

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
Пензенский государственный университет
Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К курсовой работе
по курсу «Программирование»
на тему «Инженерный калькулятор»

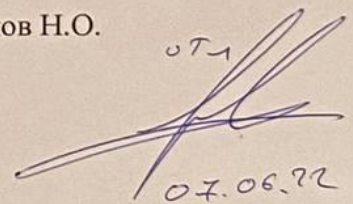
Выполнил:

студент группы 21ВВ2
Брянцев А.Г.

Принял:

Голотенков Н.О.

07.06.22



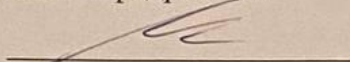
Пенза 2022

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет Вычислительной техники

Кафедра "Вычислительная техника"

"УТВЕРЖДАЮ"

Зав. кафедрой ВТ

 М.А. Митрохин

«___» _____ 20__ г

ЗАДАНИЕ

на курсовое проектирование по курсу

Программирование

Студенту Бряжнев Артему Тердовичу Группа 21ВВ2

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

Исходные данные (технические требования) на проектирование

Приложение: *интерактивный калькулятор*
Функции приложения: *вычисление арифметических действий (сложение, вычитание, умножение, деление), возведение в степень, извлечение квадратного корня, вычисление степенных тригонометрических функций, сохранение истории вычислений в файл, возможность вывода истории вычислений на экран.*
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 "22" февраля

Дата защиты проекта "07" 06. 2022

Руководитель Голотенков Н.О.

Задание получил "22" февраля 2022г.

Студент Францев Артём Терзевич

Содержание

1. Введение.....	2
2. <u>Постановка задачи</u>	4
3. <u>Выбор решения</u>	5
4. <u>Описание разработки программы</u>	7
5. <u>Отладка и тестирование</u>	9
6. <u>Описание программы. Функции</u>	11
6.1. <u>Иерархическая структура</u>	12
6.2. <u>Разработка функций на языке Assembler</u>	13
7. <u>Руководство пользователя</u>	15
<u>Заключение</u>	21
<u>Список используемых источников</u>	22
<u>Приложение А Листинг программы</u>	23
A.1 Файл « <u>main.cpp</u> »	23
A.2 Файл « <u>calculation.h</u> »	25
A.3 Файл « <u>check.h</u> ».....	26
A.4 Файл « <u>hello.h</u> »	27
A.5Файл « <u>calculation.cpp</u> ».....	28
A.6Файл « <u>check.cpp</u> ».....	53
A.7Файл « <u>hello.cpp</u> »	57
Приложение В.....	58
<u>Схемы функций программы</u>	58
<u>Схемы взаимодействия и данных</u>	65

Введение

Современные компьютеры используются во многих сферах. Для работы с ними программисты пишут много программного обеспечения для самых разных целей. В этом им помогают языки программирования.

Одним из самых универсальных и распространенных является язык СИ. Си – это мощный язык с самыми разными возможностями. Он достаточно прост для понимания при этом на нём можно реализовывать самые разные по сложности программы. В данной работе продемонстрированы базовые возможности языка СИ с применением языка Assembler. Разработка велась в среде Microsoft Visual Studio 2019.

1. Постановка задачи

Необходимо разработать программу «Инженерный калькулятор». Программа должна выполнять базовые функции калькулятора, а также несколько дополнительных, среди которых вычисление тригонометрических функций, а также работу с процентами.

Программа должна предоставить возможность ввода чисел, выбора функции или арифметической операции, наглядного вывода результата на экран. Для реализации этих требований нужно разработать интуитивно понятный интерфейс пользователя, определить названия и сокращения вызываемых функций, определить тип интерфейса, выделить место под результат. Изучить функции для работы с числами. Это необходимо для четкого понимания функциональности программы.

Многомодульность программы. Необходимо поделить программу на логические модули. Это повысит расширяемость программы и локализации ошибок при отладке и тестировании программы.

Использование сложных типов данных – массивов, файлов. Требуется изучить базовые и сложные типы данных. Это необходимо для структуризации данных и удобства обработки данных в коде программы.

Программа должна работать в графическом или текстовом режиме видеоподсистемы. Необходимо определиться с типом интерфейса, с элементами управления.

Устройство ввода-вывода – клавиатура. Требуется однозначно идентифицировать и выполнять те или иные действия в зависимости от действий пользователя. Это необходимо для легкого использования программы.

2. Выбор решения

Определение необходимых модулей программы

Приложение «CalculatorJack» состоит из четырёх модулей:

1. **main.cpp**
2. **check.cpp**
3. **calculation.cpp**
4. **hello.cpp**

Файл **main.cpp** является основным модулем данной курсовой работы, контролирующим управляющие элементы интерфейса и вызывающем различные функции из других файлов. А также содержит функции, необходимые для реализации меню калькулятора. Он связан заголовочными файлами calculation.h, hello.h.

Файл **calculation.cpp** содержит все функции необходимые для реализации вычислений калькулятора. Он связан заголовочным файлами calculation.h и check.h.

Файл **hello.cpp** используется для представления заставки калькулятора.

Файл **check.cpp** используется для проверки вводимых данных и их частичного преобразования

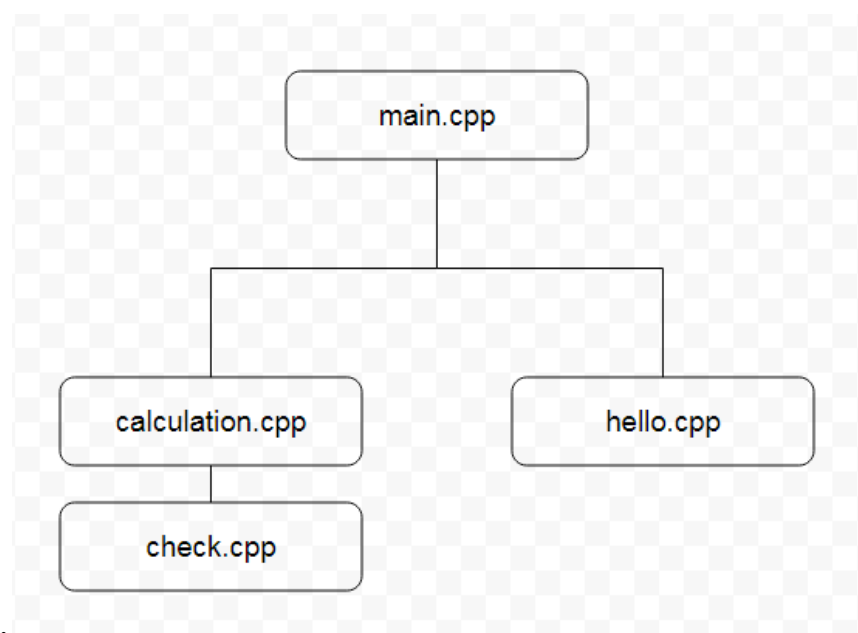


Рис.1 - Структура модулей программы

Всё вышеперечисленное удовлетворяет требованию многомодульности данной курсовой работы. Управление программой осуществляется с помощью клавиатуры. При запуске выводится заставка, после чего программа готова к работе. Программа предназначена для работы со строкой-выражением, строка вводится при помощи клавиатуры, после ввода всех запрашиваемых переменных осуществляется запуск алгоритма проверки строки и последующего ей вычисления.

Навигация по командам реализована через клавиатуру, а именно через клавиши – 1,2,3,4,5,6,7,8,9,0,q,w,e,r,t,y,u,i,o,s,c,h,a.

3. Описание разработки программы

Для написания данной программы был использован язык программирования Си. Этот язык является распространённым языком программирования. При разработке языка Си был принят компромисс между низким уровнем языка ассемблера и высоким уровнем других языков. В языке Си предусмотрено много операций, непосредственно исполняемых большинством микропроцессоров; в то же время он дает программисту возможность выражать свои алгоритмы и данные наиболее подходящими средствами. Язык Си обеспечивает возможности структурирования данных. Он позволяет разрабатывать большие, но структурированные программы, предоставляя возможность отдельной разработки подпрограмм.

Целью создателей языка Си была разработка мобильного языка, который можно было бы использовать для разработки системного программного обеспечения. Программы, написанные на языке высокого уровня, мобильны, поскольку язык должен быть одним и тем же независимо от того, на каком компьютере и в какой операционной системе он используется. Дополнительное свойство языка Си, называемое условной компиляцией, позволяет программисту

изолировать машинно-зависимые операторы и контролировать их компиляцию в другой среде. Это дополнительно повышает мобильность программного обеспечения, написанного на языке Си.

В качестве среды программирования был выбран программный продукт VisualStudio 2019.

Интерфейс калькулятора будет представлен в виде консоли. Результат будет представлен на экране, а также сохранён в документе «history.txt», появляющемся в корневой папке после первого выбора действия. Перемещение по пунктам меню будет осуществляться с помощью клавиш клавиатуры.

При неправильном вводе данных или невозможности их распознавания программой пользователь увидит сообщение о некорректном вводе данных.

Каждый раз после вывода результата на экран программа будет предлагать завершить работу зажатием клавиши **ESC**, либо же нажатием любой другой клавиши очистить консоль и повторно запросить математическое действие.

У пользователя также есть возможность просмотреть историю вычислений нажатием на клавишу 'h' и, при желании, очистить её.

Разработка программы началась с реализации набора функций, необходимых для работы с числами.

Основная программа **main.cpp** выполняется функцией **main**. В функции **main** выполняется запрос на вывод заставки и на запуск цикла обработки событий. Цикл обработки событий реагирует на вводимые с помощью клавиатуры данные и вызывает соответствующую функцию из модуля **calculation.cpp** для обработки события.

Был разработан алгоритм функции **main**. Функция отвечает за инициализацию консоли, за запрос на вывод заставки, за передачу управления в функции обработки меню. Цикл обработки событий главного окна обеспечивает активизацию меню и выход из него. После выхода из функции обработки меню функция обработки событий главного окна получает идентификатор выбранного

пункта меню. В зависимости от идентификатора функция вызывает функцию обработки выбранного пункта меню.

4. Отладка и тестирование

В качестве среды разработки была выбрана программа Visual Studio 2019. Программа предоставляет все средства, необходимые при разработке и отладке модулей и программ. В ходе отладки применялись такие возможности, как точка останова, выполнение кода по шагам, анализ содержимого локальных и глобальных переменных, анализ содержимого памяти.

Тестирование проводилось в рабочем порядке, в процессе разработки, после завершения написания программы.

5. Описание программы. Функции

Приложение **main.cpp** является основным модулем программы. После запуска программы выводится заставка. После программа ожидает выбор действия пользователем, затем управление переходит в функции. После нажатия указанных клавиш функции обрабатываются по таблице 1.

Клавиша	№ функции в переменной «С»	Действия программы
1	(49 ASCII)	Сложение
2	(50 ASCII)	Вычитание
3	(51 ASCII)	Умножение
4	(52 ASCII)	Деление
5	(53 ASCII)	Возведение в квадрат
6	(54 ASCII)	Вычисление квадратного корня
7	(55 ASCII)	Возведение в n-ую степень
8	(56 ASCII)	Вычисление корня n-ой степени
9	(57 ASCII)	Факториал числа
0	(48 ASCII)	Обратное число
f	(102 ASCII)	Вычисление факториала
o	(111 ASCII)	Вычисление натурального логарифма
i	(105 ASCII)	Вычисление десятичного логарифма
u	(117 ASCII)	Вычисление логарифма от числа ($\log_a n$)
s	(115 ASCII)	Вычисление синуса
c	(99 ASCII)	Вычисление косинуса

t	(116 ASCII)	Вычисление тангенса
y	(121 ASCII)	Вычисление котангенса

Действия с процентами:

q	(113 ASCII)	Сложение
w	(119 ASCII)	Вычитание
e	(101 ASCII)	Умножение
r	(114 ASCII)	Деление
a	(97 ASCII)	Найти n% от числа a

Просмотр истории:

h	(104 ASCII)	Просмотр истории вычислений
с (после нажатия h)	(99 ASCII)	Очистить историю вычислений

Таблица 1 - Описание состояний программы

5.1 Иерархическая структура

Иерархическая структура программы представлена на рисунке 2.

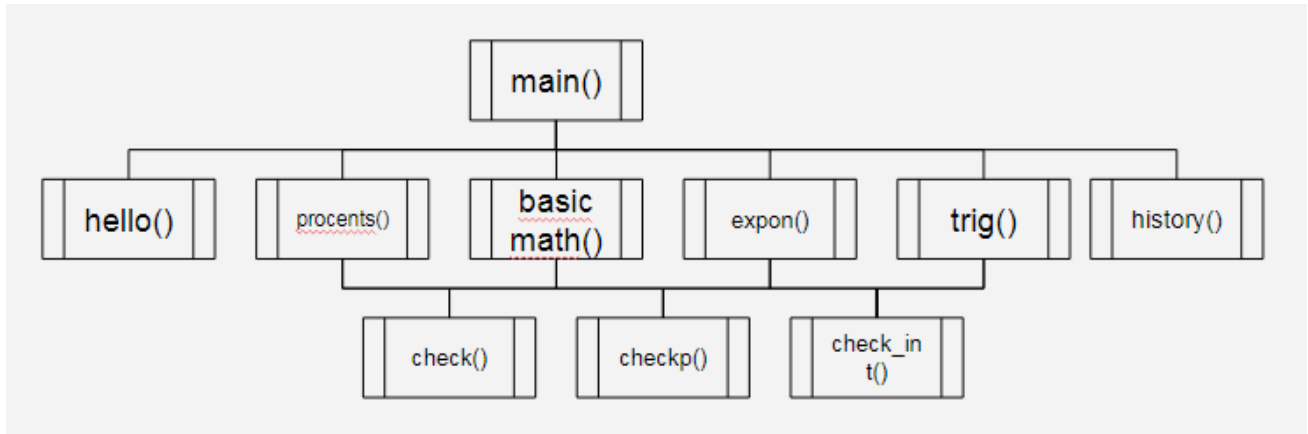


Рис.2 – Иерархическая структура программы

5.2 Разработка функций на языке Assembler

Как известно, функции, реализованные на языке ассемблера, занимают меньше места и работают быстрее. В качестве разрабатываемых функций были выбраны функции для сложения, вычитания, умножения, деления 2-х чисел; отношения единицы к числу, возведения в квадрат и вычисления квадратного корня, возведения в n-ую степень, логарифмов, синуса, косинуса и тангенса.

Описание функций выполнено на языке C, а вычисления (частично, в некоторых случаях приходилось использовать язык СИ) представляют собой ассемблерную вставку. Для реализации необходимо перед такой вставкой прописать ключевое слово «`__asm`», а сам код расположить в фигурных скобках.

Были использованы следующие ассемблерные инструкции:

- команда `mov` используется для копирования значения из приемника в источник;
- команда `call` вызывает процедуру;
- команда `add` складывает содержимое приемника и источника и сохраняет его в последнем;
- команда `cmp` используется для сравнения двух операндов;
- команда `jz` выполняет короткий переход, если первый операнд равен второму операнду при выполнении операции сравнения;
- команда `jmp` выполняет безусловный переход в указанное место;
- команда `dec` производит вычитание "1" из указанного операнда;

А также система команд сопроцессора:

- команда `finit` сбрасывает все управляющие регистры FPU в исходное состояние;
- команда `fld` загружает вещественное число в стек;

- команда `fstp` сохраняет вещественное значение с извлечением из стека в приёмнике;

- команда `fmul` выполняет умножение вещественных чисел источника и приёмника и помещает результат в приёмник;

- команда `fdiv` выполняет деление вещественных чисел (аналогично умножению);

- команда `fsqrt` извлекает квадратный корень;

- команда `fldln2` загружает в стек натуральный логарифм двойки;

- команда `fldlg2` загружает в стек десятичный логарифм двойки;

- команда `fyl2x` вычисляет значение функции $ST(0)=ST(1) \cdot \log_2 ST(0)$;

- команда `fsin` вычисляет синус;

- команда `fcos` вычисляет косинус;

6. Руководство пользователя

Программа **Jack** предназначена для математических расчётов данных. Она имеет понятный интерфейс и поддерживает множество описанных выше операций, а также интуитивно понятное управление.

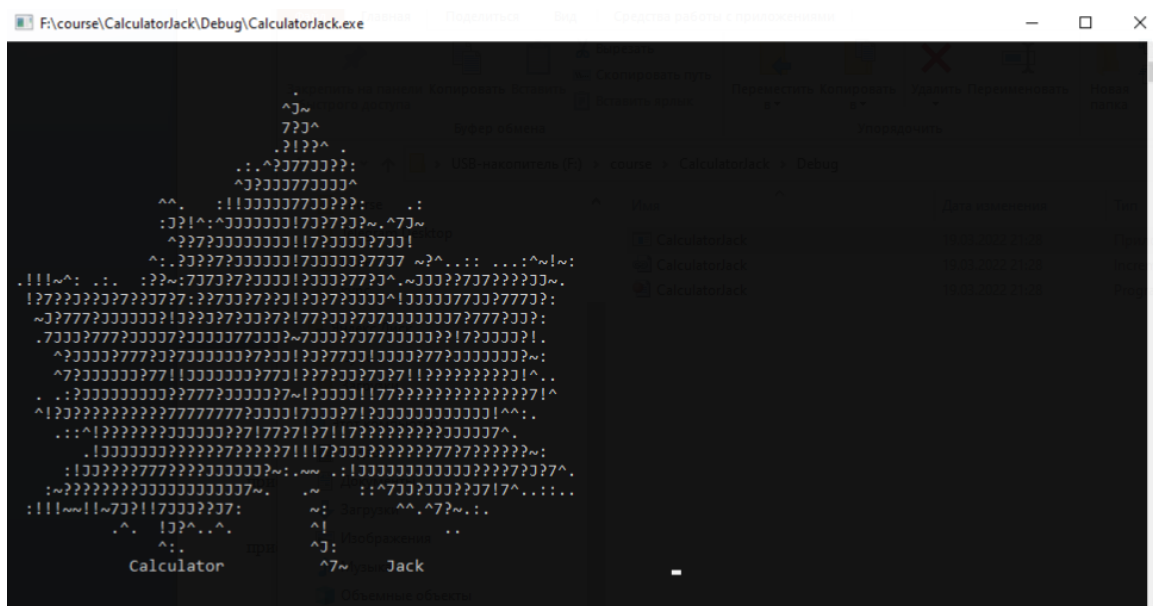


Рисунок 3.1 – заставка

Для выбора действия необходимо нажать на указанную слева клавишу:

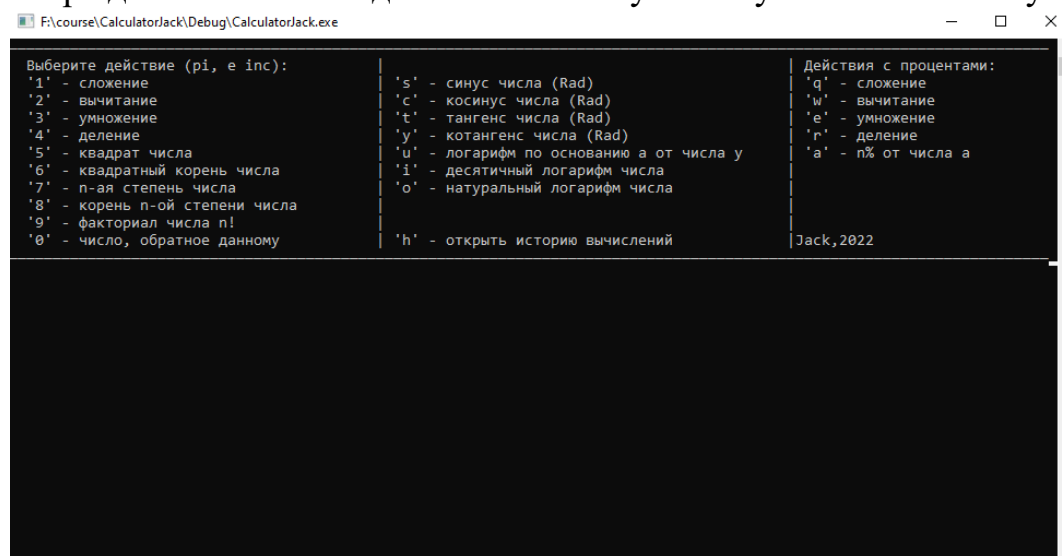


Рисунок 3.2 – окно выбора операций(главное меню)

Запись чисел осуществляется последовательным вводом переменных.

```
F:\course\CalculatorJack\Debug\CalculatorJack.exe

Выберите действие (pi, e inc):
'1' - сложение
'2' - вычитание
'3' - умножение
'4' - деление
'5' - квадрат числа
'6' - квадратный корень числа
'7' - n-ая степень числа
'8' - корень n-ой степени числа
'9' - факториал числа n!
'0' - число, обратное данному

's' - синус числа (Rad)
'c' - косинус числа (Rad)
't' - тангенс числа (Rad)
'y' - котангенс числа (Rad)
'u' - логарифм по основанию a от числа y
'i' - десятичный логарифм числа
'o' - натуральный логарифм числа
'h' - открыть историю вычислений

Деление
a =: █
```

Рисунок 3.3 – ввод

Пример вычисления деления:

```
F:\course\CalculatorJack\Debug\CalculatorJack.exe

Выберите действие (pi, e inc):
'1' - сложение
'2' - вычитание
'3' - умножение
'4' - деление
'5' - квадрат числа
'6' - квадратный корень числа
'7' - n-ая степень числа
'8' - корень n-ой степени числа
'9' - факториал числа n!
'0' - число, обратное данному

's' - синус числа (Rad)
'c' - косинус числа (Rad)
't' - тангенс числа (Rad)
'y' - котангенс числа (Rad)
'u' - логарифм по основанию a от числа y
'i' - десятичный логарифм числа
'o' - натуральный логарифм числа
'h' - открыть историю вычислений

Деление
a =: 55
b =: 11

55 / 11 =: 5

Для завершения нажмите ESC
Для нового вычисления нажмите другую клавишу: █
```

Рисунок 3.4 – вычисление деления

В случае ввода некорректных символов программа укажет на это:

```
Вычитание
a =: ф

Обнаружен недопустимый символ: ф
Повторите ввод
b =: █
```

Рисунок 3.5 – некорректный ввод

```
Число не должно превышать 18 разрядов
Повторите ввод
```

Рисунок 3.6 – предупреждение при вводе чрезмерно больших значений

'3' - умножение	't' - т
'4' - деление	'y' - к
'5' - квадрат числа	'u' - л
'6' - квадратный корень числа	'i' - д
'7' - n-ая степень числа	'o' - н
'8' - корень n-ой степени числа	
'9' - факториал числа n!	
'0' - число, обратное данному	'h' - о


```
Деление
a =: 5
b =: 0

Error! Div by zero.

Для завершения зажмите ESC
Для нового вычисления нажмите другую клавишу:
```

Рисунок 3.7 – ошибка при делении на ноль/другом некорректном вычислении

Пример

корректного

ВВОДА:

```
Логарифм по основанию a от числа y:

a =: 2
y =: 8

log2 (1*8) = 3

Для завершения зажмите ESC
Для нового вычисления нажмите другую клавишу: _
```

Рисунок 3.8 – Вычисление логарифма от числа ($\log_a n$)

В программе также возможно использование чисел e , π (с целыми коэффициентами перед ними)

```
Синус числа (радианы):  
  
a =: pi  
  
sin (1*pi) = 0,000000  
  
Для завершения зажмите ESC  
Для нового вычисления нажмите другую клавишу: _
```

Рисунок 3.19 – использование числа π

Историю вычислений можно посмотреть в файле math_history.doc, который появится в корневой папке или при нажатии 'h' в главном меню.

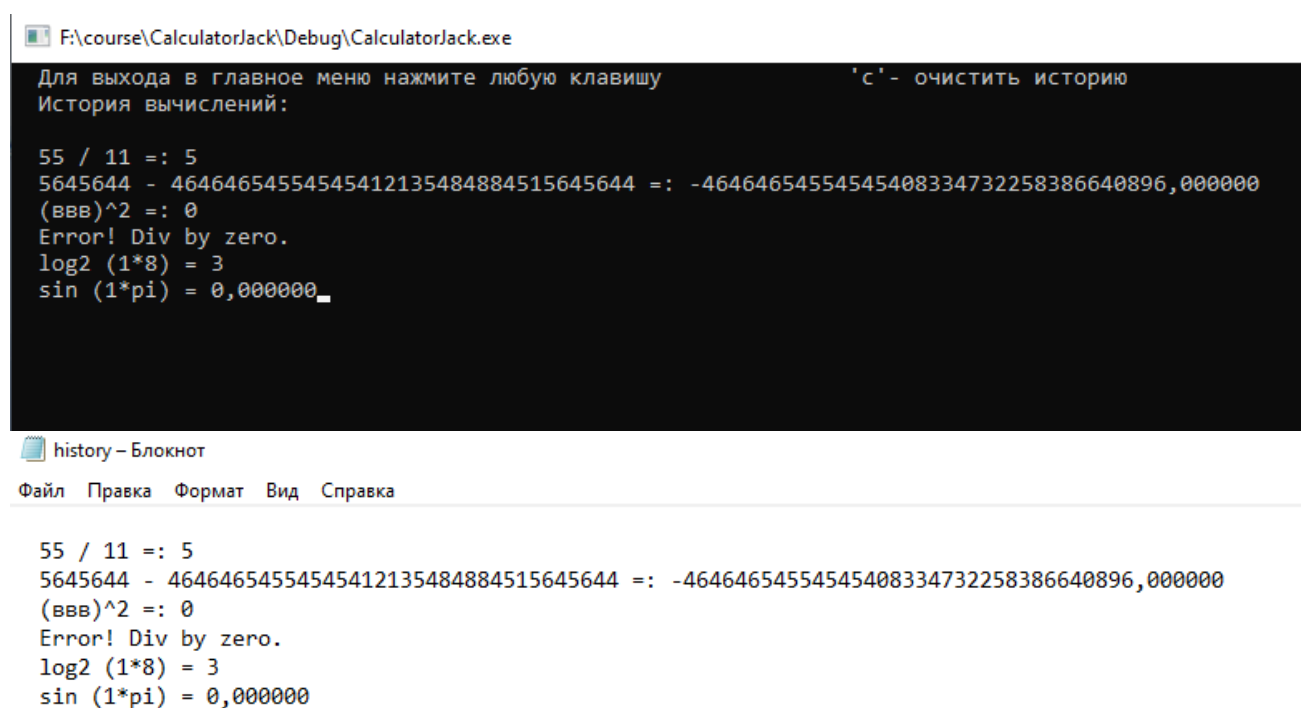


Рисунок 3.10 – история вычислений

Заключение

При выполнении данной курсовой работы были получены навыки разработки многомодульных программ. Были освоены способы обработки событий от клавиатуры. Усвоены принципы реализации интерфейса. Получены базовые навыки программирования на языке С и Ассемблер. Изучены возможности среды программирования VisualStudio 2017. Получены навыки отладки и тестирования программ.

В рамках выполнения курсовой работы была написана программа, выполняющая функции калькулятора. Программа не требует большого количества системных ресурсов, обеспечивает быструю обработку данных и мгновенный показ результата пользователю.

Программу можно улучшить, оптимизировав код для более быстрого выполнения функций.

Список используемых источников

1. Н.Ф. Богаченко, Д.Н. Лавров, Ю.С. Ракицкий -
Ассемблер в примерах и задачах
2. info-master.su/programming – Поляков А.В. – Основы программирования.
3. Х. Дейтел. П. Дейтел. Как программировать на С. 2007 г.
4. prog-cpp.ru - Основы программирования на С.

Приложение Листинг программы

A

Приложение

A.1

Файл «main.cpp»

```
#pragma once

#include"calculation.h"
#include"hello.h"
#define _CRT_SECURE_NO_WARNINGS
char c = 0;
char menu[1000];
int privet = 0;
void main()
{
    setlocale(LC_ALL, "RUS");
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    if (privet == 0)
        hello();
    privet = 1;

    FILE* file;
    FILE* mainmenu;
    mainmenu = fopen("../..../menu.txt", "r");
    while (fgets(menu, 1000, mainmenu))
    {
        printf("%s", menu);
    }
    c = _getch();

    if (c == 104)
    {
        history();
        system("cls");
        main();
    }

    __asm {
        mov bl, c
        call procents
    }
```

```

        call basicmath
        call expon
        call trigon
    }

    printf(" \n\n  Для завершения зажмите ESC \n");
    printf("    Для нового вычисления нажмите другую клавишу:
");
    c = _getch();

    if (c != 27) {

        system("cls");
        _asm {

            call main

        }
    }

    if ((file = fopen("../../history.txt", "a")) != 0) {
        fprintf(file, "\n");
        fclose(file);
    }
    getch();
}

```

Приложение

Файл «calculation.h»

A.2

```
#pragma once
void basicmath();
void expon();
void procents();
void trigon();
void history();

#include<stdlib.h>
#include<stdio.h>
#include<conio.h>
#include<locale.h>
#include<windows.h>
#include < math.h >
#include <wctype.h>
#include < wchar.h >
#define e 2.718281828459523
#define pi 3.141592653589793238462643383279
```

Приложение А.3

Файл «check.h»

```
#pragma once
int check(char vvod[999]);
int check_int(char vvod[999]);
int checkp(char vvod[999]);
int procentscheck(char vvod[999]);
```

Приложение Файл «hello.h»

A.4

```
#pragma once  
void hello();
```

Приложение

Файл «calculation.cpp»

A.5

```
#pragma once // защита от повторного включения
#include "calculation.h"
#include "check.h"

void procents()
{
    double a = 0, b = 0, y = 0, n2 = 0;
    long y_long = 0;
    long i = 0, n = 0, z = 0, g = 0;
    char c = 0, s;
    char vvod[99], str1[20], str2[20], aa[99], bb[99];
    int ef, kuf = 0;
    FILE* file;

    if ((file = fopen("../../history.txt", "r")) == 0)
        file = fopen("../../history.txt", "w");

    else
        file = fopen("../../history.txt", "a");
    _asm {
        mov c, bl
    }
    if (c == 113 || c == 119 || c == 101 || c == 114)
    {
        if (c == 113)
            printf("\n Сложение с процентами");
        if (c == 119)
            printf("\n Вычитание с процентами");
        if (c == 101)
            printf("\n Умножение с процентами");
        if (c == 114)
            printf("\n Деление с процентами");
        scan_113: printf("\n a =: ");
        scanf("%s", vvod);
        i = checkp(vvod);
        if (i == 1)
        {
            goto scan_113;
            i--;
        }
        //
        i = checkp(vvod);
        if (i == 1)
```

```

    {
        z++;
        strcpy(str1, vvod);
        a = atof(vvod);
    }
else
{
    a = atof(vvod);
    strcpy(str1, vvod);
}
//
scan_1131: printf("  b =: ");
scanf("%s", vvod);
i = checkp(vvod);
if (i == 1)
    goto scan_1131;

i = checkp(vvod);
if (i == 1)
{
    z = z + 2;
}
if (z == 0)
{
    b = atof(vvod);
    strcpy(str2, vvod);
}
if (z == 1)
{
    b = atof(vvod);
    a = b / 100 * a;
    strcpy(str2, vvod);
}
if (z == 2)
{
    b = atof(vvod);
    b = a / 100 * b;
    strcpy(str2, vvod);
}
if (z == 3)
{
    a = a / 100;
    b = atof(vvod) / 100;
    strcpy(str2, vvod);
}
_asm {
    finit

```

```

        fld a
        fld b
    }
    if (c == 113)
    {
        _asm {
            fadd
        }
        s = '+';
    }
    if (c == 119)
    {
        _asm {
            fsub
        }
        s = '-';
    }
    if (c == 101)
    {
        _asm {
            fmul
        }
        s = '*';
    }
    if (c == 114)
    {
        _asm {
            fdiv
        }
        s = '/';
        if (b == 0)
        {
            kuf = 5;
            goto error;
        }
    }
    _asm {
        fstp y
    }
    y_long = y;
    if ((y - y_long) == 0)
    {
        printf("\n    %s %c %s =: %ld ", str1, s, str2,
y_long);
        fprintf(file, "\n    %s %c %s =: %ld ", str1, s,
str2, y_long);

```

```

    }
    else
    {
        printf("\n  %s %c %s =: %lf ", str1, s, str2, y);
        fprintf(file, "\n  %s %c %s =: %lf ", str1, s,
str2, y);
    }
    error:
    if (kuf == 5)
    {
        printf("\n  Error! Div by zero.");
        fprintf(file, "\n  Error! Div by zero.");
    }
}
if (c == 97)
{
    printf("\n  Найти n-процентов от числа a");
    scan_1: printf("\n  n =: ");
    scanf("%s", vvod);
    i = check(vvod);
    if (i == INFINITY)
        goto scan_1;
    strcpy(str1, vvod);
    n2 = atof(vvod) / 100;
    scan_2: printf("\n  a =: ");
    scanf("%s", vvod);
    i = check(vvod);
    if (i == INFINITY)
        goto scan_2;
    strcpy(str2, vvod);
    a = atof(vvod);
    y_long = a;
    _asm
    {
        finit
        fld n2
        fld a
        fmul
        fstp y
    }
    y_long = y;
    if ((y - y_long) == 0)
    {
        printf("\n      %s%% (%s) =: %ld ", str1, str2,
y_long);
        fprintf(file, "\n  %s%% %c %s =: %ld ", str1, str2,
y_long);
    }
}

```

```

    }
    else
    {
        printf("\n  %s%% (%s) =: %lf ", str1, str2, y);
        fprintf(file, "\n  %s%% (%s) =: %lf ", str1, str2,
y);
    }
}
else
{
    _asm{mov bl, c}
}
fclose(file);
}

```

```

void basicmath()
{
    double a = 0, b = 0, y = 0, t = 0;
    long y_long = 0; // проверка результата на целочисленность
    long i = 0; // флаг неправильного ввода
    char c = 0, s;
    char vvod[99], str1[20], str2[20];
    int ef = 0, pf = 0, kuf = 0, k = 0, i1 = 1, i2 = 1;
    FILE *file;

    speciala:
    if (i != 0)
    {
        for (int i1 = 0; i1 < 19; i1++)
        {
            if (vvod[i1] == 'e')
            {
                ef = 1;
                break;
            }
        }
        for (int i1 = 0; i1 < 19; i1++)
        {
            if (vvod[i1] == 'p' && vvod[i1 + 1] == 'i')
            {
                pf = 1;
                break;
            }
        }
        if (ef == 1)
        {
            a = e * i;

```

```

        i1 = i;
        vvod[0] = 'e';
        vvod[1] = '\0';
        strcpy(str1, vvod);
        kuf = 1;
        ef = 0;
        goto readya;
    }
    if (pf == 1)
    {
        a = i * pi;
        i1 = i;
        vvod[0] = 'p';
        vvod[1] = 'i';
        vvod[2] = '\0';
        strcpy(str1, vvod);
        kuf = 1;
        pf = 0;
        goto readya;
    }
    if ((pf != 1) && (ef != 1))
        goto scan_1;
}
specialb:
if (i != 0)
{
    for (int i1 = 0; i1 < 19; i1++)
    {
        if (vvod[i1] == 'e')
        {
            ef = 1;
            break;
        }
    }
    for (int i1 = 0; i1 < 19; i1++)
    {
        if (vvod[i1] == 'p' && vvod[i1 + 1] == 'i')
        {
            pf = 1;
            break;
        }
    }
    if (ef == 1)
    {
        b = e * i;
        i2 = i;
        vvod[0] = 'e';

```

```

        vvod[1] = '\0';
        strcpy(str2, vvod);
        kuf = 1;
        ef = 0;
        goto readyb;
    }
    if (pf == 1)
    {
        b = i * pi;
        i2 = i;
        vvod[0] = 'p';
        vvod[1] = 'i';
        vvod[2] = '\0';
        strcpy(str2, vvod);
        kuf = 1;
        pf = 0;
        goto readyb;
    }
    if ((pf != 1) && (ef != 1))
        goto scan_2;
}
else if (k == 1) goto jump1;
else if (k == 2) goto jump2;

    if((file = fopen("../../history.txt", "r")) == 0 ) // если
файла не существует, то создаём его
        file = fopen("../../history.txt", "w");

    else
        file = fopen("../../history.txt", "a"); // иначе
дописываем

    _asm {
        mov c, bl
    }

    if ((c == 49) || (c == 50) || (c == 51) || (c == 52) || (c
== 48))
    {
        k++;
        if (c == 49)
            printf("\n Сложение ");
        if (c == 50)
            printf("\n Вычитание ");
        if (c == 51)
            printf("\n Умножение ");
        if (c == 52)

```

```

        printf("\n Деление ");
if (c == 48)
    printf("\n Число, обратное данному");
if (c == 48)
{
    a = 1;
    str1[0] = '1';
    str1[1] = '\0';
    goto f48;
}
scan_1: printf("\n a =: ");
scanf("%s", vvod);
i = check(vvod);
double finitel(i);

if ((i != 0) || (finitel == 0))
    goto speciala;
jump1:

strcpy(str1, vvod);
a = atof(vvod);

f48:
scan_2: readya: k++; printf(" b =: ");
scanf("%s", vvod);
i = check(vvod);
double finitel2(i);
if ((i != 0) || (finitel2 == 0))
    goto specialb;
jump2:
strcpy(str2, vvod);
b = atof(vvod);
readyb:
if (c == 48)
{
    _asm {
        finit
        fld a
        fld b
        fdiv
    }
    s = '/';
    if (b == 0)
    {
        k = 4;
        goto error;
    }
}

```

```

        goto f40;
    }
    _asm {
        finit
        fld a
        fld b
    }
    if (c == 49)
    {
        _asm {
            fadd
        }
        s = '+';
    }
    if (c == 50)
    {
        _asm {
            fsub
        }
        s = '-';
    }
    if (c == 51)
    {
        _asm {
            fmul
        }
        s = '*';
    }
    if (c == 52)
    {
        _asm {
            fdiv
        }
        s = '/';
        if (a / b == INFINITY)
        {
            k = 4;
            goto error;
        }
    }
}
f40:
_asm {
    fstp y
}
y_long = y;

```

```

        if (kuf = 1)
        {
            if (y_long == y)
            {
                printf("\n %d*%s %c %d*%s = %ld", i1, str1, s,
i2, str2, y_long);
                fprintf(file, "\n %d*%s %c %d*%s = %ld", i1,
str1, i2, s, str2, y_long);
            }
            else
            {
                printf("\n %d*%s %c %d*%s = %lf", i1, str1, s,
i2, str2, y);
                fprintf(file, "\n %d*%s %c %d*%s = %lf", i1,
str1, s, i2, str2, y);
            }
        }
        else
        {
            if ((y - y_long) == 0)
            {
                printf("\n %s %c %s =: %ld ", str1, s, str2,
y_long);
                fprintf(file, "\n %s %c %s =: %ld ", str1, s,
str2, y_long);
            }
            else
            {
                printf("\n %s %c %s =: %lf ", str1, s, str2,
y);
                fprintf(file, "\n %s %c %s =: %lf ", str1, s,
str2, y);
            }
            if (k == 4) {
                error:
                {
                    printf("\n Error! Div by zero.");
                    fprintf(file, "\n Error! Div by zero.");
                }
            }
        }
    }
    else
    {
        _asm {mov bl, c}
    }
    fclose(file);

```

```

}

void expon()
{
    int pf = 0, ef = 0, kuf = 0, k = 0, f55 = 0, i1 = 1, i2 =
1, y1 = 1;
    double a = 0, b = 0, y = 0, n2 = 0;
    long y_long = 0;
    long i = 0, n = 0;
    char c = 0, f[5], s[3];
    char vvod[99], str1[20], str2[20];
    FILE* file;
special2:
    if (i != 0)
    {
        for (int i1 = 0; i1 < 19; i1++)
        {
            if (vvod[i1] == 'e')
            {
                ef = 1;
                break;
            }
        }
        for (int i1 = 0; i1 < 19; i1++)
        {
            if (vvod[i1] == 'p' && vvod[i1 + 1] == 'i')
            {
                pf = 1;
                break;
            }
        }
        if (ef == 1)
        {
            a = e * i;
            vvod[0] = 'e';
            vvod[1] = '\\0';
            strcpy(str1, vvod);
            kuf = 1;
            ef = 0;
            goto readya;
        }
        if (pf == 1)
        {
            a = i * pi;
            vvod[0] = 'p';
            vvod[1] = 'i';
            vvod[2] = '\\0';

```

```

        strcpy(str1, vvod);
        kuf = 1;
        pf = 0;
        goto readya;
    }
    if ((pf != 1) && (ef != 1))
        goto scan_1;
}
else if (k == 1) goto jumpa;

    if ((file = fopen("../../history.txt", "r")) == 0) // если
    файла не существует, то создаём его
        file = fopen("../../history.txt", "w");

    else
        file = fopen("../../history.txt", "a"); // иначе
    дописываем

    _asm {
        mov c, bl
    }
    if ((c == 53) || (c == 54))
    {
        if (c == 53)
            printf("\n Квадрат числа\n ");
        if (c == 54)
            printf("\n Квадратный корень числа\n ");
    scan_1: printf("\n a =: ");
        k++;
        scanf("%s", vvod);
        i = check(vvod);

        double finitel(i);
        if ((i != 0) || (finitel == 0))
            goto special2;
    jumpa:
        strcpy(str1, vvod);
        a = atof(vvod);
    readya:
        _asm {
            finit
            fld a
        }
        if (c == 53)
        {
            _asm {

```

```

        fld a
        fmul a
    }
    f[0] = '\0';
    s[0] = '^';
    s[1] = '2';
    s[2] = '\0';
}
if (c == 54)
{
    _asm {
        fsqrt
    }
    s[0] = '\0';
    f[0] = 's';
    f[1] = 'q';
    f[2] = 'r';
    f[3] = 't';
    f[4] = '\0';
}
_asm
fstp y
    y_long = y;
double finite(y);
if (finite != y)
    goto error2;
if (kuf = 1)
{
    if (y_long == y)
    {
        printf("\n    %s(%d*%s)%s = %ld", f, i, str1, s,
y_long);
        fprintf(file, "\n    %s(%d*%s)%s = %ld", f, i,
str1, s, y_long);
    }
    else
    {
        printf("\n    %s(%d*%s)%s = %lf", f, i, str1, s,
y);
        fprintf(file, "\n    %s(%d*%s)%s = %lf", f, i,
str1, s, y);
    }
}
else
{
    if ((y - y_long) == 0)
    {

```

```

        printf("\n   %s(%s)%s =: %ld ", f, str1, s,
y_long);
        fprintf(file, "\n   %s(%s)%s =: %ld ", f, str1,
s, y_long);
    }
    else
    {
        printf("\n   %s(%s)%s =: %lf ", f, str1, s, y);
        fprintf(file, "\n   %s(%s)%s =: %lf ", f, str1,
s, y);
    }
    error2:
    if (finite != y)
    {
        printf("\n   Error!");
        fprintf(file, "\n   Error!");
    }
}
}

```

```

if ((c == 55) || (c == 56))
{
    specialaa:
    if (i != 0)
    {
        for (int i1 = 0; i1 < 19; i1++)
        {
            if (vvod[i1] == 'e')
            {
                ef = 1;
                break;
            }
        }
        for (int i1 = 0; i1 < 19; i1++)
        {
            if (vvod[i1] == 'p' && vvod[i1 + 1] == 'i')
            {
                pf = 1;
                break;
            }
        }
        if (ef == 1)
        {
            a = e * i;
            i1 = i;
            vvod[0] = 'e';

```

```

        vvod[1] = '\0';
        strcpy(str1, vvod);
        kuf = 1;
        ef = 0;
        goto readyaa;
    }
    if (pf == 1)
    {
        a = i * pi;
        i1 = i;
        vvod[0] = 'p';
        vvod[1] = 'i';
        vvod[2] = '\0';
        strcpy(str1, vvod);
        kuf = 1;
        pf = 0;
        goto readyaa;
    }
    if ((pf != 1) && (ef != 1))
        goto scan_3;
}
specialnn:
    if (i != 0)
    {
        for (int i1 = 0; i1 < 19; i1++)
        {
            if (vvod[i1] == 'e')
            {
                ef = 1;
                break;
            }
        }
        for (int i1 = 0; i1 < 19; i1++)
        {
            if (vvod[i1] == 'p' && vvod[i1 + 1] == 'i')
            {
                pf = 1;
                break;
            }
        }
        if (ef == 1)
        {
            n2 = e * i;
            i2 = i;
            vvod[0] = 'e';
            vvod[1] = '\0';
            strcpy(str2, vvod);

```

```

        kuf = 1;
        ef = 0;
        goto readynn;
    }
    if (pf == 1)
    {
        n2 = i * pi;
        i2 = i;
        vvod[0] = 'p';
        vvod[1] = 'i';
        vvod[2] = '\\0';
        strcpy(str2, vvod);
        kuf = 1;
        pf = 0;
        goto readynn;
    }
    if ((pf != 1) && (ef != 1))
        goto scan_4;
}
else if (k == 1) goto jumpaa;
else if (k == 2) goto jumpnn;
if (c == 55)
    printf("\\n  n-ая степень числа\\n ");
if (c == 56)
    printf("\\n  квадрат n-ой степенени числа\\n ");
scan_3: printf("\\n  a =: ");
scanf("%s", vvod);
k++;
i = check(vvod);

double finitelaa(i);
if ((i != 0) || (finitelaa == 0))
    goto specialaa;
jumpaa:
    strcpy(str1, vvod);
    a = atof(vvod);
readyaa:
    k++;
scan_4: printf("  n =: ");
scanf("%s", vvod);
i = check(vvod);
double finitelnn(i);
if ((i != 0) || (finitelnn == 0))
    goto specialnn;
jumpnn:
    strcpy(str2, vvod);
    n2 = atof(vvod);

```

```

readynn:
    if (c == 55)
    {
        y = pow(a, n2);
        goto f55;
        f55++;
    }
    if (c == 56)
    {
        if (n2 == 0)
        {
            k = 7;
            goto errorn;
        }
        y = pow(a, 1 / n2);
        goto f56;
    }
f55:
    y_long = y;
    if (kuf = 1)
    {
        if (y_long == y)
        {
            printf("\n  pow(%d*%s, (%d*%s))  =:  %ld",  i1,
str1, i2, str2, y_long);
            fprintf(file, "\n  pow(%d*%s, (%d*%s))  =:  %ld",
i1, str1, i2, str2, y_long);
        }
        else
        {
            printf("\n  pow(%d*%s, (%d*%s))  =:  %lf",  i1,
str1, i2, str2, y);
            fprintf(file, "\n  pow(%d*%s, (%d*%s))  =:  %ld",
i1, str1, i2, str2, y);
        }
    }
    else
    {
        if ((y - y_long) == 0)
        {
            printf("\n  pow(%s, (%s))  =:  %ld ", str1, str2,
y_long);
            fprintf(file, "\n  pow(%s, (%s))  =:  %ld ", str1,
str2, y_long);
        }
        else
        {

```

```

        printf("\n  pow(%s,(%s))  =: %lf ", str1, str2,
y);
        fprintf(file, "\n  pow(%s,(%s))  =: %lf ", str1,
str2, y);
    }
    if (f55 != 0)
        f56:
        y_long = y;
        if (kuf = 1)
        {
            if (y_long == y)
            {
                printf("\n  pow(%d*%s,(1/%d*%s))  =: %ld", i1,
str1, i2, str2, y_long);
                fprintf(file, "\n    pow(%d*%s,(1/%d*%s))  =:
%ld", i1, str1, i2, str2, y_long);
            }
            else
            {
                printf("\n  pow(%d*%s,(1/%d*%s))  =: %lf", i1,
str1, i2, str2, y);
                fprintf(file, "\n    pow(%d*%s,(1/%d*%s))  =:
%ld", i1, str1, i2, str2, y);
            }
        }
        else
            if ((y - y_long) == 0)
            {
                printf("\n  pow(%s,(1/%s))  =: %ld ", str1,
str2, y_long);
                fprintf(file, "\n    pow(%s,(1/%s))  =: %ld ",
str1, str2, y_long);
            }
            else
            {
                printf("\n  pow(%s,(1/%s))  =: %lf ", str1,
str2, y);
                fprintf(file, "\n    pow(%s,(1/%s))  =: %lf ",
str1, str2, y);
            }
        }

    if (k == 7)
        errorn:
    {
        printf("\n  Error! Div by zero.");
        fprintf(file, "\n  Error! Div by zero.");

```

```

    }
}

if (c == 57)
{
    int l = 0;
    printf("\n факториал числа n!\n ");
scan_7: printf("  n =: ");
    scanf("%s", vvod);
    i = check_int(vvod);
    if (i != 0)
        goto scan_7;
    strcpy(str1, vvod);

    y = atof(vvod);
    y_long = y;
    if (y_long != y)
    {
        errorf:
            printf("\n  Вводимое число не должно содержать
дробную часть!");
            l++;
    }

    long n = y;
    long n1 = y;
    if (n == 0)
    {
        n = 1;
        goto endl1;
    }
    int big1 = 0;
    if ((n > 16) || (n < -16))
    {
        big1 = 1;
        goto big;
    }
    if ((y < 0) && (l == 0))
    {
        n = abs(atol(vvod));
        _asm
        {
            finit

            mov ebx, n
            mov eax, ebx
            dec ebx

```

```

        START :
        cmp ebx, 0
        jz END
        imul ebx
        dec ebx
        jmp START
        END :
        mov n, eax
    }
    printf("\n (%d)! = -%ld ", n1, n);
    fprintf(file, "\n (%d)! = -%ld ", n1, n);
}
else
    if (l == 0)
    {
        asm
        {
            finit

            mov ebx, n
            mov eax, ebx
            dec ebx
            START1 :
            cmp ebx, 0
            jz END1
            imul ebx
            dec ebx
            jmp START1
            END1 :
            mov n, eax
        }
        endl1:
        printf("\n !(%d) = %ld ", n1, n);
        fprintf(file, "\n !(%d) = %ld ", n1, n);
    }
big:
    if (big1 == 1)
    {
        printf("\n          Конечный      результат      слишком
велик!");
        fprintf(file, "\n      Конечный      результат      слишком
велик!");
    }
}
if ((c == 105) || (c == 111) || (c == 117))
{
    special:

```

```

if (i != 0)
{
    for (int i1 = 0; i1 < 19; i1++)
    {
        if (vvod[i1] == 'e')
        {
            ef = 1;
            break;
        }
    }
    for (int i1 = 0; i1 < 19; i1++)
    {
        if (vvod[i1] == 'p' && vvod[i1 + 1] == 'i')
        {
            pf = 1;
            break;
        }
    }
    if (ef == 1)
    {
        y = e * i;
        i1 = i;
        vvod[0] = 'e';
        vvod[1] = '\\0';
        strcpy(str1, vvod);
        kuf = 1;
        ef = 0;
        goto ready;
    }
    if (pf == 1)
    {
        y = i * pi;
        i1 = i;
        vvod[0] = 'p';
        vvod[1] = 'i';
        vvod[2] = '\\0';
        strcpy(str1, vvod);
        kuf = 1;
        pf = 0;
        goto ready;
    }
    if ((pf != 1) && (ef != 1))
        goto scan_105;
}
speciala:
    if (i != 0)
    {

```

```

for (int i1 = 0; i1 < 19; i1++)
{
    if (vvod[i1] == 'e')
    {
        ef = 1;
        break;
    }
}
for (int i1 = 0; i1 < 19; i1++)
{
    if (vvod[i1] == 'p' && vvod[i1 + 1] == 'i')
    {
        pf = 1;
        break;
    }
}
if (ef == 1)
{
    a = e * i;
    y1 = a;
    i2 = i;
    vvod[0] = 'e';
    vvod[1] = '\\0';
    strcpy(str1, vvod);
    kuf = 1;
    ef = 0;
    goto notlog;
}
if (pf == 1)
{
    a = i * pi;
    y1 = a;
    i2 = i;
    vvod[0] = 'p';
    vvod[1] = 'i';
    vvod[2] = '\\0';
    strcpy(str1, vvod);
    kuf = 1;
    pf = 0;
    goto notlog;
}
if ((pf != 1) && (ef != 1))
    goto scan_105;
}
else if (k == 1) goto jump1;
else if (k == 2) goto jump2;
if (c == 105)

```

```

    {
        printf("\nДесятичный логарифм числа:\n ");
        k++;
        goto notlog;
    }
    if (c == 111)
    {
        printf("\nНатуральный логарифм числа:\n ");
        k++;
        goto notlog;
    }
    if (c == 117)
        printf("\n Логарифм по основанию а от числа
y:\n ");
        k++;
scan_117: printf("\n а =: ");
scanf("%s", vvod);
i = check(vvod);
double finitel2(i);
if ((i != 0) || (finitel2 == 0))
    goto speciala;
jump1:
    strcpy(str1, vvod);
    a = atof(vvod);
notlog:
    k++;
scan_105: printf("\n y =: ");
scanf("%s", vvod);
i = check(vvod);
double finitel(i);
if ((i != 0) || (finitel == 0))
    goto special;
jump2:
    strcpy(str1, vvod);
    y = atof(vvod);
ready:
    f[0] = 'l';
    if (c == 117)
        y1 = a;
    if (c == 105)
    {
        _asm
        {
            FLDLG2
            fld y
            FYL2X
            fstp y

```

```

    }
    f[1] = 'g';
    f[2] = '\\0';
}
if (c == 111)
{
    _asm
    {
        FLDLN2
        fld y
        FYL2X
        fstp y
    }
    f[1] = 'n';
    f[2] = '\\0';
}
if (c == 117)
{
    _asm
    {
        FLDLN2
        fld y
        FYL2X
        fstp y

        FLDLN2
        fld a
        FYL2X
        fst a

        fld y
        fld a
        fdiv
        fst y
    }
    f[1] = 'o';
    f[2] = 'g';
    f[3] = '\\0';
}
y_long = y;
double finite(y);
if (finite != y)
    goto error1;
if (kuf == 1 && c != 117)
{

```

```

        if (y_long == y)
        {
            printf("\n %s (%d*%s) = %ld", f, i1, str1,
y_long);
            fprintf(file, "\n %s (%d*%s) = %ld", f, i1,
str1, y_long);
        }
        else
        {
            printf("\n %s (%d*%s) = %lf", f, i1, str1,
y);
            fprintf(file, "\n %s (%d*%s) = %lf", f, i1,
str1, y);
        }
    }
    if (kuf != 1 && c != 117)
    if (y_long == y)
    {
        printf("\n %s(%s) = %ld", f, str1, y_long);
        fprintf(file, "\n %s(%s) = %ld", f, str1,
y_long);
    }
    else
    {
        printf("\n %s(%s) = %lf", f, str1, y);
        fprintf(file, "\n %s(%s) = %lf", f, str1, y);
    }
error1:
    if (finite != y)
    {
        printf("\n Error!");
        fprintf(file, "\n Error!");
    }
    if (c == 117)
        if (kuf == 1)
        {
            if (y_long == y)
            {
                printf("\n %s%ld(%d*%s) = %ld", f, y1,
i2, str1, y_long);
                fprintf(file, "\n %s%ld(%d*%s) = %ld",
f, y1, i2, str1, y_long);
            }
            else

```

```

        {
            printf("\n %s%lf(%d*%s) = %lf", f, y1,
i2, str1, y);
            fprintf(file, "\n %s%lf(%d*%s) = %lf",
f, y1, i2, str1, y);
        }
    }
    else
        if (y_long == y)
        {
            printf("\n %s%ld (%d*%s) = %ld", f, y1,
i2, str1, y_long);
            fprintf(file, "\n %s%ld (%d*%s) = %ld",
f, y1, i2, str1, y_long);
        }
        else
        {
            printf("\n %s%lf (%d*%s) = %lf", f, y1,
i2, str1, y);
            fprintf(file, "\n %s%lf (%d*%s) = %lf",
f, y1, i2, str1, y);
        }
    }
    else
    {
        _asm {mov bl, c}
    }
    fclose(file);
}

void trigon()
{
    double a = 0, b = 0, y = 0;
    long y_long = 0; // проверка результата на целочисленность
    long i = 0; // флаг неправильного ввода
    char c, s[4];
    char vvod[99], str1[20], str2[20];
    int kuf, ef = 0, pf = 0, k = 0, r = 0;
    FILE* file;
    special:
    if (i != 0)
    {
        for (int i1 = 0; i1 < 19; i1++)
        {
            if (vvod[i1] == 'e')
            {
                ef = 1;

```

```

        break;
    }
}
for (int i1 = 0; i1 < 19; i1++)
{
    if (vvod[i1] == 'p' && vvod[i1 + 1] == 'i')
    {
        pf = 1;
        break;
    }
}
if (ef == 1)
{
    y = e * i;
    vvod[0] = 'e';
    vvod[1] = '\\0';
    strcpy(str1, vvod);
    kuf = 1;
    ef = 0;
    goto ready;
}
if (pf == 1)
{
    y = i * pi;
    vvod[0] = 'p';
    vvod[1] = 'i';
    vvod[2] = '\\0';
    strcpy(str1, vvod);
    kuf = 1;
    pf = 0;
    goto ready;
}
if ((pf != 1) && (ef != 1))
    goto scan_115;
}
else if (k == 1) goto ready;

if ((file = fopen("../../history.txt", "r")) == 0) // если
файла не существует, то создаём его
    file = fopen("../../history.txt", "w");

else
    file = fopen("../../history.txt", "a"); // иначе
дописываем;

```

```

_asm {
    mov c, bl
}

if ((c == 115) || (c == 99) || (c == 116) || (c == 121))
{
    if (c == 115)
        printf("\n Синус числа (радианы):\n ");
    if (c == 99)
        printf("\n Косинус числа (радианы):\n ");
    if (c == 116)
        printf("\n Тангенс числа (радианы):\n ");
    if (c == 121)
        printf("\n Котангенс числа (радианы):\n ");
    scan_115: printf("\n  a =: ");
    scanf("%s", vvod);
    i = check(vvod);
    double finitel(i);
    k++;
    if ((i != 0) && (finitel != 0))
        goto special;
    strcpy(str1, vvod);
    y = atof(vvod);

    ready:
    double y1 = y;
    _asm
    {
        finit
        fld y
    }
    if (c == 115)
    {
        _asm
        fsin
        s[0] = 's';
        s[1] = 'i';
        s[2] = 'n';
        s[3] = '\0';
    }
    if (c == 99)
    {
        _asm
        fcos
        s[0] = 'c';
        s[1] = 'o';
        s[2] = 's';
    }
}

```

```

        s[3] = '\\0';
    }
    if (c == 116)
    {
        s[0] = 't';
        s[1] = 'a';
        s[2] = 'n';
        s[3] = '\\0';
        _asm {
            fsin
            fstp y
            fld y1
            fcos
            fst y1
            fld y
            fld y1
            fdiv
        }
    }
    if (c == 121)
    {
        _asm {
            fcos
            fstp y
            fld y1
            fsin
            fst y1
            fld y
            fld y1
            fdiv
        }
        s[0] = 'c';
        s[1] = 'o';
        s[2] = 't';
        s[3] = '\\0';
    }
    _asm
    fstp y
    y_long = y;
    if ((y > 10) || (y < -10))
    {
        r++;
        goto error;
    }
    if (kuf = 1)
    {
        if (y_long == y)

```

```

        {
            printf("\n %s (%d*%s) = %ld", s, i, str1,
y_long);
            fprintf(file, "\n %s (%d*%s) = %ld", s, i,
str1, y_long);
        }
        else
        {
            printf("\n %s (%d*%s) = %lf", s, i, str1, y);
            fprintf(file, "\n %s (%d*%s) = %lf", s, i,
str1, y);
        }
    }
    else
    {
        if (y_long == y)
        {
            printf("\n %s (%s) = %ld", s, str1, y_long);
            fprintf(file, "\n %s (%s) = %ld", s, str1,
y_long);
        }
        else
        {
            printf("\n %s (%s) = %lf", s, str1, y);
            fprintf(file, "\n %s (%s) = %lf", s, str1, y);
        }
        error:
        if (r != 0)
        {
            printf("\n Error! Div by zero.");
            fprintf(file, "\n Error! Div by zero.");
        }
    }
}
else
{
    _asm {mov bl, c}
}
fclose(file);

}
void history()
{
    FILE* file;
    if ((file = fopen("../../history.txt", "r")) == 0)
        file = fopen("../../history.txt", "w");
    file = fopen("../../history.txt", "r");

```

```

char his[100];
int k = 0;
char c = 0;
system("cls");
printf(" Для выхода в главное меню нажмите любую клавишу
'с'- очистить историю\n");
printf(" История вычислений:\n");
while (fgets(his, 100, file))
{
    printf("%s", his);
}
c = _getch();
if (c == 99)
{
    file = fopen("../../history.txt", "wb");
    fprintf(file, "\n -----");
    fclose(file);
    history();
}
fclose(file);
}

```

Приложение Файл «check.cpp»

A.6

```
#include "calculation.h"
#include "check.h"

int check(char vvod[999])
{
    int i = 0;
    int il = 0;
    int k = 1;
    int n = 1;
    while (vvod[i] != 0)
    {
        if (i >= 19)
        {
            printf("\n Число не должно превышать 18 разрядов",
vvod[i]);
            printf("\n Повторите ввод\n", vvod[i]);
            il = INFINITY;
        }
        if (vvod[i] == 101 || (vvod[i] == 112 && vvod[i + 1] ==
105))
            goto ohno;
        if ((vvod[i] <= 47 || vvod[i] >= 58) && vvod[i] != 44
&& vvod[i] != 45 && vvod[i] != 112 && vvod[i] != 105)
        {
            printf("\n Обнаружен недопустимый символ: %c",
vvod[i]);
            printf("\n Повторите ввод\n", vvod[i]);
            ohno:
            il = INFINITY;
        }

        if (il == 1)
            break;

        if (vvod[i] == 101)
        {
            if (vvod[0] == 'e') {
                il = 1;
                break;
            }
        }
    }
}
```

```

    }
    if (i != 0)
        while (i != k - 1)
        {
            if (vvod[i - k] == 49)
                i1 = i1 + 1 * n;
            if (vvod[i - k] == 50)
                i1 = i1 + 2 * n;
            if (vvod[i - k] == 51)
                i1 = i1 + 3 * n;
            if (vvod[i - k] == 52)
                i1 = i1 + 4 * n;
            if (vvod[i - k] == 53)
                i1 = i1 + 5 * n;
            if (vvod[i - k] == 54)
                i1 = i1 + 6 * n;
            if (vvod[i - k] == 55)
                i1 = i1 + 7 * n;
            if (vvod[i - k] == 56)
                i1 = i1 + 8 * n;
            if (vvod[i - k] == 57)
                i1 = i1 + 9 * n;
            k++;
            n = n * 10;
        }

    if (vvod[0] == '-')
        i1 = -i1;
    if ((vvod[0] == '-') && (vvod[1] == 'e'))
        i1 = -1;
}
if (vvod[i] == 112 && (vvod[i + 1] == 105))
{
    if (vvod[0] == 'p' && vvod[1] == 'i')
    {
        i1 = 1;
        break;
    }
    if (i != 0)
        while (i != k - 1)
        {
            if (vvod[i - k] == 49)
                i1 = i1 + 1 * n;
            if (vvod[i - k] == 50)
                i1 = i1 + 2 * n;
            if (vvod[i - k] == 51)
                i1 = i1 + 3 * n;

```

```

        if (vvod[i - k] == 52)
            i1 = i1 + 4 * n;
        if (vvod[i - k] == 53)
            i1 = i1 + 5 * n;
        if (vvod[i - k] == 54)
            i1 = i1 + 6 * n;
        if (vvod[i - k] == 55)
            i1 = i1 + 7 * n;
        if (vvod[i - k] == 56)
            i1 = i1 + 8 * n;
        if (vvod[i - k] == 57)
            i1 = i1 + 9 * n;
        k++;
        n = n * 10;
    }
    if (vvod[0] == '-')
        i1 = -i1;
    if ((vvod[0] == '-') && (vvod[1] == 'p'))
        i1 = -1;
}
i++;
}
return i1;
}
int check_int(char vvod[999])
{
    int i = 0;
    int i1 = 0;

    while (vvod[i] != 0)
    {
        if ((vvod[i] <= 47 || vvod[i] >= 58) && vvod[i] != 44
&& vvod[i] != 45)
        {
            printf("\n    Обнаружен недопустимый символ: %c",
vvod[i]);
            printf("\n    Повторите ввод\n", vvod[i]);
            i1 = 1;
        }
        if (i1 == 1)
            break;
        i++;
    }

    return i1;
}
int checkp(char vvod[999])

```

```

{
    int i = 0;
    int i1 = 0;

    while (vvod[i] != 0)
    {
        if (i >= 19)
        {
            printf("\n Число не должно превышать 18 разрядов",
vvod[i]);
            printf("\n Повторите ввод\n", vvod[i]);
            i1 = 1;
        }

        if ((vvod[i] <= 47 || vvod[i] >= 58) && vvod[i] != 44
&& vvod[i] != 45 && vvod[i] != 37)
        {
            printf("\n Обнаружен недопустимый символ: %c",
vvod[i]);
            printf("\n Повторите ввод\n", vvod[i]);
            i1 = 1;
        }
        if (i1 == 1)
            break;

        i++;
    }

    return i1;
}

```

Приложение Файл «hello.cpp»

A.7

```
#include "hello.h"
#include<stdio.h>
#include<conio.h>
#include<windows.h>
#include <ctime>

void hello()
{
    FILE* hello;
    hello = fopen("../hello.txt", "r");
    char hi[1000];
    while (fgets(hi, 1000, hello))
    {
        printf("%s", hi);
    }

    Sleep(1500);
    system("cls");
}
```

Приложение Схемы функций программы

В

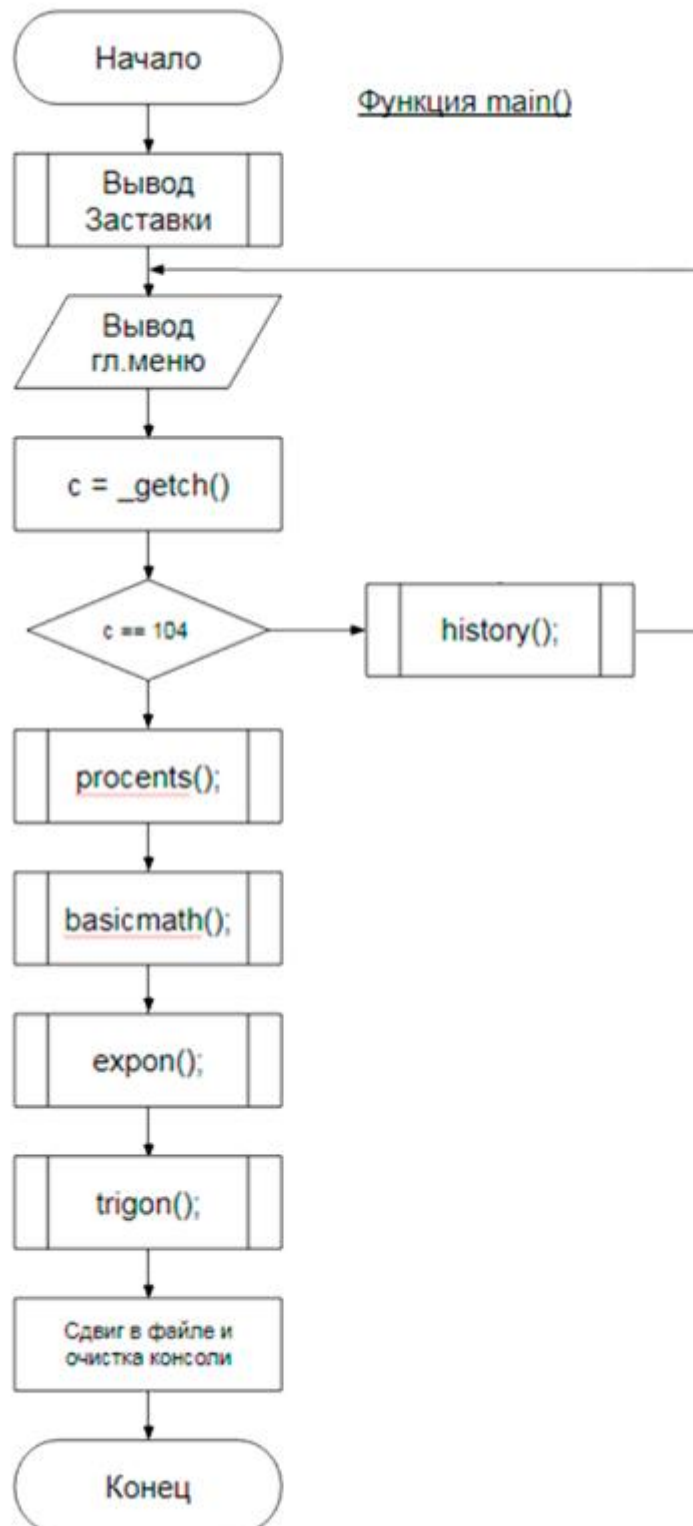


Рисунок 1 – функция `main`



Рисунок 2 – функция basicmath

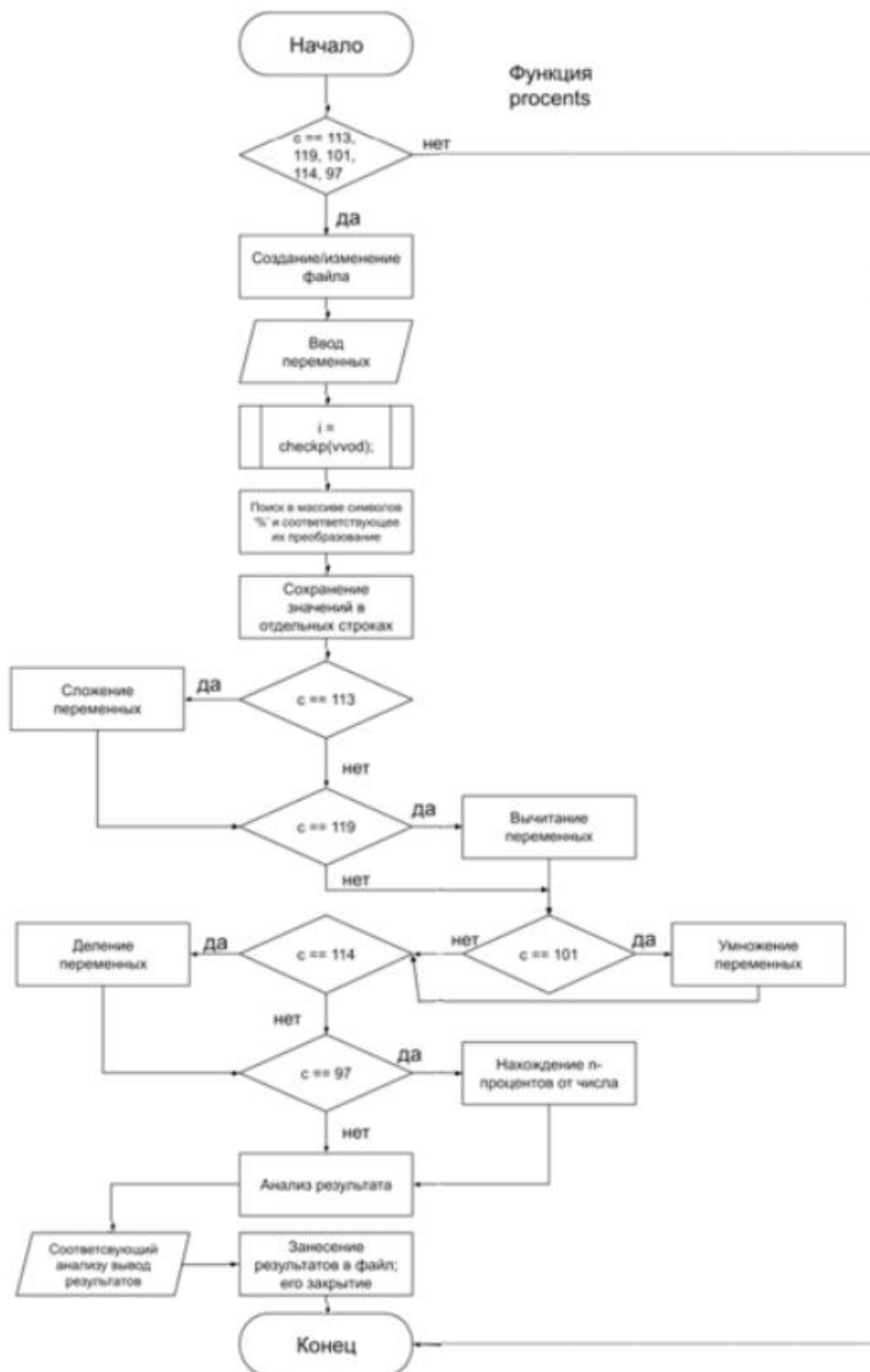


Рисунок 2 – функция procents

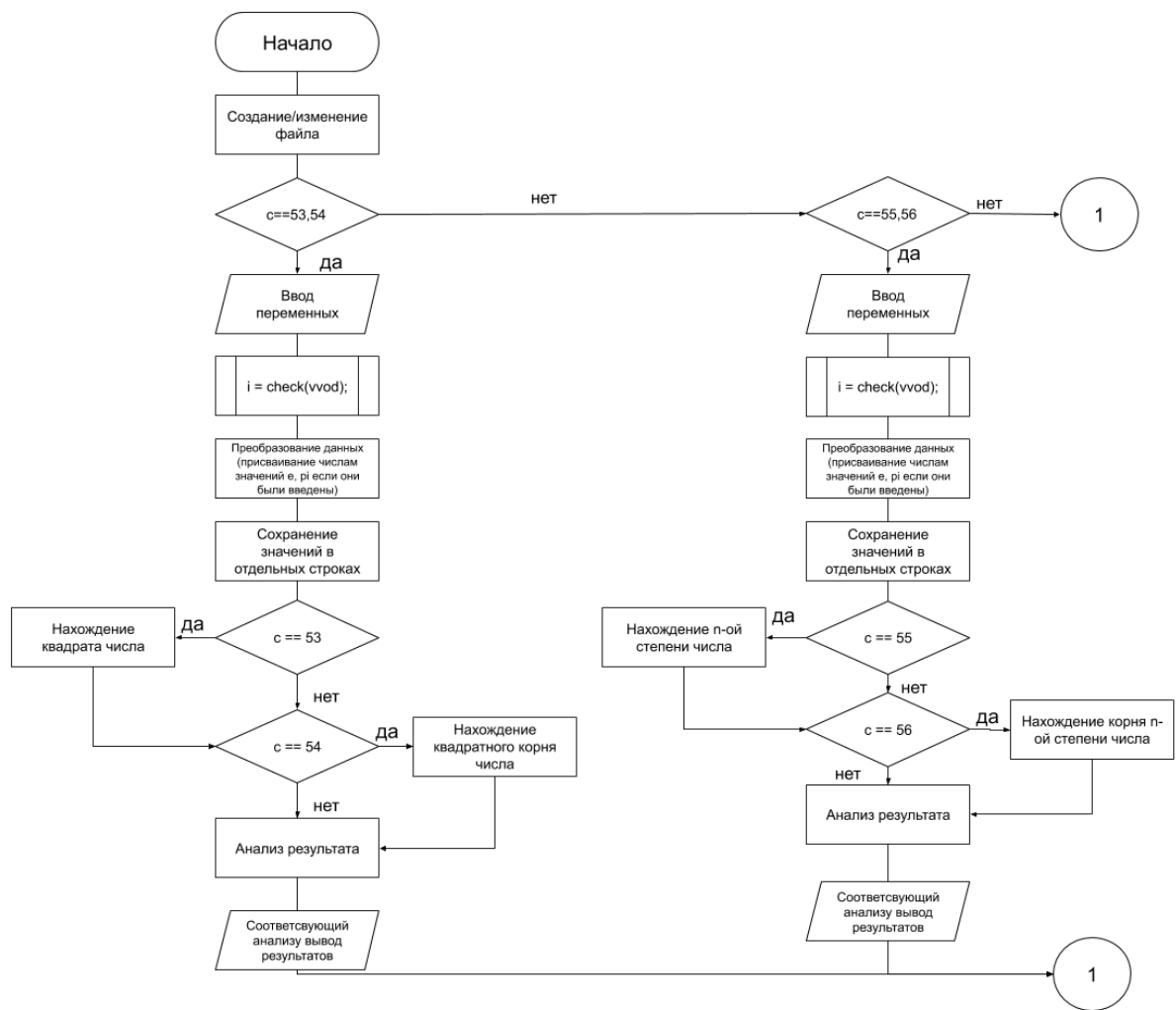


Рисунок 3.1 – функция exron

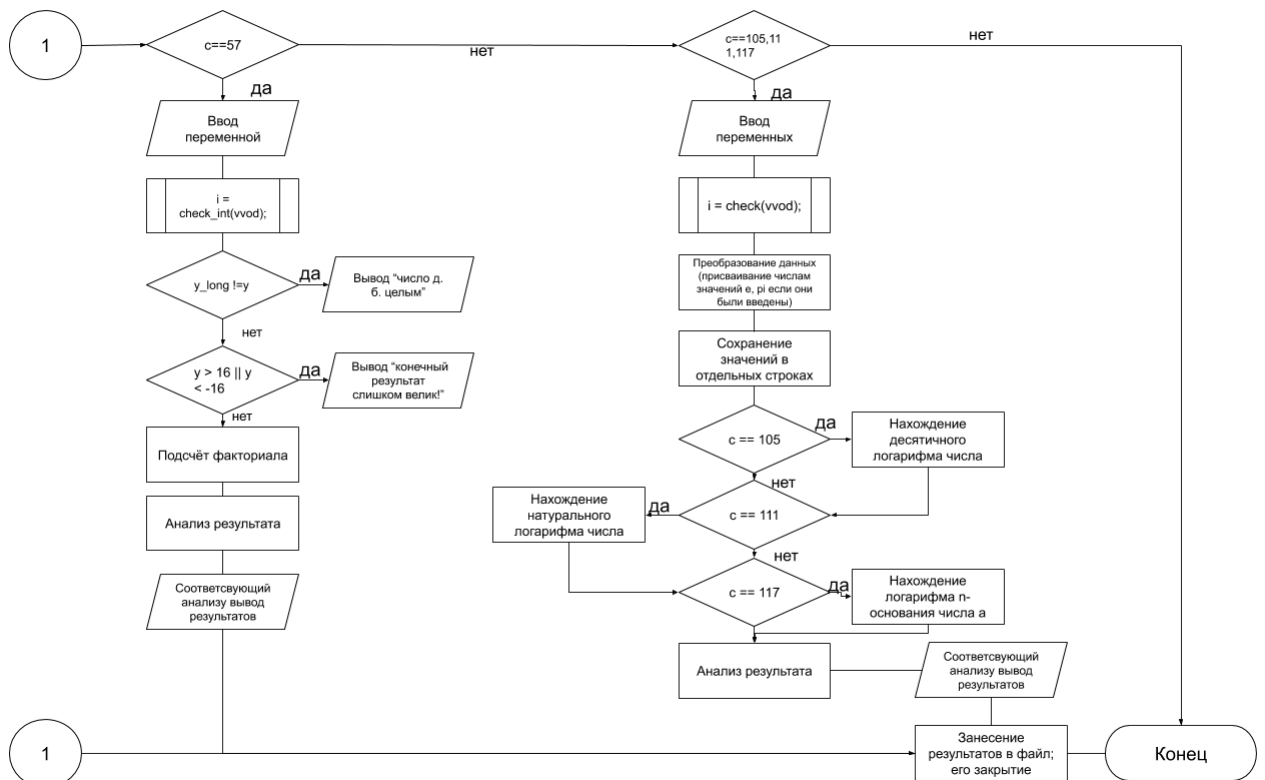


Рисунок 3.2 – функция expon

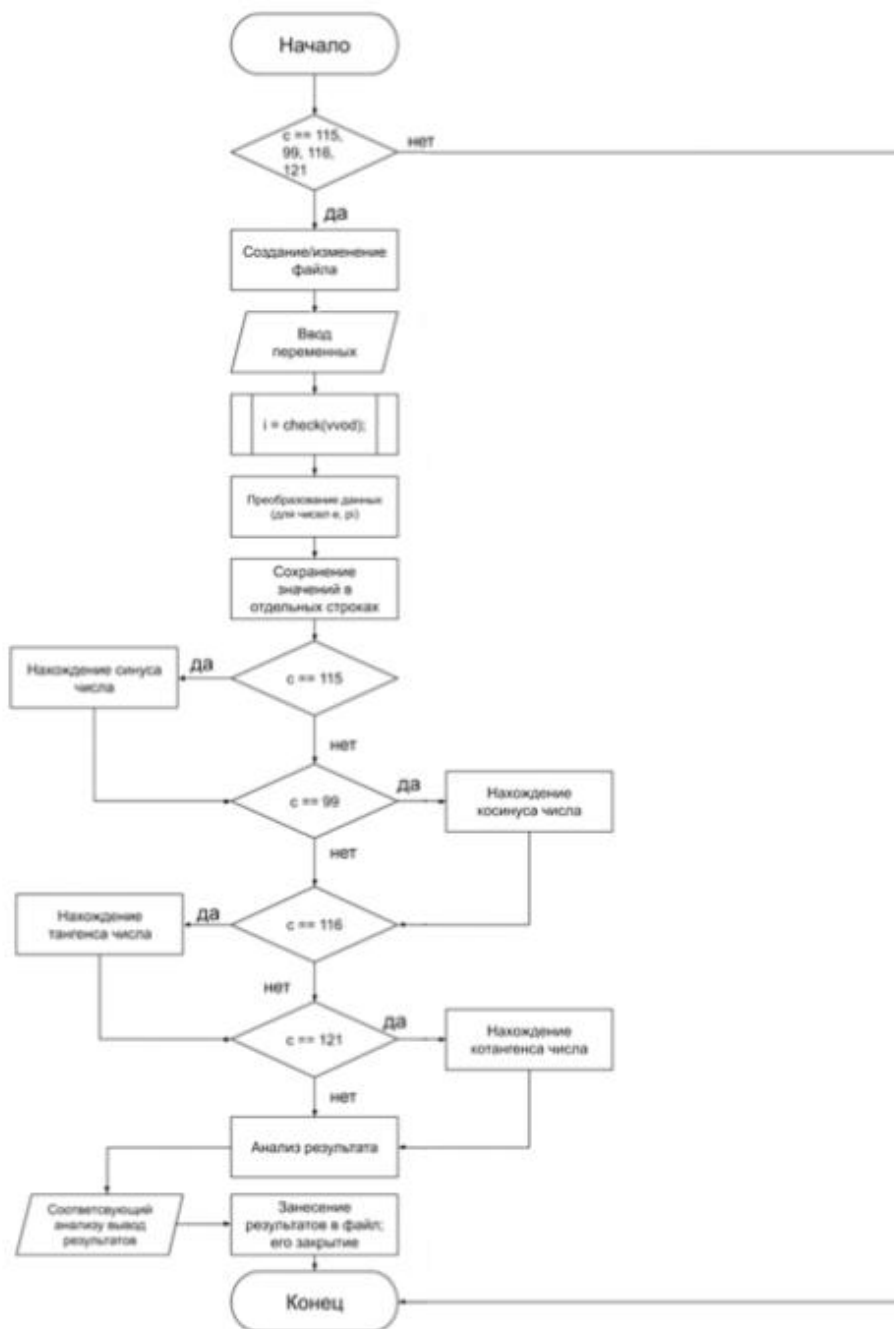


Рисунок 4 – функция trigon



Рисунок 5 – функция history

Схемы взаимодействия и данных

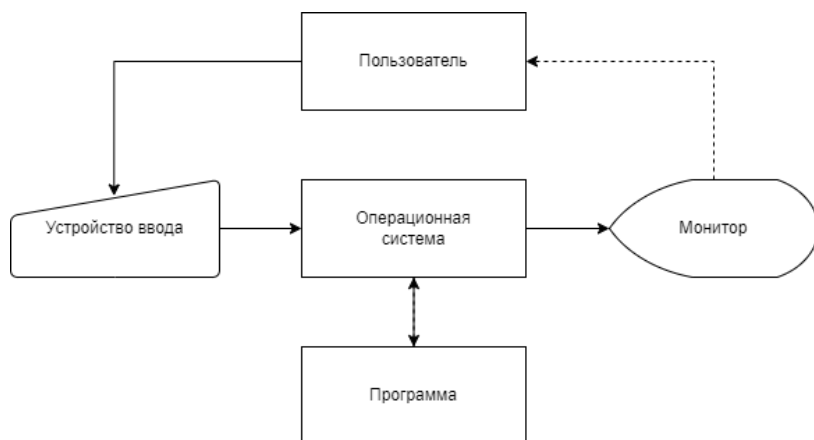


Рисунок 6 – схема взаимодействия

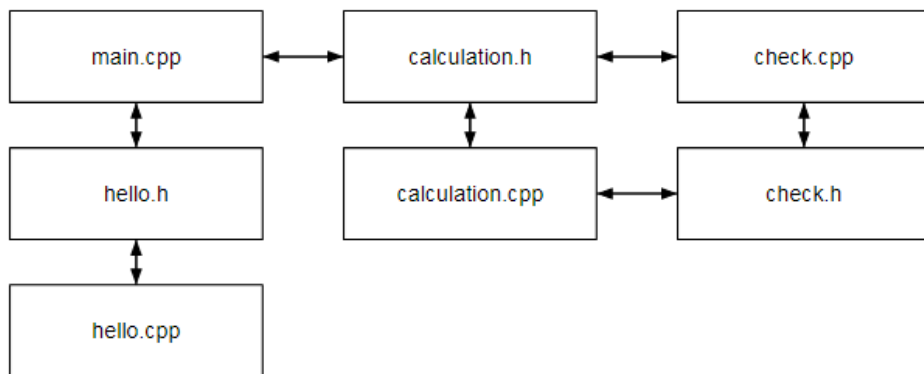


Рисунок 7 – схема данных программы