Методичка по C++

Переменные • Типы данных • Ввод/вывод • Операции • Условия • Логика

Цель

Эта методичка повторяет материал первого занятия. Ее можно открыть дома и быстро восстановить весь базис: как устроена программа на C++, как объявлять переменные, читать данные через cin, выводить через cout, делать математические операции и писать условия if/else с логическими операторами && и ||.

Дата: 10.02.2026

Содержание

1. Как выполняется программа на C++

C++ — компилируемый язык. Это значит, что ваш код сначала переводится компилятором в машинный код, и только потом запускается как программа. Поэтому ошибки типа данных и синтаксиса чаще всего ловятся уже на этапе компиляции.

Ключевая мысль

В C++ вы не можете «написать как-нибудь» и надеяться, что программа сама догадается. Типы данных и структура кода здесь играют решающую роль.

Мини-цепочка работы

- Пишем код (.cpp)
- Компилируем (получаем .exe / бинарник)
- Запускаем программу
- Читаем ввод, выводим результат

2. Каркас программы на C++

Почти любая учебная программа на C++ начинается с одинакового каркаса. Важно не просто переписывать его, а понимать назначение каждой строки.

2.1. Подключение библиотек (#include)

Директива #include подключает готовые возможности из стандартной библиотеки. Если библиотека не подключена, связанные функции/типы будут «неизвестны» компилятору.

```
#include <iostream>  // cin, cout, endl
#include <string>    // тип string
```

2.2. Пространство имен std

Стандартная библиотека C++ лежит в пространстве имен std. Чтобы не писать std::cout и std::cin каждый раз, на начальном этапе обычно добавляют строку ниже:

```
using namespace std;
```

> **Замечание**
> В больших проектах часто избегают using namespace std; и пишут std::cout. Но для обучения это нормально: проще читать код и быстрее писать.

2.3. Функция main — точка входа

Функция main() — это старт программы. Исполнение начинается именно здесь. Тип int означает, что функция возвращает целое число (обычно 0 — «все ок»).

```
int main() {
    // здесь ваш код
    return 0;
}
```

3. Переменные и типы данных

Переменная — это именованная область памяти. В C++ у каждой переменной есть тип, и тип задается при объявлении. Это называется строгой типизацией.

3.1. Основные типы, которые вы прошли

Тип	Для чего	Пример
int	Целые числа	int age = 18;
float	Дробные числа (обычно меньше точность)	float x = 2.5f;
double	Дробные числа (выше точность)	double pi = 3.14159;
char	Один символ	char grade = 'A';
string	Строка (текст)	string name = "Ali";
bool	Логическое значение	bool ok = true;

3.2. Объявление и инициализация

Объявить переменную — значит сказать компилятору: «создай место в памяти вот такого типа». Инициализация — это присваивание стартового значения.

```
int age;            // объявили (значение пока не задано)
age = 18;           // присвоили значение

double price = 199.99;  // объявили и сразу задали значение
```

Важно

Не используйте переменную, если вы не задали ей значение. В C++ неинициализированная переменная может содержать «мусор».

4. Ввод и вывод (cin и cout)

В учебных программах чаще всего используются потоки ввода/вывода: cin для ввода и cout для вывода.

4.1. Вывод: cout

```
cout << "Привет!" << endl;
cout << "Возраст: " << age << "\n";
```

Оператор << «вставляет» данные в поток вывода. Можно выводить текст и значения переменных, соединяя их цепочкой.

Разница между endl и \n

- endl — перенос строки + принудительная «очистка» буфера (flush). Иногда это медленнее.
- \n — просто символ переноса строки. Чаще достаточно именно его.

4.2. Ввод: cin

```
int age;
cout << "Сколько вам лет? ";
cin >> age;
```

Про пробелы во вводе

cin >> читает до первого пробела. Поэтому строка «Да Нет» будет считана как «Да». Для полного ввода строки с пробелами нужен getline, но это будет на следующих уроках.

5. Математические операции

Базовые арифметические операции в C++ работают привычно: +, -, *, /, %. Но важно помнить про типы данных: int / int дает int (целая часть).

5.1. Пример с int и double

```
int a = 10;
int b = 3;

cout << a / b << endl;      // 3 (целочисленное деление)
cout << a / 3.0 << endl;    // 3.33333... (деление на double)
```

5.2. Остаток от деления (%)

```
int x = 17;
cout << x % 2 << endl;  // 1 (нечетное)
```

Часто используется для проверки четности:

```
if (x % 2 == 0) {
    cout << "Четное";
} else {
    cout << "Нечетное";
}
```

6. Операторы сравнения

Операторы сравнения дают результат типа bool: true или false. Они используются в условиях if и в логических выражениях.

Оператор	Смысл	Пример
>	больше	a > b
<	меньше	a < b
>=	больше или равно	age >= 18
<=	меньше или равно	x <= 100
==	равно	day == 1
!=	не равно	password != 1234

Самая частая ошибка

Не путайте = и ==. Один знак равно (=) — это присваивание. Два (==) — сравнение.

7. Условия: if / else if / else

Условный оператор позволяет программе принимать решения. Если условие истинно — выполняется один блок кода, иначе — другой.

7.1. Базовый if

```
if (age >= 18) {
    cout << "Доступ разрешен";
}
```

7.2. if / else

```
if (age >= 18) {
    cout << "Доступ разрешен";
} else {
    cout << "Доступ запрещен";
}
```

7.3. Цепочка else if

Цепочка else if проверяется сверху вниз. Как только найдено истинное условие — остальные ветки не проверяются.

```
int day;
cout << "Введите день недели 1-7: ";
cin >> day;

if (day == 1) {
    cout << "Понедельник";
} else if (day == 2) {
    cout << "Вторник";
} else if (day == 3) {
    cout << "Среда";
} else if (day == 4) {
    cout << "Четверг";
} else if (day == 5) {
    cout << "Пятница";
} else if (day == 6) {
    cout << "Суббота";
} else if (day == 7) {
    cout << "Воскресенье";
} else {
    cout << "ОШИБКА: введите число от 1 до 7!";
}
```

8. Логические операторы: &&, ||, !

Логические операторы объединяют несколько условий в одно. Результат всегда bool (true/false).

8.1. && (И / AND)

Выражение A && B истинно только тогда, когда истинны и A, и B.

```
if (login == true_login && password == true_password) {
    cout << "Здравствуйте " << true_login;
} else {
    cout << "Неправильный логин или пароль";
}
```

8.2. || (ИЛИ / OR)

Выражение A || B истинно, если истинно хотя бы одно из условий.

```
if (age >= 18 || passport == "Да") {
    cout << "Доступ разрешен";
} else {
    cout << "Ты еще маленький или найди паспорт";
}
```

8.3. ! (НЕ / NOT)

```
bool hasTicket = false;

if (!hasTicket) {
    cout << "Нет билета!";
}
```

Лайфхак для чтения условий

Сначала прочитай каждое простое условие отдельно (age >= 18), затем проговори словами: «возраст 18+ И/ИЛИ паспорт Да» — и только потом смотри на код.

9. Полный пример из урока (разбор)

Ниже пример программы, которая спрашивает возраст и наличие паспорта и дает доступ по логике ИЛИ.

```
#include <iostream>
#include <string>

using namespace std;

int main() {
    int age;
    cout << "Сколько вам лет? ";
    cin >> age;

    string passport;
    cout << "У вас есть паспорт? (Да/Нет): ";
    cin >> passport;

    if (age >= 18 || passport == "Да") {
        cout << "Доступ разрешен";
    } else {
        cout << "Ты еще маленький или найди паспорт";
    }

    return 0;
}
```

Почему здесь стоит ||, а не &&?

- Требование: доступ дать, если возраст 18+ ИЛИ есть паспорт.
- Если бы было &&, то пришлось бы быть и взрослым, и с паспортом одновременно (условие стало бы жестче).

Вариант 2: логин и пароль (AND)

```
string login = "qwerty";
int password = 123;

string true_login;
int true_password;

cout << "Введите ваш логин: ";
cin >> true_login;

cout << "Введите ваш пароль: ";
cin >> true_password;

if (login == true_login && password == true_password) {
    cout << "Здравствуйте " << true_login;
} else {
    cout << "Неправильный логин или пароль";
}
```

10. Частые ошибки и чек-лист

10.1. Ошибки, которые встречаются чаще всего

- Путают = и == (присваивание vs сравнение).
- Ставят ; после if: if (x > 0); { ... } — это ломает логику.
- Пишут string без кавычек: passport == Да (нужно "Да").
- Используют переменную без ввода/инициализации (в ней может быть мусор).
- Думают, что int / int дает дробь (нет, это целая часть).

10.2. Чек-лист перед запуском

- Подключены ли нужные библиотеки (#include , #include)?
- Есть ли using namespace std; (или использованы std::)?
- Внутри main() объявлены все переменные?
- Каждой переменной задано значение до использования (через cin или =)?
- В условиях использованы правильные операторы (==, !=, >=, &&, ||)?

11. Практика (10 задач)

Задачи сделаны так, чтобы повторить ввод/вывод, математику, сравнения и логику. Решай строго на C++ с cin/cout.

11.1. Математика и переменные (1-5)

- Сумма: Ввести a и b. Вывести a + b.
- Арифметика: Ввести a и b. Вывести a + b, a - b, a * b, a / b (для дроби делай деление через double).
- Среднее: Ввести 3 числа. Вывести среднее арифметическое.
- Конвертация: Ввести метры. Вывести сантиметры.
- Прямоугольник: Ввести ширину и высоту. Вывести площадь и периметр.

11.2. Условия и логика (6-10)

- Знак числа: Ввести x. Вывести Positive / Negative / Zero.
- Возраст: Ввести age. Если age >= 18 — доступ разрешен, иначе — запрещен.
- Четность: Ввести x. Если x % 2 == 0 — Even, иначе — Odd.
- Диапазон (&&): Ввести x. Если x в [10; 100] — In range, иначе — Out of range. Обязательно используй &&.
- Проверка (||): Ввести x. Если x < 0 или x > 100 — Invalid value, иначе — Valid value. Обязательно используй ||.

Совет для самопроверки

Проверяй каждую задачу на 3-5 разных вводах (включая границы: 0, 1, -1, 10, 100, 101). Это быстро показывает ошибки в условиях.