

Ravesli [Ravesli](#)


- [Уроки по C++](#)
- [OpenGL](#)
- [SFML](#)
- [Qt5](#)
- [RegExr](#)
- [Ассемблер](#)
- [Купить .PDF](#)


Урок №14. Почему функции — полезны, и как их эффективно использовать?

 [Юрий](#) |

- [Уроки C++](#)

|

 Обновл. 19 Сен 2020 |

 46389

[17](#)

Теперь, когда мы уже знаем, что такое [функции](#) и зачем они нужны, давайте более подробно рассмотрим, почему они так полезны.

Зачем использовать функции?

Начинающие программисты часто спрашивают: «А можно ли обходиться без функций и весь код помещать непосредственно в функцию `main()`?». Если вашего кода всего 10-20 строк, то можно. Если же серьезно, то функции предназначены для упрощения кода, а не для его усложнения. Они имеют ряд преимуществ, которые делают их чрезвычайно полезными в нетривиальных программах.

➔ **Структура.** Как только программы увеличиваются в размере/сложности, сохранять весь код внутри `main()` становится трудно. Функция — это как мини-программа, которую мы можем записать отдельно от головной программы, не заморачиваясь при этом об остальных частях кода. Это позволяет разбивать сложные задачи на более мелкие и простые, что кардинально снижает общую сложность программы.

➔ **Повторное использование.** После объявления функции, её можно вызывать много раз. Это позволяет избежать дублирования кода и сводит к минимуму вероятность возникновения ошибок при копировании/вставке кода. Функции также могут использоваться и в других программах, уменьшая объем кода, который нужно писать с нуля каждый раз.

➔ **Тестирование.** Поскольку функции убирают лишний код, то и тестировать его становится проще. А так как функция — это самостоятельная единица, то нам достаточно протестировать её один раз, чтобы убедиться в её работоспособности, а затем мы можем её повторно использовать много раз без необходимости проводить тестирование (до тех пор, пока не внесем изменения в эту функцию).

- ➔ **Модернизация.** Когда нужно внести изменения в программу или расширить её функционал, то функции являются отличным вариантом. С их помощью можно внести изменения в одном месте, чтобы они работали везде.
- ➔ **Абстракция.** Для того, чтобы использовать функцию, нам нужно знать её имя, данные ввода, данные вывода и где эта функция находится. Нам не нужно знать, как она работает. Это очень полезно для написания кода, понятного другим (например, Стандартная библиотека C++ и всё, что в ней находится, созданы по этому принципу).

Каждый раз, при вызове `std::cin` или `std::cout` для ввода или вывода данных, мы используем функцию из Стандартной библиотеки C++, которая соответствует всем вышеперечисленным концепциям.

Эффективное использование функций

Одной из наиболее распространенных проблем, с которой сталкиваются новички, является понимание того, где, когда и как эффективно использовать функции. Вот **несколько основных рекомендаций при написании функций**:

- ➔ *Рекомендация №1:* Код, который появляется более одного раза в программе, лучше переписать в виде функции. Например, если мы получаем данные от пользователя несколько раз одним и тем же способом, то это отличный вариант для написания отдельной функции.
- ➔ *Рекомендация №2:* Код, который используется для сортировки чего-либо, лучше записать в виде отдельной функции. Например, если у нас есть список вещей, которые нужно отсортировать — пишем функцию сортировки, куда передаем несортированный список и откуда получаем отсортированный.
- ➔ *Рекомендация №3:* Функция должна выполнять одно (и только одно) задание.
- ➔ *Рекомендация №4:* Когда функция становится слишком большой, сложной или непонятной — её следует разбить на несколько подфункций. Это называется **рефакторингом кода**.

При изучении языка C++ вам предстоит написать много программ, которые будут включать следующие три подзадания:

- ➔ Получение данных от пользователя.
- ➔ Обработка данных.
- ➔ Вывод результата.

Для простых программ (менее, чем 30 строк кода) частично или все эти три подзадания можно записать в функции `main()`. Для более сложных программ (или просто для практики) каждое из этих трех подзаданий является хорошим вариантом, чтобы написать отдельные функции.

Новички часто комбинируют обработку ввода и вывод результата в одной функции. Тем не менее, это нарушает **правило «одного задания»**. Функция, которая обрабатывает значение, должна возвращать его в `caller`, а дальше уже пускай `caller` сам решает, что ему с ним делать.

Оценить статью:



(615 оценок, среднее: 4,95 из 5)

[← Урок №13. Параметры и аргументы функций](#)[Урок №15. Локальная область видимости →](#)

Комментариев: 17

1. *prmx:*[19 апреля 2020 в 15:23](#)

Спасибо, этот материал мне очень помогает

[Ответить](#)2. *Андрей:*[19 февраля 2020 в 21:23](#)

Блин, это круто.

Я сначала недооценил этот сайт, думал скучно и непонятно, а оказалось, что толстенные Страуструпы не нужны)

Как новичок, я доволен, ибо получается, функции я освоил))

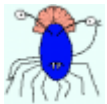
Люблю тебя)

как заработаю на c++ — тебе отправлю)

И по твоему сайту людей учить буду))

[Ответить](#)1. *Юрий:*[19 февраля 2020 в 23:48](#)

Ну всё, жду твоего комментария на последнем уроке курса по C++!))

[Ответить](#)3. *Владимир:*[16 января 2020 в 13:44](#)

Юрий, Спасибо Вам за уроки!

[Ответить](#)4. *Дмитрий:*[10 октября 2019 в 18:58](#)

Спасибо за такое обучения на C++ все понятно

[Ответить](#)



1. *Юрий:*

[10 октября 2019 в 22:31](#)

Пожалуйста)

[Ответить](#)



5. *Vortm4x:*

[15 мая 2019 в 08:09](#)

А существует ли в c++ $\log(x)$ с произвольным основанием (типа $\log_4(x)$) или нужно пользоваться свойствами логарифмов

$\log_4(x) = \log(4) / \log(x)$?

[Ответить](#)



6. *Александр:*

[8 апреля 2019 в 13:21](#)

Читаю уроки и душа радуется! Мне так давно хотелось освоить этот язык. Юра спасибо огромное и долгие лета!

[Ответить](#)



1. *Юрий:*

[8 апреля 2019 в 14:48](#)

Пожалуйста 😊

[Ответить](#)



7. *Михаил:*

[25 января 2019 в 23:23](#)

Спасибо Вам большое за такой чудесный материал. Все просто четко и ясно изложено. Еще раз огромное Вам спасибо!!!

[Ответить](#)



1. *Юрий:*

[26 января 2019 в 13:33](#)

Спасибо и вам, что читаете 😊

[Ответить](#)



1. *Waldemar:*

[27 марта 2019 в 20:57](#)

Юрий, скажите, если я, например, в функции сделаю return 0, вместо return нужной константы, у меня он выведет в итоге 0?

[Ответить](#)



8. *Алёна:*

[1 ноября 2018 в 18:53](#)

Юра, СПАСИБО! Училась в 5 школах и 3 институтах, более структурированных и доходчивых лекций не видела. Без вот этой пафосной "научной лексики", даже словарь сделал для чайников! Ты делаешь очень ценное дело — даешь бесплатное образование, так еще и качество выдаешь на порядок лучше, чем многие преподаватели престижных ВУЗов и частных школ. Когда стану богатой, дам тебе много денег).

[Ответить](#)



1. *Юрий:*

[1 ноября 2018 в 20:03](#)

Спасибо за отзыв, реально приятно 😊 Именно это и есть целью этих уроков — простота и качество в одном флаконе. Сам столкнулся с тем, что информации в книгах/учебниках/многих сайтах вроде бы много, но способ изложения хромает. А когда нашел качественный ресурс LearnCpp, то уже не удержался и решил переводить/поделиться уроками с Рунетом.

[Ответить](#)



2. *84.88.8G.80:*

[5 декабря 2018 в 17:45](#)

Полностью согласен с Аленой.

[Ответить](#)



9. *Семён:*

[24 февраля 2018 в 22:00](#)

Спасибо ! Очень полезно и интересно , но и все уроки нужно подкреплять практикой и что бы хоть один раз получилось вызвать функцию , написать параметры и аргументы .

[Ответить](#)



1. *Дима:*

[5 июня 2018 в 15:25](#)

В предыдущих уроках их предостаточно.

Пример с ОК — круть

[Ответить](#)

Добавить комментарий

Ваш E-mail не будет опубликован. Обязательные поля помечены *

Имя *

Email *

Комментарий

☐ Сохранить моё Имя и E-mail. Видеть комментарии, отправленные на модерацию

☐ Получать уведомления о новых комментариях по электронной почте. Вы можете [подписаться](#) без комментирования.

[TELEGRAM](#)  [КАНАЛ](#)



[ПАБЛИК](#) 

ТОП СТАТЬИ

- [📖 Словарь программиста. Сленг, который должен знать каждый кодер](#)
- [🔌 Урок №1. Введение в программирование](#)
- [🔗 70+ бесплатных ресурсов для изучения программирования](#)
- [🎮 Урок №1: Введение в создание игры «Same Game»](#)
- [⚙️ Урок №4. Установка IDE \(Интегрированной Среды Разработки\)](#)

- [Ravesli](#)
- - [О проекте](#) -
- - [Пользовательское Соглашение](#) -
- - [Все статьи](#) -
- Copyright © 2015 - 2020