МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

Московский технический университет связи и информатики

Кафедра «Информатика»

Лабораторный практикум по дисциплине

Введение в информационные технологии

Лабораторная работа № 4

«Разработка консольных проектов с использованием функций C++»

Выполнил:

студент группы БПС2601

Миронов М.А.

Проверил:

доц. Рюмин Ю.А.

Москва 2026

СОДЕРЖАНИЕ

1 Цель работы 3

2 Общее задание 4

3 Индивидуальное задание 6

4 Формализация и уточнение задания 7

4.1 Математическая модель 7

4.2 Типы данных и область допустимых значений 7

4.3 Способы реализации функций 8

5 Схемы алгоритмов решения задачи 9

6 Программные проекты 12

6.1 Проект 1 — функция перед main 12

6.2 Проект 2 — функция после main 13

6.3 Проект 3 — выходной параметр по ссылке 13

6.4 Проект 4 — глобальные переменные 14

6.5 Проект 5 — раздельная компиляция 15

7 Результаты выполнения проектов 17

8 Эксперименты с функциями 18

8.1 Первый проект 18

8.2 Второй проект 18

8.3 Третий проект 19

8.4 Четвёртый проект 19

9 Доказательство правильности результатов 20

10 Вывод 21

1 Цель работы

Изучить структуру консольных проектов и правила определения, объявления и вызова функций C++. Реализовать вычисление арифметического выражения с использованием различных способов передачи данных и расположения функций, а также объединить пять программных проектов в одно решение.

2 Общее задание

1. Изучить структуру программного кода консольных проектов и правила определения, объявления и вызова функций C++.
2. Выбрать индивидуальный вариант задания из таблицы 4.1 методических указаний.
3. Решить задачу вычисления заданного арифметического выражения с использованием функций C++. Для этого разработать три варианта схем алгоритмов и соответствующих процедур:
   - схему алгоритма процедуры с входными параметрами и возвращаемым значением;
   - схему алгоритма процедуры с входными и выходными параметрами и без возвращаемого значения;
   - схему алгоритма процедуры без параметров и без возвращаемого значения.

   Написать программные коды трёх функций и функции main в соответствии со схемами алгоритмов.
4. Создать консольное решение, содержащее пять проектов. Каждый проект должен содержать одну из разработанных функций и главную функцию main, в которой выполняются ввод исходных данных, вызов соответствующей функции и вывод результата. Требуются следующие способы реализации:
   - функция с параметрами и возвращаемым значением, определение которой записано перед функцией main;
   - функция с параметрами и возвращаемым значением, определение которой записано после функции main;
   - функция с параметрами и без возвращаемого значения;
   - функция без параметров и без возвращаемого значения с глобальными переменными;
   - функция с параметрами и возвращаемым значением, определения которой и функции main находятся в разных файлах.

Каждый способ должен быть реализован в отдельном проекте, а все пять проектов должны быть объединены в одном решении.

5. Выполнить созданные проекты и получить результаты. Убедиться в идентичности и правильности результатов, полученных при выполнении каждого из пяти проектов.
6. Провести эксперименты, описанные в примере выполнения задания. Внесение изменений в программный код выполнять путём комментирования исходного кода с последующим удалением комментария для возврата к исходному состоянию. После внесения изменений выполнять повторную компиляцию и перестроение решения.

3 Индивидуальное задание

Вариант 16. Создать решение из пяти консольных проектов для вычисления арифметического выражения

$$b = \frac{\log_{10} x - \sin^2(xy)}{0.8 \cdot (\ln(1-x))^2} \tag{1}$$

при $x = 0.501$ и $y = 1.7$. Вычисление реализовать при помощи функций с различными способами обмена данными и различным расположением определений. Провести эксперименты с параметрами, прототипами, ссылками и областями видимости переменных. Сравнить результаты всех проектов и проверить их правильность.

4 Формализация и уточнение задания

4.1 Математическая модель

В числителе вычисляется разность десятичного логарифма $\log_{10} x$ и квадрата синуса произведения xy. В знаменателе находится произведение числа 0.8 на квадрат натурального логарифма $\ln(1 - x)$. Аргумент тригонометрической функции задаётся в радианах.

Для проверки результата выражение представим через промежуточные значения:

$$\text{numerator} = \log_{10} x - \sin^2(xy), \tag{2}$$

$$\text{denominator} = 0.8 \cdot (\ln(1 - x))^2, \tag{3}$$

$$b = \frac{\text{numerator}}{\text{denominator}}. \tag{4}$$

В программных проектах используется следующий оператор:

```
b = (log10(x) - pow(sin(x * y), 2)) / (0.8 * pow(log(1 - x), 2));
```

4.2 Типы данных и область допустимых значений

Исходные значения x, y и результат b имеют вещественный тип double. Назначение переменных приведено в таблице 1.

Таблица 1 — Типы данных программных проектов

Переменные	Тип	Назначение
x, y	double	Исходные вещественные данные
b	double	Результат вычисления выражения

Для существования десятичного логарифма требуется $x > 0$, а для существования натурального логарифма — $1 - x > 0$, то есть $x < 1$. При выполнении этих условий знаменатель не равен нулю. Область допустимых значений:

$$0 < x < 1, \quad y \in \mathbb{R}. \tag{5}$$

Исходные данные варианта принадлежат этой области. Переполнение и ошибки ввода в учебной программе отдельно не обрабатываются.

4.3 Способы реализации функций

Назначение пяти проектов приведено в таблице 2.

Таблица 2 — Состав решения из пяти проектов

№	Функция	Способ реализации
1	func1	Параметры и возвращаемое значение; определение перед main
2	func2	Прототип перед main; определение после main
3	func3	Входные параметры по значению; результат передаётся по ссылке
4	func4	Без параметров и возвращаемого значения; глобальные переменные
5	func5	Функция и main находятся в разных файлах

5 Схемы алгоритмов решения задачи

Главная функция во всех проектах выполняет ввод x и y, вызывает функцию вычисления и выводит результат. Её блок-схема представлена на рисунке 1.

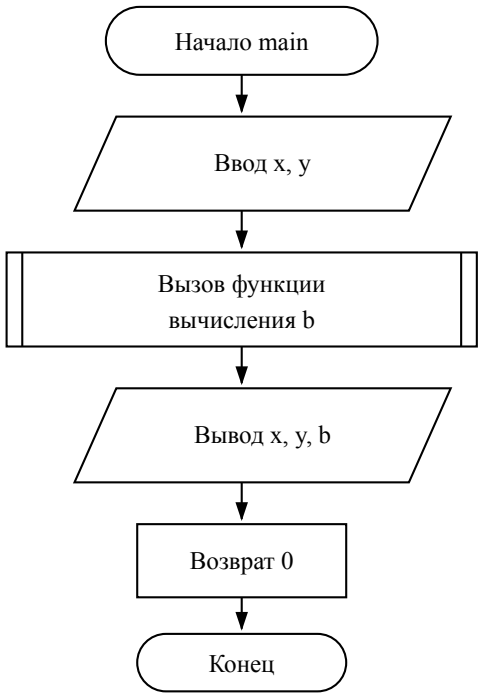


Рисунок 1 — Блок-схема главной функции main

Функции func1, func2 и func5 имеют одинаковый алгоритм: получают два входных параметра, вычисляют b и возвращают его оператором return. Схема приведена на рисунке 2.

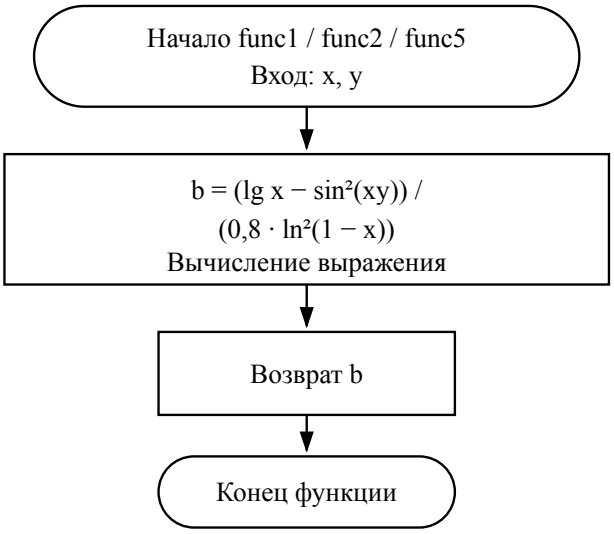


Рисунок 2 — Блок-схема функции с возвращаемым значением

Функция func3 не возвращает значение оператором return. Результат записывается в выходной параметр, переданный по ссылке. Алгоритм показан на рисунке 3.

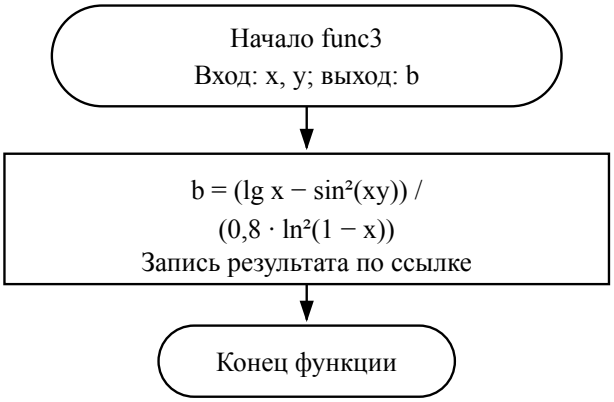


Рисунок 3 — Блок-схема функции с выходным параметром по ссылке

Функция func4 использует глобальные переменные x, y и b, поэтому параметры и возвращаемое значение не требуются. Её схема приведена на рисунке 4.

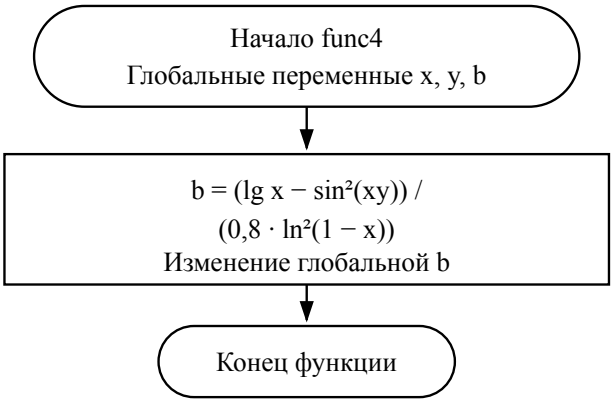


Рисунок 4 — Блок-схема функции с глобальными переменными

6 Программные проекты

Все проекты объединены в одно решение CMake. Для сборки используется стандарт C++20. Исполняемые файлы получают имена pr1, pr2, pr3, pr4 и pr5.

6.1 Проект 1 — функция перед main

В листинге 1 определение func1 расположено перед main, поэтому отдельный прототип не требуется. Функция получает копии значений x и y, вычисляет локальную переменную b и возвращает её.

```cpp
#include <clocale>
#include <cmath>
#include <iomanip>
#include <iostream>

using namespace std;

double func1(double x, double y)
{
    double b;
    b = (log10(x) - pow(sin(x * y), 2)) / (0.8 * pow(log(1 - x), 2));
    return b;
}

int main()
{
    setlocale(LC_ALL, "");

    double x, y, b;

    cout << "Введите x: ";
    cin >> x;
    cout << "Введите y: ";
    cin >> y;

    b = func1(x, y);

    cout << fixed << setprecision(10);
    cout << "x = " << x << ", y = " << y << endl;
    cout << "b = " << b << endl;

    return 0;
}
```

Листинг 1 — Исходный код первого проекта

6.2 Проект 2 — функция после main

Во втором проекте перед main записан прототип double func2(double, double);. Он сообщает компилятору имя функции, тип возвращаемого значения и типы параметров. Определение функции находится после main. Код приведён в листинге 2.

```
#include <clocale>
#include <cmath>
#include <iomanip>
#include <iostream>

using namespace std;

double func2(double, double);

int main()
{
    setlocale(LC_ALL, "");

    double x, y, b;

    cout << "Введите x: ";
    cin >> x;
    cout << "Введите y: ";
    cin >> y;

    b = func2(x, y);

    cout << fixed << setprecision(10);
    cout << "x = " << x << ", y = " << y << endl;
    cout << "b = " << b << endl;

    return 0;
}

double func2(double x, double y)
{
    double b;
    b = (log10(x) - pow(sin(x * y), 2)) / (0.8 * pow(log(1 - x), 2));
    return b;
}
```

Листинг 2 — Исходный код второго проекта

6.3 Проект 3 — выходной параметр по ссылке

Функция func3 имеет тип void. Параметры x и y передаются по значению, а b — по ссылке double&. Поэтому присваивание внутри функции изменяет переменную b в main. Исходный код представлен в листинге 3.

```cpp
#include <clocale>
#include <cmath>
#include <iomanip>
#include <iostream>

using namespace std;

void func3(double, double, double&);

int main()
{
    setlocale(LC_ALL, "");

    double x, y, b;

    cout << "Введите x: ";
    cin >> x;
    cout << "Введите y: ";
    cin >> y;

    func3(x, y, b);

    cout << fixed << setprecision(10);
    cout << "x = " << x << ", y = " << y << endl;
    cout << "b = " << b << endl;

    return 0;
}

void func3(double x, double y, double& b)
{
    b = (log10(x) - pow(sin(x * y), 2)) / (0.8 * pow(log(1 - x), 2));
}
```

Листинг 3 — Исходный код третьего проекта

6.4 Проект 4 — глобальные переменные

В четвёртом проекте переменные x, y и b объявлены вне функций. Функция func4 не имеет параметров и возвращаемого значения: она читает глобальные x и y и записывает результат в глобальную b. Код приведён в листинге 4.

```cpp
#include <clocale>
#include <cmath>
#include <iomanip>
#include <iostream>

using namespace std;

void func4();

double x, y, b;

int main()
{
    setlocale(LC_ALL, "");

    cout << "Введите x: ";
    cin >> x;
    cout << "Введите y: ";
    cin >> y;

    func4();

    cout << fixed << setprecision(10);
    cout << "x = " << x << ", y = " << y << endl;
    cout << "b = " << b << endl;

    return 0;
}

void func4()
{
    b = (log10(x) - pow(sin(x * y), 2)) / (0.8 * pow(log(1 - x), 2));
}
```

Листинг 4 — Исходный код четвёртого проекта

6.5 Проект 5 — раздельная компиляция

В пятом проекте main находится в файле main.cpp, а определение func5 — в файле func.cpp. Прототип в main.cpp позволяет вызвать функцию до компоновки объектных файлов. Содержимое файлов приведено в листингах 5 и 6.

```cpp
#include <clocale>
#include <iomanip>
#include <iostream>

using namespace std;

double func5(double, double);

int main()
{
    setlocale(LC_ALL, "");

    double x, y, b;

    cout << "Введите x: ";
    cin >> x;
    cout << "Введите y: ";
    cin >> y;

    b = func5(x, y);

    cout << fixed << setprecision(10);
    cout << "x = " << x << ", y = " << y << endl;
    cout << "b = " << b << endl;

    return 0;
}
```

Листинг 5 — Файл main.cpp пятого проекта

```cpp
#include <cmath>

using namespace std;

double func5(double x, double y)
{
    double b;
    b = (log10(x) - pow(sin(x * y), 2)) / (0.8 * pow(log(1 - x), 2));
    return b;
}
```

Листинг 6 — Файл func.cpp пятого проекта

7 Результаты выполнения проектов

Решение собрано в Linux компилятором GCC 16.2.0 с параметрами предупреждений -Wall, -Wextra и -Wpedantic. Все пять проектов скомпилированы и скомпонованы без ошибок и предупреждений. При вводе $x = 0.501$ и $y = 1.7$ получены результаты, приведённые в таблице 3.

Таблица 3 — Результаты выполнения пяти проектов

Проект	Функция	b	Код возврата
pr1	func1	−2,2408213917	0
pr2	func2	−2,2408213917	0
pr3	func3	−2,2408213917	0
pr4	func4	−2,2408213917	0
pr5	func5	−2,2408213917	0

Значения полностью совпали. Следовательно, изменение расположения функции и способа обмена данными не повлияло на результат вычисления.

8 Эксперименты с функциями

8.1 Первый проект

При замене вызова func1(x, y) на func1(y, x) результат стал NaN. В этом вызове первый параметр получил значение 1.7, поэтому нарушилось условие $x < 1$ и выражение $\ln(1 - x)$ оказалось не определено для вещественных чисел. Эксперимент показывает, что порядок фактических параметров имеет значение.

После добавления x++ перед return b значение x в main осталось равным 0.501. Параметр передаётся по значению, поэтому функция изменяет только свою локальную копию. Вызовы с выражениями и константами также выполнены; результаты приведены в таблице 4.

Таблица 4 — Вызовы функции с различными фактическими параметрами

Вызов	Результат
func1(x, y)	$b = -2.2408213917$
func1(x + 0.1, y - 0.2)	$b = -1.2381611339$
func1(0.501, 1.7)	$b = -2.2408213917$

После переименования формальных параметров x и y в a и c с соответствующим изменением формулы результат остался равным -2.2408213917. Имена формальных параметров локальны для определения функции и не связаны с именами переменных в main.

8.2 Второй проект

После комментирования прототипа func2 компиляция завершилась с ошибкой:

```
error: ‘func2’ was not declared in this scope
```

При определении функции после main прототип обязателен, поскольку при обработке вызова компилятор ещё не встретил объявление func2. После восстановления прототипа проект снова собрался без ошибок.

8.3 Третий проект

Передача выражений во входные параметры допустима, однако выражение b + 1 нельзя передать в выходной параметр типа double&. Компилятор выдал ошибку:

```
error: cannot bind non-const lvalue reference of type ‘double&’
to an rvalue of type ‘double’
```

Выражение создаёт временное значение, а неконстантная ссылка должна быть связана с изменяемой переменной. После удаления & в определении func3 проект компилируется, но результат в main не изменяется: функция получает копию b. Компилятор дополнительно сообщает, что локальная b используется без инициализации. Поэтому для выходного параметра необходима ссылка.

8.4 Четвёртый проект

При добавлении локальных переменных double x, y, b; в main эти имена скрывают глобальные переменные. Ввод выполняется в локальные x и y, но func4 продолжает использовать глобальные переменные. Функция изменяет глобальную b, тогда как выводится неинициализированная локальная b. Компилятор выдал предупреждение ‘b’ is used uninitialized. Таким образом, добавление локальных переменных нарушает обмен данными с func4.

9 Доказательство правильности результатов

Правильность проверена независимым расчётом средствами Python с использованием функций math.log10, math.sin и math.log. Для исходных данных варианта получены значения, приведённые в таблице 5.

Таблица 5 — Независимый расчёт арифметического выражения

Величина	Значение
Числитель	−0,8662699759
Знаменатель	0,3865859096
Контрольное значение b	−2,2408213917

Контрольный результат совпал с результатами всех пяти проектов с точностью до десяти знаков после запятой. Равенство результатов подтверждает правильность записи формулы, передачи параметров, возврата значения и компоновки отдельных файлов.

10 Вывод

В ходе лабораторной работы были изучены правила определения, объявления и вызова функций C++. Для варианта 16 формализовано арифметическое выражение, определена область допустимых значений и построены блок-схемы главной функции и трёх способов организации вычислений.

Создано одно решение из пяти консольных проектов. В нём реализованы функции до и после main, функция с выходным параметром по ссылке, функция с глобальными переменными и раздельная компиляция функции и main. Дополнительно проведены эксперименты с порядком и именами параметров, прототипом функции, ссылкой и областью видимости переменных.

Все проекты успешно собраны и выполнены. Получено значение $b = -2.2408213917$, одинаковое для всех способов реализации. Независимый расчёт подтвердил правильность результата, а эксперименты показали практическое значение прототипов, ссылок и областей видимости.