



Урок 4. C++: Контракт процедуры и параметры

Часть 1. Материалы ученика

Программирование на Python и C++ · 8 класс

Урок 4 из 33 · 45 минут · Модуль «Документирование программы. Процедуры с параметрами»

Уровень: начинающий программист; C++17.

Программа: <https://schoolotzyv.ru/blog/python-cpp-8-klass-programma>

Печать разделителя кажется мелочью, пока один вызов не печатает лишний символ, а другой забывает перевод строки. Зафиксируем точный контракт процедуры.

Результат урока: Описать параметры и побочный эффект void-процедуры, проверить нулевое повторение и отличить её от функции с результатом.

01. Проблемная ситуация

0–5 минут. Процедуре поручили «печатать линию из n знаков». Ученик для $n=0$ не печатает ничего; другой печатает пустую строку. Кто прав? Запишите, какой пункт контракта снимает неоднозначность. В нашем варианте перевод строки обязателен даже при $n=0$.

02. Понятия и разобранный пример

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

void означает отсутствие возвращаемого значения

void-процедура в C++ не возвращает значение, пригодное для присваивания. Она может выполнять полезное действие, например печатать. Контракт должен описывать это действие точно: число символов, их порядок и перевод строки. Комментарии над объявлением не исполняются и не заменяют проверку входа.

В `print_line(int n, char mark)` параметры `n` и `mark` являются локальными копиями аргументов. Вызов `print_line(4, '#')` связывает `n` с 4, `mark` с одним символом `#`. Символ пишется в одинарных кавычках; строка в двойных кавычках имеет другой тип и не является аргументом `char`.

Цикл `i=0; i<n` выполняется n раз для $n \geq 0$. При $n=0$ тело не выполняется, но следующий за циклом вывод перевода строки остаётся. Контракт допускает только n от 0 до 20; текущая процедура доверяет вызывающему коду. Ввод пользователя, если он есть, нужно проверять до вызова.

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 печатает `####` и перевод строки; второй — пустую строку. Всего две строки вывода. Попытка `int result=print_line(4, '#')`; не компилируется: у void нет возвращаемого числа. Не исправляйте это случайным `return 0`: сначала решите, нужен ли результат.

```
#include <iostream>
// n: целое 0...20; mark: один печатаемый символ.
// Печатает n символов mark и перевод строки; без ввода.
void print_line(int n, char mark) {
    for (int i=0; i<n; ++i) std::cout << mark;
    std::cout << '\n';
}

int main() {
    print_line(4, '#');
    print_line(0, '*');
}
```



Аргументы	4 и символ #
Цикл	$i = 0, 1, 2, 3$
Вывод	#### и перевод строки
Граница $n=0$	Пустая строка, не ноль строк

`print_line(4, '#')`: четыре повторения и один перевод строки; при $n=0$ остаётся только перевод строки.

03. Практика 1. Проверяем материал

Два разных символа и нулевой случай

14–25 минут. Напишите `print_pair(n, left, right)`: печатает n пар символов и один перевод строки. n целое $0 \dots 10$, `left/right` — печатаемые `char`. Для `(3, '<', '>')` нужна строка `<><><>`.

Вызов	Точный вид строки
<code>print_line(1, '*')</code>	*
<code>print_line(3, '+')</code>	+++
<code>print_line(0, '.')</code>	Пустая строка
<code>print_line(20, '#')</code>	20 знаков #

2 минуты — контракт; 4 — код; 3 — тесты $n=0, 1, 3, 10$ с символами `<` и `>`; 2 — подсчёт длины без перевода строки. Продукт: исходник и четыре строки журнала. Критерии: $2n$ символов, верный порядок пары, отдельный перевод строки, отсутствие внутреннего ввода.

04. Практика 2. Перенос способа

Функция вычисляет, процедура показывает

25–33 минуты. Нужна функция `pair_length(n)`, возвращающая число напечатанных символов без перевода строки при n от 0 до 10. Она ничего не печатает. За 2 минуты оформите контракт, за 4 реализуйте, за 2 проверьте $n=0, 1, 3, 10$. Объясните различие типов `void` и `int`.

05. Обсуждение и границы вывода

33–40 минут. Должна ли `print_pair` сама возвращать $2*n$? Одна пара предлагает совместить печать и вычисление; другая разделяет их. Сравните удобство тестов и возможность использовать длину без печати. Обоснуйте решение для библиотеки подпрограмм.

06. Индивидуальная проверка

40–45 минут. Для `print_pair(4, '[', ']')` назовите строку, число символов без перевода, число повторений цикла и тип функции `pair_length`. Четыре ответа — четыре балла.



Часть 2. Ответы и материалы учителя

Практика 1: разбор и критерии

Длины 0,2,6,20. Для $n=1$ строка $<>$; $n=3$ — $<><><>$. Условие $i \leq n$ ошибочно добавит пару, особенно заметную при $n=0$. Контракт печати не обещает возвращать длину.

```
#include <iostream>
#include <initializer_list>
// n в 0...10. Печатает n пар left,right и перевод строки.
void print_pair(int n, char left, char right) {
    for (int i=0; i<n; ++i) std::cout << left << right;
    std::cout << '\n';
}

int main() {
    for (int n : {0,1,3,10}) print_pair(n, '<', '>');
}
```

Практика 2: разбор и критерии

Функция возвращает $2*n$: 0,2,6,20. Печать делает main. Так длину можно использовать в следующем расчёте, не разбирая текст на экране.

```
#include <iostream>
#include <initializer_list>
// Вернуть число символов в n парах; n в 0...10.
int pair_length(int n) { return 2*n; }

int main() {
    for (int n : {0,1,3,10})
        std::cout << pair_length(n) << "\n";
}
```

Обсуждение: возможный вывод

Совмещать действия допустимо при явном контракте, но чистая функция длины удобна без побочного вывода. Для учебной библиотеки разделение делает обязанности яснее. Важна согласованность типа и обещанного поведения, а не запрет на все комбинированные функции.

Индивидуальная проверка: ответ и критерии

`[ ] [ ] [ ]`; 8; 4; int. Пустой вызов $n=0$ печатает одну пустую строку. При ошибке в повторениях протрассируйте $i=0...3$.

Учителю: подготовка и план на 45 минут

Подготовьте компьютер, редактор, папку с правом записи, распечатку ученической части PDF, бумагу и ручку. Компилятор G++ с поддержкой C++17 и редактор. Сборка: `g++ -std=c++17 -Wall -Wextra -pedantic main.cpp -o lesson`. Запуск `./lesson` в Linux/macOS или `.\lesson.exe` в Windows (при сборке задайте `-o lesson.exe`). Установку и проверку запуска учитель выполняет заранее. Все программы, данные и ожидаемые ответы даны; регистрация детей и интернет на занятии не нужны.

Уровень — начинающий программист, 8 класс. Предпосылки: `void`, вызов процедуры, `for` и символьные литералы. В паре автор прогнозирует и вводит код, проверяющий ведёт журнал, затем роли меняются. Без компьютера выполните трассировку полного кода на бумаге и сопоставьте с готовыми результатами; учитель демонстрирует подготовленный запуск. Бумажный вариант не считается фактическим запуском ученика. Перед практикой учитель выдаёт готовый файл разобранного примера, чтобы дети тратили время на изменение и проверку, а не длинное перепечатывание. 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
Документирование программы. Процедуры с параметрами	Разбирает пример и создаёт две подпрограммы	Исходники, прогнозы и протокол проверки
Составление и отладка программ	Сравнивает фактический результат с контрактом	Объяснение границ и самостоятельный ответ

Источники и права на изображения

Исходная программа курса

<https://schoolotzyv.ru/blog/python-cpp-8-klass-programma>

C++: возвращение из функции

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

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: 18.10.2026.