

Министерство образования Российской Федерации
Пензенский государственный университет
Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе
по курсу «Программирование»
на тему «Калькулятор»

*Хорошо
01.06.2026
Г. Генералов*

Выполнил:

студент группы 25BVB3
Махотин Д. А.

Принял:

к.т.н., доцент Генералова А.А.

Пенза, 2026

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет Вычислительной техники

Кафедра "Вычислительная техника"

"УТВЕРЖДАЮ"

Зав. кафедрой ВТ

М.А. Митрохин

«19» февраля 2026г

ЗАДАНИЕ

на курсовое проектирование по курсу

Программирование

Студенту Махотин Денис Александрович Группа 25ВВВЗ

Тема проекта: *Разработка программы сложной структуры методом нисходящего программирования.*

Исходные данные (технические требования) на проектирование
Программа "Калькулятор". Компьютер осуществляет расчёт арифметических действий. Пользователь вводит числа и выполняет операцию. Компьютер работает с целыми и десятичными числами. В файле сохраняется история последних действий.

Обязательные требования к программе:

1. Многомодульность.
2. Использование сложных типов данных – структур, списков, файлов.
3. Режим работы видеосистемы – текстовый / графический.
4. Устройство ввода информации – клавиатура, мышь.
5. Пользовательский интерфейс должен быть построен на основе меню и панели инструментов.
6. Наличие заставки.
7. Операционная система MS Windows.
8. Язык программирования – Си и Ассемблер.
9. Среда разработки ПО – Microsoft Visual Studio.

Объем работы по курсу

1. Расчетная часть

Разработка программы.

2. Графическая часть

Схема данных, схема ресурсов системы, схема работы системы, иерархическая структура программы, схема взаимодействия программы.

3. Экспериментальная часть

Отладка программы.

Срок выполнения проекта по разделам

В соответствии с графиком.

Дата выдачи задания “19” февраля 20

Дата защиты проекта “26” июня 26

Руководитель А. Генералов А.Ф.

Задание получил “19” Феврале 2026г.

Студент Мах/ - Махотин Денис Александрович.

Содержание

Введение.....	5
Постановка задачи	6
Выбор решения.....	7
Описание разработки программы	8
Отладка и тестирование	8
Описание программы.....	9
Руководство пользователя.....	19
Заключение.....	23
Приложение А Листинги программы.....	24
Приложение В Результаты работы программы	29
Приложение С Результат работы программы	30

Введение

На современном этапе развития вычислительной техники программирование остаётся одной из ключевых дисциплин, формирующих инженерное мышление и практические навыки разработки программного обеспечения. Несмотря на повсеместное распространение высокоуровневых языков и автоматизированных средств разработки, понимание принципов работы процессора, управления памятью и взаимодействия программного кода с аппаратным обеспечением является обязательным требованием для квалифицированного специалиста.

Арифметические вычисления лежат в основе практически всех прикладных задач: от инженерных расчётов и научных симуляций до финансовой аналитики и обработки сигналов. Точность и скорость выполнения операций с вещественными числами напрямую зависят от выбранной архитектуры вычислений. В процессорах семейства x86 для работы с числами с плавающей запятой исторически используется математический сопроцессор x87 FPU, реализующий стек регистров и набор специализированных команд для сложения, вычитания, умножения и деления.

В рамках данной курсовой работы разрабатывается программа-калькулятор, реализующая базовые арифметические операции с поддержкой целых и десятичных чисел. Отличительной особенностью проекта является использование встроенных ассемблерных вставок (inline assembly) для выполнения вычислений, что позволяет продемонстрировать принципы низкоуровневого управления сопроцессором и соответствует учебным требованиям по изучению архитектуры ЭВМ.

Программа оснащена текстовым меню, экранной заставкой с информацией об авторе и руководителе, механизмом накопления истории вычислений с сохранением в файл, а также поддержкой неограниченного количества операндов до ввода завершающего символа =. Реализован режим деления с остатком, при котором результат выводится в виде частного и остатка отдельно.

Актуальность работы обусловлена необходимостью закрепления навыков работы с ассемблерными вставками в среде современных компиляторов, изучения особенностей вычислений с плавающей запятой, а также формирования компетенций по проектированию пользовательского интерфейса и работе с файловой системой в рамках консольных приложений.

Цель курсовой работы – разработка и отладка программы-калькулятора с интеграцией ассемблерных инструкций x87 FPU, обеспечивающей корректное выполнение арифметических операций, ведение истории расчётов и соответствие требованиям технического задания.

Постановка задачи

Необходимо разработать консольную программу-калькулятор на языке C/C++, выполняющую арифметические операции над вещественными числами. Программа должна соответствовать следующим требованиям:

Интерфейс:

- При запуске выводится заставка с указанием темы курсовой работы, ФИО студента и преподавателя.
- Основное меню содержит 5 пунктов арифметических действий и пункт выхода.
- Навигация осуществляется вводом цифр от 0 до 5 с подтверждением клавишей Enter.

Ввод данных:

- После выбора операции пользователю предлагается вводить числа по одному.
- Ввод завершается при вводе символа =.
- Количество операндов не ограничено (минимум 2 для выполнения операции).
- Поддерживаются целые и десятичные числа (включая экспоненциальную запись).

Вычисления:

- Сложение, вычитание, умножение, деление реализуются через встроенные ассемблерные вставки.
- Каждая asm-вставка должна содержать минимальное количество инструкций (2–4 строки).
- Деление с остатком (пункт 5) выводит частное и остаток отдельно.
- Обработка деления на ноль должна быть предусмотрена до вызова asm-блока.

История операций:

- Результаты всех успешных вычислений сохраняются в файл history.txt.
- На экране после каждого расчёта выводится краткая история последних действий.

Локализация:

- Программа должна корректно отображать русский текст в консоли Windows.
- Используется кодовая страница 1251 и соответствующая локаль.

Технические ограничения:

- Компиляция в среде Microsoft Visual Studio.
- Платформа: Win32 (x86), так как inline-asm не поддерживается в x64.
- Использование стандартной библиотеки C и встроенного ассемблера MSVC.

Многомодульность программы. Программа разделена на логические модули (main.c, calc_asm.c, io.c, io.h, consts.h, calc_asm.h). Это упрощает отладку, тестирование отдельных блоков и соответствует принципам структурного программирования. Использование сложных типов данных – массивов, файлового ввода-вывода, парсинга строк. Режим работы видеосистемы – текстовый. Устройство ввода-вывода – клавиатура. Интерфейс построен на основе меню.

Выбор решения

При запуске программы пользователь видит приветственный экран с информацией о курсовой работе. Далее выводится главное меню с шестью пунктами:

Сложение – пользователь вводит числа, завершает ввод символом =, программа последовательно складывает операнды.

Вычитание – аналогично, выполняется последовательное вычитание.

Умножение – последовательное перемножение введенных чисел.

Деление – последовательное деление первого числа на остальные.

Деление с остатком – вычисляется частное и остаток. Частное округляется вниз, остаток вычисляется через `fmod`.

Выход – завершение работы программы.

Для выполнения арифметических операций выбран сопроцессор x87 FPU. Данный выбор обусловлен следующими факторами:

- x87 нативно поддерживает формат `double` (64-бит IEEE 754), что обеспечивает высокую точность вычислений.

- Синтаксис `MSVC __asm { ... }` позволяет напрямую обращаться к переменным C, минуя ручное управление адресами.

- В отличие от `SSE/AVX`, x87 не требует выравнивания данных и работает с отдельными скалярными значениями, что упрощает реализацию и отладку в учебных целях.

Программа реализована в многомодульной архитектуре: вычисления вынесены в отдельный модуль `calc_asm.c`, ввод/вывод и работа с историей – в `io.c`, главный цикл управления – в `main.c`. Это позволяет изолировать ассемблерные вставки, упростить проверку корректности вычислений и избежать ошибок линковки.

История операций сохраняется в текстовый файл в режиме дополнения ("a"). Это позволяет накапливать данные между запусками программы. Для отображения на экране используется функция, читающая последние 5 записей.

Ввод чисел реализован через посимвольное чтение строк с последующим парсингом `strtod`. Это исключает ошибки переполнения буфера и позволяет корректно обрабатывать невалидный ввод без аварийного завершения.

Обработка деления на ноль выполняется на уровне C до передачи данных в `asm`-блок, так как сопроцессор x87 генерирует исключение FPU, которое в консольном приложении может привести к неопределённому поведению.

Описание разработки программы

Разработанная программа состоит из следующих модулей, реализованных нисходящим методом:

`main.c` – основной файл. Главный цикл, меню, обработка выбора, координация модулей.

`calc_asm.c / .h` – модуль вычислений. Функция `calc_asm()` содержит 4 блока встроенного ассемблера (`fld`, `fadd/fsb`, `fmul/fdiv`, `fstp`).

`io.c / .h` – модуль ввода-вывода. Функции `show_splash()`, `show_menu()`, `read_numbers()`, `save_history()`, `show_recent_history()`, очистка буфера.

`consts.h` – централизованное хранение констант (`MAX_NUMS`, `HISTORY_FILE` и др.).

Каждый модуль компилируется отдельно и линкуется в единый исполняемый файл. Разработка велась нисходящим методом: сначала спроектирован главный цикл и меню, затем реализован парсер ввода, после чего добавлены `asm`-блоки и модуль истории. Каждый блок тестировался сразу после написания, что минимизировало накопление ошибок.

Архитектура вычислений итеративная: результат предыдущей операции становится первым операндом для следующей. Это позволяет обрабатывать неограниченное количество чисел без динамического выделения памяти.

Для поддержки русского языка в начале `main()` вызывается `setlocale(LC_ALL, "Russian")` и `system("chcp 1251 > nul")`. Файл исходного кода сохранён в кодировке Windows-1251 для корректной компиляции в MSVC.

Отладка и тестирование

В качестве среды разработки была выбрана программа Microsoft Visual Studio 2019. Программа обладает всеми средствами необходимыми при разработке и отладке программы. Для отладки использовались несколько возможностей Visual Studio: точка останова, трассировка, анализ содержимого переменных.

Тестирование проводилось во время разработки и также после завершения разработки. В ходе нее было выявлено огромное количество проблем, связанных с работой с файлами, работой с памятью, работой с полями структуры.

Описание программы

Приложение calculator.exe является основным модулем программы. При запуске происходит вывод заставки, после чего пользователь выбирает пункт меню. Описание состояний программы представлено в таблице 1.

Клавиши, вызывающее событие	Действие пользователя	Действие программы
1. Enter	Выбран пункт «Сложение»	Запускается диалог ввода чисел, выполняется сложение.
2. Enter	Выбран пункт «Вычитание»	Запускается диалог ввода чисел, выполняется вычитание.
3. Enter	Выбран пункт «Умножение»	Запускается диалог ввода чисел, выполняется умножение.
4. Enter	Выбран пункт «Деление»	Запускается диалог ввода чисел, выполняется деление.
5. Enter	Выбран пункт «Деление с остатком»	Запускается диалог ввода чисел, вычисляются частное и остаток, выводятся отдельно.
0. Enter	Выбран пункт «Выход»	Программа завершает работу, сохраняя при этом историю.

Таблица 1.

Клавиши, вызывающее событие	Действие пользователя	Действие программы
-----------------------------	-----------------------	--------------------

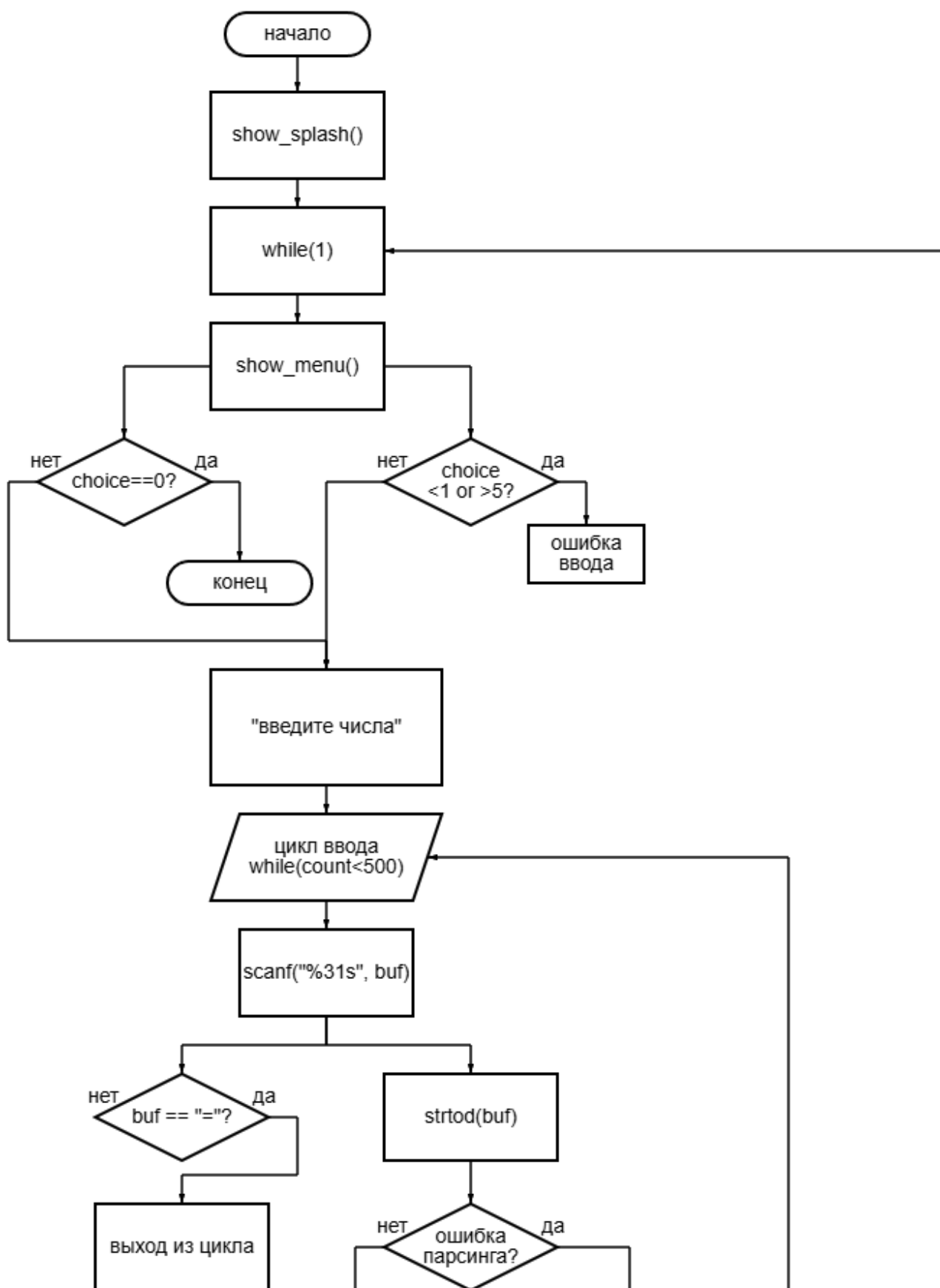
1. Число + Enter	Ввод операнда	Число парсится через strtod, сохраняется в массив.
2. «=» + Enter	Завершение ввода	Цикл прерывается, начинается вычисление.
3. Неверный формат	Ввод текста/символов	Выводится «[!] Некорректный ввод», запрос повторного действия.
4. <2 чисел + «=»	Недостаточно операндов	Ошибка, возврат в главное меню.

Таблица 2. – Ввод операндов

Операция	Float load	Floating-point add	Floating-point store & pop	Описание
1. Сложение	fld a	fadd b	fstp res	Загрузка, сложение, сохранение.
2. Вычитание	fld a	fsub b	fstp res	Загрузка, вычитание, сохранение.
3. Умножение.	fld a	fmul b	fstp res	Загрузка, умножение, сохранение.
4. Деление.	fld a	fdiv b	fstp res	Загрузка, деление, сохранение.

Таблица 3 – Ассемблерные вставки.

Схема программы. Функция Main



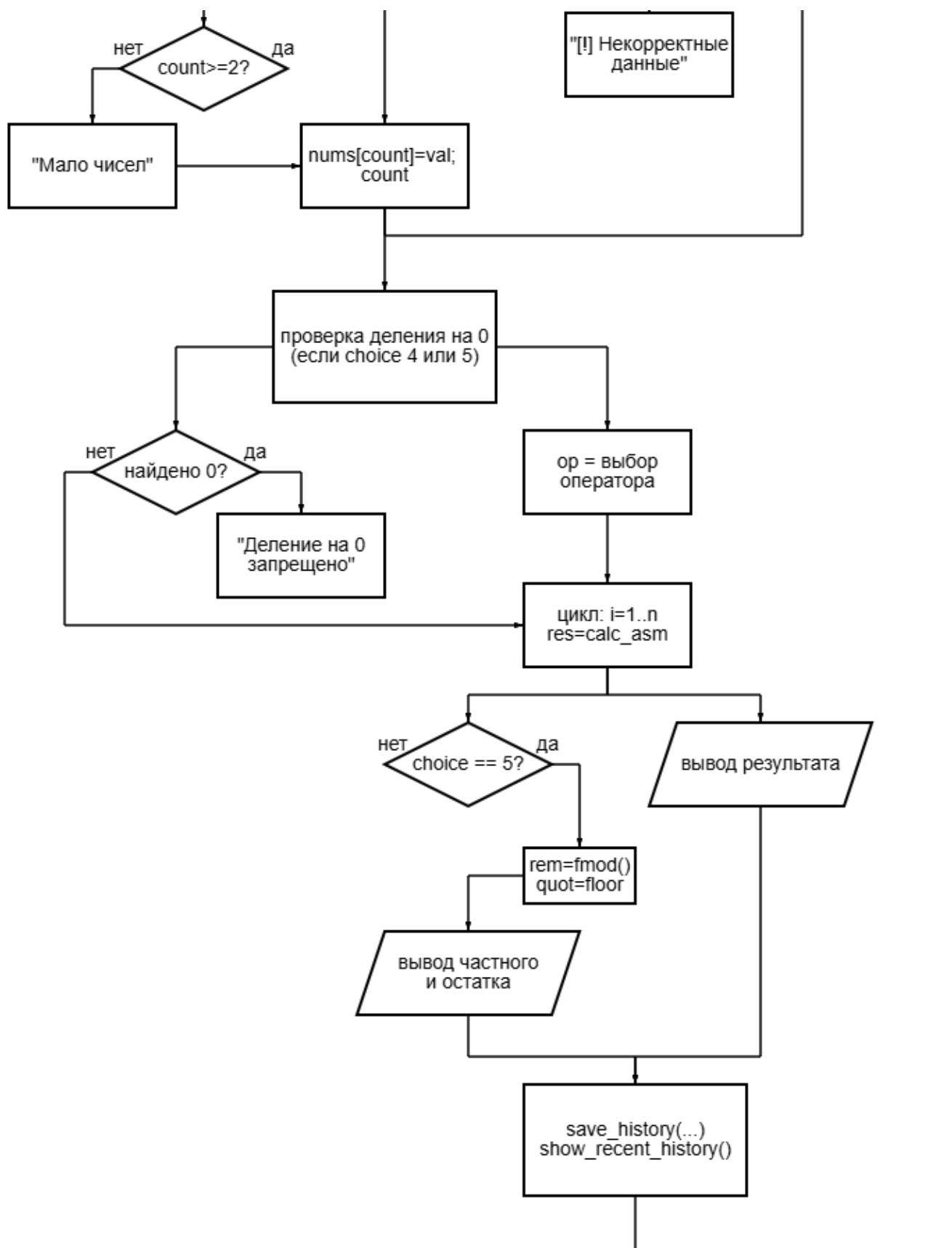


Рисунок - функция `main`

Схема программы. Функция calc_asm.

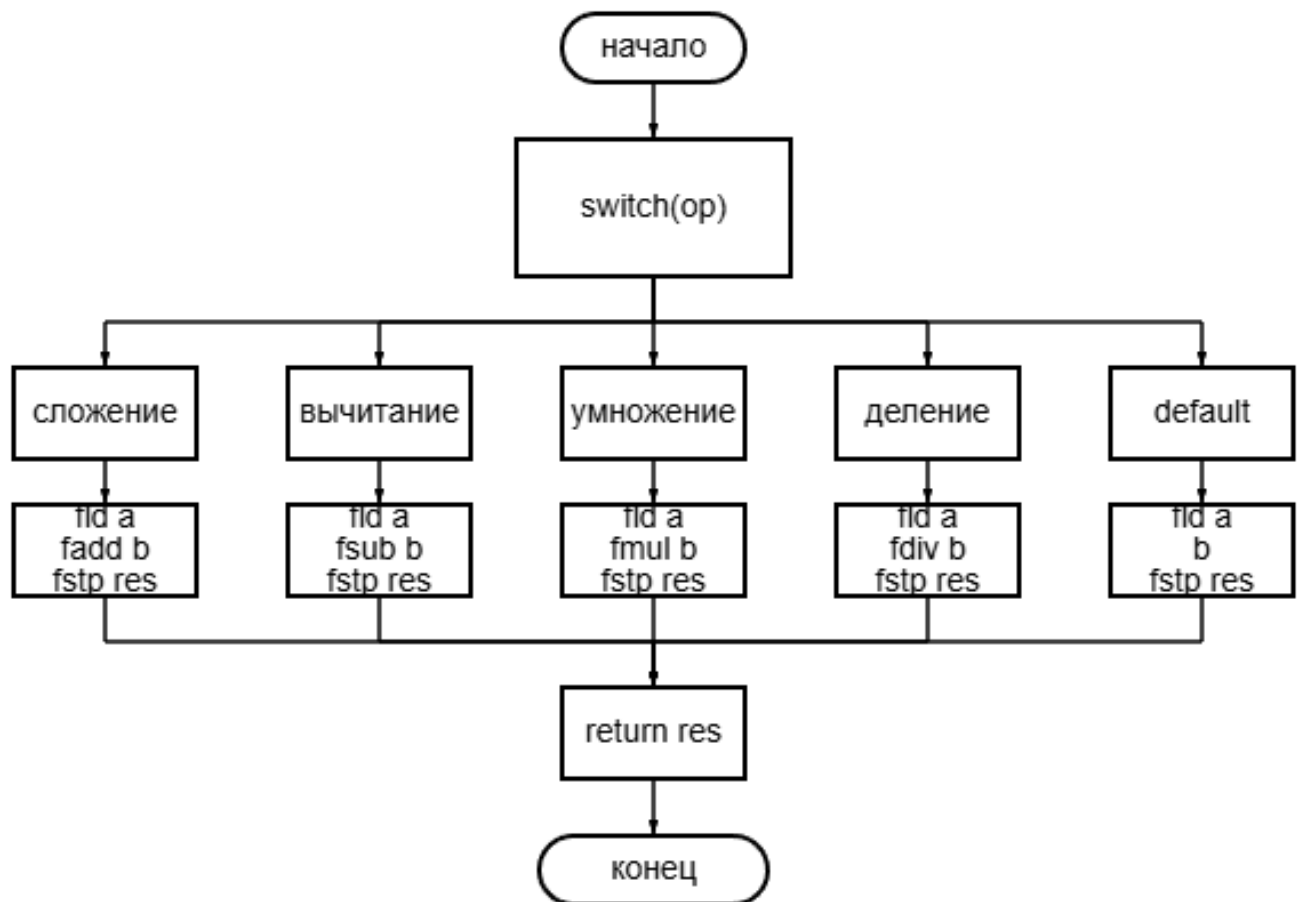


Рисунок - Функция calc_asm

Схема программы. Функция save_history.

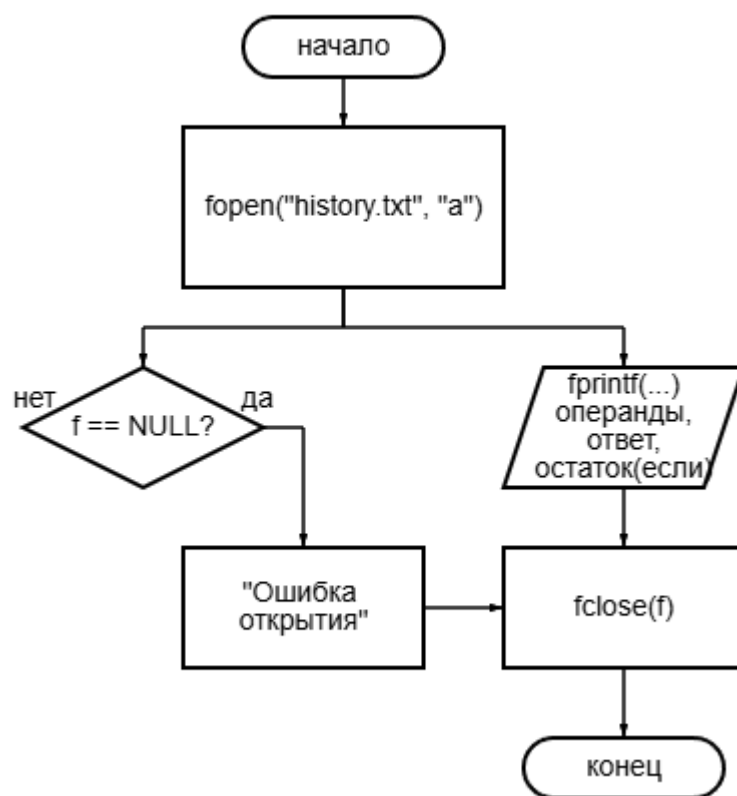


Рисунок - Функция save_history

Схема программы. Функция show_recent_history.

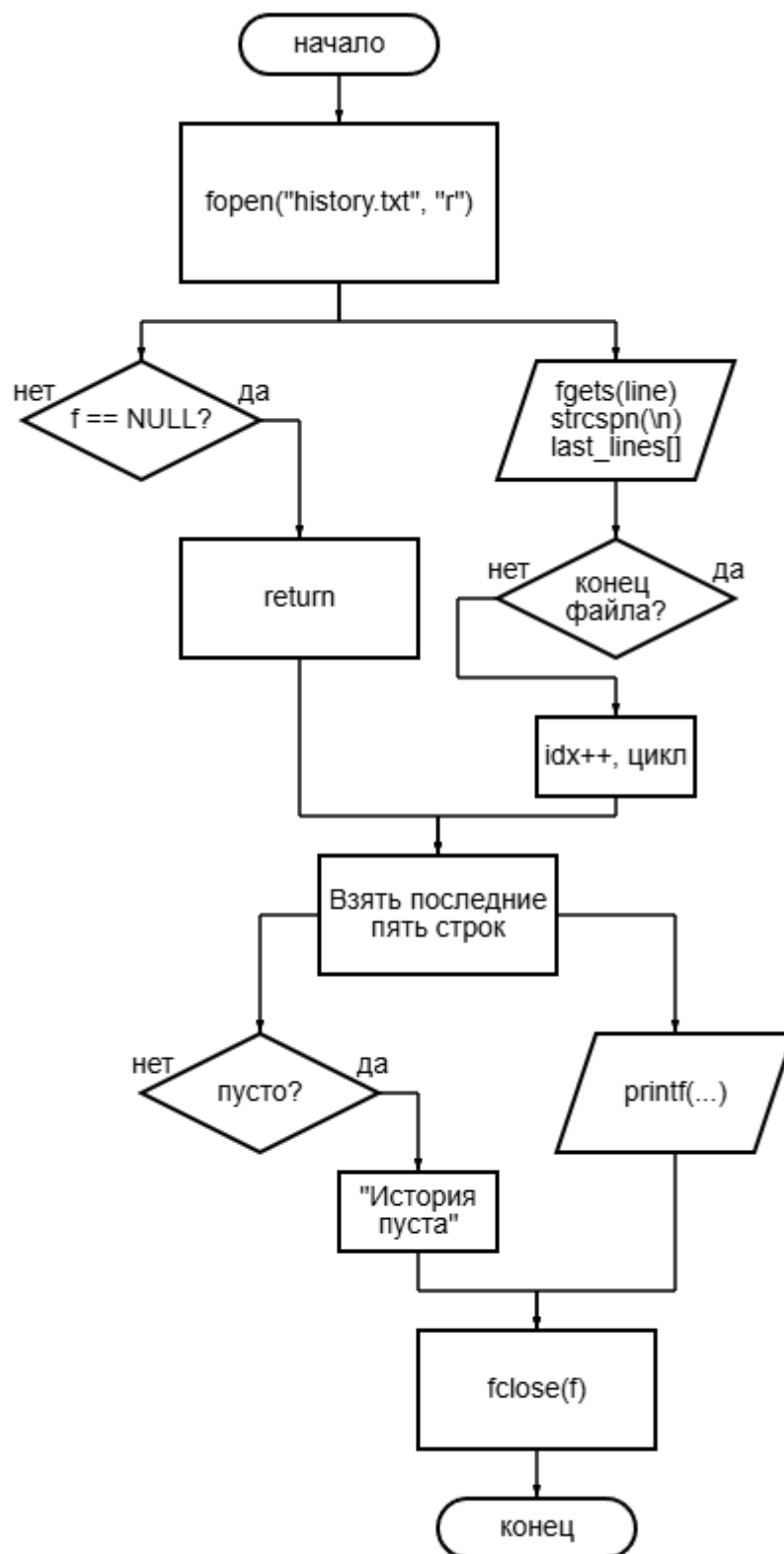


Рисунок - Функция show_recent_history

Схема программы. Функция read_numbers.

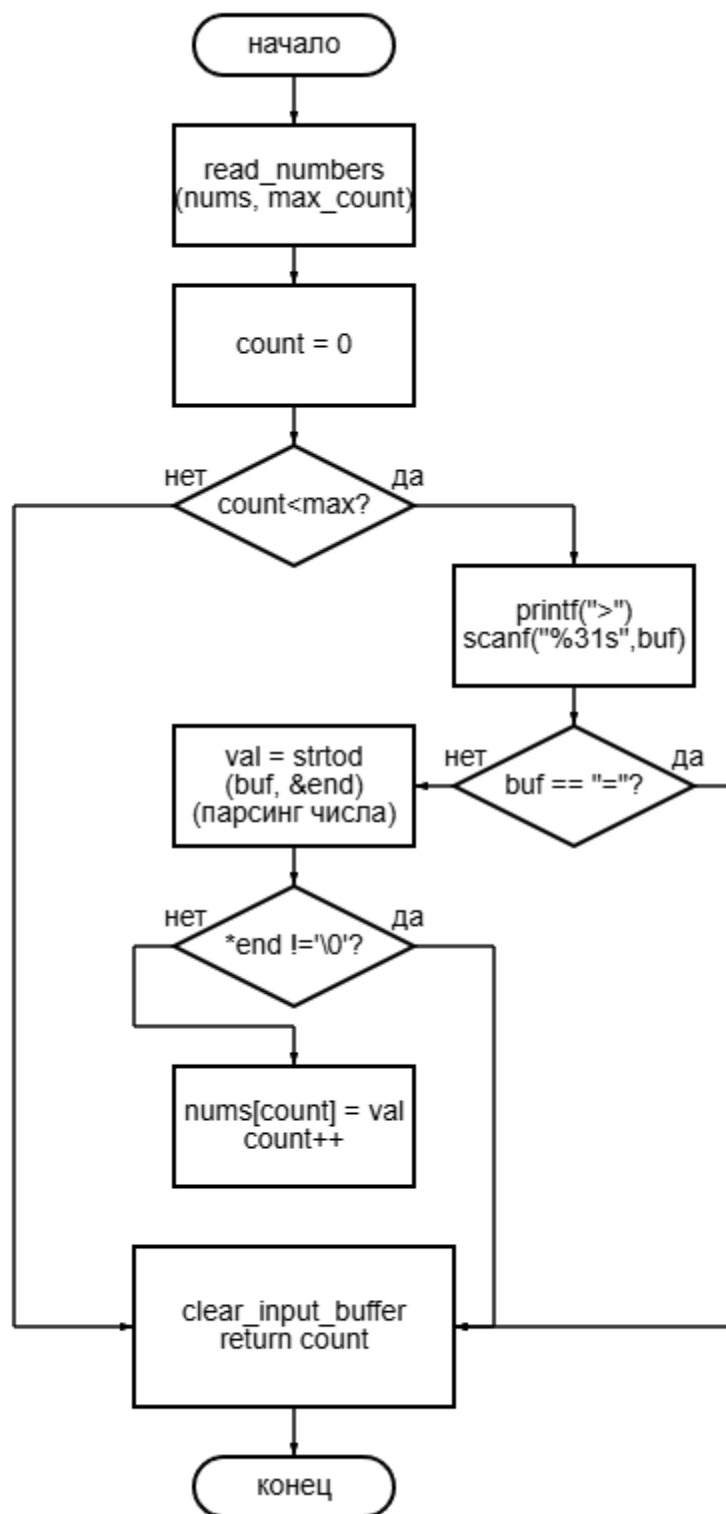


Рисунок - Функция read_numbers

Схема программы. Функция `check_division_by_zero`.

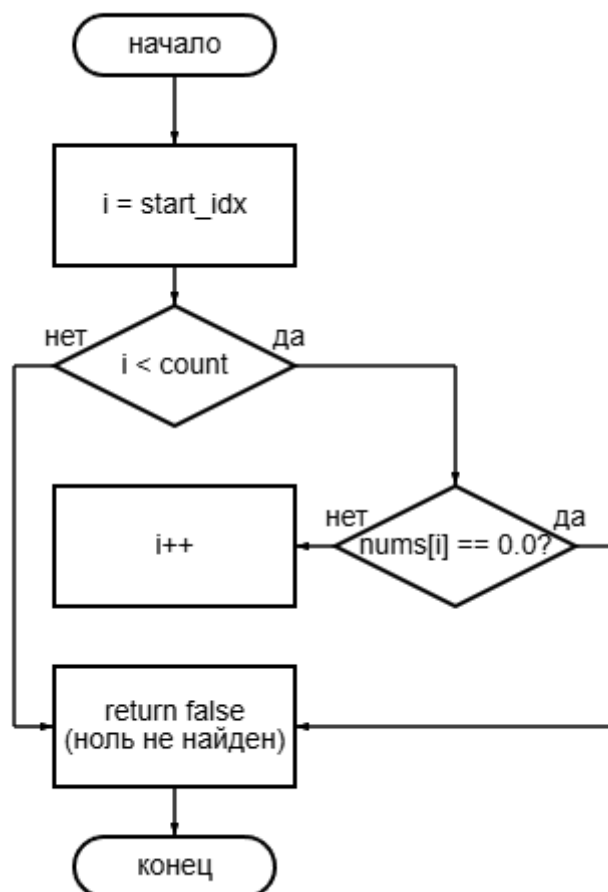


Рисунок - Функция `check_division_by_zero`

Схема программы. Функция `show_splash`.

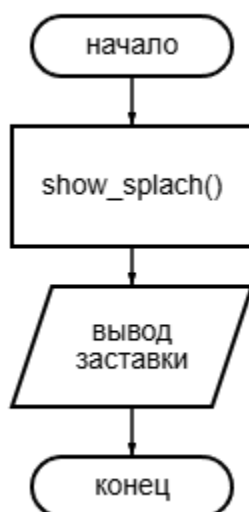


Рисунок - Функция `show_splash`

Схема программы. Функция show_menu.



Рисунок - Функция show_menu

Иерархическая схема программы.

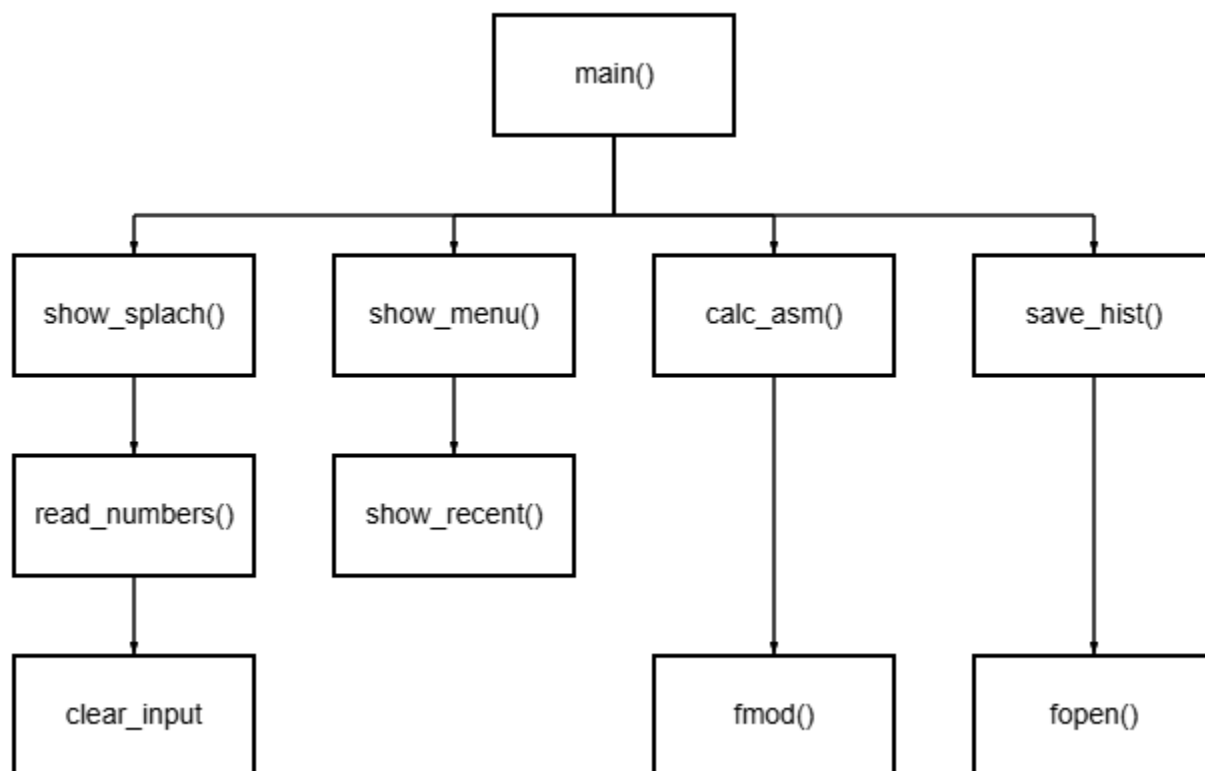


Рисунок – Иерархическая схема.

Руководство пользователя

Программа calculator.exe предназначена для выполнения арифметических операций над вещественными числами с использованием низкоуровневых инструкций процессора. Программа поддерживает сложение, вычитание, умножение, деление и деление с остатком. Результаты сохраняются в файл, история отображается на экране.

Запуск программы:

Запустите calculator.exe из командной строки или через проводник. Откроется консольное окно с заставкой.

Навигация:

Используйте цифры 1–5 для выбора операции, 0 для выхода. Подтверждайте выбор клавишей Enter.

Ввод чисел:

После выбора операции программа предложит вводить числа по одному. Введите число, нажмите Enter. Повторите для всех операндов. Для завершения расчёта введите символ = и нажмите Enter.

```
=====
                        КУРСОВАЯ РАБОТА: КАЛЬКУЛЯТОР
Выполнил студент группы 25BVB3: Махотин Денис Александрович
Принял:  к.т.н., доцент Генералова А.А.
=====

Выберите действие:
 1 - Сложение
 2 - Вычитание
 3 - Умножение
 4 - Деление
 5 - Деление с остатком
 0 - Выход
Ваш выбор:
```

Демонстрация сложения.

```
Выберите действие:
1 - Сложение
2 - Вычитание
3 - Умножение
4 - Деление
5 - Деление с остатком
0 - Выход
Ваш выбор: 1

Вводите числа по одному. Для завершения введите '=':
> 123
> 52134
> =

>>> Результат: 52257
```

Демонстрация вычитания (при этом вычитаем большее число, чем само вычитаемое).

```
Выберите действие:
1 - Сложение
2 - Вычитание
3 - Умножение
4 - Деление
5 - Деление с остатком
0 - Выход
Ваш выбор: 2

Вводите числа по одному. Для завершения введите '=':
> 323
> 404
> =

>>> Результат: -81
```

Демонстрация умножения.

```
Выберите действие:
1 - Сложение
2 - Вычитание
3 - Умножение
4 - Деление
5 - Деление с остатком
0 - Выход
Ваш выбор: 3

Вводите числа по одному. Для завершения введите '=':
> 123
> 321
> =

>>> Результат: 39483
```

Демонстрация деления.

```
Выберите действие:
 1 - Сложение
 2 - Вычитание
 3 - Умножение
 4 - Деление
 5 - Деление с остатком
 0 - Выход
Ваш выбор: 4

Вводите числа по одному. Для завершения введите '=':
> 404
> 3
> =

>>> Результат: 134,667
```

Демонстрация деления на ноль и реакцию самой программы.

```
Выберите действие:
 1 - Сложение
 2 - Вычитание
 3 - Умножение
 4 - Деление
 5 - Деление с остатком
 0 - Выход
Ваш выбор: 4

Вводите числа по одному. Для завершения введите '=':
> 404
> 0
> =

[ОШИБКА] Деление на ноль!
```

Демонстрация деления с остатком.

```
C:\Users\dench\source\repos\kursach2\Debug\kursach2.exe
 2 - Вычитание
 3 - Умножение
 4 - Деление
 5 - Деление с остатком
 0 - Выход
Ваш выбор: 5

Вводите числа по одному. Для завершения введите '=':
> 12
> 5
> =

>>> Частное: 2
>>> Остаток: 2
```

Демонстрация умножения нескольких операндов.

Выберите действие:

- 1 - Сложение
- 2 - Вычитание
- 3 - Умножение
- 4 - Деление
- 5 - Деление с остатком
- 0 - Выход

Ваш выбор: 3

Вводите числа по одному. Для завершения введите '=':

> 2
> 6
> 12
> 24
> 48
> =

>>> Результат: 165888

Демонстрация истории ответов.

> 0
> 12
> 24
> 48
> =

>>> Результат: 165888

--- История последних действий ---

| Ответ: -81 | Ответ: 39483 | Ответ: 134,667 | Ответ: 2 | Ответ: 165888

Заключение

При выполнении данной курсовой работы были получены навыки разработки многомодульных программ. Была освоена работа с файлами, изучены функции работы с консолью и оперативной памятью. Также были получены основные навыки отладки и тестирования программ и программирования в среде Visual Studio 2019 на языках Си/C++ и Assembler.

В рамках курсовой работы была написана программа для работы с расчетом арифметических действий.

В дальнейшем программу можно улучшить, путём добавления графического интерфейса, что позволит упростить пользование данной программой.

Список используемых источников

1. Керниган Б. Ритчи Д. Язык программирования С. 1985 г.
2. MSDN.
3. Прата С. Язык программирования C++. 2019 г.
4. А.А. Тюгашев. Языки программирования. Учебное пособие. 2018 г.
5. Руслан Аблязов. Программирование на ассемблере на платформе x86-64. 2016 г.

Приложение А

Листинги программы

Проект calculator.cpp

Файл main.c

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <locale.h>
#include "io.h"
#include "calc_asm.h"

int main(void) {
    setlocale(LC_ALL, "Russian");
    system("chcp 1251 > nul");
    show_splash();

    double nums[MAX_NUMS]; int choice;
    while (1) {
        show_menu();
        if (scanf("%d", &choice) != 1) {
            printf("Ошибка: введите цифру от 0 до 5.\n");
            clear_input_buffer(); continue;
        }
        clear_input_buffer();
        if (choice == 0) break;
        if (choice < 1 || choice > 5) { printf("Неверный пункт меню.\n\n");
        continue; }

        int count = read_numbers(nums, MAX_NUMS);
        if (count < 2) { printf("Ошибка: нужно минимум 2 числа.\n\n");
        continue; }

        if (choice == 4 || choice == 5) {
            if (check_division_by_zero(nums, count, 1)) {
                printf("\n[ОШИБКА] Деление на ноль!\n\n"); continue;
            }
        }

        double res = nums[0];
        char op = (choice == 5) ? '/' : "+-*/"[choice - 1];
        for (int i = 1; i < count; i++) res = calc_asm(res, nums[i], op);

        if (choice == 5) {
            double rem = fmod(nums[0], nums[1]);
            for (int i = 2; i < count; i++) rem = fmod(rem, nums[i]);
            double quotient = floor(res);
```

```

        printf("\n>>> Частное: %g\n>>> Остаток: %g\n", quotient, rem);
        save_history("Деление с ост.", nums, count, quotient, rem, 1);
    }
    else {
        const char* names[] = { "Сложение", "Вычитание", "Умножение",
"Деление" };
        printf("\n>>> Результат: %g\n", res);
        save_history(names[choice - 1], nums, count, res, 0.0, 0);
    }
    show_recent_history();
}
printf("Программа завершена.\n");
return 0;

}

```

Файл calc_asm.c

```

#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include "calc_asm.h"

double calc_asm(double a, double b, char op) {
    double res = 0.0;

    switch (op) {
        case '+':
            __asm {
                fld a
                fadd b
                fstp res
            }
            break;

        case '-':
            __asm {
                fld a
                fsub b
                fstp res
            }
            break;

        case '*':
            __asm {
                fld a
                fmul b
                fstp res
            }
            break;

        case '/':
            __asm {

```

```

        fld a
        fdiv b
        fstp res
    }
    break;

default:
    printf("Неизвестная операция!\n");
    break;
}

return res;

}

```

Файл io.c

```

#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include "io.h"

void show_splash(void) {

printf("=====\n");
    printf("                КУРСОВАЯ РАБОТА: КАЛЬКУЛЯТОР                \n");
    printf("    Выполнил студент группы 25ВВВ3: Махотин Денис Александрович\n");
    printf("                Принял:  к.т.н., доцент Генералова А.А.                \n");

printf("=====\n\n");
}

void show_menu(void) {
    printf("Выберите действие:\n");
    printf("  1 - Сложение\n");
    printf("  2 - Вычитание\n");
    printf("  3 - Умножение\n");
    printf("  4 - Деление\n");
    printf("  5 - Деление с остатком\n");
    printf("  0 - Выход\n");
    printf("Ваш выбор: ");
}

void clear_input_buffer(void) {
    int c; while ((c = getchar()) != '\n' && c != EOF);
}

int read_numbers(double* nums, int max_count) {
    char buf[32]; int count = 0;

```

```

printf("\nВводите числа по одному. Для завершения введите '=':\n");
while (count < max_count) {
    printf("> ");
    if (scanf("%31s", buf) != 1) break;
    if (strcmp(buf, "=") == 0) break;
    char* end; double val = strtod(buf, &end);
    if (*end != '\0') { printf("  [!] Некорректный ввод.\n"); continue; }
    nums[count++] = val;
}
clear_input_buffer();
return count;
}

bool check_division_by_zero(double* nums, int count, int start_idx) {
    for (int i = start_idx; i < count; i++) if (nums[i] == 0.0) return true;
    return false;
}

void save_history(const char* op_name, double* vals, int count, double res,
double rem, int has_rem) {
    FILE* f = fopen(HISTORY_FILE, "a");
    if (!f) { printf("Ошибка открытия history.txt\n"); return; }
    fprintf(f, "[%s] Операнды: ", op_name);
    for (int i = 0; i < count; i++) fprintf(f, "%g ", vals[i]);
    fprintf(f, "| Ответ: %g", res);
    if (has_rem) fprintf(f, " | Остаток: %g", rem);
    fprintf(f, "\n");
    fclose(f);
}

void show_recent_history(void) {
    FILE* f = fopen(HISTORY_FILE, "r");
    if (!f) return;
    printf("\n--- История последних действий ---\n");
    char line[MAX_LINE], output[MAX_OUTPUT] = "", last_lines[5][MAX_LINE];
    int idx = 0;
    while (fgets(line, sizeof(line), f)) {
        line[strcspn(line, "\n")] = '\0';
        strcpy(last_lines[idx % 5], line); idx++;
    }
    fclose(f);
    int start = (idx < 5) ? 0 : idx - 5;
    int total = (idx < 5) ? idx : 5;
    for (int i = 0; i < total; i++) {
        int line_idx = (start + i) % 5;
        char* res_ptr = strstr(last_lines[line_idx], "| Ответ:");
        if (res_ptr) {
            char temp[256] = "";
            char* rem_ptr = strstr(res_ptr, "| Остаток:");
            if (rem_ptr) { strncpy(temp, res_ptr, rem_ptr - res_ptr);
            temp[rem_ptr - res_ptr] = '\0'; }
            else strcpy(temp, res_ptr);

```

```

        if (strlen(output) + strlen(temp) + 3 < sizeof(output)) {
            if (strlen(output) > 0) strcat(output, " ");
            strcat(output, temp);
        }
    }
}
printf("%s\n", strlen(output) > 0 ? output : "История пуста");
printf("-----\n\n");

}

```

Файл consts.h

```

#ifndef CONSTS_H
#define CONSTS_H

#define MAX_NUMS    500
#define MAX_BUF     32
#define MAX_LINE    512
#define MAX_OUTPUT  1024
#define HISTORY_FILE "history.txt"

#endif

```

Файл io.h

```

#ifndef IO_H
#define IO_H

#include <stdbool.h>
#include "consts.h"

void show_splash(void);
void show_menu(void);
int read_numbers(double* nums, int max_count);
void save_history(const char* op_name, double* vals, int count, double
res, double rem, int has_rem);
void show_recent_history(void);
void clear_input_buffer(void);
bool check_division_by_zero(double* nums, int count, int start_idx);

#endif

```

Файл calc_asm.h

```

#ifndef CALC_ASM_H
#define CALC_ASM_H
double calc_asm(double a, double b, char op);

#endif

```

Приложение В

Результаты работы программы

Приложение В.1 – Информационная заставка.

```
=====
                КУРСОВАЯ РАБОТА: КАЛЬКУЛЯТОР
Выполнил студент группы 25ВВВ3: Махотин Денис Александрович
Принял: к.т.н., доцент Генералова А.А.
=====
```

Приложение В.2 – Главное меню.

```
Выберите действие:
1 - Сложение
2 - Вычитание
3 - Умножение
4 - Деление
5 - Деление с остатком
0 - Выход
Ваш выбор:
```

Приложение С

Результат работы программы

Файл history.txt

```
[Вычитание] Операнды: 321 123 | Ответ: 198
[Деление с ост.] Операнды: 123 52 | Ответ: 2 | Остаток: 19
[Сложение] Операнды: 123 -123 | Ответ: 0
[Деление] Операнды: 312 12 | Ответ: 26
[Деление] Операнды: 5 4 | Ответ: 1,25
[Умножение] Операнды: 123 213 | Ответ: 26199
[Умножение] Операнды: 123 123 123 | Ответ: 1,86087e+06
[Деление] Операнды: 1234 123 123 | Ответ: 0,0815652
[Деление с ост.] Операнды: 1234 12 | Ответ: 102 | Остаток: 10
```