Климович А.Д.

правильный способ изучить PYTHON с примерами и задачами

Издательство АСТ
г. Москва

УДК 004.38
ББК 32.973.2
К49

Климович, Александр Дмитриевич.

К49 Правильный способ изучить PYTHON с примерами и задачами / А. Д. Климович. — Москва : Издательство АСТ, 2026. — 448 с. — (Программирование от А до С++).

ISBN 978-5-17-178635-9.

Перед вами практическое руководство по программированию на одном из самых востребованных сегодня языков — Python, которое формирует не просто знание синтаксиса, а системное мышление разработчика, способного решать сложные задачи в любой области.

Книга строится на принципе глубокого понимания вместо механического запоминания. Каждая концепция раскрывается через трехмерную модель: что представляет собой конструкция, почему она работает именно так и когда ее следует применять. От первых команд и типов данных до объектно-ориентированного программирования и работы с современными технологиями — читатель проходит путь от обычного пользователя компьютера до создателя реальных программных воплощений.

Особое внимание уделено культуре качественного кода: правильные практики именования, структурирования, документирования и тестирования интегрированы в каждый раздел с самого начала. Книга развивает алгоритмическое мышление — умение декомпозировать сложные проблемы, создавать абстракции и находить элегантные решения.

Автор рассматривает Python не как самоцель, а как инструмент формирования вычислительного мышления, которое применимо далеко за пределами сферы программирования. В книге также затрагиваются этические аспекты разработки и ответственность программиста в создании инклюзивного и безопасного программного продукта.

Данное издание предназначено для всех, кто хочет освоить программирование на качественном уровне — от абсолютных новичков до тех, кто стремится систематизировать свои знания и развить профессиональное мышление разработчика.

УДК 004.38
ББК 32.973.2

ISBN 978-5-17-178635-9

Содержание

ПРЕДИСЛОВИЕ

Добро пожаловать в мир программирования

Вы держите в руках не просто учебник по языку программирования Python — это приглашение в удивительный мир создания программного обеспечения, где абстрактные идеи превращаются в работающие решения реальных проблем. Программирование сегодня стало одним из самых мощных инструментов человеческого интеллекта, позволяющим автоматизировать процессы, анализировать огромные объемы данных, создавать интерактивные приложения и даже моделировать сложные системы.

Эта книга создана с глубоким убеждением, что программирование должно преподаваться не как набор технических приемов или заучивание синтаксиса, а как дисциплина мышления, формирующая способность к систематическому решению проблем. Каждый программист, независимо от специализации, должен понимать фундаментальные принципы вычислений, алгоритмические подходы к решению задач и культуру создания качественного кода.

ФИЛОСОФИЯ ИЗУЧЕНИЯ ПРОГРАММИРОВАНИЯ

Программирование как способ мышления

Многие люди воспринимают программирование исключительно как техническую дисциплину — изучение синтаксиса языка, запоминание функций и методов. Такой подход приводит к формированию поверхностных знаний, которые быстро устаревают с появлением новых технологий. В действительности программирование представляет собой фундаментальный способ мышления, который можно применить к решению задач в любой области человеческой деятельности.

Программистское мышление характеризуется несколькими ключевыми особенностями. Во-первых, это системный подход к декомпозиции сложных проблем — умение разбивать сложную задачу на множество более простых подзадач, каждую из которых можно решить независимо. Во-вторых, это алгоритмическое мышление — способность представить процесс решения в виде последовательности четко определенных шагов. В-третьих, это абстрактное мышление — умение выделять общие закономерности и создавать универсальные решения, применимые к широкому классу задач.

Понимание вместо запоминания

Традиционные методы обучения программированию часто строятся по принципу «покажи и повтори»: студенту демонстрируют пример кода, а затем просят написать нечто похожее. Такой подход может дать быстрые результаты на начальном этапе, но создает ложное ощущение понимания и не формирует устойчивых навыков.

В этой книге мы следуем принципиально иному подходу. Каждая концепция объясняется с трех позиций: *что* представляет собой данная конструкция, *почему* она работает именно таким образом и *когда* ее следует применять. Такая трехмерная модель понимания позволяет не просто запомнить синтаксис, но и развить интуицию относительно эффективных способов решения различных классов задач.

Например, изучая циклы, мы не ограничиваемся демонстрацией синтаксиса `for` и `while`. Мы исследуем концептуальную природу итерации, анализируем различные паттерны (patterns) использования циклов, обсуждаем производительность различных подходов и рассматриваем альтернативные способы достижения того же результата. Такой подход формирует глубокое понимание, которое остается актуальным независимо от изменений в технологиях.

Культура качественного кода

Одним из наиболее важных аспектов профессионального программирования является культура создания качественного кода. К сожалению, многие учебные материалы игнорируют этот аспект, сосредотачиваясь исключительно на функциональности программ. В результате у начинающих программистов часто формируются вредные привычки, от которых потом крайне сложно избавиться.

Качественный код характеризуется не только корректностью работы, но и читаемостью, поддерживаемостью, эффективностью и элегантностью решения. Читаемость подразумевает, что код должен быть понятен другим программистам (и вам самим через несколько месяцев). Поддерживаемость означает возможность легкого внесения изменений и расширения функциональности. Эффективность касается разумного использования вычислительных ресурсов. Элегантность отражает красоту и простоту решения — качество, которое приходит с опытом и глубоким пониманием предметной области.

В этой книге вопросы качества кода не выносятся в отдельную главу, а интегрированы в каждый раздел. С самых первых примеров демонстрируются правильные практики именования переменных (variables), структурирования кода, написания комментариев (comments) и документации (documentation). Такой подход позволяет сформировать правильные привычки с самого начала обучения.

Почему Python — идеальный первый язык

Философия простоты и читаемости

Создатели Python руководствовались философией «код должен быть написан в том числе для чтения людьми, а не только для выполнения компьютерами». Эта философия кардинально отличает Python от многих других языков программирования, где синтаксические конструкции могут быть крайне сложными для понимания. Принцип читаемости кода в Python не является случайным — он отражает глубокое понимание того, что программирование в первую очередь является интеллектуальной деятельностью, требующей ясности мышления.

Синтаксис Python максимально приближен к естественному английскому языку. Приведем пример решения одной из задач в Python:

```
if age >= 21:
    print("Вы можете голосовать")
else:
    print("Вы пока не можете голосовать")
```

Даже человек, никогда не изучавший программирование, может интуитивно понять смысл этого кода. Такая естественность синтаксиса позволяет сосредоточиться на изучении алгоритмического мышления, а не на запоминании сложных синтаксических правил.

Мультипарадигмальность как преимущество

Python поддерживает несколько парадигм программирования: процедурную, объектно-ориентированную и функциональную. Для начинающего программиста это представляет огромную ценность, поскольку позволяет естественным образом переходить от простых линейных программ к более сложным архитектурным решениям.

На начальном этапе обучения студенты работают в процедурной парадигме, создавая простые последовательности команд. По мере накопления опыта они начинают группировать код в функции (functions), что соответствует функциональной парадигме. Затем, при решении более сложных задач, естественным образом приходит понимание необходимости объектно-ориентированного программирования (object-oriented programming). Такая эволюция подходов происходит органично, без резких переходов и концептуальных скачков.

Обширная экосистема и практическое применение

Python обладает одной из самых богатых экосистем среди всех языков программирования. Стандартная библиотека Python содержит модули для решения широчайшего спектра задач: от работы с файлами и сетевыми протоколами до математических вычислений и обработки данных. Дополнительные библиотеки, доступные через систему управления пакетами (package management system), расширяют возможности языка практически до бесконечности.

Эта особенность Python имеет огромное педагогическое значение. Студенты с самого начала обучения могут работать с реальными данными, создавать веб-приложения (web applications), анализировать статистику, обрабатывать изображения или даже создавать простые игры. Возможность быстро получать впечатляющие результаты поддерживает мотивацию к обучению и демонстрирует практическую ценность приобретаемых знаний.

Плавная кривая обучения

Python спроектирован таким образом, что базовые концепции можно изучать постепенно, без необходимости сразу погружаться в сложные технические детали. Например, при изучении функций студенту не нужно сразу понимать особенности управления памятью или сложности системы типов. Эти продвинутые темы можно изучать по мере необходимости, когда базовые навыки уже сформированы.

Одновременно Python не скрывает сложность искусственно. По мере развития навыков программирования студент может изучать все более продвинутые