



Урок 26. C++: условие и цикл с предусловием

Часть 1. Материалы ученика

Программирование на Python и C++ · 7 класс

Урок 26 из 34 · 45 минут · Модуль «Циклы: состояние и завершение»

Уровень: начинающий программист; C++17.

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

При цели 1 начальное значение уже подходит. Почему цикл, который всегда делает первый шаг, даёт лишнюю работу?

Результат урока: Записать `while` с проверкой до тела, проследить состояние и обосновать завершение.

01. Проблемная ситуация

Начинаем с 1 и удваиваем до значения не меньше цели. За 5 минут выпишите состояния для целей 1, 8 и 9. Отдельно посчитайте выполненные удвоения, а не количество записанных чисел.

02. Понятия и разобранный пример

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

Условие проверяется перед каждой итерацией

`while` (условие) повторяет тело, пока условие истинно; первый тест предшествует первому телу. Если начальное состояние уже подходит, тело выполняется ноль раз. Фигурные скобки объединяют обновления; точка с запятой сразу после `while` создала бы пустое тело.

В `doubling value` хранит текущее число, `steps` — количество удвоений. Условие `value < target` продолжает работу только при недостаточном `value`. После выхода `value ≥ target`. Предыдущее значение, если шаги были, меньше `target`: это проверка минимальности.

Ввод `target` ограничен 1...1000000. Последнее удвоенное значение не больше 1048576, оно помещается в `int` используемой сборки с минимум 32 битами. Если использовать 16-битный `int`, этот контракт требует другого типа. На нашей сборке проверено 32 бита или больше; переносимый выбор здесь можно заменить на `long long`. Рост положительного `value` гарантирует достижение ограниченной цели.

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

Для `target=9`: 1→2→4→8→16, четыре обновления. Вывод 4 16. Для 8 — 3 8; для 1 — 0 1. Ошибка `<=` даёт лишнее удвоение на точной степени двойки.

```
#include <iostream>
```

```
int main() {
    int target;
    if (!(std::cin >> target) || target < 1 || target > 1000000) {
        std::cout << "error\n"; return 0;
    }
    int value = 1, steps = 0;
    while (value < target) {
        value *= 2;
        ++steps;
    }
    std::cout << steps << " " << value << "\n";
}
```



C++: условие и цикл с предусловием

Старт	value=1, steps=0
Проверка	value < target?
Обновление	value*=2; steps++
Выход	value≥target; лишнего шага нет

Цель 9: состояния 1,2,4,8,16; сравнение идёт до обновления, на выходе 4 шага.

03. Практика 1. Проверяем материал

Первое достижение цели удвоением

14–25 минут. Реализуйте удвоение по описанию, закрыв код примера до первой попытки.

Цель	Вывод steps value
1	0 1
2	1 2
8	3 8
9	4 16
1000000	20 1048576
0	error

2 минуты — трассировка 9; 4 — код; 3 — таблица тестов; 2 — объяснение остановки и минимальности. Продукт: исходник, фактические выводы и состояние перед последним обновлением. Критерии: ноль шагов для 1, точная граница 8, корректное увеличение счётчика, понятная причина завершения.

04. Практика 2. Перенос способа

Убывающее состояние и округление половины

25–33 минуты. Новый контракт: value от 0 до 1000000, limit от 1 до 1000000. Пока value больше limit, заменяйте его половиной с округлением вверх: $(value+1)/2$. Выведите число замен и итог. За 2 минуты трассируйте 13 3, за 4 реализуйте, за 2 проверьте 12 3, 0 1, 5 5 и 5 0. Критерии: сравнение >, округление вверх, отсутствие лишнего шага.

05. Обсуждение и границы вывода

33–40 минут. Почему фраза «в цикле есть счётчик» не доказывает завершение? Сравните удвоение и деление пополам: какая величина приближается к выходу и почему не может застрять?

06. Индивидуальная проверка

40–45 минут. Для цели 17 назовите число удвоений, итог и предыдущее состояние. Для половинного алгоритма объясните запрет limit=0. Четыре пункта по одному баллу.



Часть 2. Ответы и материалы учителя

Практика 1: разбор и критерии

Ответы в таблице. Для 9 предыдущее состояние $8 < 9$, итог $16 \geq 9$. Удаление $value *= 2$ оставит условие истинным бесконечно, если $target > 1$. Счётчик сам по себе не меняет проверяемый $value$. Полный исходник совпадает с разобранным вариантом.

```
#include <iostream>
```

```
int main() {
    int target;
    if (!(std::cin >> target) || target < 1 || target > 1000000) {
        std::cout << "error\n"; return 0;
    }
    int value = 1, steps = 0;
    while (value < target) {
        value *= 2;
        ++steps;
    }
    std::cout << steps << " " << value << "\n";
}
```

Практика 2: разбор и критерии

$13 \rightarrow 7 \rightarrow 4 \rightarrow 2$: ответ 3 2. Для $12 \ 3 \rightarrow 2 \ 3$; $0 \ 1 \rightarrow 0 \ 0$; $5 \ 5 \rightarrow 0 \ 5$; $5 \ 0 \rightarrow$ error. При $value > limit \geq 1$ имеем $value \geq 2$ и новая половина строго меньше старой. Поэтому состояние убывает до остановки. $(value + 1)$ не переполняется в данном диапазоне.

```
#include <iostream>
```

```
int main() {
    int value, limit;
    if (!(std::cin >> value >> limit) || value < 0 || value > 1000000
        || limit < 1 || limit > 1000000) {
        std::cout << "error\n"; return 0;
    }
    int steps = 0;
    while (value > limit) {
        value = (value + 1) / 2;
        ++steps;
    }
    std::cout << steps << " " << value << "\n";
}
```

Обсуждение: возможный вывод

Для роста нужны положительный старт и конечная цель; для уменьшения — строгое убывание при истинном условии. При $limit = 0$ округление вверх удержало бы 1 навсегда, поэтому такой $limit$ отвергается. Доказательство относится к контракту, а не к виду ключевого слова.

Индивидуальная проверка: ответ и критерии

5 шагов, итог 32, предыдущее 16. При $limit = 0$ состояние 1 обновляется снова в 1 и цикл не остановится.

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: сравнения, `if`, присваивание и целое деление. 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
Циклы с условием. Цикл с предусловием	Записать <code>while</code> с проверкой до тела, проследить состояние и обосновать завершение.	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>

Рабочий текст стандарта C++: циклы

<https://eel.is/c++draft/stmt.iter>

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.