

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра информационной безопасности

КУРСОВАЯ РАБОТА
по дисциплине «Программирование»
Тема: Приложение «Калькулятор»

Студент гр. 3364

Юсфи.А.

Преподаватель

Халиуллин Р.А.

Санкт-Петербург

2023

ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ

Студент Юсфи.А.

Группа 3364

Тема проекта: Приложение «Калькулятор»

Исходные данные:

Разработать на языке программирования С или С++ (по выбору студента) приложение для выполнения простых расчетов с использованием основных арифметических операций (сложение, вычитание, умножение, деление). Значения для расчета (не менее двух значений) и арифметическая операция вводятся пользователем. Поддержка вещественных значений (значений с плавающей точкой) не обязательна. Допускается возникновение целочисленного переполнения в результате выполнения арифметической операции. Ввод/вывод значений должен осуществляться в десятичной системе счисления. Приложение должно иметь консольный или графический интерфейс (по выбору студента). В интерфейсе приложения допускается использовать буквы латинского алфавита для транслитерации букв алфавита русского языка. Интерфейс приложения должен быть интуитивно понятным и содержать подсказки для пользователя. В исходном коде приложения должны быть реализованы функции. В исходном коде приложения должны быть реализованы проверки аргументов реализованных функций и проверки возвращаемых функциями значений (для всех функций — как сторонних, так и реализованных). Приложение должно корректно обрабатывать ошибки, в том числе ошибки ввода/вывода, выделения/освобождения памяти и т. д.

Содержание пояснительной записки:

- АННОТАЦИЯ.
- SUMMARY.
- СОДЕРЖАНИЕ.
- ВВЕДЕНИЕ
- ЗАКЛЮЧЕНИЕ
- СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Дата выдачи задания: 12.11.2023

Дата сдачи реферата: 25.12.2023

Дата защиты реферата: 25.12.2023.

Студент

Юсфи.А.

Преподаватель

Халиуллин Р.А.

АННОТАЦИЯ

Курсовой проект включает в себя разработку приложения-калькулятора на языке программирования C с основными арифметическими операциями. Методы исследования включают в себя анализ существующих решений, определение функциональных требований и выбор технологий, таких как C. Результаты включают успешную реализацию функций сложения, вычитания, умножения и деления с надежной обработкой ошибок. Приложение поддерживает ввод пользователем двух операндов и выбранной операции. Интерфейс разработан для удобства использования, с интуитивными подсказками. Проект обеспечивает стабильную функциональность, успешно достигая цели предоставления надежного приложения-калькулятора.

SUMMARY

The course project involves developing a C calculator application with basic arithmetic operations. Research methods include analyzing existing solutions, defining functional requirements, and selecting technologies like C. Results include the successful implementation of addition, subtraction, multiplication, and division functions with proper error handling. The application supports user input for two operands and the chosen operation. The interface is designed for ease of use, featuring intuitive prompts. The project ensures stable functionality, effectively meeting the goal of providing a reliable calculator application.

СОДЕРЖАНИЕ

Введение	6-8
1. Теоретическая часть	9
1.1. Описание (определение).	9
1.2. История калькуляторов.	9
2. Реализация программы :	10-15
2.1. Основные сведения о программном обеспечении.	10
2.2. Реализованные функции.	10-15
3. Результаты тестирования программы:	16-17
3.1. Тестирование программы с помощью база без проверки на наличие каких-либо ошибок.	16
3.2. Проверка входного значения.	16
3.3 Проверка деления на Ноль.	17
Заключение	18
Список использованных источников	19
Приложение А блок схема алгоритма	20-21
Приложение В исходный код программы	22-25

ВВЕДЕНИЕ

Цель работы:

Создание простого калькулятора на языке программирования C.

Основные задачи и методы их решения:

Валидация ввода:

Задача:

Гарантировать корректность ввода чисел и операций пользователем.

Метод:

Использование функции `buffer()` для очистки вводного буфера и циклов проверки ввода для чисел и операций.

Арифметические функции:

Задача:

Реализовать базовые арифметические операции (сложение, вычитание, умножение, деление).

Метод:

Определение отдельных функций (`addition`, `subtraction`, `multiplication`, `division`) для каждой операции с проверкой возможных ошибок (например, NaN).

Основной цикл программы:

Задача:

Обеспечить многократное выполнение операций до тех пор, пока пользователь не решит завершить работу.

Метод:

Использование цикла `while` в функции `main()`, который продолжается, пока `continueFlag` равен 'y' или 'Y'.

Обработка ошибок:

Задача:

Обеспечить корректное завершение программы при возникновении ошибок.

Метод:

Вывод соответствующих сообщений об ошибке и использование ненулевых кодов завершения для индикации проблем.

Завершение программы:

Задача:

Предоставить пользователю возможность завершить программу при необходимости.

Метод:

Вывод сообщения о завершении программы при вводе 'q' или отказе от продолжения после каждого расчета.

Пользовательский опыт:

Описание:

Положительный пользовательский опыт является ключевым для удобства использования калькулятора. Четкие инструкции, интуитивно понятные подсказки и прямолинейные операции способствуют бесперебойному взаимодействию с пользователем.

Применение:

Калькулятор предоставляет четкие и лаконичные подсказки, направляя пользователя на каждом этапе процесса вычисления.

Механизмы обратной связи, такие как сообщения об ошибках и отображение результатов, гарантируют, что пользователь всегда в курсе и может легко понять ответы программы.

Пример: после каждого вычисления пользователю предоставляется результат, а затем предлагается возможность выполнить новое вычисление или завершить программу, увеличивая гибкость и вовлеченность пользователя.

1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1. ОПИСАНИЕ (ОПРЕДЕЛЕНИЕ):

Калькулятор — это устройство или инструмент, используемый для выполнения арифметических операций. Хотя этот термин может относиться к простым инструментам, таким как счеты, или к более совершенным электронным устройствам, основная функция остается прежней: помогать в математических вычислениях.

1.2. История калькуляторов:

Калькуляторы имеют долгую историю, эволюционируя от простых счетных инструментов, таких как счеты, до сложных электронных устройств. Ключевые вехи включают:

1.2.1. Счеты (около 2400 г. до н.э.):

одно из самых ранних известных вычислительных устройств.

1.2.2. Механические калькуляторы (начиная с 17 века):

Примечательны такие устройства, как калькулятор Паскаля и арифмометр Чарльза Ксавье Томаса.

1.2.3. Электромеханические калькуляторы (середина 20 века):

В 1940-х и 1950-х годах появились электронные калькуляторы, в которых использовались вакуумные трубки, а позже и транзисторы.

1.2.4. Электронные калькуляторы (1960-е годы и далее):

1960-е и 1970-е годы стали свидетелями появления карманных электронных калькуляторов, которые стали более доступными по цене.

2. Реализация программы

1.1. Основные сведения о программном обеспечении:

Курсовая работа была выполнена мной на языке программирования Си. Использовались операционная система **Windows**, и среда разработки **Clion** от **Jetbrains Community** 2023 версии 2023.3.2 с, и используйте **GitHub**. Протестировано в среде IDE и с использованием (**exe**).

1.2. Реализованные функции:

Функция **Buffer** (Буфер):

Описание:

Функция **buffer()** необходима для очистки входного буфера. Это гарантирует удаление любых оставшихся или нежелательных символов в буфере перед тем, как программа продолжит выполнение.

Применение:

Когда пользователь вводит символ или строку вместо числа, функция **buffer()** очищает входной буфер. Это предотвращает ошибочную интерпретацию программой недопустимого ввода и позволяет запросить новый ввод.

Пример: если пользователь по ошибке вводит символ "А" вместо числа, функция **buffer** гарантирует, что этот символ будет очищен, и программа сможет запросить действительный числовой ввод.

Арифметические функции:

Описание:

Арифметические функции (сложение, вычитание, умножение, деление) являются основой функциональности

калькулятора. Они отвечают за выполнение основных математических операций и обеспечивают точность результатов.

Применение:

Сложение:

Проверяет вводные числа и выполняет операцию сложения. Функция возвращает сумму двух чисел.

Вычитание:

Проверяет входные значения и вычисляет разницу между ними, возвращая результат.

Умножение:

Проверяет вводные данные и вычисляет произведение указанных чисел, предоставляя результат.

Деление:

Обеспечивает, что знаменатель не равен нулю, а затем выполняет операцию деления, возвращая частное.

Пример:

Сложение: Ввод чисел 5 и 3. Результат: 8

Вычитание: Ввод чисел 7 и 4. Результат: 3

Умножение: Ввод чисел 6 и 2. Результат: 12

Деление: Ввод чисел 8 и 2. Результат: 4

Главная функция:

Описание:

Главная функция действует как организатор калькулятора, управляя взаимодействием с пользователем, вызовом арифметических функций и отображением результатов.

Применение:

Функция получает ввод от пользователя для чисел и операций.

Она вызывает соответствующую арифметическую функцию на основе выбранной операции.

Отображает вычисленный результат пользователю и предлагает варианты для дальнейших вычислений или завершения работы программы.

Пример:

Пользователь вводит первое число: 10

Пользователь выбирает операцию: '+'

Пользователь вводит второе число: 5

Результат отображается: 15

Пользователь получает запрос на продолжение или завершение.

Основная функция(main):

Функция `main` служит основным драйвером программы калькулятора. Она организует взаимодействие с пользователем, вызывает необходимые арифметические функции и управляет общим потоком программы.

1. Инициализация:

- Переменные, такие как `Num1`, `Num2`, `result`, `operation` и `continueFlag`, объявлены для хранения пользовательских вводов и результатов.

- Пользователю отображается приветственное сообщение.

2. Цикл взаимодействия с пользователем:

- Цикл ``while`` (``while (continueFlag == 'y' || continueFlag == 'Y')``) гарантирует, что калькулятор продолжит работу до тех пор, пока пользователь не решит выйти.

3. Обработка ввода:

- В рамках цикла программа:
 - Предлагает пользователю ввести первое число.
 - Проверяет вводимые данные, чтобы убедиться, что это допустимое числовое значение.
 - Просит пользователя выбрать арифметическую операцию (``+``, ``-``, ``*``, ``/``) или завершить (``q``).
 - Предлагает пользователю ввести второе число и проверяет правильность ввода аналогично первому числу.

4. Арифметическая операция:

- На основе выбранной операции программа:
 - Вызывает соответствующую арифметическую функцию (``сложение``, ``вычитание``, ``умножение``, ``деление``), передавая введенные числа.
 - Получает результат от арифметической функции и сохраняет его в переменной ``результат``.

5. Отображение результата:

- Вычисленный результат затем отображается пользователю с помощью ``printf``.

6. Обработка ошибок:

- Если во время проверки ввода или арифметических операций возникают какие-либо ошибки, отображаются соответствующие

сообщения об ошибках, и программа может завершиться с ненулевым статусом.

7. Запрос на продолжение:

- После отображения результата программа спрашивает пользователя, желает ли он выполнить другое вычисление. Если пользователь решает продолжить, цикл повторяется снова; в противном случае программа завершается.

Резюме:

Функция ``main`` инкапсулирует основную функциональность калькулятора, обрабатывая взаимодействие с пользователем, проверку ввода, арифметические операции и отображение результатов. Она предоставляет структурированную и интерактивную среду для выполнения пользователями базовых арифметических вычислений до тех пор, пока они не решат выйти из программы.

Обработка ошибок:

Описание:

Обработка ошибок неотъемлема для надежности программы. Это гарантирует, что программа может корректно обрабатывать неожиданный ввод или условия, предоставляя обратную связь пользователю при необходимости.

Применение:

Код включает проверки для валидации ввода пользователем, такие как гарантия того, что знаменатель не равен нулю или обнаружение нечисловых символов.

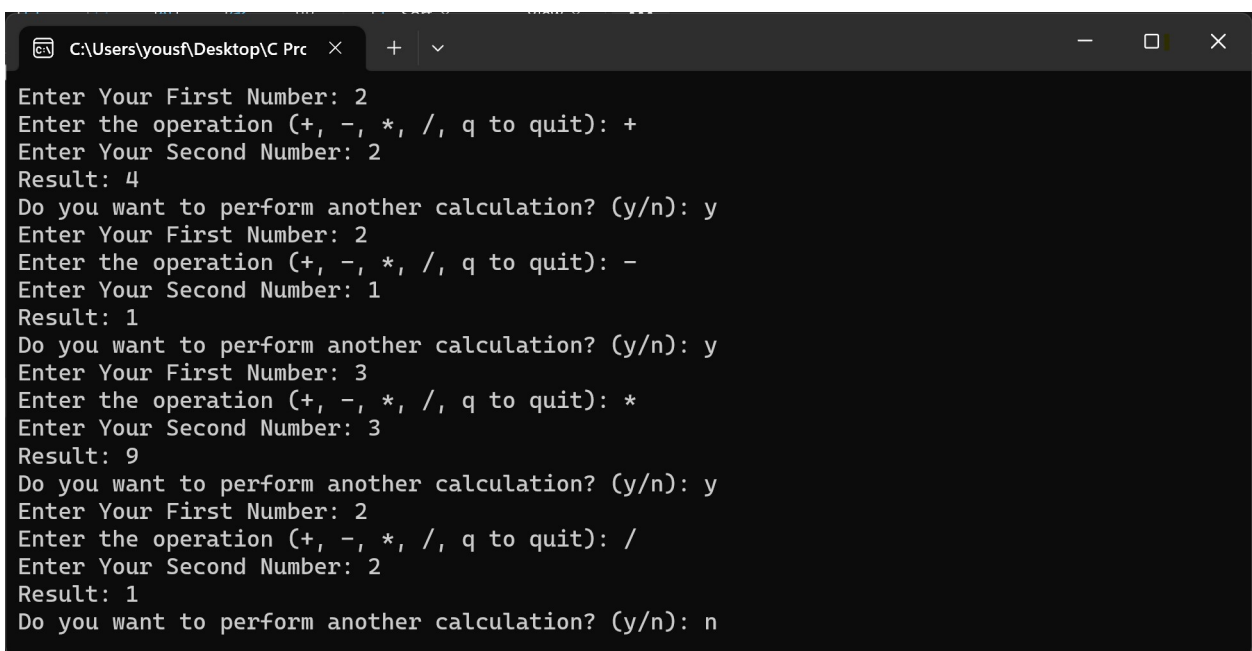
Если обнаружена ошибка, программа отображает описательное сообщение об ошибке, направляя пользователя на корректировку их ввода.

Пример: если пользователь пытается разделить на ноль, программа обнаруживает эту ситуацию, отображает сообщение об ошибке ("Нельзя делить на ноль.") и предлагает пользователю ввести действительный знаменатель.

3. РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ ПРОГРАММЫ

3.1. Тестирование программы с помощью basic без проверки на наличие каких либо ошибок:

- На этом шаге мы протестировали программу, используя (.exe) . и мы протестировали с простым и допустимым значением, чтобы подтвердить, что он работает нормально, в качестве первого шага.

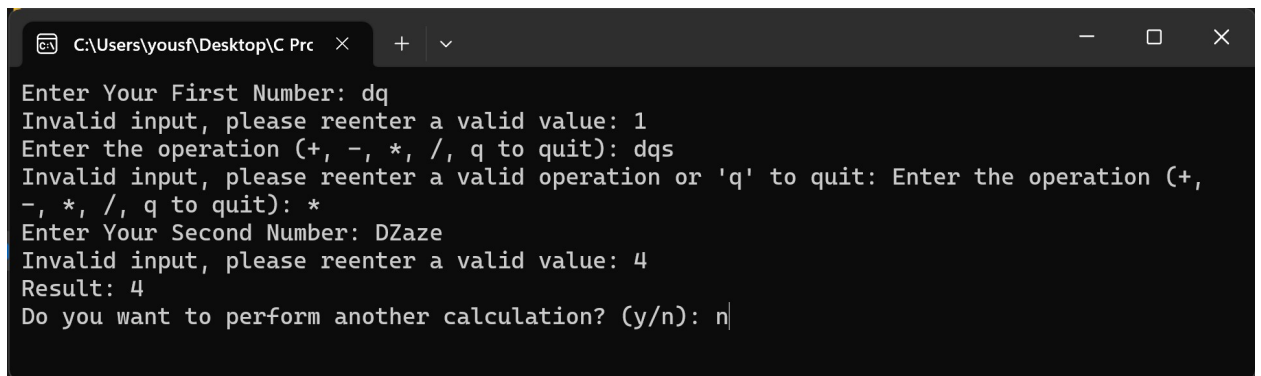


```
C:\Users\yousf\Desktop\C Prc
Enter Your First Number: 2
Enter the operation (+, -, *, /, q to quit): +
Enter Your Second Number: 2
Result: 4
Do you want to perform another calculation? (y/n): y
Enter Your First Number: 2
Enter the operation (+, -, *, /, q to quit): -
Enter Your Second Number: 1
Result: 1
Do you want to perform another calculation? (y/n): y
Enter Your First Number: 3
Enter the operation (+, -, *, /, q to quit): *
Enter Your Second Number: 3
Result: 9
Do you want to perform another calculation? (y/n): y
Enter Your First Number: 2
Enter the operation (+, -, *, /, q to quit): /
Enter Your Second Number: 2
Result: 1
Do you want to perform another calculation? (y/n): n
```

Рисунок 1: Протестированный подтвержденный ввод.

3.2. Проверка входного значения:

Проверяя допустимый ввод в качестве примера, мы попытались ввести вместо первого числового алфавита, но код проверяется и дает нам возможность ввести еще раз и для второго числа тоже два. и мы проверили арифметическую операцию, если ввод не из допустимого, также проверяем код и возвращаемся к у нас есть шанс внести свой вклад еще раз .

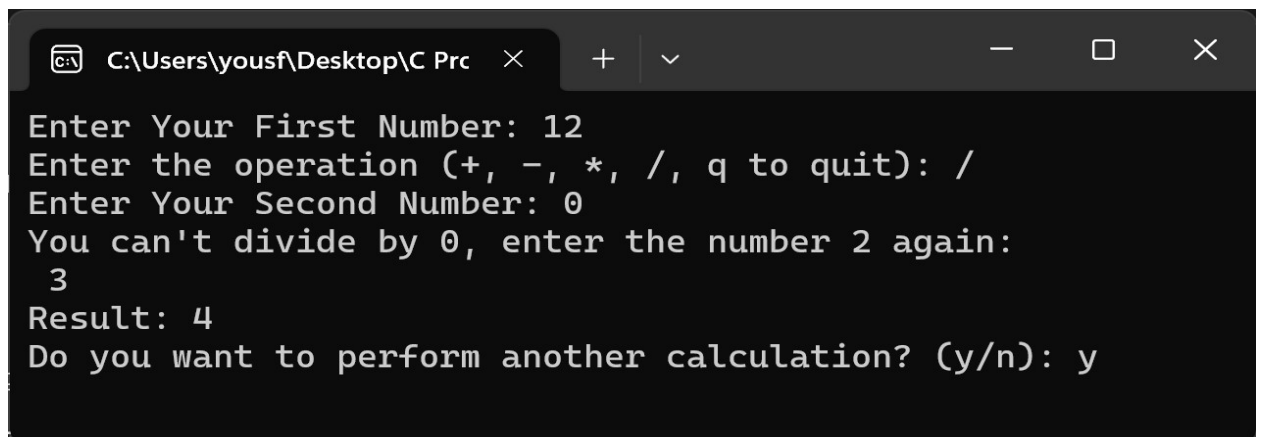


```
C:\Users\yousf\Desktop\C Prc > Enter Your First Number: dq
Invalid input, please reenter a valid value: 1
Enter the operation (+, -, *, /, q to quit): dqs
Invalid input, please reenter a valid operation or 'q' to quit: Enter the operation (+, -, *, /, q to quit): *
Enter Your Second Number: DZaze
Invalid input, please reenter a valid value: 4
Result: 4
Do you want to perform another calculation? (y/n): n
```

рисунок 2: Проверка входного значения

3.3. Проверка деления на Ноль :

В общем обычный калькулятор, когда мы делим на 0, калькулятор выдает нам ошибку, но в моем коде я проверил и даю возможность пользователю снова ввести второе число .



```
C:\Users\yousf\Desktop\C Prc > Enter Your First Number: 12
Enter the operation (+, -, *, /, q to quit): /
Enter Your Second Number: 0
You can't divide by 0, enter the number 2 again:
3
Result: 4
Do you want to perform another calculation? (y/n): y
```

Рисунок 3 : Проверка деления на ноль

Я хочу добавить, что причина, по которой я не запустил опцию n (No), означает, что пользователь не хочет снова использовать калькулятор, поэтому при запуске программа закрылась .

ЗАКЛЮЧЕНИЕ

Проект целостно выполнил свою цель — создание приложения-калькулятора на языке программирования C, предоставляющего базовый функционал арифметических операций. В процессе анализа альтернативных решений и выбора технологий, включая использование C, были сделаны обоснованные решения. Результаты работы включают успешную реализацию операций сложения, вычитания, умножения и деления с учетом различных сценариев исключительных ситуаций.

Приложение обеспечивает ввод пользователем двух операндов и выбора операции, а также аккуратно обрабатывает возможные ошибки, такие как деление на ноль или неверный ввод. Интуитивно понятный интерфейс и простота использования соответствуют поставленной цели обеспечения удобства для конечного пользователя.

В итоге, проект демонстрирует успешное соответствие своей цели, предоставляя стабильное, надежное и функциональное приложение-калькулятор.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. "Calculating Machines and Instruments: History, Technology, and Applications" by Geoffrey W.A. Dummer.
2. Standard C Library Functions Table, By Name [Электронный ресурс]. – URL:
https://www.ibm.com/support/knowledgecenter/ssw_ibm_i_74/rtref/stalib.htm
(дата обращения: 21.12.2020).
3. C/C++ Refences – All C Functions [Электронный ресурс]. – URL:
https://doc.bccnsoft.com/docs/cppreference_en/all_c_functions.html (дата обращения: 13.12.2020).
4. geeksforgeeks -[Электронный ресурс].
URL:<https://www.geeksforgeeks.org/c-programming-language/>

БЛОК СХЕМА АЛГОРИТМА

Для блока схемы я сделал две схемы, одну для человека, который не разбирается в программировании. а во второй подробно описано, как все работает, и главное, как это работает с деталями.

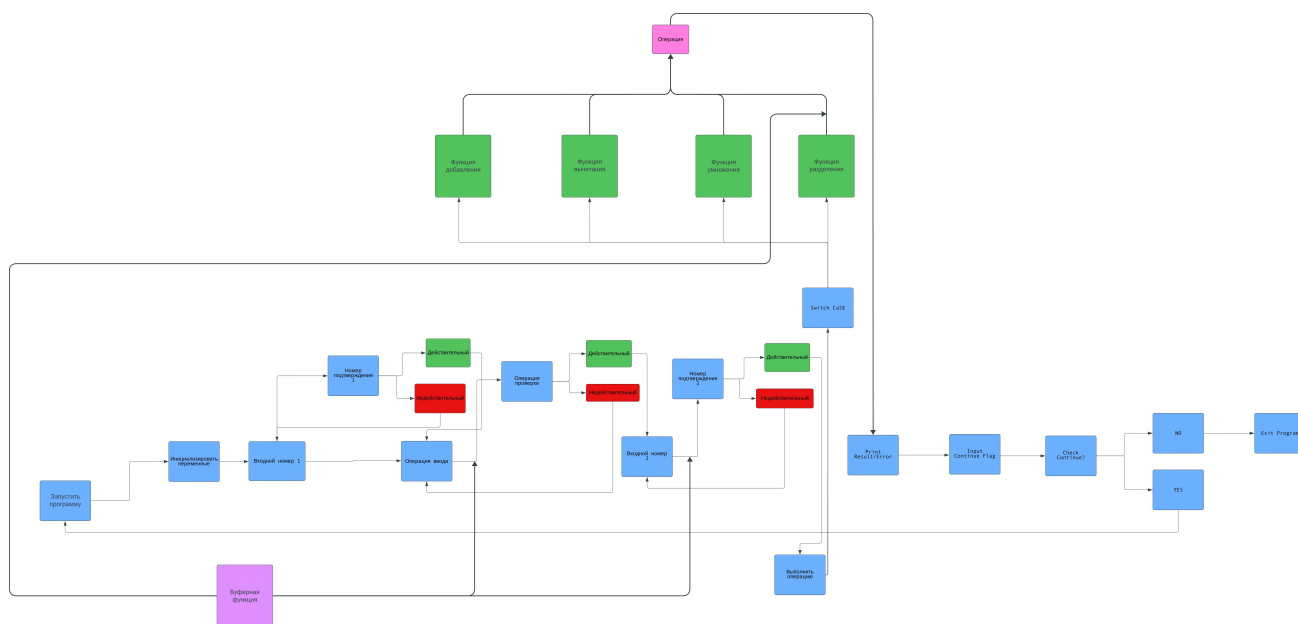
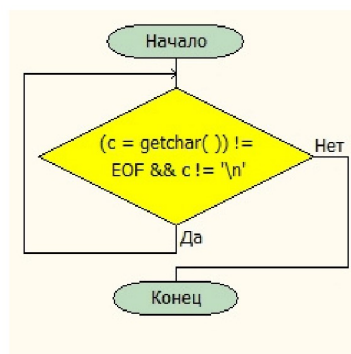
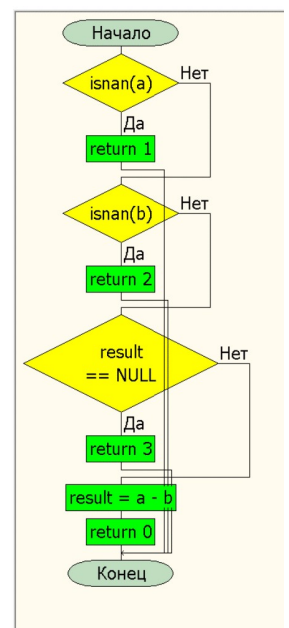
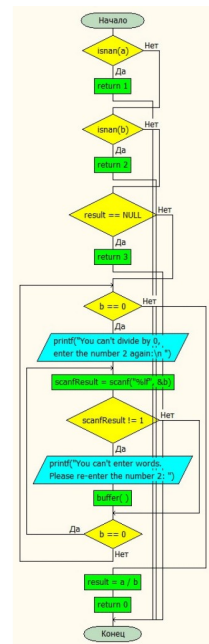
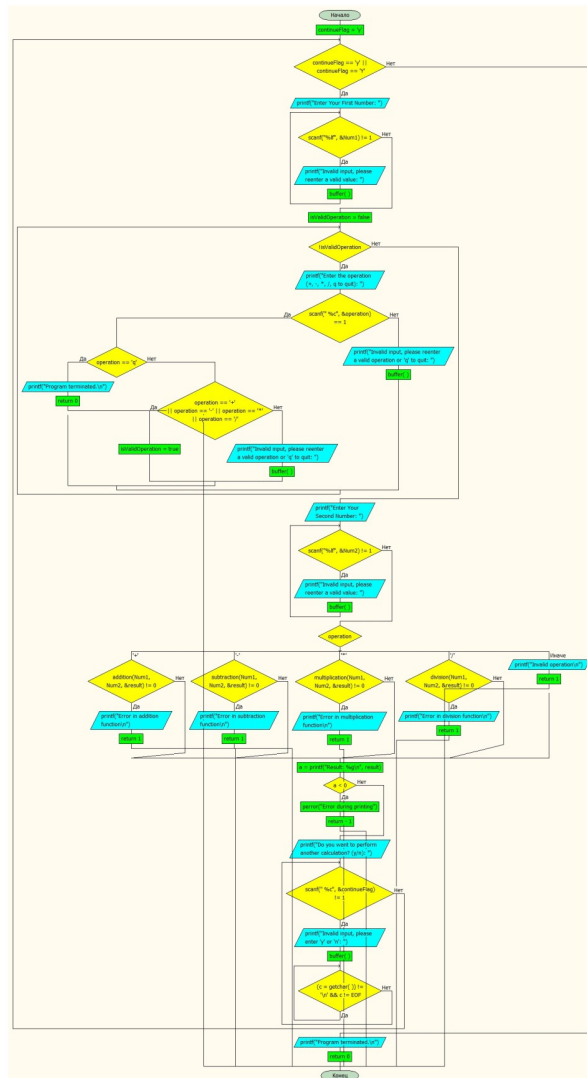
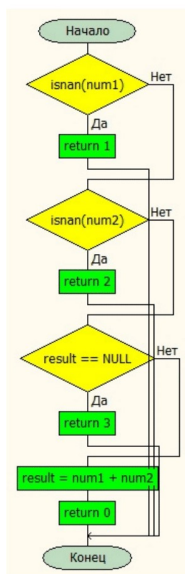
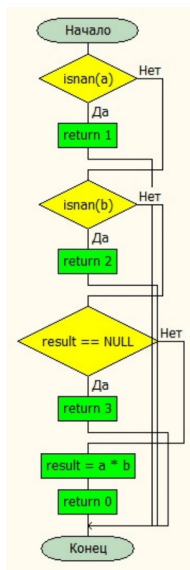


Рисунок 1 простая схема блока

Рисунок 2 детали блока схема



ПРИЛОЖЕНИЕ В

ИСХОДНЫЙ КОД ПРОГРАММЫ

```
#include <stdio.h>
#include <math.h>
// Function to clear the input buffer
void buffer(void)
{
    int c;
    while ((c = getchar()) != EOF && c != '\n');
}
// Declare the function of addition
double addition(double num1, double num2, double *result)
{
    // Check if num1 is NaN
    if (isnan(num1)) return 1;
    // Check if num2 is NaN
    if (isnan(num2)) return 2;
    // Check if result is NULL
    if (result == NULL) return 3;
    *result = num1 + num2;
    return 0;
}
// Declare the function of subtraction
double subtraction(double a, double b, double *result)
{
    if (isnan(a)) return 1;
    if (isnan(b)) return 2;
    if (result == NULL) return 3;
    *result = a - b;
    return 0;
}
// Declare the function of multiplication
double multiplication(double a, double b, double *result)
{
    if (isnan(a)) return 1;
    if (isnan(b)) return 2;
    if (result == NULL) return 3;
    *result = a * b;
    return 0;
}
// Declare the function of division
double division(double a, double b, double *result)
{
    if (isnan(a)) return 1;
    if (isnan(b)) return 2;
    if (result == NULL) return 3;
    // Check for division by zero
    while (b == 0)
    {
        printf("You can't divide by 0, enter the number 2 again:\n ");
    }
}
```

```

do {
    int scanfResult = scanf("%lf", &b);
    if (scanfResult != 1) {
        printf("You can't enter words. Please re-enter the number 2: ");
        buffer();
    }
} while (b == 0);
}
*result = a / b;
return 0;
}

int main() {
    double Num1, Num2, result;
    char operation;
    char continueFlag = 'y';
    printf("Welcome !! For source Code Check My github @sher0u\n ");
    while (continueFlag == 'y' || continueFlag == 'Y')
    {
        // Get the first number
        printf("Enter Your First Number: ");
        while (scanf("%lf", &Num1) != 1)
        {
            printf("Invalid input, please reenter a valid value: ");
            buffer();
        }
        // buffer(); // Uncomment this line if you encounter issues with subsequent inputs
        // Get the operation from the user
        bool isValidOperation = false;
        // Get the operation from the user
        while (!isValidOperation) {
            printf("Enter the operation (+, -, *, /, q to quit): ");
            if (scanf(" %c", &operation) == 1) {
                if (operation == 'q') {
                    printf("Program terminated.\n");
                    return 0;
                } else if (operation == '+' || operation == '-' || operation == '*' || operation == '/') {
                    isValidOperation = true; // Exit the loop if a valid operation is entered
                } else {
                    printf("Invalid input, please reenter a valid operation or 'q' to quit: ");
                    buffer();
                }
            } else {
                printf("Invalid input, please reenter a valid operation or 'q' to quit: ");
                buffer();
            }
        }
        // buffer(); // Uncomment this line if you encounter issues with subsequent inputs
        // Get the second number
        printf("Enter Your Second Number: ");
        while (scanf("%lf", &Num2) != 1)
        {
            printf("Invalid input, please re enter a valid value: ");

```

```

    buffer();
}
// buffer(); // Uncomment this line if you encounter issues with subsequent inputs
// now let's make the switch case for the arithmetic operation
switch (operation)
{
    case '+':
        if (addition(Num1, Num2, &result) != 0)
        {
            printf("Error in addition function\n");
            return 1;
        }
        break;
    case '-':
        if (subtraction(Num1, Num2, &result) != 0)
        {
            printf("Error in subtraction function\n");
            return 1;
        }
        break;
    case '*':
        if (multiplication(Num1, Num2, &result) != 0)
        {
            printf("Error in multiplication function\n");
            return 1;
        }
        break;
    case '/':
        if (division(Num1, Num2, &result) != 0)
        {
            printf("Error in division function\n");
            return 1;
        }
        break;
    default:
        printf("Invalid operation\n");
        return 1; // this will exit with an error code
}
int a;
a = printf("Result: %g\n", result); // we use %g to print the integer if the correct form and the
float                                     // in the right form

if (a < 0) {
    perror("Error during printing");
    return -1;
}
// Ask if the user wants to continue
printf("Do you want to perform another calculation? (y/n): ");
while (scanf(" %c", &continueFlag) != 1) {
    printf("Invalid input, please enter 'y' or 'n': ");
    buffer(); // Clear invalid input

```



```

        // Consume any remaining characters in the buffer including the newline
        int c;
        while ((c = getchar()) != '\n' && c != EOF);
    }
}
printf("Program terminated.\n");
return 0;
}
/*
Yousfi abdelakder / github :@sher0u /mail : abdelkader.yousfi.pr@mail.ru

* Tested to work.
*/

```