|  |  |
| --- | --- |
| Пермский национальный исследовательский политехнический университет  Кафедра информационные технологии и автоматизированные системы | |
|  |  |
|  |  |
| Лабораторная работа №7  "Перегрузка функций в С++"  Вариант №9 | |
|  |  |
|  | Выполнил:  Студент группы ИВТ-24-2б  А.С. Ваулин  Проверил:  Доцент кафедры ИТАС  О.А. Полякова |
|  |  |
| Пермь 2025 | |

# Постановка задачи

Написать перегруженные функции и основную программу, которая их вызывает.

а) для сложения десятичных дробей;

б) для сложения обыкновенных дробей.

# Анализ задачи

**1. Цель программы:**

Основная цель программы - предоставить пользователю простой консольный калькулятор для выполнения операции сложения. Калькулятор должен уметь складывать два типа чисел:

* **Десятичные дроби:** Обычные числа с плавающей точкой.
* **Обыкновенные дроби:** Числа в формате числитель/знаменатель, с возможностью упрощения результата и выделения целой части.

Программа должна быть интерактивной, предоставляя пользователю меню для выбора типа операции и запрашивая необходимые входные данные.

**2. Функциональность:**

Программа реализует следующие основные функции:

* **Выбор типа операции:** Предоставляет пользователю меню для выбора между сложением десятичных и обыкновенных дробей, а также опцию выхода из программы.
* **Сложение десятичных дробей:** Реализует операцию сложения двух десятичных чисел с использованием стандартной операции сложения + для типа double.
* **Сложение обыкновенных дробей:** Реализует сложение двух обыкновенных дробей, включая:
  + Нахождение общего знаменателя.
  + Сложение числителей.
  + Упрощение полученной дроби путем нахождения наибольшего общего делителя (НОД) числителя и знаменателя и деления на него обоих.
  + Выделение целой части из неправильной дроби (если числитель по модулю больше или равен знаменателю по модулю).
* **Интерактивный ввод данных:** Запрашивает у пользователя необходимые входные данные для каждой операции (десятичные числа или числители и знаменатели дробей) через консольный ввод.
* **Вывод результатов:** Выводит результаты сложения в консоль в понятном формате: десятичные дроби выводятся как обычные числа, обыкновенные дроби выводятся в формате "числитель/знаменатель" или "целая часть и числитель/знаменатель" (если есть целая часть).

**3. Ввод:**

Программа принимает следующие типы входных данных от пользователя через консоль:

* **Выбор операции:** Целое число (1, 2 или 0), соответствующее выбору операции из меню.
* **Десятичные числа:** Два числа типа double для операции сложения десятичных дробей.
* **Числители и знаменатели обыкновенных дробей:** Четыре целых числа типа int: числитель и знаменатель первой дроби, и числитель и знаменатель второй дроби.

**4. Вывод:**

Программа выводит следующие типы данных в консоль:

* **Меню операций:** Текстовое меню с вариантами операций и опцией выхода.
* **Приглашения для ввода данных:** Текстовые сообщения, запрашивающие ввод чисел или числителей/знаменателей.
* **Результат сложения десятичных дробей:** Число типа double, представляющее сумму.
* **Результат сложения обыкновенных дробей:** Дробь, представленная в виде:
  + "числитель/знаменатель" (для правильных дробей или нуля).
  + "целая часть и числитель/знаменатель" (для неправильных дробей).
* **Сообщения об ошибках:** Текстовые сообщения, например, "Знаменатель не может быть равен 0!" или "Неверный выбор. Пожалуйста, выберите 1, 2 или 0."
* **Сообщение о выходе:** Текстовое сообщение "Выход из программы."

**5. Алгоритмы и структуры данных:**

* **Алгоритм Евклида:** Используется для нахождения наибольшего общего делителя (НОД) в функции gcd(). Это стандартный и эффективный алгоритм для нахождения НОД двух целых чисел.
* **Алгоритм сложения десятичных дробей:** Стандартная операция сложения для чисел с плавающей точкой, предоставляемая языком C++.
* **Алгоритм сложения обыкновенных дробей:** Реализован вручную, включает приведение к общему знаменателю, сложение числителей и упрощение результата.
* **Структура данных pair<int, int>:** Используется для возвращения результата сложения обыкновенных дробей в виде пары (числитель, знаменатель).
* **Цикл do-while:** Используется для организации основного меню программы и повторения действий до тех пор, пока пользователь не выберет выход.
* **Оператор switch:** Используется для обработки выбора пользователя в меню операций, обеспечивая более читаемую структуру кода по сравнению с if-else if-else в данном случае.

# Блок-схема

начало

double add(double num1, double num2)

#include <iostream>

#include <cmath>

#include <windows.h>

using namespace std;

return num1 + num2;

{

}

int gcd(int a, int b)

while (b)

a %= b;

swap(a, b);

return a;

{

}

{

}

1

0

pair<int, int> add(int num1\_num, int num1\_den, int num2\_num, int num2\_den)

int result\_num = (num1\_num \* num2\_den) + (num2\_num \* num1\_den);

int result\_den = num1\_den \* num2\_den;

int common\_divisor = gcd(abs(result\_num), abs(result\_den));

result\_num /= common\_divisor;

result\_den /= common\_divisor;

return make\_pair(result\_num, result\_den);

{

}

int main()

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

int choice;

do

cout << "Выберите операцию:" << endl;

cout << "1. Сложение десятичных дробей" << endl;

cout << "2. Сложение обыкновенных дробей" << endl;

cout << "0. Выход" << endl;

cout << "Ваш выбор: ";

cin >> choice;

α

κ

{

{

α

switch

(

)

choice

case 1:

double decimal1, decimal2, decimal\_result;

cout << "Введите первое десятичное число: ";

cin >> decimal1;

cout << "Введите второе десятичное число: ";

cin >> decimal2;

decimal\_result = add(decimal1, decimal2);

cout << "Сумма десятичных дробей: " << decimal\_result << endl;

cout << endl;

break;

γ

β

{

β

case 2:

int num1\_num, num1\_den, num2\_num, num2\_den;

pair<int, int> fraction\_result;

cout << "Введите числитель первой дроби: ";

cin >> num1\_num;

cout << "Введите знаменатель первой дроби (не 0): ";

cin >> num1\_den;

cout << "Введите числитель второй

дроби: ";

cin >> num2\_num;

cout << "Введите знаменатель второй дроби (не 0): ";

cin >> num2\_den;

if (num1\_den == 0 || num2\_den == 0)

δ

ε

ί

1

0

δ

ε

cout << "Знаменатель не может быть равен 0!" << endl;

fraction\_result = add(num1\_num, num1\_den, num2\_num, num2\_den);

int whole\_part = 0;

int fractional\_num = fraction\_result.first;

int fractional\_den = fraction\_result.second;

if (abs(fractional\_num) >= abs(fractional\_den))

whole\_part = fractional\_num / fractional\_den;

fractional\_num %= fractional\_den;

cout << "Сумма обыкновенных дробей: ";

if (whole\_part != 0)

ζ

η

θ

{

1

0

{

}

1

0

ζ

η

cout << whole\_part;

if (fractional\_num != 0)

cout << " и " << abs(fractional\_num) << "/" << abs(fractional\_den);

cout << fractional\_num << "/" << fractional\_den;

cout << endl;

θ

cout << endl;

break;

γ

}

{

}

1

0

ί

case 0:

cout << "Выход из программы." << endl;

break;

default:

cout << "Неверный выбор. Пожалуйста, выберите 1, 2 или 0." << endl;

cout << endl;

break;

while (choice != 0);

κ

return 0;

γ

}

}

}

# Код программы

#include <iostream>

#include <cmath>

#include <windows.h>

using namespace std;

// Функция для сложения десятичных дробей

double add(double num1, double num2)

{

return num1 + num2;

}

// Функция для нахождения наибольшего общего делителя (НОД)

int gcd(int a, int b)

{

while (b)

{

a %= b;

swap(a, b);

}

return a;

}

// Функция для сложения обыкновенных дробей

pair<int, int> add(int num1\_num, int num1\_den, int num2\_num, int num2\_den)

{

int result\_num = (num1\_num \* num2\_den) + (num2\_num \* num1\_den);

int result\_den = num1\_den \* num2\_den;

// Упрощение дроби

int common\_divisor = gcd(abs(result\_num), abs(result\_den));

result\_num /= common\_divisor;

result\_den /= common\_divisor;

return make\_pair(result\_num, result\_den);

}

int main()

{

// Установка русской кодировки консоли (Windows-1251)

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

int choice;

do

{

cout << "Выберите операцию:" << endl;

cout << "1. Сложение десятичных дробей" << endl;

cout << "2. Сложение обыкновенных дробей" << endl;

cout << "0. Выход" << endl;

cout << "Ваш выбор: ";

cin >> choice;

switch (choice)

{

case 1:

{

// Сложение десятичных дробей

double decimal1, decimal2, decimal\_result;

cout << "Введите первое десятичное число: ";

cin >> decimal1;

cout << "Введите второе десятичное число: ";

cin >> decimal2;

decimal\_result = add(decimal1, decimal2); // Вызов перегруженной функции для double

cout << "Сумма десятичных дробей: " << decimal\_result << endl;

cout << endl;

break;

}

case 2:

{

// Сложение обыкновенных дробей

int num1\_num, num1\_den, num2\_num, num2\_den;

pair<int, int> fraction\_result;

cout << "Введите числитель первой дроби: ";

cin >> num1\_num;

cout << "Введите знаменатель первой дроби (не 0): ";

cin >> num1\_den;

cout << "Введите числитель второй дроби: ";

cin >> num2\_num;

cout << "Введите знаменатель второй дроби (не 0): ";

cin >> num2\_den;

if (num1\_den == 0 || num2\_den == 0)

{

cout << "Знаменатель не может быть равен 0!" << endl;

}

else

{

fraction\_result = add(num1\_num, num1\_den, num2\_num, num2\_den); // Вызов перегруженной функции для int

int whole\_part = 0;

int fractional\_num = fraction\_result.first;

int fractional\_den = fraction\_result.second;

if (abs(fractional\_num) >= abs(fractional\_den))

{

whole\_part = fractional\_num / fractional\_den;

fractional\_num %= fractional\_den;

}

cout << "Сумма обыкновенных дробей: ";

if (whole\_part != 0)

{

cout << whole\_part;

if (fractional\_num != 0)

{

cout << " и " << abs(fractional\_num) << "/" << abs(fractional\_den);

}

}

else

{

cout << fractional\_num << "/" << fractional\_den;

}

cout << endl;

}

cout << endl;

break;

}

case 0:

{

cout << "Выход из программы." << endl;

break;

}

default:

{

cout << "Неверный выбор. Пожалуйста, выберите 1, 2 или 0." << endl;

cout << endl;

break;

}

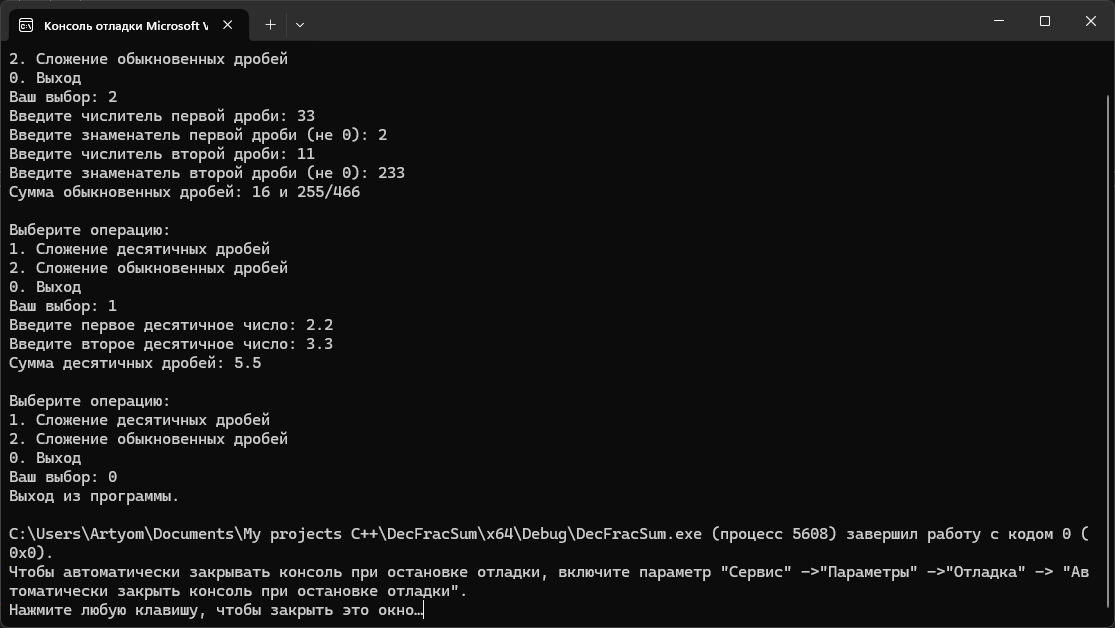
}

} while (choice != 0);

return 0;

}

# Результат



Git: <https://github.com/Artivaa/DecFracSum>

# Постановка задачи

Написать функцию max с переменным числом параметров, которая находит минимальное из чисел типа int или из чисел типа double, тип параметров определяется с помощью первого параметра функции. Написать вызывающую функцию main, которая обращается к функции min не менее трех раз с количеством параметров 5, 10, 12. Решить задачу, используя функции с переменным числом параметров.

# Анализ задачи

Задача состоит в создании функции min, которая:

* **Принимает переменное количество аргументов.** Это означает, что функция должна работать с разным числом входных значений.
* **Находит минимальное значение среди этих аргументов.** Функциональность должна быть аналогична стандартной функции min, но для произвольного набора чисел.
* **Работает с типами int и double.** Функция должна быть обобщенной и поддерживать как целые, так и вещественные числа. Тип данных определяется первым аргументом.

**2. Выбор инструментария C++:**

Для решения задачи необходимо использовать возможности C++ для работы с переменным числом аргументов и обобщенным программированием. Наиболее подходящие инструменты:

* **Шаблоны (Templates):** Для обеспечения обобщенности и работы с разными типами данных (int, double и потенциально другими числовыми типами). Шаблоны позволяют создать функцию, которая будет инстанцироваться компилятором для конкретных типов при вызове.
* **Variadic Templates (Шаблоны с переменным числом параметров):** Для обработки переменного количества аргументов. Variadic templates позволяют функции принимать произвольное число аргументов разных (или одинаковых) типов.

# Блок=схема

начало

template<typename T, typename... Args>

T min(T first, Args... args)

#include <iostream>

using namespace std;

T min\_val = first;

((min\_val = (args < min\_val) ? args : min\_val), ...);

return min\_val;

int main()

cout << min(1, 3, 2, 5, 4) << endl;

cout << min(5.2, 3.33, 9.22, 1.11, 4.2, 6.5, 2.45, 8.34, 7.04, 10.430) << endl;

cout << min(12, 15, 9, 17, 22, 11, 5, 8, 13, 10, 4, 20) << endl;

return 0;

# Код программы

#include <iostream>

using namespace std;

template<typename T, typename... Args>

T min(T first, Args... args)

{

T min\_val = first;

((min\_val = (args < min\_val) ? args : min\_val), ...);

return min\_val;

}

int main()

{

// Вызов с 5 параметрами (int)

cout << min(1, 3, 2, 5, 4) << endl;

// Вызов с 10 параметрами (double)

cout << min(5.2, 3.33, 9.22, 1.11, 4.2, 6.5, 2.45, 8.34, 7.04, 10.430) << endl;

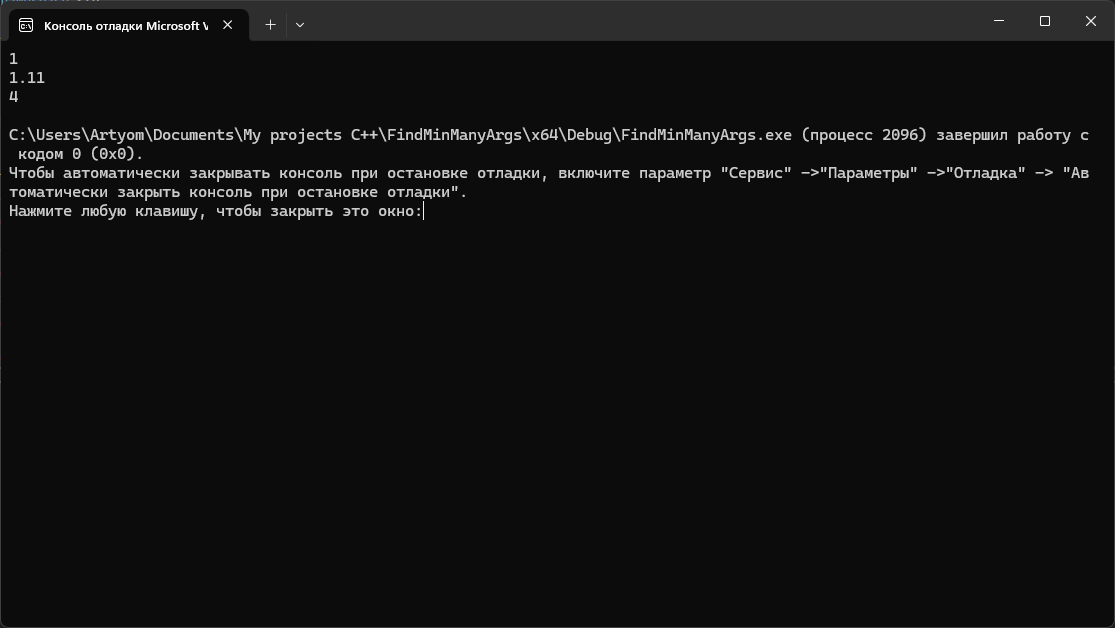
// Вызов с 12 параметрами (int)

cout << min(12, 15, 9, 17, 22, 11, 5, 8, 13, 10, 4, 20) << endl;

return 0;

}

# Результат



Git: <https://github.com/Artivaa/FindMinManyArgs>