Васильев А.Н.

ПРОГРАММИРОВАНИЕ НА C

В ПРИМЕРАХ И ЗАДАЧАХ

РОССИЙСКИЙ КОМПЬЮТЕРНЫЙ БЕСТСЕЛЛЕР

Москва
2020

УДК 004.43
ББК 32.973.2-018.2
В19

Васильев, Алексей Николаевич.

В19 Программирование на С в примерах и задачах / А.Н. Васильев. — Москва : Эксмо, 2020. — 560 с.— (Российский компьютерный бестселлер).

ISBN 978-5-699-89518-2

Книга включает в себя полный перечень сведений о языке Си, представленный в рамках обучающей методики от лучшего российского автора учебников по языкам программирования Алексея Васильева. В каждой главе читатель найдет подробный разбор примеров, а также задачи для самостоятельного решения и комментарии автора.

УДК 004.43
ББК 32.973.2-018.2

ISBN 978-5-699-89518-2

Производственно-практическое издание

РОССИЙСКИЙ КОМПЬЮТЕРНЫЙ БЕСТСЕЛЛЕР

Васильев Алексей Николаевич

ПРОГРАММИРОВАНИЕ НА С В ПРИМЕРАХ И ЗАДАЧАХ

Директор редакции *Е. Капьёв*
Ответственный редактор *Е. Истомина*
Художественный редактор *В. Брагина*

В оформлении обложки использованы иллюстрации:
pavel7tymoshenko / Shutterstock.com
Используется по лицензии от Shutterstock.com

ООО «Издательство «Эксмо»
123308, Москва, ул. Зорге, д. 1. Тел.: 8 (495) 411-68-86.
Home page: www.eksmo.ru E-mail: info@eksmo.ru
Өндіруші: «ЭКСМО» АҚБ Баспасы, 123308, Мәскеу, Ресей, Зорге көшесі, 1 үй.
Тел.: 8 (495) 411-68-86.
Home page: www.eksmo.ru E-mail: info@eksmo.ru.
Тауар белгісі: «Эксмо»
Интернет-магазин : www.book24.ru

Интернет-магазин : www.book24.kz
Интернет-дукен : www.book24.kz
Импортёр в Республику Казахстан ТОО «РДЦ-Алматы».
Қазақстан Республикасындағы импорттаушы «РДЦ-Алматы» ЖШС.
Дистрибьютор и представитель по приему претензий на продукцию,
в Республике Казахстан: ТОО «РДЦ-Алматы»
Қазақстан Республикасында дистрибьютор және өнім бойынша арыз-талаптарды
қабылдаушының өкілі «РДЦ-Алматы» ЖШС,
Алматы қ., Домбровский көш., 3«а», литер Б, офис 1.
Тел.: 8 (727) 251-59-90/91/92; E-mail: RDC-Almaty@eksmo.kz
Өнімнің жарамдылық мерзімі шектелмеген.
Сертификация туралы ақпарат сайтта: www.eksmo.ru/certification

Сведения о подтверждении соответствия издания согласно законодательству РФ
о техническом регулировании можно получить на сайте Издательства «Эксмо»
www.eksmo.ru/certification
Өндірген мемлекет: Ресей. Сертификация қарастырылмаған

Подписано в печать 17.02.2020.
Формат 70x100$^{1}/_{16}$. Печать офсетная. Усл. печ. л. 45,37.
Доп. тираж 1000 экз. Заказ

12+

ISBN 978-5-699-89518-2

ОГЛАВЛЕНИЕ

Вступление. Язык C и процедурное программирование 5
Языки и стандарты 5
Парадигмы программирования 8
Структура и концепция книги 11
Компиляторы и среды разработки 13
Обратная связь 41
Благодарности 42
Об авторе 42

Глава 1. Знакомство с языком C 43
Первая программа 43
Использование переменных и базовые типы данных 49
Знакомство с указателями 65
Операторы 69
Знакомство с условным оператором 81
Знакомство с операторами цикла 87
Знакомство с функциями 96
Знакомство с массивами 105
Консольный ввод и вывод 112
Примеры решения задач 120
Резюме 140
Задачи для самостоятельного решения 141

Глава 2. Управляющие инструкции 143
Условный оператор if 143
Оператор выбора switch 161
Операторы цикла while и do-while 177
Оператор цикла for 193
Инструкция безусловного перехода goto 203
Примеры решения задач 205
Резюме 219
Задачи для самостоятельного решения 221

Глава 3. Массивы, указатели и динамическое выделение памяти 223
Указатели 223
Динамическое выделение памяти 248
Одномерные массивы 256

Двумерные массивы 271
Символьные массивы 281
Разные операции с массивами 290
Примеры решения задач 305
Резюме 327
Задачи для самостоятельного решения 329

Глава 4. Функции 331
Создание функции 331
Аргументы функции 340
Результат функции 362
Рекурсия 377
Указатель на функцию 382
Главная функция программы 396
Функция с переменным количеством аргументов 401
Примеры решения задач 406
Резюме 430
Задачи для самостоятельного решения 431

Глава 5. Структуры и объединения 434
Знакомство со структурами 434
Операции с экземплярами структуры 438
Структуры и функции 442
Структуры и массивы 448
Структуры и указатели 461
Вложенные структуры 470
Некоторые особые свойства структур 476
Объединения 484
Примеры решения задач 496
Резюме 512
Задачи для самостоятельного решения 513

Глава 6. Заключительные замечания 515
Директивы препроцессора 515
Оператор typedef 530
Перечисления 536
Битовые поля 537
Файловый ввод и вывод 540
Резюме 557

Вступление
ЯЗЫК С И ПРОЦЕДУРНОЕ ПРОГРАММИРОВАНИЕ

Все люди делятся на две категории: тех, кто знает язык С, и тех, кто не умеет программировать.

Книга посвящена языку программирования С. Именно С, а не С++. Разница в названиях небольшая, но какая огромная разница в базовых подходах! Имеется в виду не столько синтаксис языков С и С++ (хотя здесь тоже есть различия), а сама концепция программирования.

Общепринятой является точка зрения, что язык программирования С++ представляет собой расширение языка С для поддержания парадигмы объектно-ориентированного программирования (ООП). Данный контекст позиционирует язык С как подмножество языка С++. Сказанное отчасти действительно справедливо. Но все не так просто. В какой-то момент язык С++ на самом деле был прямым расширением языка С, но затем, образно выражаясь, их дороги разошлись. Не очень сильно разошлись, но разошлись. И тенденции такие, что уже вряд ли сойдутся (хотя и загадывать не стоит). Как бы там ни было, представление о языке С как о «неполном» или «не до конца укомплектованном» языке С++, является некорректным. Язык С — совершенно самостоятельный язык программирования, со своими преимуществами и недостатками, и разумеется, абсолютно самодостаточный. Он простой, лаконичный, емкий и исключительно эффективный. Язык С однозначно оправдает время и усилия, затраченные на его изучение.

Языки и стандарты

Интеллигентный человек должен знать несколько иностранных языков. Ну, или хотя бы несколько языков программирования.

В истории развития языков С и С++ есть определенные знаковые вехи. Определяются они выходом очередного стандарта языка. Ведь что такое, по большому счету, язык программирования? Это набор синтак-

сических правил, которым должен соответствовать программный код, плюс компилятор, позволяющий преобразовать программный код в понятные для компьютера инструкции.

НА ЗАМЕТКУ

Вообще-то, в этом определении можно было ограничиться «набором правил», однако без компилятора весь процесс теряет смысл. Действительно, какой смысл писать программный код, если компьютер его не понимает и в принципе не может понять? Очевидно, что смысла нет. Поэтому за каждым языком программирования стоит целая технология, связанная с обработкой и имплементацией программных кодов. Это такой огромный айсберг, и в этом айсберге язык, как набор формальных правил, олицетворяет собой лишь вершину.

Что касается синтаксиса языка программирования (будь то C или C++), то синтаксические правила определяются нормативными документами, которые обычно называются стандартами языка. Время от времени (но не очень часто) стандарты изменяются (кем и как — в данном случае не важно). Понятно, что внести изменения в документ легко. Но этого мало. Необходимо, чтобы были компиляторы, поддерживающие новые стандарты языка.

ПОДРОБНОСТИ

Компилятор — специальная программа, предназначенная для преобразования исходного кода, написанного на языке вроде C или C++, в последовательность инструкций, которые собственно и выполняются компьютером. Обычно после того, как программа написана, ее необходимо скомпилировать. Запускается программа на выполнение только после успешной компиляции.

НА ЗАМЕТКУ

Существуют языки программирования (например, Python), в которых программы не *компилируются*, а *интерпретируются*. Принципиальное отличие интерпретируемых программ от компилируемых состоит в том, что при интерпретации «преобразование» команд программы в исполнительный код выполняется команда за командой, одновременно с их выполнением. Что касается языков C и C++, то они относятся к компилируемым языкам. В частности, программа, написанная на языке C, компилируется в исполнительный файл, который можно запустить на выполнение. Выполняется исполнительный файл под управлением операционной системы.

Если стандарт языка один для всех, то компиляторы разрабатываются разными компаниями. После изменения стандарта языка разработчикам компиляторов необходимо внести коррективы в программное обеспечение. Это процесс небыстрый. Поэтому далеко не все компиляторы в полной мере поддерживают самый последний (а значит, и единственно правильный) стандарт языка программирования. Кроме того, не все фирмы-разработчики ставят перед собой задачу строго соответствовать стандарту. Отсюда нередки ситуации, когда тот или иной компилятор имеет некоторые «особенности».

Переходя к вопросам более приземленным и относящимся непосредственно к языку программирования С, отметим, что существует несколько исторически значимых стандартов данного языка. Важной реперной точкой является стандарт С89 языка С от 1989 года. Затем был стандарт С99 от 1999 года, и наконец, стандарт С11 от 2011 года. В чем особенность стандарта С89? Это своеобразная точка бифуркации, когда язык С, с одной стороны, был включен в состав языка С++, а с другой, продолжил свое независимое существование. То, что произошло с языком С после стандарта С89, мало связано с развитием языка С++. Другими словами, изменения в язык С вносились независимо от изменений языка С++ (который тоже периодически модифицируется).

НА ЗАМЕТКУ

Конечно, некоторая корреляция в процессах усовершенствования языков С и С++ имеется. Но те, кто изучал математическую статистику, знают, что наличие корреляции не обязательно означает наличие функциональной связи.

Таким образом, язык С++ представляет собой расширение языка С стандарта С89, с учетом тех изменений, которые вносились уже непосредственно в язык С++. Что касается языка С, то у него своя судьба. И его «кооперация» с языком С++ на ранних этапах развития не отменяет его уникальности и самобытности.

НА ЗАМЕТКУ

На момент написания книги последним стандартом языка С++ является стандарт С++14 от 2014 года. Предыдущие стандарты этого языка С++11 и С++99 соответственно появились в 2011 и 1999 годах.

Резюме здесь такое: на сегодняшний день действующий стандарт языка C — стандарт C11 от 2011 года. Именно на него мы будем ориентироваться в книге. Вместе с тем важно понимать, что не все описанное и задекларированное в стандарте C11 обязательно будет реализовано в каждом компиляторе. Поэтому выбор «правильного» компилятора представляет собой задачу важную и нетривиальную. Ее мы обсудим немного позже.

ЯЗЫК C++

Поскольку дороги C и C++ разошлись, между ними существуют различия. Имеется фундаментальное отличие, связанное с тем, что C++ поддерживает парадигму ООП, а язык C является процедурным языком программирования. Но даже если вывести за скобки этот факт, то отличия наблюдаются уже на нижнем, базовом уровне. Здесь проявляются последствия независимого «плавания» языков. В случаях, когда различия между C и C++ особенно существенны, мы будем выделять соответствующие комментарии в специальном блоке. Это важно не только с точки зрения информационной, но еще и в силу того обстоятельства, что для многих программистов язык C становится трамплином к изучению языка C++. Поэтому разумно сразу выделить «тонкие моменты», связанные с несоответствием стандартов этих языков.

Парадигмы программирования

Нет такой безумной идеи, которую нельзя было бы реализовать в виде программного кода.

Выше отмечался факт «процедурности» языка C и «объектной ориентированности» языка C++. Пришло время внести ясность.

Итак, *процедурное* и *объектно-ориентированное* программирование (сокращенно *ООП*) — две различные парадигмы программирования. Парадигма программирования фактически определяет способ организации программного кода в программе. Чтобы понять разницу между процедурным программированием и ООП, желательно абстрагироваться от конкретики и посмотреть на программу как на набор команд или инструкций, предназначенных для хранения, преобразования и обработки данных. При такой интерпретации программы весь программный код можно разбить на две категории: код, через который реализуются данные, и код, который предназначен для работы с этими данными. В рамках концепции процедурного программирования код для обработки данных

оформляется в виде процедур и функций. Программа же представляет собой набор инструкций, в которых процедуры и функции вызываются для оперирования данными. Данные, в свою очередь, хранятся в переменных, массивах и прочих конструкциях. Таким образом, данные и код для их обработки существуют независимо друг от друга. Их «встреча» происходит в программе. «Место» и «время» встречи определяет программист, составляющий программный код.

В рамках концепции ООП данные и коды для обработки этих данных группируются в отдельных блоках или конструкциях, которые называются *объектами*. Обычно объект содержит в себе и переменные (*поля* объекта), и функции для работы с ними (*методы* объекта). Программа в таком случае представляет собой набор взаимодействующих друг с другом объектов.

Если сравнивать концепции ООП и процедурного программирования, то главное их отличие можно сформулировать так: если в процедурном программировании связывание данных и кода для обработки данных выполняется непосредственно при выполнении программы, то в ООП такое связывание выполняется уже на этапе создания кода для реализации данных и их обработки.

Исторически более ранней является концепция процедурного программирования. ООП появилось несколько позже. Причины появления ООП просты и прозаичны. Дело в том, что по мере увеличения объема программного кода чисто технически становится все сложнее корректно «стыковать» данные с вызовами процедур и функций. Здесь речь идет об ограниченных возможностях человека, а не компьютера. Проще говоря, чем больше по объему код, тем выше вероятность для программиста ошибиться при его написании. При использовании концепции ООП за счет объединения данных и функционального кода на ранних этапах проектирования вероятность ошибки уменьшается.

НА ЗАМЕТКУ

Написание надежных кодов не означает написание компактных кодов. Использование концепции ООП если и упрощает жизнь программиста, то это не означает, что программного кода писать нужно меньше. Скорее, наоборот.

В процедурном программировании программа состоит из процедур, функций и переменных (в самом широком смысле данного понятия). В ООП программа состоит из классов и объектов.

ПОДРОБНОСТИ

В ООП класс — это некоторый шаблон, на основе которого создаются объекты. Класс содержит описание того, что должно быть в объекте. Если провести аналогию и отождествить объект со зданием, то класс представляет собой план или проект здания. Понятно, что по одному проекту можно построить несколько однотипных зданий. Точно так же на основе одного и того же класса можно создать несколько объектов. Вместе с тем, далеко не во всех языках, поддерживающих парадигму ООП, объекты создаются на основе классов. Например, в сценарном языке JavaScript классы как таковые отсутствуют, а объекты создаются или «с нуля», или на основе уже существующих объектов. Но это все детали, которые в нашем деле несущественны.

В языке C++ можно создавать программы как с классами и объектами, так и без классов и объектов. Проще говоря, язык C++ поддерживает и парадигму процедурного программирования, и парадигму ООП. В языке C поддерживается только парадигма процедурного программирования. Поэтому при создании программы на языке C мы не сможем прибегнуть ни к помощи классов, ни к помощи объектов. С другой стороны, необходимость в такой помощи возникает намного реже, чем может показаться. Если же речь идет о не очень больших программах, то применение такой «тяжелой артиллерии», как ООП, вообще представляется занятием малоперспективным. Кроме того, в языке C имеется огромное количество встроенных средств для написания программы почти любой сложности. Так что недостатка в средствах не будет.

НА ЗАМЕТКУ

Как отмечалось выше, язык C++ является переходным в том смысле, что позволят выбирать стиль программирования на собственный вкус. Но такой демократизм в мире ООП не является нормой. Скорее, это исключение из правил. Например, другие наиболее популярные языки программирования Java и C# являются полностью объектно-ориентированными. Написание даже самой простой программы в Java или C# подразумевает описание по меньшей мере одного класса.

Что важно вынести из сказанного выше? Важно понять, что процедурный характер языка C на самом деле не несет ни положительной, ни отрицательной нагрузки. Просто вот такой язык программирования. А то, что это хороший язык программирования, сомневаться не приходится.

Структура и концепция книги

Лучший способ разобраться в проблеме —
написать книгу.

Как отмечалось выше, книга посвящена языку программирования С. Соответственно, в книге описывается синтаксис языка, и приводятся примеры составления программ на языке С.

Учиться программированию лучше и проще на примерах. Можно долго рассуждать о принципах, концепциях и подходах при составлении программ, но если все это не подкреплено прикладными навыками, результат будет малоутешительным. Материал книги подбирался и составлялся с учетом этого обстоятельства.

Каждая глава книги посвящена определенной теме. Вначале обычно дается теоретическая справка, после которой приводятся небольшие примеры, относящиеся к обсуждаемому аспекту проблемы. В конце каждой главы (за исключением последней) есть примеры решения задач, там же приводится резюме, призванное помочь читателю в закреплении материала главы. Еще одно назначение резюме — выделить основные, наиболее важные моменты, связанные с тематикой главы.

Завершает каждую главу (кроме последней) список задач для самостоятельного решения (имеются в виду задачи, предполагающие написание программного кода). Условия задач аналогичны тем, что рассматривались в соответствующей главе. Так что во многих отношениях задачи для самостоятельного решения представляют собой вариацию на основе задач уже рассмотренных. Это дает надежду, что книга может использоваться для изучения языка программирования С даже без преподавателя, то есть в режиме самообучения.

Значительная часть рассмотренных в книге программ имеют непосредственное отношение к математике и числовым расчетам. Причин для такого особого внимания к математическим задачам две:

- рассматриваемые в книге задачи сами по себе представляют практический интерес и очень часто встречаются в рамках различных университетских курсов (в первую очередь речь идет, конечно, о числовых методах);
- алгоритмы, используемые при решении прикладных математических задач, весьма показательны в плане применения возможностей языка программирования С.

НА ЗАМЕТКУ

При рассмотрении математических задач, если в этом есть необходимость, приводится краткая справка по сути задачи. Так что специальной математической подготовки читателю иметь не обязательно.

Помимо математических задач рассматриваются и другие. Таким образом, даже если читатель не планирует в дальнейшем посвятить себя решению проблем прикладного математического характера, есть все основания полагать, что книга для него все равно будет полезной.

Как отмечалось ранее, при изложении материала книги особое внимание уделяется сравнению языков C и C++. Важный в данном аспекте материал обычно выделяется специальным образом.

ЯЗЫК C++

Информация, важная для тех, кто планирует впоследствии изучать язык C++ или уже знаком с этим языком, выводится в специальных блоках — таких, как этот.

Особое внимание к языку C++ не случайно. Истоки языка C++, как мы помним, связаны именно с языком C. Но точка зрения, согласно которой то, что имеет место в языке C, справедливо и для языка C++, на сегодня является иллюзией. А иллюзии обходятся дорого. Поэтому мы сделаем все, чтобы их избежать.

Поскольку в книге речь идет о программных кодах, а их нужно каким-то образом тестировать, компилировать и запускать на выполнение, далее проанализируем некоторые моменты, связанные с программным обеспечением, полезным при написании программ на языке C.

НА ЗАМЕТКУ

Представленный далее материал носит технических характер и касается в основном методов установки, настройки и использования программного обеспечения, необходимого или полезного при написании программных кодов на языке C. Речь будет идти о нескольких компиляторах и средах разработки. Ориентирован соответствующий материал на читателей, малознакомых с программным обеспечением, используемым при написании программ. Поэтому не исключено, что для некоторых читателей данный материал покажется

чрезмерным. В таком случае следующие несколько разделов читатель может пропустить или бегло просмотреть.
Также важно отметить, что далее неявно предполагается использование операционной системы Windows. Объяснение такое: во-первых, предметом книги все же является язык программирования, а не особенности реализации сопутствующих технологий в разных операционных системах. Во-вторых, имеется подозрение, что пользователи альтернативных к Windows операционных систем, раз уж они смогли разобраться с этими самыми операционными системами, также найдут в себе силы для освоения программного обеспечения, используемого при создании программ на языке C.

Компиляторы и среды разработки

Если программа компилируется, то это уже повод, чтобы сделать ее коммерческой.

Итак, для программирования на языке C нам нужен по меньшей мере компилятор, который бы «понимал» язык C. Большинство современных компиляторов для языка C позволяют компилировать программы и на языке C++.

ПОДРОБНОСТИ

Пожалуй, правильнее было бы сказать, что компиляторы для языка C++ позволяют компилировать программы, написанные на языке C. Но в данном случае это не так уж важно. Важно то, что речь идет о компиляторах сразу для двух языков: C и C++.

Компиляторы

Многие вполне приличные компиляторы свободно загружаются и легко устанавливаются на компьютер. В принципе, в компиляторах недостатка нет (особенно если учесть не только свободно распространяемые, но и коммерческие). Мы остановимся на компиляторе *MinGW*.

НА ЗАМЕТКУ

Речь идет о проекте MinGW, в рамках которого разрабатывается и поддерживается коллекция программных утилит, которые можно задействовать при компиляции программ на языках C и C++.

Для установки программного обеспечения необходимо перейти на страницу проекта www.mingw.org. На рис. В.1 показано окно браузера, открытое на данной странице.

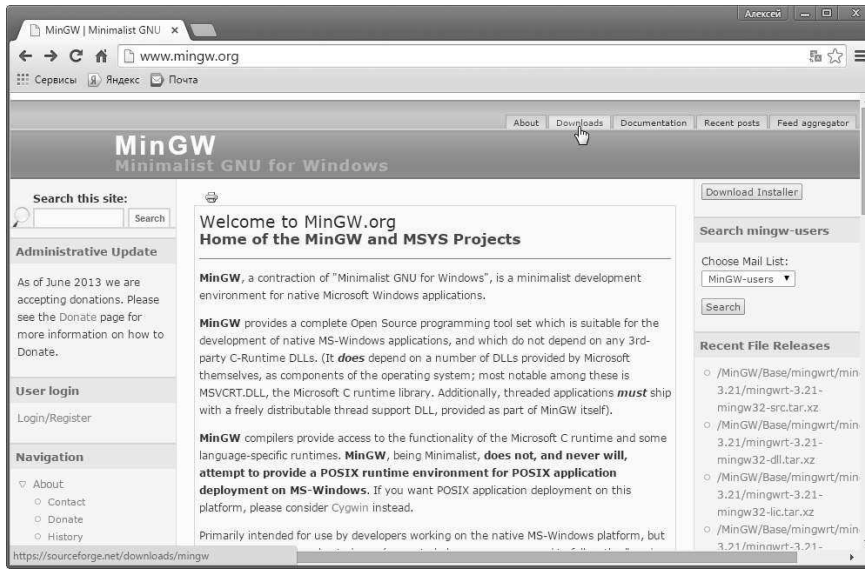

Рис. В.1. Страница проекта MinGW в окне браузера

Необходимое программное обеспечение (а нас интересует, напомним, компилятор) обычно без проблем загружается и устанавливается на компьютер.

НА ЗАМЕТКУ

Программное обеспечение для загрузки обычно представлено на вкладке **Downloads**. Но поскольку веб-страницы имеют свойство часто изменять свою «внешность», теоретически возможны отличия в том, что увидит читатель, выйдя на страницу проекта, от того, что показано на рис. В.1. Тем не менее хочется верить, что поиск и установка программного обеспечения не займут много времени и усилий.

Если компилятор установлен, мы в принципе можем приступать к программированию. Для этого нам придется набрать программный код и откомпилировать его. Но если мы используем только компилятор, то компилировать придется «вручную», а набирать программный код в текстовом редакторе. Это вполне допустимо, но очень неудобно.