



Урок 25. C++: вещественные числа и точность

Часть 1. Материалы ученика

Программирование на Python и C++ · 7 класс

Урок 25 из 34 · 45 минут · Модуль «Числа: представление и вывод»

Уровень: начинающий программист; C++17.

Программа: <https://schoolotzyv.ru/blog/python-cpp-7-klass-programma>

Программа печатает 0.30, но внутри хранится приближение. Означают ли две цифры на экране точность вычисления?

Результат урока: Различить целое и вещественное деление, объяснить округление и выбрать формат под задачу.

01. Проблемная ситуация

За 5 минут предскажите $7/2$ и $7.0/2$. Затем сравните утверждения «выведено 0.30» и «в памяти точно 0.30». Запишите, какую проверку вывода и какую проверку вычислений вы бы сделали.

02. Понятия и разобранный пример

5–14 минут: 4 минуты на понятия, 3 на пример, 2 на объяснение способа партнёру.

Типы операндов и формат не одно и то же

При двух целых операндах / даёт целое частное: $7/2$ равно 3. Хотя результат потом присвоить `double`, потерянная половина не вернётся. В $7.0/2$ участвует вещественный операнд, получается 3.5. Типы определяются до присваивания результата.

В распространённом двоичном формате `double` число 0.5 представимо точно, а 0.1 — конечным двоичным приближением. На используемой сборке с `binu64` сумма 0.1 и 0.2 при выводе 17 значащих цифр видна как 0.30000000000000004. Это наблюдение о данной реализации; не обещаем одинаковую последнюю цифру для любого формата `double`.

`fixed` вместе с `setprecision(2)` задаёт две цифры после десятичной точки при выводе, а не изменяет хранимую переменную. Без `fixed` `setprecision` задаёт значащие цифры. Для дискретных сотых, например учебных цен в копейках, разумно хранить целое число сотых и проверять верхнюю границу произведения. Ввод вещественных тестов здесь — конечные числа с точкой, не NaN/inf.

Отступы набирайте обычными пробелами: 4 на уровень. При копировании с веб-страницы замените неразрывные пробелы обычными; полный код с обычными пробелами есть в PDF.

На проверенной сборке первые две строки: 0.30000000000000004 и 0.30. Последняя строка: 3 3.50, потому что `fixed` и точность 2 остаются активными. Число 3 слева целое и этим форматом не меняется.

```
#include <iostream>
#include <iomanip>

int main() {
    double a = 0.1, b = 0.2;
    std::cout << std::setprecision(17) << a + b << "\n";
    std::cout << std::fixed << std::setprecision(2)
              << a + b << "\n";
    std::cout << 7 / 2 << " " << 7.0 / 2 << "\n";
}
```



C++: вещественные числа и точность

Целые операнды	$7/2 = 3$
Вещественный операнд	$7.0/2 = 3.5$
Формат вывода	fixed, precision 2 $\rightarrow$ 3.50
Контроль смысла	Единицы и диапазон проверяются отдельно

$7/2 \rightarrow 3$; $7.0/2 \rightarrow 3.5$. Формат fixed с точностью 2 меняет вид вывода, не исходную величину.

03. Практика 1. Проверяем материал

Скорость с переводом сантиметров в метры

14–25 минут. Программа принимает конечные cm и seconds: $0 \leq \text{cm} \leq 1000$, $0 < \text{seconds} \leq 1000$. Выведите скорость в м/с с двумя знаками.

Сантиметры и секунды	Вывод скорости, м/с
230 3	0.77
100 4	0.25
0 5	0.00
1000 1	10.00
100 0	error

2 минуты — единицы и прогноз таблицы; 4 — программа; 3 — все тесты и отрицательное время; 2 — объясните, почему cm/seconds не даёт м/с. Продукт: исходник и журнал. Критерии: перевод /100.0, ненулевое время, fixed и два знака, корректные границы.

04. Практика 2. Перенос способа

Точные сотые единицы через целые числа

25–33 минуты. Учебный заказ: count одинаковых единиц, цена каждой cents копеек. Ввод два целых 0...100000. Требуется точная сумма в рублях с двумя цифрами копеек. Не используйте double. За 2 минуты оцените максимум, за 4 напишите код, за 2 проверьте 3 105, 1 7, 0 999 и 100000 100000. Критерии: long long до умножения, нулевое дополнение копеек, точный итог.

05. Обсуждение и границы вывода

33–40 минут. Можно ли исправить ошибку единиц только setprecision(2)? Сравните две причины: 230/3 вместо 230/100/3 и вывод длинного приближения для правильного вычисления. Какой тест различит эти ошибки?

06. Индивидуальная проверка

40–45 минут. Укажите результаты $9/4$ и $9.0/4$; запишите скорость для 150 см за 2 с с двумя знаками; объясните, делает ли fixed вычисления точными. По одному баллу за четыре элемента.



Часть 2. Ответы и материалы учителя

Практика 1: разбор и критерии

$230/100/3=0.7666\dots$ м/с, на экране 0.77. 100 см за 4 с дают 0.25 м/с. Нулевое расстояние допустимо, нулевое/отрицательное время нет. Полный код ниже. Округлённые 0.77 не подставляют вместо точного промежуточного результата в дальнейший расчёт.

```
#include <iostream>
#include <iomanip>

int main() {
    double cm, seconds;
    if (!(std::cin >> cm >> seconds) || cm < 0 || cm > 1000
        || seconds <= 0 || seconds > 1000) {
        std::cout << "error\n"; return 0;
    }
    double v = cm / 100.0 / seconds;
    std::cout << std::fixed << std::setprecision(2) << v << "\n";
}
```

Практика 2: разбор и критерии

Ответы: 3.15; 0.07; 0.00; 100000000.00. Максимум 10^{10} копеек, поэтому long long нужен уже у множителей. setw(2) задаёт ширину следующего поля, setfill('0') заполняет ведущий разряд. Это формат учебных целых цен, не финансовая рекомендация.

```
#include <iostream>
#include <iomanip>

int main() {
    long long count, cents;
    if (!(std::cin >> count >> cents) || count < 0 || count > 100000
        || cents < 0 || cents > 100000) {
        std::cout << "error\n"; return 0;
    }
    long long total = count * cents;
    std::cout << total / 100 << "." << std::setw(2)
        << std::setfill('0') << total % 100 << "\n";
}
```

Обсуждение: возможный вывод

Формат не исправляет масштаб: ошибочные 76.67 и правильные 0.77 различаются из-за единиц. Длинное представление не всегда означает плохой алгоритм, а красивый короткий вывод не доказывает правильность. Тест 100 см за 1 с должен дать 1.00 м/с.

Индивидуальная проверка: ответ и критерии

2; 2.25; 0.75 м/с; нет, fixed задаёт формат вывода. Полный ответ 4 балла.

Учителю: подготовка и план на 45 минут

Подготовьте компьютер, рабочую папку с правом записи, редактор, распечатку ученической части PDF, бумагу и ручку. Локальный компилятор G++ с C++17 или настроенная учителем IDE; сборка g++ -std=c++17 -Wall -Wextra -pedantic main.cpp -o lesson. В Windows задайте lesson.exe. Запуск ./lesson в Linux/macOS, .\lesson.exe в Windows. Для урока 31 дополнительно обычный браузер: открывать локальный SVG, не C++-исходник. Все входы и ожидаемые результаты даны. Установку и проверку запуска учитель выполняет до занятия; детские аккаунты и интернет не нужны.

Уровень — начинающий программист, 7 класс. Предпосылки: целое деление, типы int и long long, единицы длины и времени. В паре автор вводит и объясняет, проверяющий сопоставляет с прогнозом, затем роли меняются. Без компьютера трассируйте полный код на бумаге, отмечайте координаты по схеме и проверяйте готовые результаты; учитель демонстрирует заранее собранный пример. Бумажная трассировка не выдаётся за реальную компиляцию или запуск ученика. План 45 минут включает чтение, ввод, сборку и обсуждение. В графических задачах сначала прогноз, затем сравнение. Матрица программы предназначена учителю.



Минуты	Работа	Результат
0–5	Проблема	Вопрос и первое решение
5–14	Понятия и пример	Способ проверки
14–25	Практика 1	Готовый продукт и проверка
25–33	Практика 2	Перенос способа
33–40	Обсуждение	Аргумент и ограничение
40–45	Индивидуальная проверка	Самостоятельный ответ

07. Соответствие исходной программе

Пункт программы	Действие ученика	Проверяемый результат
Обработка вещественных чисел. Особенности представления в памяти	Различить целое и вещественное деление, объяснить округление и выбрать формат под задачу.	Самостоятельный продукт и тесты, описанные в практиках
Составлять и отлаживать простые диалоговые программы	Сверяет прогноз с фактическим запуском; объясняет ошибочный вариант	Журнал тестов и индивидуальный ответ по критериям

Источники и права на изображения

Исходная программа курса

<https://schoolotzyv.ru/blog/python-cpp-7-klass-programma>

C++: форматирование потоков

<https://eel.is/c++draft/iostream.format>

C++: вещественные типы

<https://eel.is/c++draft/basic.fundamental>

Тексты, учебные данные, реплики, программы и схемы подготовлены для уроков портала «Школы России». Числовые серии — учебные модели, не результаты проведённых классом опытов. Схемы авторские, сторонние фотографии не используются. Методические ссылки приведены для учителя; ученик получает все материалы здесь. Проверка первичных источников: 17.10.2026.