



Урок 30. C++: процедура и параметры

Часть 1. Материалы ученика

Программирование на Python и C++ · 7 класс

Урок 30 из 34 · 45 минут · Модуль «Процедуры: повторяемое действие»

Уровень: начинающий программист; C++17.

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

Заголовки отчёта требуют разные разделительные полосы. Как изменить правило перевода строки в одном месте, а не в каждой копии?

Результат урока: Выделить печать строки в void-процедуре, передать ширину и символ и объяснить передачу по значению.

01. Проблемная ситуация

Нужны две строки: `###` и `*****`. За 5 минут опишите общее действие и два изменяемых параметра. Решите, относится ли перевод строки к каждому символу или ко всей готовой полосе.

02. Понятия и разобранный пример

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

Определение функции и её вызов

В C++ слово `void` указывает, что функция не возвращает значение результата через `return`. Наша процедура всё же выполняет действие — вывод. Определение `line` размещено до `main`, поэтому компилятор знает имя и параметры к моменту вызовов.

`width` типа `int` и `mark` типа `char` — локальные параметры. Аргументы вызова вычисляются и передаются по значению: функция получает копии. Символ записывают в одинарных кавычках, например `'#'`; строку — в двойных. Цикл печатает `width` символов, затем один перевод строки.

Определение не рисует полосы само по себе. Каждый вызов выполняет тело заново со своими аргументами. Внутри `for` используется `i < width`: для ширины 3 индексы 0,1,2 дают три символа. Контракт вызывающей части проверяет положительную ширину; не обещаем автоматической проверки самой `line`.

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

Первый вызов `line(3,'#')` печатает `###` и перевод строки; второй `line(5,'*')` печатает `*****`. Если `cout << "\n"` поместить внутрь цикла, вместо горизонтальной полосы получится столбец.

```
#include <iostream>
void line(int width, char mark) {
    for (int i = 0; i < width; ++i) std::cout << mark;
    std::cout << "\n";
}
int main() {
    line(3, '#');
    line(5, '*');
}
```

C++: процедура и параметры

Определение	<code>line(width,mark)</code>
Первый вызов	<code>3 и '#' → ###</code>
Второй вызов	<code>5 и '*' → *****</code>
Общий шаг	один перевод строки после полосы

Вызовы `line(3,'#')` и `line(5,'*')` используют одно тело, но разные локальные значения `width` и `mark`.



03. Практика 1. Проверяем материал

Прямоугольник как несколько вызовов

14–25 минут. Ввод $w=1\dots 20, h=1\dots 10$. Напечатайте прямоугольник из #, пользуясь готовой `line`, без копирования внутреннего цикла в `main`.

w h	Проверяемый результат
1 1	одна строка #
4 2	две строки ####
20 10	10 строк по 20 символов
0 2	error

2 минуты — прогноз строк и символов; 4 — код; 3 — таблица тестов; 2 — объяснение двух разных циклов. Продукт: исходник и журнал числа строк/длины. Критерии: одно определение `line`, `h` вызовов, ширина `w`, отказ вне контракта.

04. Практика 2. Перенос способа

Копия параметра не изменяет исходную ширину

25–33 минуты. Ввод `w` от 3 до 20. Выведите `line(w, '+')`, затем `line(w-2, '-')`, затем `width=` с исходным `w`. За 2 минуты прогноз для 5; за 4 код; за 2 тесты 3,5,20,2. Поясните, меняет ли выражение `w-2` само `w`. Критерии: две разные длины, общий инструмент, исходная переменная сохранена.

05. Обсуждение и границы вывода

33–40 минут. Когда короткая повторяемая процедура полезнее трёх копий? Предложите изменение общего правила и тест, который проверяет, что изменение коснулось всех вызовов. Можно ли считать любую новую функцию улучшением?

06. Индивидуальная проверка

40–45 минут. Для `w=6` предскажите `line(w-2, '@')` и значение `w` после вызова. Назовите роль `void` и место перевода строки. Четыре пункта.



Часть 2. Ответы и материалы учителя

Практика 1: разбор и критерии

Для 4 2 две строки #####. Внешний row считает строки, внутренний i — символы. Всего w·h символов #, но h переводов строки. Возврат значения от line для этой задачи не нужен. Полный код ниже.

```
#include <iostream>
void line(int width, char mark) {
    for (int i = 0; i < width; ++i) std::cout << mark;
    std::cout << "\n";
}
int main() {
    int w, h;
    if (!(std::cin >> w >> h) || w < 1 || w > 20
        || h < 1 || h > 10) {
        std::cout << "error\n"; return 0;
    }
    for (int row = 0; row < h; ++row) line(w, '#');
}
```

Практика 2: разбор и критерии

Для 5: +++++, затем ---, затем width=5. Для 3 средняя строка содержит один -. Выражение w-2 лишь вычисляет аргумент; присваивания w нет. Параметр width локален каждому вызову. Значение 2 отвергается, чтобы width-2 оставалась положительной.

```
#include <iostream>
void line(int width, char mark) {
    for (int i = 0; i < width; ++i) std::cout << mark;
    std::cout << "\n";
}
int main() {
    int w;
    if (!(std::cin >> w) || w < 3 || w > 20) {
        std::cout << "error\n"; return 0;
    }
    line(w, '+');
    line(w - 2, '-');
    std::cout << "width=" << w << "\n";
}
```

Обсуждение: возможный вывод

Если перевод строки или правило символов меняется, одно тело сокращает риск несогласованных копий. Но чрезмерно общая процедура с непонятными параметрами ухудшает ясность. Проверяют каждый вызов с разными аргументами и не смешивают число строк с шириной.

Индивидуальная проверка: ответ и критерии

@@@@; w остаётся 6; void означает отсутствие возвращаемого значения; перевод строки выполняется после цикла символов. Каждый элемент 1 балл.

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: for, целые значения и символ char. 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
Процедуры. Процедуры с параметрами	Выделить печать строки в void-процедуру, передать ширину и символ и объяснить передачу по значению.	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++: определение функции

<https://eel.is/c++draft/dcl.fct.def.general>

C++: вызов и аргументы

<https://eel.is/c++draft/expr.call>

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.