

# Оглавление

<b>Предисловие от издательства .....</b>	<b>11</b>
Отзывы и пожелания.....	11
Список опечаток.....	11
Нарушение авторских прав .....	11
<b>Предисловие от главного редактора русского перевода .....</b>	<b>12</b>
<b>Истории читателей.....</b>	<b>16</b>
<b>Введение .....</b>	<b>17</b>
Соглашения.....	17
Специальные шрифты .....	17
Специальные блоки.....	18
Исходный код.....	18
Компиляция программ .....	18
Как вам следует читать эту книгу? .....	20
Личные замечания .....	20
Благодарности .....	20
Сиппи .....	20
<b>Редакторы русского перевода.....</b>	<b>21</b>
<b>Обо мне .....</b>	<b>22</b>
<b>О ЯЗЫКЕ C++.....</b>	<b>24</b>
<b>1. Исторический контекст .....</b>	<b>25</b>
1.1 C++98.....	25
1.2 C++03.....	26
1.3 TR1 .....	26
1.4 C++11.....	26
1.5 C++14.....	26
1.6 C++17.....	26

**2. Стандартизация ..... 27**

2.1 Стадия 3 .....	28
2.2 Стадия 2 .....	28
2.3 Стадия 1 .....	28

**КРАТКИЙ ОБЗОР C++20 ..... 30****3. C++20 ..... 31**

3.1 Большая четверка .....	32
3.1.1 Концепты .....	32
3.1.2 Модули .....	33
3.1.3 Библиотека диапазонов .....	34
3.1.4 Корутины .....	35
3.2 Ядро языка .....	37
3.2.1 Оператор трехстороннего сравнения .....	37
3.2.2 Назначенная инициализация .....	38
3.2.3 <code>constexpr</code> и <code>constexpr</code> .....	40
3.2.4 Улучшения работы с шаблонами .....	41
3.2.5 Улучшения лямбд .....	42
3.2.6 Новые атрибуты .....	42
3.3 Стандартная библиотека .....	43
3.3.1 <code>std::span</code> .....	43
3.3.2 Улучшения контейнеров .....	44
3.3.3 Арифметические утилиты .....	44
3.3.4 Календарь и временные зоны .....	44
3.3.5 Библиотека для форматированного вывода .....	45
3.4 Параллельность .....	46
3.4.1 Атомарные операции .....	46
3.4.2 Семафоры .....	47
3.4.3 Защелки и барьеры .....	47
3.4.4 Кооперативное прерывание .....	48
3.4.5 <code>std::jthread</code> .....	50
3.4.6 Синхронные выходные потоки .....	51

**ПОДРОБНО ПРО C++20 ..... 54****4. Ядро языка ..... 55**

4.1 Концепты .....	55
4.1.1 Два неправильных подхода .....	56
4.1.2 Преимущества концептов .....	62
4.1.3 Длинная, длинная история .....	62
4.1.4 Исползования концептов .....	63
4.1.5 Ограниченные или неограниченные заполнители .....	75
4.1.6 Сокращенные шаблонные функции .....	78
4.1.7 Предопределенные концепты .....	82
4.1.8 Определение концептов .....	88

4.1.9	Применение концептов.....	96
4.2	Модули.....	108
4.2.1	Для чего нужны модули?.....	108
4.2.2	Преимущества использования модулей .....	114
4.2.3	Простой пример использования модулей.....	115
4.2.5	Экспорт из модуля .....	120
4.2.6	Рекомендации по структуре модуля.....	121
4.2.7	Блок интерфейса модуля и блок реализации модуля .....	122
4.2.8	Подмодули и разделы модулей .....	125
4.2.9	Шаблоны в модулях .....	129
4.2.10	Линковка на уровне модулей .....	132
4.2.11	Заголовочные блоки .....	134
4.3	Оператор трехстороннего сравнения.....	136
4.3.1	Упорядочение до C++20 .....	136
4.3.2	Упорядочение начиная со стандарта C++20.....	138
4.3.3	Категории сравнения.....	141
4.3.4	Создаваемый компилятором оператор трехстороннего сравнения.....	143
4.3.5	Переписывание выражений.....	148
4.3.6	Задаваемые пользователем и создаваемые автоматически операторы сравнения.....	151
4.4	Назначенная инициализация .....	154
4.4.1	Агрегированная инициализация .....	154
4.4.2	Именованная инициализация членов класса.....	156
4.5	consteval и constinit.....	161
4.5.1	constexpr.....	161
4.5.2	constinit.....	163
4.5.3	Выполнение функций.....	164
4.5.4	Инициализация переменных.....	166
4.5.5	Исправляем проблему порядка статической инициализации .....	167
4.6	Улучшение работы с шаблонами .....	173
4.6.1	Условный явный конструктор.....	173
4.6.2	Нетипизированные параметры шаблона .....	176
4.7	Улучшения лямбд.....	180
4.7.1	Шаблонные параметры для лямбд.....	180
4.7.2	Определение неявного копирования указателя this .....	184
4.7.3	Лямбды в контекстах без выполнения. Использование конструктора по умолчанию и копирования для лямбд без состояния .....	186
4.8	Новые атрибуты .....	190
4.8.1	[[nodiscard("reason")]].....	191
4.8.2	[[likely]] и [[unlikely]] .....	195
4.8.3	[[no_unique_address]] .....	196
4.9	Дополнительные улучшения.....	199
4.9.1	volatile .....	199
4.9.2	Оператор цикла for с инициализацией на основе диапазона .....	201
4.9.3	Виртуальная функция с constexpr .....	202

4.9.4 Новый символьный тип для utf8-строк: <code>char8_t</code> .....	204
4.9.5 Использование <code>using enum</code> в локальной области видимости.....	205
4.9.6 Инициализаторы по умолчанию для битовых полей .....	206

## 5. Стандартная библиотека .....209

5.1 Библиотека диапазонов.....	210
5.1.1 Концепты <code>ranges</code> и <code>views</code> .....	211
5.1.2 Работа алгоритмов непосредственно со всем контейнером .....	212
5.1.3 Композиция функций.....	216
5.1.4 Отложенное выполнение .....	218
5.1.5 Определение видов.....	221
5.1.6 Аромат Python.....	224
5.2 <code>std::span</code> .....	230
5.2.1 Статическая и динамическая длина .....	230
5.2.2 Автоматический вывод размера непрерывной последовательности объектов .....	232
5.2.3 Создание <code>std::span</code> из указателя и размера .....	233
5.2.4 Изменение объектов, к которым происходит обращение через ссылку.....	235
5.2.5 Обращение к элементам <code>std::span</code> .....	236
5.2.6 Постоянный диапазон изменяемых элементов .....	238
5.3 Улучшения контейнеров .....	241
5.3.1 Контейнеры и алгоритмы со спецификатором <code>constexpr</code> .....	241
5.3.2 <code>std::array</code> .....	242
5.3.3 Последовательное удаление из контейнеров .....	244
5.3.4 <code>contains</code> для ассоциативных контейнеров .....	249
5.3.5 Проверка строки на наличие префикса и суффикса .....	252
5.4 Арифметические функции.....	255
5.4.1 Безопасное сравнение целых чисел .....	255
5.4.2 Математические константы.....	260
5.4.3 Вычисление середины отрезка и линейная интерполяция .....	262
5.4.4 Работа с битами .....	263
5.5 Календарные зоны и часовые пояса.....	269
5.5.1 Время дня .....	270
5.5.2 Календарные даты .....	273
5.5.3 Часовые пояса .....	289
5.6 Библиотека форматирования .....	296
5.6.1 Функции форматирования.....	296
5.6.2 Форматная строка.....	298
5.6.3 Задаваемые пользователем типы.....	306
5.7 Дальнейшие улучшения .....	312
5.7.1 <code>std::bind_front</code> .....	312
5.7.2 <code>std::is_constant_evaluated</code> .....	314
5.7.3 <code>std::source_location</code> .....	316

<b>6. Параллельность .....</b>	<b>318</b>
6.1 Корутины .....	319
6.1.1 Функция-генератор .....	320
6.1.2 Характеристики .....	323
6.1.3 Фреймворк .....	325
6.1.4 Ожидаемые и ожидающие объекты .....	328
6.1.5 Исполняемый поток процессов .....	330
6.1.6 <code>co_return</code> .....	334
6.1.7 <code>co_yield</code> .....	336
6.1.8 <code>co_await</code> .....	339
6.2 Атомарные переменные .....	349
6.2.1 <code>std::atomic_ref</code> .....	349
6.2.2 Атомарный умный указатель .....	358
6.2.3 Расширения <code>std::atomic_flag</code> .....	362
6.2.4 Расширения <code>std::atomic</code> .....	370
6.3 Семафоры .....	374
6.4 Защелки и барьеры .....	379
6.4.1 <code>std::latch</code> .....	379
6.4.2 <code>std::barrier</code> .....	385
6.5 Координированное прерывание .....	389
6.5.1 <code>std::stop_source</code> .....	390
6.5.2 <code>std::stop_token</code> .....	391
6.5.3 <code>std::stop_callback</code> .....	391
6.6 <code>std::jthread</code> .....	398
6.6.1 Автоматическое присоединение .....	399
6.6.2 Кооперативное прерывание <code>std::jthread</code> .....	401
6.7 Синхронизированные потоки вывода .....	404
<b>7. Практические примеры .....</b>	<b>413</b>
7.1 Быстрая синхронизация потоков .....	414
7.1.1 Условные переменные .....	415
7.1.2 <code>std::atomic_flag</code> .....	417
7.1.3 <code>std::atomic&lt;bool&gt;</code> .....	420
7.1.4 Семафоры .....	422
7.1.5 Общая статистика .....	424
7.2 Вариации объектов <code>future</code> .....	425
7.2.1 Ленивый объект <code>future</code> .....	428
7.2.2 Выполнение на другом потоке .....	431
7.3 Модификация и обобщение генератора .....	436
7.3.1 Изменения .....	440
7.3.2 Обобщение .....	443
7.4 Различные потоковые архитектуры, основанные на задачах .....	447
7.4.1 Прозрачная архитектура ожидающего потока задач .....	447
7.4.2 Автоматическое возобновление ожидающей задачи .....	450
7.4.3 Автоматическое возобновление ожидающего объекта на отдельном потоке .....	453

<b>ЭПИЛОГ .....</b>	<b>458</b>
<b>ДОПОЛНИТЕЛЬНАЯ ИНФОРМАЦИЯ .....</b>	<b>460</b>
<b>8. C++23 и не только .....</b>	<b>461</b>
8.1 C++23 .....	462
8.1.1 Библиотека сопрограмм .....	462
8.1.2 Модуляризированная стандартная библиотека .....	476
8.1.3 Исполнители .....	479
8.1.4 Сетевая библиотека .....	483
8.2 C++23 или позже .....	485
8.2.1 Контракты .....	485
8.2.2 Рефлексия .....	488
8.2.3 Сопоставление с образцом .....	492
8.3 Дополнительная информация о стандарте C++23 .....	496
<b>9. Дополнительное тестирование .....</b>	<b>497</b>
<b>10. Глоссарий .....</b>	<b>510</b>

# Предисловие от издательства

## Отзывы и пожелания

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв на нашем сайте [www.dmkpress.com](http://www.dmkpress.com), зайдя на страницу книги и оставив комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу [dmkpress@gmail.com](mailto:dmkpress@gmail.com); при этом укажите название книги в теме письма.

Если вы являетесь экспертом в какой-либо области и заинтересованы в издании новой книги, заполните форму на нашем сайте по адресу [http://dmkpress.com/authors/publish\\_book/](http://dmkpress.com/authors/publish_book/) или напишите в издательство по адресу [dmkpress@gmail.com](mailto:dmkpress@gmail.com).

## Список опечаток

Хотя мы приняли все возможные меры для того, чтобы обеспечить высокое качество наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг – возможно, ошибку в основном тексте или программном коде, – мы будем очень благодарны, если вы сообщите нам о ней. Сделав это, вы избавите других читателей от непонимания текста и поможете нам улучшить последующие издания этой книги.

Если вы найдете какие-либо ошибки в коде, пожалуйста, сообщите о них главному редактору по адресу [dmkpress@gmail.com](mailto:dmkpress@gmail.com), и мы исправим их в следующих тиражах.

## Нарушение авторских прав

Пиратство в сети Интернет по-прежнему является насущной проблемой. Издательство «ДМК Пресс» очень серьезно относится к вопросам защиты авторских прав и лицензирования. Если вы знаете о незаконной публикации какой-либо из наших книг в сети Интернет, пожалуйста, пришлите нам ссылку на интернет-ресурс, чтобы мы могли применить санкции.

Ссылку на подозрительные материалы можно прислать по адресу [dmkpress@gmail.com](mailto:dmkpress@gmail.com).

Мы высоко ценим любую помощь по защите наших авторов, благодаря которой мы можем предоставлять вам качественные материалы.

## Благодарности

Александр Богомолов, Руслан Нуржанов, Alexey Vinogradov.

# Предисловие от главного редактора русского перевода

*Дорогие друзья!*

Книга, которую вы держите в руках, открывает серию «Книжная полка Истового Инженера», которая издается при поддержке компании YADRO. Это издание подготовлено к публикации Московским институтом электроники и математики им. А. Н. Тихонова НИУ ВШЭ совместно с «ДМК Пресс».

Если вы интересуетесь цифровой электроникой, разработкой на ПЛИС, проектированием на языках описания аппаратуры Verilog или VHDL, то вы, скорее всего, уже знакомы с книгами серии «Цифровой синтез», такими как: Д. Харрис и С. Л. Харрис «Цифровая схемотехника и архитектура компьютера»; «Цифровой синтез: практический курс» (под. ред. А. Ю. Романова и Ю. В. Панчула), Ф. Бруно «Программирование FPGA для начинающих» и др. Эта книга несколько отличается по тематике и в первую очередь ориентирована на программистов, работающих с языками высокого уровня.

## Почему я взялся редактировать русский перевод книги Р. Гримма «С++20 в деталях»?

На самом деле я сильно сомневался. Но меня убедила моя коллега Ирина Романова, она же и выполнила первую вычитку и редактуру этой книги. Без ее помощи я бы не согласился редактировать данный материал. Остальные мотивы описаны далее.

## Как я использую С++?

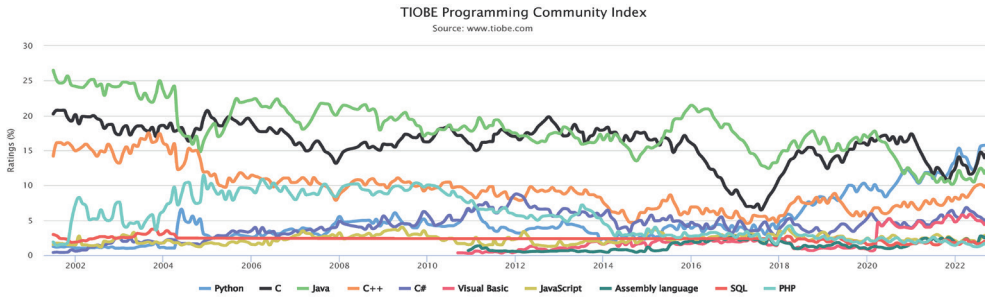
Хотя я применяю С++ в своей практике постоянно, использую я его обычно как язык системного программирования, а также для программирования встраиваемых систем (таких как микроконтроллеры), т. е. вполне могу обходиться подмножеством С; а если нужна объектно-ориентированность, то вполне достаточно тех возможностей, которые предоставляет стандарт С++98.

## Зачем изучать С++20?

Зачем же тогда изучать все эти С++11, потом С++14 и т. д., а теперь и С++20? (Ведь на подходе уже и следующий стандарт!)

Ответ: это очень интересно! С++ – это один из ведущих языков. По рейтингу ТIOV С/С++ – все еще самый популярный язык в мире. Популярнее, чем Python!





C++ – довольно «старый» язык, который развивается, меняется, создает новые методы и принципы программирования прямо у нас на глазах. Это (во многом) отражение прошлого и будущего всех классических высокоуровневых языков программирования. Каждый новый стандарт – это новые парадигмы и «фишки» программирования. При этом предлагают, описывают и внедряют их обычные программисты в открытом комьюнити.

Поэтому даже если вы заядлый «питонист» или никогда не планируете использовать новшества C++, все равно стоит делать как я: следить за выходом новых стандартов и хотя бы пролистывать релизы о нововведениях в язык и другую литературу по теме – потому что понимать ход современной передовой мысли разработчиков программного кода, знать новые приемы и возможности ключевого языка общения всех программистов очень важно для саморазвития любого, кто любит и практикует разработку программ.

## Зачем нужен русский перевод этой книги?

Очень часто в среде программистов встречаю мнение, что изучать специализированную литературу по программированию надо на языке оригинала: английском. В целом я с этим согласен – сам большинство информации по теме получаю на английском. Но это не потому, что читать на английском удобней, а потому, что на русском почти всегда нужной информации нет. Если хотите читать оригинал, пожалуйста, читайте его. Но если кто-то хочет быстро познаться с новшествами C++20 или разобраться более детально с какими-то тонкостями этих новшеств на примерах – эта книга для вас.

## Почему именно книга Р. Гримма?

Потому что это признанный автор, а его книги очень популярны. Потому что Р. Гримм сам из «тусовки» разработчиков новых стандартов C++, и у него сравнительно легкая и интересная подача. Надеюсь, нам удалось это сохранить и в русском переводе. А еще в этой книге довольно информативные и показательные примеры с небольшим количеством ошибок. Этим могут похвастаться далеко не все издания в данной области, даже издания именитых авторов!

Еще для себя я выработал один неформальный принцип оценки зарубежных книг, рассчитанных на большую аудиторию: это то, как она оформлена. Чем больше вложено в оформление издания, тем выше вероятность того (хоть и не всегда!), что содержание книги будет на уровне. Дело в том, что западное

общество довольно инфантильно (чего только стоят «График Шму» или «Принцип печенья Орео!»), поэтому книге без «картинок» довольно сложно добиться популярности. Объединение «картинок» и хорошего содержания является фактором, обеспечивающим успех книги.

Данная книга соответствует этому критерию. Хотя, на мой взгляд, персонаж книги, Сиппи, довольно безобразна, а подписи к рисункам катастрофически неинформативны. Такое представление персонажа я (не художник) тоже выдать могу! Но не включаю его в серьезную книгу. Хотя... а почему бы и нет?



Было желание убрать все эти бессмысленные рисунки из книги, но, в конце концов, было решено смириться с этим.

## Для кого эта книга?

Давайте вернемся к серьезному тону и, наконец, ответим на вопрос: кому будет интересна эта книга? Программирование C++ по ней вы не изучите – для этого есть классические книги Х. Дейтела, Б. Страуструпа, С. Прата и др. Для того чтобы воспринимать материал, нужно уже знать концепцию объектно-ориентированного программирования и иметь хотя бы небольшой опыт разработки программ на C++ или «близких по духу» высокоуровневых языках – вроде C#, Java, Python. Неплохо бы знать такие концепции, как лямбда-функции и многопоточные/параллельные программы. Стандарт C++20 глубоко вы по этой книге тоже не изучите. Она нужна для первого быстрого и легкого знакомства с новшествами, которые появились в новом стандарте языка. Заодно читатели смогут узнать немного о причинах появления этих новшеств, принципах и идеях, которые в них заложены. Также в книге очень неплохие и понятные примеры программ. Остальные тонкости вы при необходимости узнаете на практике или изучив стандарт.

## Ошибки, терминология и другие замечания

Проблема перевода такого рода книг состоит в том, что довольно часто в русском языке нет устоявшегося аналога английскому термину или прямой русский перевод выглядит глупо и неестественно. В основном это так, потому что сами английские термины являются «новоделами». Например, тот же coroutine. Все уже смирились говорить и писать «корутин», притом что в русском языке есть тоже иностранное, но более привычное слово «сопрограмма», которое имеет приблизительно то же значение. В русском переводе осуществлена попытка соблюсти баланс, обеспечивающий читабельность и понятность текста: какие-то термины используются в форме, более привычной русскоязычному читателю, а какие-то вообще не переведены. По той же причине сознательно не переведены надписи на рисунках и комментарии в листингах.

Эту книгу прочитали два научных редактора, один корректор и еще два студента, но, безусловно, она не идеальна. Я буду очень признателен тем внимательным читателям, которые обнаружат в данном издании какие-либо ошибки или опечатки и сообщат о них по адресу [a.romanov@hse.ru](mailto:a.romanov@hse.ru) или [dmkpress@gmail.com](mailto:dmkpress@gmail.com) (книги постоянно перепечатываются, и в каждом новом тираже все найденные ошибки и недочеты исправляются).

Также вы можете присылать свои рисунки, изображающие Сиппи. Я пока еще не придумал, что с ними делать, но (возможно) получится импровизированная выставка ☺

**Александр Юрьевич Романов,**  
к. т. н., доцент ДКИ МИЭМ НИУ ВШЭ,  
преподаватель курсов «Проектирование систем на кристалле»,  
«Системное проектирование цифровых устройств»  
и «Системы искусственного интеллекта»  
г. Москва, Россия

# Истории читателей



Сандор Дарго, старший программист в Amadeus

«С++ 20 в деталях» – это именно та книга, которая вам сейчас нужна, если вы хотите погрузиться в последнюю версию С++. Это полное руководство, в котором Рейнер рассматривает не просто передовые возможности С++20, но и небольшие добавления в язык. Эта книга содержит множество примеров кода, так что даже если у вас еще нет доступа к последним компиляторам, у вас все равно будет хорошее представление о том, чего вы можете ожидать от различных возможностей языка. Крайне рекомендую прочитать.



Адриан Там, директор по Data Science, Synchro Inc

С++ очень сильно развился с момента его появления на свет. С++20 – это практически новый язык. Конечно, это книга не предназначена для того, чтобы научить вас наследованию или перегрузке, но если вы хотите привести свои знания С++ к современному уровню, то это очень подходящая книга. Вы удивитесь тому, сколько всего нового из С++20 вошло в базовый язык С++. Эта книга дает ясные объяснения и краткие примеры. Структура книги позволит вам использовать ее как справочник в дальнейшем. Она поможет вам узнать все современные возможности языка.

# Введение

Моя книга по C++ – это и учебник, и справочное руководство. Она научит вас C++20 и даст вам детальную информацию о новом стандарте C++. Наиболее выдающимися возможностями C++20 являются следующие:

- **концепты** – изменяют способ, которым мы думаем о программах с шаблонами. Они являются семантическими категориями для параметров шаблонов. Они позволяют вам ясно выражать свои намерения через систему типов. Если что-то пойдет не так, то компилятор даст вам понятное сообщение об ошибке;
- **модули** – позволяют обойти ограничения заголовочных файлов. Они очень многообещающи. Например, разделение заголовочного файла и исходного файла становится таким уже устаревшим, как и препроцессор. В результате вы получаете более быструю компиляцию и легкий способ создания пакетов;
- новая **библиотека диапазонов** (ranges library) поддерживает применение алгоритмов к контейнерам, включающим алгоритмы, организованные как конвейер (pipe), и «ленивое» (lazy) применение алгоритмов к бесконечным потокам данных;
- благодаря **корутинам** (coroutines) асинхронное программирование в C++ становится широко распространенным. Корутины образуют основу для кооперативных задач, циклов обработки событий, бесконечных потоков данных или конвейеров.

Конечно, это далеко не все. Вот еще некоторые новые возможности C++20:

- генерируемые автоматически операторы сравнения;
- библиотеки для работы с датами и временными зонами;
- библиотека format;
- «виды» (view) непрерывных блоков памяти;
- улучшенные прерываемые потоки;
- атомарные умные указатели;
- семафоры;
- примитивы управления, такие как защелки (latch) и барьеры (barrier).

## Соглашения

Есть всего несколько соглашений по форматированию этой книги:

### Специальные шрифты

#### *Курсив*

Используется *курсив* для выделения цитат.

#### **Жирный**

Используется **жирный** шрифт для выделения имен.

Моноширинный

Используется моноширинный шрифт для кода, инструкций, ключевых слов и имен типов, переменных, функций и классов.

## Специальные блоки

Блоки содержат подсказки, предупреждения и дополнительную информацию.



### Подсказка

Этот блок предоставляет дополнительную информацию по текущему материалу и подсказки по компиляции программ.



### Предупреждения

Блоки предупреждений должны помочь вам избегать ошибок.



### Собранная информация

Этот блок в конце каждого раздела содержит важную информацию, которую стоит запомнить.

## Исходный код

Примеры исходного кода, приведенные в книге, являются завершенными. Это значит, что если у вас есть подходящий компилятор, то вы можете их откомпилировать и выполнить. Имя исходного файла находится в заголовке каждого примера кода. Исходный код использует четыре пробела для табуляции (отступов). Иногда используется два пробела из соображений размещения на странице.

Кроме того, я не большой любитель директив namespace вроде `using namespace std`, поскольку они делают код более сложным для чтения и могут засорять пространства имен. Соответственно, я применяю их только тогда, когда это улучшает читаемость кода (например, `using namespace std::chrono_literals`). Иногда для более удобного размещения кода на странице я использую `using`, например `using std::chrono::system_clock`.

Таким образом, только в случае необходимости иногда делаются следующие отступления от общих правил оформления кода:

- табуляция на два пробела, а не на четыре;
- используется директива `using namespace std`.

## Компиляция программ

Поскольку C++20 – это новый стандарт, то многие примеры могут быть откомпилированы только при помощи подходящих компиляторов. Я использую последние версии GCC<sup>1</sup>, Clang<sup>2</sup> и MSVC<sup>3</sup>. При компиляции программы вы должны указать используемый стандарт C++. Это значит, что при использовании GCC

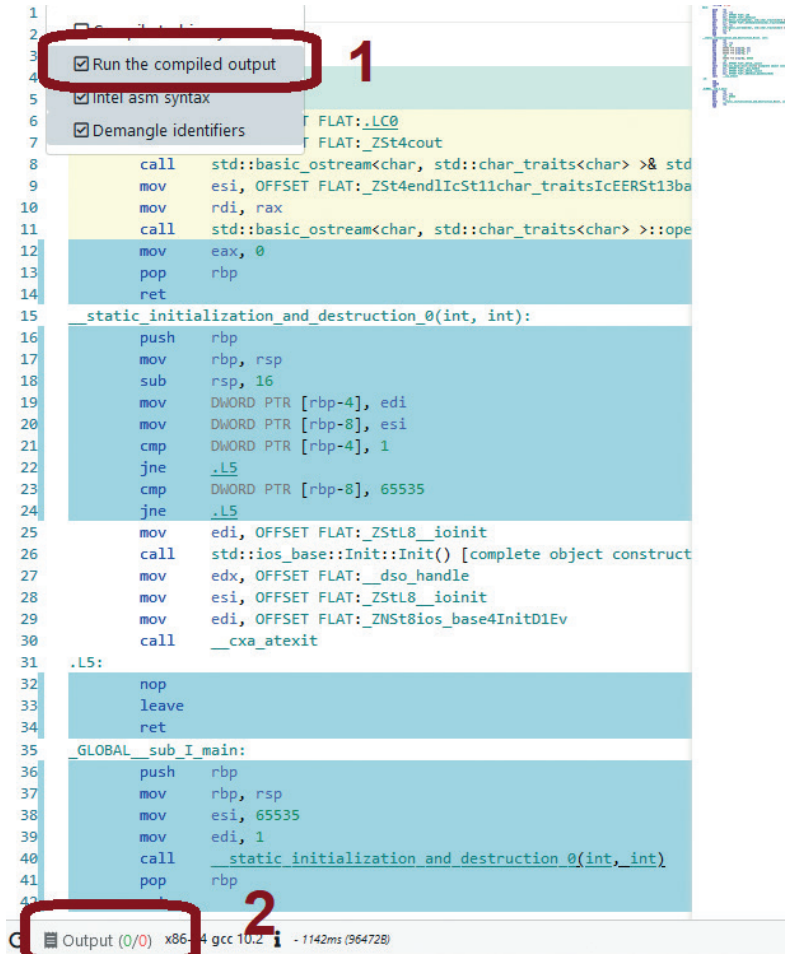
<sup>1</sup> <https://gcc.gnu.org>.

<sup>2</sup> <https://clang.llvm.org>.

<sup>3</sup> [https://en.wikipedia.org/Microsoft\\_Visual\\_C%2B%2B](https://en.wikipedia.org/Microsoft_Visual_C%2B%2B).

или Clang вы должны указать флаг `-std=c++20` и для MSVC флаг `/std:c++latest`. При применении параллельности (concurrency) для GCC и Clang необходимо указать, что вам требуется линковка с библиотекой `pthread` при помощи опции `-pthread`.

Если у вас нет в распоряжении подходящего компилятора C++, то вы можете использовать онлайн-компилятор вроде [Wandbox](https://wandbox.org)<sup>1</sup> или [Compiler Explorer](https://godbolt.org)<sup>2</sup>. При применении [Compiler Explorer](https://godbolt.org) с выбором компилятора GCC или Clang вы также можете выполнить откомпилированную программу. Для этого, во-первых, вам нужно выбрать `Run the compiled output` и, во-вторых, открыть окно `Output`.



### Run code in the Compiler Explorer

Вы можете получить дополнительную информацию о совместимости с C++20 для различных компиляторов на [cppreference.com](https://en.cppreference.com/w/cpp/compiler_support)<sup>3</sup>.

<sup>1</sup> <https://wandbox.org>.

<sup>2</sup> <https://godbolt.org>.

<sup>3</sup> [https://en.cppreference.com/w/cpp/compiler\\_support](https://en.cppreference.com/w/cpp/compiler_support).



## Как вам следует читать эту книгу?

Если вы незнакомы со стандартом C++20, то начните с краткого обзора для получения общей картины.

После получения общей картины вы можете переходить к ключевым возможностям языка (core language). Рассказ о каждой новой возможности языка является самодостаточным, но наилучшим вариантом будет прочтение всей книги от начала до конца. При первом прочтении вы можете пропустить то, что не упомянуто в кратком обзоре.

## Личные замечания

### Благодарности

Я просил читателей проверить на ошибки мой англоязычный блог `Modernescpp`<sup>1</sup> и получил гораздо больше откликов, чем ожидал. Огромное спасибо вам всем. Вот имена тех, кто внес в это свой вклад: Bob Bird, Nicola Bombace, Dave Burchill, Sandor Dargo, James Drobina, Frank Grimm, Kilian Henneberger, Ivan «espkk» Kondakov, Péter Kardos, Rakesh Mane, Jonathan O'Connor, John Plaiice, Iwan Smith, Paul Targosz, Steve Vinoski и Greg Wagner.

Особенное спасибо моей дочери Джульетте и моей жене Беатрис. Джульетта поправила мой язык и исправила множество опечаток. Беатрис создала персонажа Сиппи (Cippi, девочка на рисунках) и иллюстрации к этой книге.

### Сиппи



Позвольте представить вам Сиппи, которая будет сопровождать вас в этой книге. Надеюсь, она вам понравится.

Я Сиппи (по аналогии с Пеппи Длинныйчулок), любознательная, умная и женственная!

<sup>1</sup> <http://modernescpp.com>.



# Редакторы русского перевода



Романов Александр Юрьевич (<https://www.hse.ru/staff/a.romanov>) – главный научный редактор русского перевода данной книги, доцент Московского института электроники и математики им. А. Н. Тихонова Национального исследовательского университета «Высшая школа экономики» (МИЭМ НИУ ВШЭ). С 2014 г. работает в МИЭМ НИУ ВШЭ, где возглавляет лабораторию САПР (<https://miem.hse.ru/edu/ce/cadsystem>), специализирующуюся на проектной деятельности, а также разработку цифровых систем на ПЛИС/микроконтроллерах, робототехнических комплексов, аппаратных реализаций систем искусственного интеллекта, многопроцессорных систем, систем удаленного доступа к ла-

бораторному оборудованию и т. д. В 2015 г. защитил диссертацию в Институте проблем проектирования в микроэлектронике РАН (г. Зеленоград), является автором более 150 научных статей, патентов и книг. А. Ю. Романов преподает C++ с 2009 г. в качестве лекционного и практического курса для студентов и постоянно использует данный язык в практической работе.



Романова Ирина Ивановна (<https://www.hse.ru/staff/iromanova>) – научный редактор русского перевода – занимается преподаванием компьютерных и инженерных дисциплин с 2010 г. В настоящее время работает старшим преподавателем в Московском институте электроники и математики им. А. Н. Тихонова Национального исследовательского университета «Высшая школа экономики» (МИЭМ НИУ ВШЭ), ведущий преподаватель и лектор дисциплины «Информатика» для студентов 1-го курса, а также соавтор известной книги «Цифровой синтез: практический курс» (М.: ДМК Пресс, 2020). Автор более 30 научных статей. Специализируется на методике преподавания компьютерных дисциплин студентам младших курсов.

# Обо мне

Я работал в качестве архитектора программных систем, руководителя группы (team lead) и лектора начиная с 1999-го. В 2002 году я создавал тренинги для обучения. Я начал вести занятия с 2002 года. Моими первыми тренингами были занятия по управлению проприетарным программным обеспечением, но вскоре я начал обучать языкам Python и C++. В мое свободное время мне нравится писать статьи о C++, Python и Haskell. Также мне нравится выступать на различных конференциях. Каждую неделю я пишу что-то в моем англоязычном блоге ModernesCpp и в блоге на немецком языке<sup>1</sup>, размещенном у Heise Developer.

С 2016 года я выступал в качестве независимого инструктора с семинарами по современному C++ и Python. Я написал несколько книг на разных языках о современном C++ и, в частности, по параллелизму. В связи с моей профессией я всегда ищу лучшие пути обучения современному C++.



Райнер Гримм

---

<sup>1</sup> <https://www.grimm-jaud.de/index.php/blog>.



# О ЯЗЫКЕ C++

Конец ознакомительного фрагмента.

Приобрести книгу можно

в интернет-магазине

«Электронный универс»

[e-Univers.ru](http://e-Univers.ru)