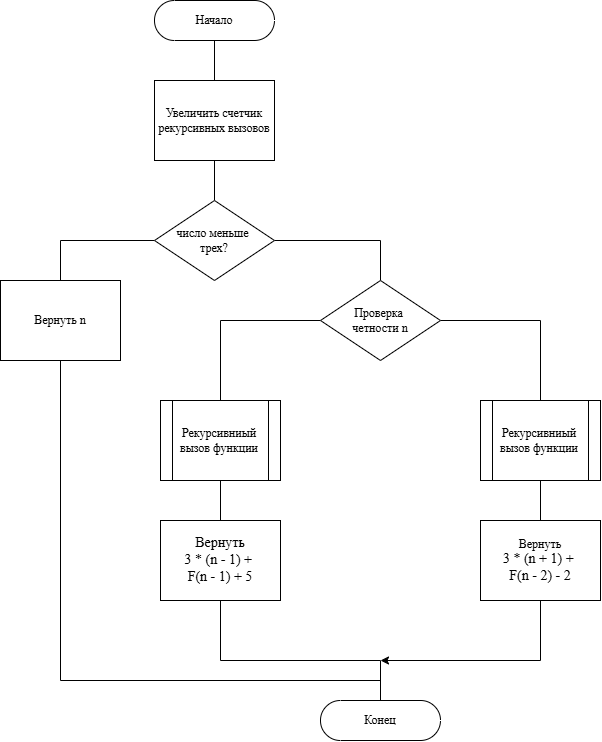


Рисунок Б.4 - Блок схема метода int Formulas::F2\_recursion класса Form1as

Приложение В

(Обязательное)

Листинг В.1 — Код файла MyForm.cpp  
#include "MyForm.h"

using namespace System;

using namespace System::Windows::Forms;

[STAThreadAttribute]

int main(array<String^>^ args) {

Application::SetCompatibleTextRenderingDefault(false);

Application::EnableVisualStyles();

блякакойжеразяпеределоваю::MyForm form;

Application::Run(% form);

}

Листинг В.2 — Код файла MyForm.h

#pragma once

#include <Windows.h>

#include <mmsystem.h>

#include "dz2.h"

#include "formulas.h"

#include <chrono>

#pragma comment(lib, "winmm.lib")

using namespace System::Media;

namespace блякакойжеразяпеределоваю {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Сводка для MyForm

/// </summary>

public ref class MyForm : public System::Windows::Forms::Form

{

public:

MyForm(void)

{

InitializeComponent();

this->FormBorderStyle = System::Windows::Forms::FormBorderStyle::Sizable;

this->MaximizeBox = true;

// Разрешаем изменение размера элементов

this->button1->Anchor = System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Left | System::Windows::Forms::AnchorStyles::Right;

this->button2->Anchor = System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Left | System::Windows::Forms::AnchorStyles::Right;

this->label2->Anchor = System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Left | System::Windows::Forms::AnchorStyles::Right;

this->label2->AutoSize = true; // Автоматическое изменение размера для текста

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~MyForm()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::Button^ button2;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Panel^ panelTask1;

private: System::Windows::Forms::TextBox^ txtInputN;

private: System::Windows::Forms::Label^ labelResult;

private: System::Windows::Forms::Button^ btnRecursion;

private: System::Windows::Forms::Button^ buttonhome;

private: System::Windows::Forms::Button^ buttonаf;

private: System::Windows::Forms::Button^ button3;

private: System::Windows::Forms::Button^ button4;

protected:

private:

int F9\_loop(int n) { return Formulas::F9\_loop(n); }

int F2\_loop(int n) { return Formulas::F2\_loop(n); }

int F9\_recursion(int n) {

static int counter = 0;

counter = 0;

return Formulas::F9\_recursion(counter, n);

}

int F2\_recursion(int n) {

static int counter = 0;

counter = 0;

return Formulas::F2\_recursion(counter, n);

}

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container^ components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(MyForm::typeid));

this->label1 = (gcnew System::Windows::Forms::Label());

this->button1 = (gcnew System::Windows::Forms::Button());

this->button2 = (gcnew System::Windows::Forms::Button());

this->label2 = (gcnew System::Windows::Forms::Label());

this->panelTask1 = (gcnew System::Windows::Forms::Panel());

this->button3 = (gcnew System::Windows::Forms::Button());

this->labelResult = (gcnew System::Windows::Forms::Label());

this->buttonаf = (gcnew System::Windows::Forms::Button());

this->button4 = (gcnew System::Windows::Forms::Button());

this->buttonhome = (gcnew System::Windows::Forms::Button());

this->btnRecursion = (gcnew System::Windows::Forms::Button());

this->txtInputN = (gcnew System::Windows::Forms::TextBox());

this->panelTask1->SuspendLayout();

this->SuspendLayout();

//

// label1

//

this->label1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 19.8F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label1->Location = System::Drawing::Point(15, 32);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(358, 55);

this->label1->TabIndex = 0;

this->label1->Text = L"Выберите задание";

this->label1->Click += gcnew System::EventHandler(this, &MyForm::label1\_Click);

//

// button1

//

this->button1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 12, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->button1->Location = System::Drawing::Point(12, 113);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(350, 50);

this->button1->TabIndex = 1;

this->button1->Text = L"Задание №1";

this->button1->UseVisualStyleBackColor = true;

this->button1->Click += gcnew System::EventHandler(this, &MyForm::button1\_Click);

//

// button2

//

this->button2->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 12, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->button2->Location = System::Drawing::Point(12, 194);

this->button2->Name = L"button2";

this->button2->Size = System::Drawing::Size(350, 50);

this->button2->TabIndex = 1;

this->button2->Text = L"Задание №2";

this->button2->UseVisualStyleBackColor = true;

this->button2->Click += gcnew System::EventHandler(this, &MyForm::button2\_Click);

//

// label2

//

this->label2->AccessibleRole = System::Windows::Forms::AccessibleRole::None;

this->label2->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 12, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label2->Location = System::Drawing::Point(120, 280);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(179, 37);

this->label2->TabIndex = 2;

this->label2->Text = L" ";

this->label2->Click += gcnew System::EventHandler(this, &MyForm::label2\_Click);

//

// panelTask1

//

this->panelTask1->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>((((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Bottom)

| System::Windows::Forms::AnchorStyles::Left)

| System::Windows::Forms::AnchorStyles::Right));

this->panelTask1->Controls->Add(this->button3);

this->panelTask1->Controls->Add(this->labelResult);

this->panelTask1->Controls->Add(this->buttonаf);

this->panelTask1->Controls->Add(this->button4);

this->panelTask1->Controls->Add(this->buttonhome);

this->panelTask1->Controls->Add(this->btnRecursion);

this->panelTask1->Controls->Add(this->txtInputN);

this->panelTask1->Location = System::Drawing::Point(2, 0);

this->panelTask1->Name = L"panelTask1";

this->panelTask1->Size = System::Drawing::Size(379, 351);

this->panelTask1->TabIndex = 3;

this->panelTask1->Visible = false;

this->panelTask1->Paint += gcnew System::Windows::Forms::PaintEventHandler(this, &MyForm::panelTask1\_Paint);

//

// button3

//

this->button3->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>(((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Left)

| System::Windows::Forms::AnchorStyles::Right));

this->button3->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10.2F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->button3->Location = System::Drawing::Point(130, 90);

this->button3->Name = L"button3";

this->button3->Size = System::Drawing::Size(110, 40);

this->button3->TabIndex = 4;

this->button3->Text = L"Цикл";

this->button3->UseVisualStyleBackColor = true;

this->button3->Click += gcnew System::EventHandler(this, &MyForm::button3\_Click\_1);

//

// labelResult

//

this->labelResult->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>((((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Bottom)

| System::Windows::Forms::AnchorStyles::Left)

| System::Windows::Forms::AnchorStyles::Right));

this->labelResult->BackColor = System::Drawing::SystemColors::InactiveBorder;

this->labelResult->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 9, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->labelResult->Location = System::Drawing::Point(14, 194);

this->labelResult->Name = L"labelResult";

this->labelResult->Size = System::Drawing::Size(346, 142);

this->labelResult->TabIndex = 3;

this->labelResult->Text = L" ";

this->labelResult->Click += gcnew System::EventHandler(this, &MyForm::labelResult\_Click);

//

// buttonаf

//

this->buttonаf->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>(((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Left)

| System::Windows::Forms::AnchorStyles::Right));

this->buttonаf->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10.2F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->buttonаf->Location = System::Drawing::Point(188, 136);

this->buttonаf->Name = L"buttonаf";

this->buttonаf->Size = System::Drawing::Size(168, 40);

this->buttonаf->TabIndex = 1;

this->buttonаf->Text = L"Формула";

this->buttonаf->UseVisualStyleBackColor = true;

this->buttonаf->Click += gcnew System::EventHandler(this, &MyForm::buttonаf\_Click);

//

// button4

//

this->button4->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10.2F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->button4->Location = System::Drawing::Point(14, 136);

this->button4->Name = L"button4";

this->button4->Size = System::Drawing::Size(168, 40);

this->button4->TabIndex = 1;

this->button4->Text = L"Задание №2";

this->button4->UseVisualStyleBackColor = true;

this->button4->Click += gcnew System::EventHandler(this, &MyForm::button4\_Click);

//

// buttonhome

//

this->buttonhome->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Right));

this->buttonhome->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10.2F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->buttonhome->Location = System::Drawing::Point(246, 90);

this->buttonhome->Name = L"buttonhome";

this->buttonhome->Size = System::Drawing::Size(110, 40);

this->buttonhome->TabIndex = 1;

this->buttonhome->Text = L"Домой";

this->buttonhome->UseVisualStyleBackColor = true;

this->buttonhome->Click += gcnew System::EventHandler(this, &MyForm::button3\_Click);

//

// btnRecursion

//

this->btnRecursion->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 9, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->btnRecursion->Location = System::Drawing::Point(14, 90);

this->btnRecursion->Name = L"btnRecursion";

this->btnRecursion->Size = System::Drawing::Size(110, 40);

this->btnRecursion->TabIndex = 1;

this->btnRecursion->Text = L"Рекурсия";

this->btnRecursion->UseVisualStyleBackColor = true;

this->btnRecursion->Click += gcnew System::EventHandler(this, &MyForm::btnRecursion\_Click);

//

// txtInputN

//

this->txtInputN->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>(((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Left)

| System::Windows::Forms::AnchorStyles::Right));

this->txtInputN->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 13.8F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->txtInputN->Location = System::Drawing::Point(14, 27);

this->txtInputN->Multiline = true;

this->txtInputN->Name = L"txtInputN";

this->txtInputN->Size = System::Drawing::Size(350, 40);

this->txtInputN->TabIndex = 0;

//

// MyForm

//

this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->BackColor = System::Drawing::SystemColors::InactiveBorder;

this->ClientSize = System::Drawing::Size(382, 353);

this->Controls->Add(this->panelTask1);

this->Controls->Add(this->label2);

this->Controls->Add(this->button2);

this->Controls->Add(this->button1);

this->Controls->Add(this->label1);

this->Icon = (cli::safe\_cast<System::Drawing::Icon^>(resources->GetObject(L"$this.Icon")));

this->Name = L"MyForm";

this->Text = L"MyForm";

this->panelTask1->ResumeLayout(false);

this->panelTask1->PerformLayout();

this->ResumeLayout(false);

}

#pragma endregion

private: System::Void label1\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e) {

PlaySound(L"C:\\Users\\adida\\Downloads\\sound.wav", NULL, SND\_FILENAME | SND\_ASYNC);

//is->label2->Text = L"Загрузка...";

panelTask1->Visible = true;

}

private: System::Void button2\_Click(System::Object^ sender, System::EventArgs^ e) {

PlaySound(L"C:\\Users\\adida\\Downloads\\sound.wav", NULL, SND\_FILENAME | SND\_ASYNC);

dz2^ form2 = gcnew dz2();

form2->Show();

//his->label2->Text = L"Загрузка...";

}

private: System::Void label2\_Click(System::Object^ sender, System::EventArgs^ e) {

}

private: System::Void panelTask1\_Paint(System::Object^ sender, System::Windows::Forms::PaintEventArgs^ e) {

}

private: System::Void labelResult\_Click(System::Object^ sender, System::EventArgs^ e) { //leble для задания 1 вывод

}

private: System::Void button3\_Click(System::Object^ sender, System::EventArgs^ e) {

PlaySound(L"C:\\Users\\adida\\Downloads\\sound.wav", NULL, SND\_FILENAME | SND\_ASYNC);

panelTask1->Visible = false;

//panelTask2->Visible = false;

}

private: System::Void button4\_Click(System::Object^ sender, System::EventArgs^ e) {

PlaySound(L"C:\\Users\\adida\\Downloads\\sound.wav", NULL, SND\_FILENAME | SND\_ASYNC);

dz2^ form2 = gcnew dz2();

form2->Show();

}

private: System::Void btnRecursion\_Click(System::Object^ sender, System::EventArgs^ e) {

PlaySound(L"C:\\Users\\adida\\Downloads\\sound.wav", NULL, SND\_FILENAME | SND\_ASYNC);

try {

int n = Convert::ToInt32(txtInputN->Text);

if (n > 20000) {

labelResult->Text = "Ошибка! Введите число меньше или равное 20000.";

return;

}

int counterV9 = 0;

int counterV2 = 0;

String^ result = "";

// === Формула 9 ===

auto startV9 = std::chrono::high\_resolution\_clock::now();

int resV9 = Formulas::F9\_recursion(counterV9, n);

auto endV9 = std::chrono::high\_resolution\_clock::now();

auto timeV9 = std::chrono::duration\_cast<std::chrono::microseconds>(endV9 - startV9);

// === Формула 2 ===

auto startV2 = std::chrono::high\_resolution\_clock::now();

int resV2 = Formulas::F2\_recursion(counterV2, n);

auto endV2 = std::chrono::high\_resolution\_clock::now();

auto timeV2 = std::chrono::duration\_cast<std::chrono::microseconds>(endV2 - startV2);

result += "=== Формула 9 ===\n";

result += "Результат: " + resV9.ToString() + "\n";

result += "Вызовов рекурсии: " + counterV9.ToString() + "\n";

result += "Время: " + timeV9.count().ToString() + " мкс\n\n";

result += "=== Формула 2 ===\n";

result += "Результат: " + resV2.ToString() + "\n";

result += "Вызовов рекурсии: " + counterV2.ToString() + "\n";

result += "Время: " + timeV2.count().ToString() + " мкс";

labelResult->Text = result;

}

catch (...) {

labelResult->Text = "Ошибка! Введите целое число.";

}

}

private: System::Void buttonаf\_Click(System::Object^ sender, System::EventArgs^ e) {

PlaySound(L"C:\\Users\\adida\\Downloads\\sound.wav", NULL, SND\_FILENAME | SND\_ASYNC);

labelResult->Text =

"Формула 9:\n"

"F(n) = 1, при n = 1\n"

"F(n) = n + 2\*F(n-1), если n чётное\n"

"F(n) = 1 + 3\*F(n-2), если n > 1 и нечётное\n\n"

"Формула 2:\n"

"F(n) = n, при n < 3\n"

"F(n) = 3\*(n-1) + F(n-1) + 5, если n больше или равно 3 и чётное\n"

"F(n) = 3\*(n+1) + F(n-2) - 2, если n больше или равно 3 и нечётное";

}

private: System::Void button3\_Click\_1(System::Object^ sender, System::EventArgs^ e) {

PlaySound(L"C:\\Users\\adida\\Downloads\\sound.wav", NULL, SND\_FILENAME | SND\_ASYNC);

try {

int n = Convert::ToInt32(txtInputN->Text);

// Проверка лимита

if (n < 0 || n > 10000000) {

labelResult->Text = "Ошибка! Число должно быть от 0 до 10,000,000.";

return; // Прерываем выполнение

}

String^ result = "";

auto startV9 = std::chrono::high\_resolution\_clock::now();

int resV9\_loop = F9\_loop(n);

auto endV9 = std::chrono::high\_resolution\_clock::now();

auto timeV9 = std::chrono::duration\_cast<std::chrono::microseconds>(endV9 - startV9);

auto startV2 = std::chrono::high\_resolution\_clock::now();

int resV2\_loop = F2\_loop(n);

auto endV2 = std::chrono::high\_resolution\_clock::now();

auto timeV2 = std::chrono::duration\_cast<std::chrono::microseconds>(endV2 - startV2);

result += "=== Формула 9 ===\n";

result += "Цикл: " + resV9\_loop.ToString() + "\n";

result += "Время: " + timeV9.count().ToString() + " мкс\n\n";

result += "=== Формула 2 ===\n";

result += "Цикл: " + resV2\_loop.ToString() + "\n";

result += "Время: " + timeV2.count().ToString() + " мкс";

labelResult->Text = result;

}

catch (...) {

labelResult->Text = "Ошибка! Введите целое число от 0 до 10,000,000.";

}

}

private: System::Void panelTask2\_Paint(System::Object^ sender, System::Windows::Forms::PaintEventArgs^ e) {

}

};

}

Листинг В.3 — Код файла formulas.cpp

#include "formulas.h"

int Formulas::F9\_loop(int n) {

if (n == 1) return 1;

int\* dp = new int[n + 1];

dp[1] = 1;

for (int i = 2; i <= n; ++i) {

dp[i] = (i % 2 == 0) ? i + 2 \* dp[i - 1] : 1 + 3 \* dp[i - 2];

}

int res = dp[n];

delete[] dp;

return res;

}

int Formulas::F2\_loop(int n) {

if (n < 3) return n;

int\* dp = new int[n + 1];

dp[1] = 1;

dp[2] = 2;

for (int i = 3; i <= n; ++i) {

dp[i] = (i % 2 == 0) ? 3 \* (i - 1) + dp[i - 1] + 5 : 3 \* (i + 1) + dp[i - 2] - 2;

}

int res = dp[n];

delete[] dp;

return res;

}

int Formulas::F9\_recursion(int& counter, int n) {

counter++;

if (n == 1) return 1;

return (n % 2 == 0) ? n + 2 \* F9\_recursion(counter, n - 1)

: 1 + 3 \* F9\_recursion(counter, n - 2);

}

int Formulas::F2\_recursion(int& counter, int n) {

counter++;

if (n < 3) return n;

return (n % 2 == 0) ? 3 \* (n - 1) + F2\_recursion(counter, n - 1) + 5

: 3 \* (n + 1) + F2\_recursion(counter, n - 2) - 2;

}

int Formulas::SumDigits(int n) {

if (n == 0) return 0;

return (n % 10) + SumDigits(n / 10);

}

Листинг В.4 — Код файла formulas.h

// formulas.h

#pragma once

class Formulas {

public:

static int F9\_loop(int n);

static int F2\_loop(int n);

static int F9\_recursion(int& counter, int n);

static int F2\_recursion(int& counter, int n);

static int SumDigits(int n);

};

Листинг В.5 — Код файла dz2.h

#pragma once

#include "formulas.h"

#include <Windows.h>

#include <mmsystem.h>

#include <chrono>

#pragma comment(lib, "winmm.lib")

namespace блякакойжеразяпеределоваю {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Сводка для dz2

/// </summary>

public ref class dz2 : public System::Windows::Forms::Form

{

public:

dz2(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~dz2()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::TextBox^ textBox11;

protected:

private: System::Windows::Forms::Button^ buttonres;

private: System::Windows::Forms::Button^ buttonFormul;

private: System::Windows::Forms::Label^ labelresalt;

private: System::Windows::Forms::Button^ Home1;

protected:

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(dz2::typeid));

this->textBox11 = (gcnew System::Windows::Forms::TextBox());

this->buttonres = (gcnew System::Windows::Forms::Button());

this->buttonFormul = (gcnew System::Windows::Forms::Button());

this->labelresalt = (gcnew System::Windows::Forms::Label());

this->Home1 = (gcnew System::Windows::Forms::Button());

this->SuspendLayout();

//

// textBox11

//

this->textBox11->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>(((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Left)

| System::Windows::Forms::AnchorStyles::Right));

this->textBox11->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 16.2F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->textBox11->Location = System::Drawing::Point(14, 27);

this->textBox11->Multiline = true;

this->textBox11->Name = L"textBox11";

this->textBox11->Size = System::Drawing::Size(350, 40);

this->textBox11->TabIndex = 0;

//

// buttonres

//

this->buttonres->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>(((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Left)

| System::Windows::Forms::AnchorStyles::Right));

this->buttonres->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 12, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->buttonres->Location = System::Drawing::Point(14, 89);

this->buttonres->Name = L"buttonres";

this->buttonres->Size = System::Drawing::Size(349, 40);

this->buttonres->TabIndex = 1;

this->buttonres->Text = L"Расчитать";

this->buttonres->UseVisualStyleBackColor = true;

this->buttonres->Click += gcnew System::EventHandler(this, &dz2::buttonres\_Click);

//

// buttonFormul

//

this->buttonFormul->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>(((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Left)

| System::Windows::Forms::AnchorStyles::Right));

this->buttonFormul->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 12, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->buttonFormul->Location = System::Drawing::Point(14, 135);

this->buttonFormul->Name = L"buttonFormul";

this->buttonFormul->Size = System::Drawing::Size(170, 40);

this->buttonFormul->TabIndex = 1;

this->buttonFormul->Text = L"Формула";

this->buttonFormul->UseVisualStyleBackColor = true;

this->buttonFormul->Click += gcnew System::EventHandler(this, &dz2::buttonFormul\_Click);

//

// labelresalt

//

this->labelresalt->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>((((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Bottom)

| System::Windows::Forms::AnchorStyles::Left)

| System::Windows::Forms::AnchorStyles::Right));

this->labelresalt->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 10.2F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->labelresalt->Location = System::Drawing::Point(11, 178);

this->labelresalt->Name = L"labelresalt";

this->labelresalt->Size = System::Drawing::Size(352, 166);

this->labelresalt->TabIndex = 2;

//

// Home1

//

this->Home1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 12, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->Home1->Location = System::Drawing::Point(193, 135);

this->Home1->Name = L"Home1";

this->Home1->Size = System::Drawing::Size(170, 40);

this->Home1->TabIndex = 3;

this->Home1->Text = L"Домой";

this->Home1->UseVisualStyleBackColor = true;

this->Home1->Click += gcnew System::EventHandler(this, &dz2::Home1\_Click);

//

// dz2

//

this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(382, 353);

this->Controls->Add(this->Home1);

this->Controls->Add(this->labelresalt);

this->Controls->Add(this->buttonFormul);

this->Controls->Add(this->buttonres);

this->Controls->Add(this->textBox11);

this->Icon = (cli::safe\_cast<System::Drawing::Icon^>(resources->GetObject(L"$this.Icon")));

this->Name = L"dz2";

this->Text = L"dz2";

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void buttonres\_Click(System::Object^ sender, System::EventArgs^ e) {

PlaySound(L"C:\\Users\\adida\\Downloads\\sound.wav", NULL, SND\_FILENAME | SND\_ASYNC);

try {

int number = Convert::ToInt32(textBox11->Text);

int sum = Formulas::SumDigits(number);

labelresalt->Text = "Сумма цифр: " + sum.ToString();

}

catch (...) {

labelresalt->Text = "Ошибка! Введите число.";

}

}

private: System::Void buttonFormul\_Click(System::Object^ sender, System::EventArgs^ e) {

PlaySound(L"C:\\Users\\adida\\Downloads\\sound.wav", NULL, SND\_FILENAME | SND\_ASYNC);

labelresalt->Text =

"Формула задания 2:\n"

"F(n) = n % 10 + F(n / 10)\n"

"Базовый случай: F(0) = 0\n\n"

"Пример для 2024:\n"

"2 + 0 + 2 + 4 = 8\n"

"Глубина рекурсии: 5 вызовов";

}

private: System::Void Home1\_Click(System::Object^ sender, System::EventArgs^ e) {

PlaySound(L"C:\\Users\\adida\\Downloads\\sound.wav", NULL, SND\_FILENAME | SND\_ASYNC);

this->Close();

}

};

}

Листинг В.6 — Код файла Test.cpp

#include "gtest/gtest.h"

#include "formulas.h"

TEST(F2\_RecursiveTest, BasicCases) {

int counter = 0;

EXPECT\_EQ(Formulas::F2\_recursion(counter, 1), 1);

EXPECT\_EQ(Formulas::F2\_recursion(counter, 2), 2);

}

TEST(F2\_RecursiveTest, EvenNumber) {

int counter = 0;

EXPECT\_EQ(Formulas::F2\_recursion(counter, 4), 25);

}

TEST(F2\_RecursiveTest, OddNumber) {

int counter = 0;

EXPECT\_EQ(Formulas::F2\_recursion(counter, 3), 11);

}

TEST(F2\_RecursiveTest, EdgeCases) {

int counter = 0;

EXPECT\_EQ(Formulas::F2\_recursion(counter, 0), 0);

}

TEST(F2\_LoopTest, BasicCases) {

EXPECT\_EQ(Formulas::F2\_loop(1), 1);

EXPECT\_EQ(Formulas::F2\_loop(2), 2);

}

TEST(F2\_LoopTest, EvenNumber) {

EXPECT\_EQ(Formulas::F2\_loop(4), 25);

}

TEST(F2\_LoopTest, OddNumber) {

EXPECT\_EQ(Formulas::F2\_loop(3), 11);

}

TEST(F2\_LoopTest, EdgeCases) {

EXPECT\_EQ(Formulas::F2\_loop(0), 0);

}

TEST(F9\_RecursiveTest, BasicCases) {

int counter = 0;

EXPECT\_EQ(Formulas::F9\_recursion(counter, 1), 1);

EXPECT\_EQ(Formulas::F9\_recursion(counter, 2), 4);

}

TEST(F9\_RecursiveTest, OddNumber) {

int counter = 0;

EXPECT\_EQ(Formulas::F9\_recursion(counter, 3), 4);

}

TEST(F9\_LoopTest, BasicCases) {

EXPECT\_EQ(Formulas::F9\_loop(1), 1);

EXPECT\_EQ(Formulas::F9\_loop(2), 4);

}

TEST(F9\_LoopTest, OddNumber) {

EXPECT\_EQ(Formulas::F9\_loop(5), 13);

}

TEST(SumDigitsTest, Basic) {

EXPECT\_EQ(Formulas::SumDigits(2024), 8);

EXPECT\_EQ(Formulas::SumDigits(-456), -15);

}

int main(int argc, char\*\* argv) {

testing::InitGoogleTest(&argc, argv);

return RUN\_ALL\_TESTS();

}