Предисловие

Если вы находитесь в книжном магазине и ищете краткую историю об этой книге, то вот и она.

- *Python* — мощный компьютерный язык программирования, поддерживающий множество парадигм, который оптимизирован для обеспечения высокой продуктивности программистов, читабельности кода и качества программного обеспечения.
- *Эта книга* предлагает всестороннее и доскональное введение в сам язык Python. Ее цель в том, чтобы помочь вам справиться с основами Python, прежде чем переходить к их применению в своей работе. Подобно всем предшествующим изданиям книга направлена на то, чтобы служить единым всеобъемлющим обучающим ресурсом для всех новичков в Python, будут они использовать Python 2.X, Python 3.X или обе линейки.
- *Настоящее издание* было написано во времена версий Python 3.3 и 2.7 (и в основном уточнено с учетом текущей для середины 2019 года версии Python 3.7 — *прим.пер.*), а также в значительной степени расширено, чтобы отражать общепринятую практику, сложившуюся в мире Python.

В *данном предисловии* более подробно описаны цели, границы и структура этой книги. Читать его необязательно, но оно поможет сориентироваться до того, как вы приступите к чтению книги в целом.

"Экосистема" этой книги

Python представляет собой популярный язык программирования с открытым кодом, применяемый для написания автономных программ и сценарных приложений в широком разнообразии предметных областей. Он является бесплатным, переносимым, мощным, а также относительно легким и удивительно приятным в использовании. Программисты из всех уголков индустрии программного обеспечения считают, что ориентация Python на высокую продуктивность разработчиков и качество программного обеспечения должна быть стратегическим преимуществом как в крупных, так и в малых проектах.

Независимо от того, новичок вы в программировании или же профессиональный разработчик, эта книга направлена на то, чтобы ознакомить вас с языком Python быстрее, чем могут более ограниченные подходы. После чтения книги вы должны знать о языке Python достаточно для его применения в любых выбранных прикладных областях.

Книга задумывалась как руководство, в котором особый акцент делается на самом *языке Python*, а не на его специфических приложениях. Она входит в следующий набор:

- *Learning Python* (`http://www.oreilly.com/catalog/9781449355739`) — обучает самому языку Python с концентрацией внимания на основах языка, которые охватывают все предметные области;
- *Programming Python* (`http://www.oreilly.com/catalog/9780596158101`) показывает то, что можно делать с помощью Python после его изучения.

Такое разделение труда неслучайно. В то время как прикладные цели могут варьироваться от читателя к читателю, потребность в практичном обзоре основ языка остается неизменной. Книги, ориентированные на приложения, такие как *Programming Python*, логично продолжают данную книгу, используя реалистично масштабированные примеры для исследования роли Python в распространенных прикладных областях, среди которых веб-сеть, графические пользовательские интерфейсы, программные системы, базы данных и текст. Кроме того, книга *Python Pocket Reference* (`http://www.oreilly.com/catalog/9780596009403/`) предлагает не включенные здесь справочные материалы и задумана как дополнение настоящей книги.

Однако поскольку книга сосредоточена на основах, появляется возможность представить принципы Python глубже, чем многие программисты могут увидеть при начальном изучении языка. Принятый в ней восходящий подход и самостоятельные учебные примеры рассчитаны на то, чтобы постепенно обучать читателей всему языку.

Знания базового языка, получаемые в процессе чтения, применимы к любой программной системе Python, которая вам встретится — будь это популярный инструмент вроде Django, NumPy и App Engine или другая система.

Будучи основанной на трехдневном учебном курсе языка Python с повсеместными контрольными вопросами и упражнениями, книга также служит введением в язык, позволяющим каждому самостоятельно выбирать скорость изучения. Хотя ее формат лишен живого взаимодействия, принятого на занятии, это компенсируется добавочной глубиной и гибкостью, которую способна обеспечить только книга. Несмотря на то что книгу можно использовать многими способами, последовательные читатели сочтут ее приблизительно эквивалентной семестровому курсу лекций по Python.

О пятом издании

Предыдущее *четвертое издание* книги, вышедшее в 2009 году, охватывало версии Python 2.6 и 3.0[1]. В нем рассматривались многие и временами несовместимые изменения, внесенные в линейку Python 3.X. Также было предложено новое руководство по объектно-ориентированному программированию и новые главы по более сложным темам, таким как текст Unicode, декораторы и метаклассы.

[1] Недолго просуществовавшее третье издание книги, вышедшее в 2007 году, охватывало версию Python 2.5 и ее более простой и *узкий* мир Python с единственной линейкой. История этой книги доступна по ссылке `http://learning-python.com/books`. С течением лет книга выросла в размере и сложности прямо пропорционально собственному росту Python. Согласно приложению B только в версии Python 3.0 появилось 27 дополнений и 57 изменений языка, которые были отражены в книге, а в Python 3.3 тенденция продолжилась. В наши дни программисты на Python сталкиваются с двумя несовместимыми линейками, тремя основными парадигмами, изобилием расширенных инструментов и порядочной избыточностью функциональным средств, большинство из которых не удастся аккуратно разделить между линейками Python 2.X и Python 3.X. Ситуация не настолько обескураживающая, как может показаться (многие инструменты являются просто вариациями на какую-то тему), но она полностью отражена в этой всеобъемлющей книге по Python.

Пятое издание, законченное в 2013 году, является пересмотром предыдущего издания, обновленным для охвата версий *Python 3.3 и 2.7*, текущих на то время выпусков в линейках Python 3.X и Python 2.X (оно в основном уточнено с учетом текущей для середины 2019 года версии Python 3.7 — *прим.пер.*). Пятое издание включает все изменения языка в каждой линейке и доработано, чтобы улучшить представление материала. Ниже описаны основные моменты.

- Обзор *Python 2.X* обновлен для охвата таких средств, как включения словарей и множеств, которые раньше были доступны только в Python 3.X, но перенесены в Python 2.7.
- Обзор *Python 3.X* дополнен рассмотрением нового синтаксиса `yield` и `raise`; модели байт-кода `__pycache__`; пакетов пространств имен Python 3.3; режима с единым браузером в PyDoc; изменений в литералах и хранении; а также нового запускающего модуля Windows, появившегося в версии Python 3.3.
- Добавлены новые или расширены прежние *смешанные* темы, такие как JSON, `timeit`, PyPy, `os.popen`, генераторы, рекурсии, слабые ссылки, `__mro__`, `__iter__`, `super`, `__slots__`, метаклассы, дескрипторы, `random`, Sphinx и т.д. Кроме того, в целом повышена совместимость с Python 2.X как в примерах, так и в изложении материала.

В этом издании появилось новое *заключение* в виде главы 41 (об эволюции Python), два новых *приложения* (о последних изменениях в Python и новом запускающем модуле Windows), а также новая *глава* (об оценочных испытаниях: расширенный пример измерения времени выполнения кода). Краткая сводка по *изменениям Python* приведена в приложении B второго тома. Там также подводятся итоги по отличиям между линейками Python 2.X и Python 3.X в целом, которые впервые рассматривались в предыдущем издании, хотя часть из них вроде классов нового стиля охватывают обе линейки и просто стали обязательными в Python 3.X (смысл X вскоре будет объяснен).

Согласно последнему пункту в предыдущем списке данное издание также получило определенный прирост, потому что оно полнее раскрывает более *сложные возможности языка*, которые на протяжении последнего десятилетия многие из нас всячески старались игнорировать как необязательные, но теперь они становятся более популярными в коде Python. Как мы увидим, эти инструменты делают Python более мощным, однако также поднимают планку для новичков и способны расширить границы и определение Python. Поскольку вы можете столкнуться с любым из них, я непосредственно раскрываю такие инструменты в книге, а не притворяюсь, что их не существует.

Несмотря на обновления, пятое издание сохраняет бо́льшую часть структуры и содержания предыдущего издания и по-прежнему предназначено служить исчерпывающим обучающим ресурсом по линейкам Python 2.X и Python 3.X. В то время как оно ориентируется главным образом на пользователей Python 3.3 (3.7) и Python 2.7, исторический ракурс также делает его подходящим для *более старых* версий Python, которые регулярно применяются в наши дни.

Хотя будущее предсказывать невозможно, книга делает акцент на основах, которые были действительными на протяжении более двух десятилетий и вероятно будут применимы также к *будущим* версиям Python. Пробелы в книге, которые могут возникнуть с выходом новых версий, восполнят документы о нововведениях в комплекте руководств по Python.

Линейки Python 2.X и Python 3.X

Так как это сильно влияет на содержание книги, мне необходимо заранее сказать несколько слов об истории Python 2.X/3.X. Когда в 2009 году было написано *четвертое издание* книги, Python совсем недавно стал доступным в двух разновидностях:

- версия 3.0 была первой в линейке развивающейся и несовместимой мутации языка, в общем известной как *Python 3.X*;
- версия 2.6 сохранила обратную совместимость с огромной массой существующего кода Python и была последней в линейке, коллективно известной как *Python 2.X*.

Наряду с тем, что Python 3.X был в значительной степени тем же самым языком, он почти не запускал код, написанный для предшествующих выпусков. Он:

- навязал модель Unicode с обширными последствиями для строк, файлов и библиотек;
- превратил роль итераторов и генераторов во всепроникающую как часть более полной функциональной парадигмы;
- сделал обязательными классы нового стиля, которые объединяются с типами, но становятся более мощными и сложными;
- изменил многие фундаментальные инструменты и библиотеки, а также полностью заменил или удалил другие.

Единственная мутация `print` из оператора в функцию без преувеличения нарушила работу почти всех написанных ранее программ Python. Помимо стратегического потенциала обязательные модели Unicode и классов и вездесущие генераторы Python 3.X были ориентированы на отличающуюся практику программирования.

Хотя многие оценили Python 3.X как усовершенствование и как будущее Python, линейка Python 2.X все еще очень широко использовалась и долго поддерживалась параллельно с линейкой Python 3.X. Большинство кода Python было написано с применением Python 2.X и миграция на Python 3.X казалась медленным процессом.

Современная история Python 2.X/3.X

При написании *пятого издания* книги в 2013 году появились версии Python 3.3 (Python 3.7 на момент выхода русскоязычного издания в 2019 году) и Python 2.7, но история с Python 2.X/3.X по большому счету *не изменилась*. По существу теперь Python представляет собой мир с двумя версиями, в котором многие пользователи запускают *обе* линейки Python 2.X и Python 3.X согласно своим программным целям и зависимостям. При этом для многих новичков выбор между Python 2.X и Python 3.X остается выбором существующего программного обеспечения или передового языка. Несмотря на то что в Python 3.X было перенесено немало основных пакетов Python, многие другие пакеты в настоящее время по-прежнему работают только в Python 2.X.

Для ряда наблюдателей Python 3.X теперь выглядит как *песочница*, позволяющая исследовать новые идеи, тогда как Python 2.X рассматривается как *испытанный* Python, который не обладает всеми возможностями Python 3.X, но является намного более распространенным. Другие все еще видят Python 3.X как будущее — представление, которое поддерживалось основными планами разработки в 2013 году: Python 2.7 продолжит поддерживаться, но будет последней версией в линейке Python 2.X, в то время как Python 3.3 станет очередной версией в развитии линейки Python 3.X.

С другой стороны, инициативы вроде *PyPy* (реализация Python 2.X, которая предлагает ошеломляющие улучшения производительности) представляют будущее Python 2.X, если только не откровенное отделение.

Несмотря на все мнения, по прошествии почти пяти (уже более десяти — *прим. пер.*) лет после своего выпуска Python 3.X пока еще не заменил собой Python 2.X. Как показатель, Python 2.X по-прежнему загружается чаще Python 3.X для Windows из вебсайта `python.org` вопреки тому факту, что данная оценка естественным образом искажается в пользу *новых* пользователей и *самого недавнего* выпуска. Конечно, такие статистические данные подвержены изменениям, но прошедшие пять лет показательны в смысле внедрения Python 3.X. Существующая программная база Python 2.X все еще намного превосходит языковые расширения Python 3.X. Кроме того, последняя позиция в линейке Python 2.X делает версию Python 2.7 своего рода *стандартом де-факто*, невосприимчивым к постоянному темпу изменений в линейке Python 3.X — положительная характеристика для тех, кто ищет стабильную базу, и отрицательная для тех, кому важен непрерывный рост.

Лично я думаю, что современный мир Python достаточно велик, чтобы дать пристанище *обеим* линейкам Python 3.X и Python 2.X. Похоже, они удовлетворяют разным целям и привлекательны для разных сторон, что обеспечивает превосходство перед другими семействами языков (скажем, C и C++ давно сосуществуют, хотя возможно отличаются между собой больше, чем Python 2.X и Python 3.X). Вдобавок, поскольку две линейки Python настолько подобны, навыки, обретенные в результате изучения любой из двух линеек Python, почти полностью переносятся на другую, особенно если вы прибегали к помощи ресурсов, охватывающих две версии, таких как эта книга. Фактически до тех пор, пока вы понимаете, в чем они расходятся, часто есть возможность писать код, который выполняется в обеих линейках.

В то же самое время такое разделение становится значительной дилеммой, не подающей признаков ослабления, как для программистов, так и для авторов книг. Хотя при написании книги было бы легче притвориться, что Python 2.X не существует, и раскрывать только Python 3.X, это не будет отвечать потребностям крупной пользовательской базы Python, имеющейся на сегодняшний день. С применением Python 2.X был написан огромный объем кода, который в ближайшее время никуда не денется. И наряду с тем, что некоторые новички в языке могут и должны концентрироваться на Python 3.X, любой, кому приходится использовать код, написанный в прошлом, обязан быть на связи с нынешним миром Python 2.X. Так как могут пройти годы, пока многие сторонние библиотеки и расширения будут перенесены в Python 3.X, вполне возможно, что данная ветвь окажется не совсем временной.

Раскрытие линеек Python 3.X и Python 2.X

Чтобы учесть такое разветвление и отвечать потребностям всех потенциальных читателей, книга была обновлена для раскрытия обеих линеек Python 3.X и Python 2.X. Она ориентирована на программистов, применяющих Python 2.X, программистов, использующих Python 3.X, и программистов, привязанных к двум линейкам.

То есть книгу можно применять для изучения *любой из двух* линеек Python. Хотя часто делается особый акцент на Python 3.X, попутно также отмечаются отличия и инструменты Python 2.X, рассчитанные на программистов, которые используют более старый код. Наряду с тем, что две версии в основном похожи, в ряде важных аспектов они расходятся, на что и указывается в подходящих ситуациях.

Например, в большинстве примеров я буду применять вызовы `print` из Python 3.X, но также опишу оператор `print` из Python 2.X, так что вы сможете понять смысл

раннего кода и нередко использовать переносимые приемы вывода, которые выполняются в обеих линейках. Кроме того, я буду вводить новые средства, такие как оператор `nonlocal` в Python 3.X и строковый метод `format`, который доступен, начиная с версий Python 2.6 и Python 3.0, а также отмечать, когда подобные расширения отсутствуют в старых версиях Python.

Опосредованно это издание относится и к другим версиям Python 2.X/3.X, хотя в некоторых более старых версиях Python 2.X может не получиться запустить все примеры из книги. Скажем, несмотря на то, что декораторы классов доступны, начиная с версий Python 2.6 и Python 3.0, вы не можете применять их в более старых версиях Python 2.X, которые еще не располагали таким средством. Сводку по изменениям, внесенным в Python 2.X и Python 3.X, ищите в приложении B второго тома.

Какая версия Python должна использоваться?

Выбор версии может диктоваться вашей организацией, но если вы новичок в Python и учитесь самостоятельно, то можете задаться вопросом, какую версию устанавливать. Ответ зависит от ваших целей. Ниже приведено несколько советов по выбору.

Когда выбирать Python 3.X: новые возможности, развитие

Если вы впервые изучаете Python и не нуждаетесь в работе с любым существующим кодом Python 2.X, то я рекомендую начать с линейки Python 3.X. В ней устранены давнишние недостатки и убран устаревший хлам, но одновременно сохранены все первоначальные основные идеи и добавлены элегантные новые инструменты. Например, бесшовная модель Unicode и более широкое применение генераторов и методик функционального программирования в Python 3.X многими пользователями рассматривается как ценное свойство. Многие популярные библиотеки и инструменты Python уже доступны для Python 3.X или будут доступны ко времени чтения вами книги, особенно принимая во внимание постоянное усовершенствование линейки Python 3.X. Все развитие языка происходит только в Python 3.X, что приводит к добавлению новых средств и совершенствованию Python, но также превращает определение языка в постоянно движущуюся цель — компромисс, присущий переднему краю.

Когда выбирать Python 2.X: существующий код, стабильность

Если вы будете использовать систему, основанную на Python 2.X, то линейка Python 3.X в текущий момент может не подойти. Тем не менее, вы обнаружите, что книга также решит ваши проблемы и поможет перейти на Python 3.X в будущем. Вдобавок вы окажетесь в большой компании. Все группы, которые я обучал в 2012 году, применяли только Python 2.X, и по-прежнему встречается полезное программное обеспечение на Python в форме, предназначенной только для Python 2.X. Кроме того, в отличие от Python 3.X линейка Python 2.X больше не будет изменяться — что в зависимости от обстоятельств расценивается как достоинство или как недостаток. В использовании и написании кода Python 2.X нет ничего плохого, но у вас может быть желание следить за линейкой Python 3.X и ее продолжающимся развитием. Будущее Python еще предстоит написать, и во многом оно зависит от пользователей, включая вас.

Когда выбирать обе линейки: код, нейтральный к версиям

Вероятно, лучшая новость здесь заключается в том, что главные принципы Python остаются одинаковыми в обеих линейках — Python 2.X и Python 3.X отличаются в аспектах, которые многие пользователи сочтут незначительными, и книга призвана помочь освоить обе линейки. На самом деле при условии понимания отличий между ними зачастую легко писать код, нейтральный к версиям, который выполняется под управлением обеих линеек Python, что будет постоянно встречаться в этой книге. Ищите в приложении В второго тома указания по миграции Python 2.X/3.X и советы относительно написания кода для обеих линеек и аудитории.

Независимо от того, на какой версии или версиях вы сосредоточите внимание поначалу, обретенные вами навыки будут переноситься прямо туда, куда приведет работа с Python.

Замечание относительно применения X. Для ссылки на совокупные выпуски, представленные в двух линейках, будут использоваться формы Python 3.X и Python 2.X. Скажем, *Python 3.X* включает версии от 3.0 до 3.3/3.7 и будущие выпуски, а *Python 2.X* означает все версии, начиная с 2.0 и заканчивая 2.7 (и, по всей видимости, никаких других). Более конкретные выпуски упоминаются, когда предмет обсуждения применим только к ним (например, литералы множеств в Python 2.7 или запускающий модуль и пакеты пространств имен в Python 3.3). Иногда это замечание окажется слишком общим, поскольку некоторые средства, помеченные здесь как относящиеся к Python 2.X, могут отсутствовать в ранних выпусках Python 2.X, редко используемых в наши дни, но оно подогнано к линейке Python 2.X, которой насчитывается без малого 20 лет.

Предпосылки и усилия

Установить абсолютные предпосылки для этой книги невозможно, т.к. ее полезность и ценность могут зависеть от мотивации читателя в той же степени, что и от имеющихся у него знаний. И настоящие новички, и подлинные ветераны в программировании в прошлом успешно использовали данную книгу. Если вы заинтересованы в изучении Python и хотите уделить ему соответствующее время и внимание, тогда книга наверняка вам пригодится.

Сколько времени потребуется, чтобы изучить Python? Несмотря на то что затраты будут варьироваться от ученика к ученику, книга оказывает наибольшее воздействие, когда ее *читают*. Некоторые читатели могут применять книгу как справочный ресурс, но большинство людей, стремящихся к совершенному владению Python, должны ожидать затрат по меньшей мере недель и возможно месяцев на проработку материалов книги в зависимости от того, насколько тщательно они отслеживают предлагаемые примеры. Как уже упоминалось, книга является приблизительным эквивалентом семестрового курса по самому языку Python.

Такова оценка времени изучения только самого Python и обретения навыков, требуемых для его эффективного использования. Хотя этой книги может оказаться достаточно для достижения основных целей написания сценариев, читатели, которые надеются заняться разработкой программного обеспечения во всем объеме, должны ожидать, что после прочтения данной книги им придется выделить дополнительное время на освоение более крупномасштабных проектов и возможно обратиться к книгам вроде *Programming Python* (`http://www.oreilly.com/catalog/9780596158101`).

Новость может не выглядеть приятной для тех, кто надеется на мгновенное получение квалификации, но программирование — непростое занятие (вопреки тому, что вы могли слышать о нем). Современный язык Python и программное обеспечение в целом являются достаточно сложными, чтобы заслуживать усилий, потраченных на освоение таких всеобъемлющих книг, как эта. Ниже приведено несколько указаний по работе с книгой для читателей, обладающих и не обладающих опытом программирования.

Опытные программисты

Вы имеете начальное преимущество и можете быстрее пройти ряд начальных глав, но вы не должны упустить из виду основные идеи и вам вполне вероятно придется избавиться от некоторого багажа. В общем случае исследование любых приемов программирования или написания сценариев до настоящей книги может быть полезным по причине вероятных аналогий. С другой стороны, я также обнаружил, что предшествующий опыт программирования может стать помехой из-за предположений, укоренившихся в других языках (бывших программистов на Java или C++ очень легко распознать по коду Python, который они пишут поначалу!). Правильное применение Python требует принятия надлежащего типа мышления. За счет концентрации на основных концепциях данная книга призвана помочь вам научиться писать код на Python в духе Python.

Настоящие новички

Вы тоже в состоянии изучать здесь язык Python, равно как и само программирование, но возможно придется работать несколько усерднее и дополнять книгу другими вводными ресурсами. Даже если вы еще не считаете себя программистом, то книга все равно будет для вас полезной, но вероятно темп ее изучения окажется медленнее, к тому же в ходе чтения понадобится тщательно прорабатывать все примеры и упражнения. Также имейте в виду, что в книге уделяется больше внимания обучению самому языку Python, а не основам программирования. Если вы обнаруживаете, что не понимаете материал, тогда я рекомендую предварительно ознакомиться с какой-нибудь вводной книгой по программированию и только затем приниматься за эту книгу. На веб-сайте Python есть много ссылок на полезные ресурсы для начинающих.

Формально настоящая книга призвана служить *первой книгой по Python для любого рода новичков*. Она может не быть идеальным ресурсом для того, кто никогда раньше не прикасался к компьютеру (например, в ней не тратится время на выяснение, что собой представляет компьютер), но и не делается особенно много предположений относительно вашего опыта или образования в области программирования.

С другой стороны, я не буду обижать читателей, считая их “пустышками”, что бы под этим ни понималось — посредством Python легко делать полезные вещи и в книге будет показано, каким образом. В книге Python иногда противопоставляется с такими языками, как C, C++, Java и прочими, но вы можете благополучно игнорировать такие сравнения, если в прошлом не пользовались другими языками.

Структура этой книги

Чтобы помочь вам сориентироваться, в этом разделе предлагается краткое изложение содержания и целей главных частей книги. Если вам не терпится добраться до них, тогда можете спокойно пропустить данный раздел (или взамен просмотреть ог-

лавление). Однако краткая дорожная карта для настолько объемной книги некоторым читателям наверняка покажется достоинством.

По замыслу каждая *часть* раскрывает значительную функциональную область языка и состоит из *глав*, сконцентрированных на специфических темах или аспектах этой области. Вдобавок каждая глава заканчивается *контрольными вопросами* и ответами на них, а каждая часть — более крупными *упражнениями*, решения которых представлены в приложении.

Практика имеет значение. Я настоятельно рекомендую читателям по возможности прорабатывать контрольные вопросы и упражнения, предложенные в книге, и разбирать примеры в целом. В программировании ничто не способно заменить опробование прочитанного на практике. Делаете вы это с данной книгой или с собственным проектом, но фактическое написание кода имеет решающее значение, если вы хотите придерживаться изложенных здесь идей.

В целом презентация книги будет восходящей, т.к. таков характер Python. По мере продвижения примеры и темы становятся все более сложными. Скажем, классы Python по большому счету являются всего лишь пакетами функций, которые обрабатывают встроенные типы. Как только вы освоите встроенные типы и функции, классы потребуют относительно небольшого мысленного скачка. Поскольку каждая часть построена на основе предшествующих, большинство читателей сочтут, что *последовательное чтение* имеет наибольший смысл. Ниже приведен обзор главных частей книги, с которыми вам предстоит ознакомиться. По причине большого объема книга разделена на два тома.

Часть I (том 1)

Мы начнем с общего обзора Python, который ответит на часто задаваемые вопросы — почему люди используют язык, для чего он полезен и т.д. В первой главе представлены главные идеи, лежащие в основе технологии, чтобы ввести вас в курс дела. В остальных главах этой части исследуются способы, которыми Python и программисты запускают программы. Главная цель — дать вам достаточный объем информации, чтобы вы были в состоянии работать с последующими примерами и упражнениями.

Часть II (том 1)

Далее мы начинаем тур по языку Python с исследования основных встроенных объектных типов Python и выяснения, что посредством них можно предпринимать: чисел, списков, словарей и т.д. С помощью только этих инструментов уже можно многое сделать, и они лежат в основе каждого сценария Python. Данная часть книги является самой важной, поскольку она образует фундамент для материала, рассматриваемого в оставшихся главах. Здесь мы также исследуем динамическую типизацию и ссылки — ключевые аспекты для правильного применения Python.

Часть III (том 1)

В этой части будут представлены *операторы* Python — код, набираемый для создания и обработки объектов в Python. Здесь также будет описана общая синтаксическая модель Python. Хотя часть сконцентрирована на синтаксисе, в ней затрагиваются связанные инструменты (такие как система PyDoc), концепции итерации и альтернативные способы написания кода.

Часть IV (том 1)

В этой части начинается рассмотрение высокоуровневых инструментов структурирования программ на Python. *Функции* предлагают простой способ упаковки кода для многократного использования и избегания избыточности кода. Здесь мы исследуем правила поиска в областях видимости, приемы передачи аргументов, пресловутые лямбда-функции и многое другое. Мы также пересмотрим итераторы с точки зрения функционального программирования, представим определяемые пользователем генераторы и выясним, как измерять время выполнения кода Python для оценки производительности.

Часть V (том 1)

Модули Python позволяют организовывать операторы и функции в более крупные компоненты; в этой части объясняется, каким образом создавать, применять и перезагружать модули. Мы также обсудим такие темы, как пакеты модулей, перезагрузка модулей, импортирование пакетов, появившиеся в Python 3.3 пакеты пространств имен и атрибут `__name__`.

Часть VI (том 2)

Здесь мы исследуем инструмент объектно-ориентированного программирования Python — *класс*, который является необязательным, но мощным способом структурирования кода для настройки и многократного использования, что почти естественно минимизирует избыточность. Как вы увидите, классы задействуют идеи, раскрытые к этому месту в книге, и объектно-ориентированное программирование в Python сводится главным образом к поиску имен в связанных объектах с помощью специального первого аргумента в функциях. Вы также увидите, что объектно-ориентированное программирование в Python необязательно, но большинство находит объектно-ориентированное программирование на Python более простым, чем на других языках, и оно способно значительно сократить время разработки, особенно при выполнении долгосрочных стратегических проектов.

Часть VII (том 2)

Мы завершим рассмотрение основ языка в книге исследованием модели и операторов обработки исключений Python, а также кратким обзором инструментов разработки, которые станут более полезными, когда вы начнете писать крупные программы (например, инструменты для отладки и тестирования). Хотя исключения являются довольно легковесным инструментом, эта часть помещена после обсуждения классов, поскольку теперь все определяемые пользователем исключения должны быть классами. Мы также здесь раскроем более сложные темы, такие как диспетчеры контекста.

Часть VIII (том 2)

В этой части мы рассмотрим ряд дополнительных тем: Unicode и байтовые строки, инструменты управляемых атрибутов вроде свойств и дескрипторов, декораторы функций и классов и метаклассы. Главы данной части предназначены для дополнительного чтения, т.к. не всем программистам обязательно понимать раскрываемые в них темы. С другой стороны, читатели, которые должны обрабатывать интернационализированный текст либо двоичные данные или отвечать за разработку API-интерфейсов для использования другими

программистами, наверняка найдут в этой части что-то интересное для себя. Приводимые здесь примеры крупнее большинства других примеров в книге и могут служить материалом для самостоятельного изучения.

Часть IX (том 2)

Книга завершается четырьмя приложениями, в которых приведены советы по установке и применению Python на разнообразных платформах; представлен запускающий модуль Windows, появившийся в Python 3.3; подытожены изменения, внесенные в различные версии Python; и предложены решения упражнений для каждой части. Ответы на контрольные вопросы по главам приводятся в конце самих глав.

Чем эта книга не является

Учитывая накопленную с годами довольно крупную читательскую аудиторию, неизбежно появлялись и те, кто ожидал, что книга исполнит роль, выходящую за пределы ее границ. Итак, после того, как было указано, чем эта книга является, необходимо прояснить, чем она *не* является.

- Книга представляет собой обучающее руководство, а *не* справочник.
- В книге раскрывается сам язык, а *не* приложения, стандартные библиотеки или сторонние инструменты.
- Книга предлагает всесторонний взгляд на значительную тему, а *не* расплывчатый обзор.

Поскольку перечисленные пункты являются ключевыми для содержания книги, имеет смысл раскрыть их более подробно.

Это не справочник и не руководство по специфическим приложениям

Книга является *обучающим руководством по языку*, а не справочником и не книгой о приложениях. Так задумано: *современный язык Python* с его встроенными типами, генераторами, замыканиями, включениями, Unicode, декораторами и сочетанием парадигм процедурного, объектно-ориентированного и функционального программирования превращает один лишь основной язык в значительную тему, знание которой становится необходимым условием вашей будущей работы с Python независимо от того, какой предметной областью вы будете заниматься. Тем не менее, когда вы готовы к ознакомлению с другими ресурсами, то вот несколько советов и напоминаний.

Справочные ресурсы

Как следует из предыдущего описания структуры книги, для поиска информации вы можете использовать предметный указатель и содержание, но никаких справочных приложений в книге не предусмотрено. Если вас интересуют справочные ресурсы по Python (что очень скоро случится у большинства читателей), то я рекомендую книгу *Python Pocket Reference,* `http://www.oreilly.com/catalog/9780596009403/` (*Python. Карманный справочник, 5-е изд,* `http://www.williamspublishing.com/Books/978-5-8459-1912-0.html`), а также другие справочники, которые легко найти, и стандартные справоч-

ные руководства по Python, поддерживаемые на `http://www.python.org`. Последние бесплатны, всегда актуальны и доступны как в онлайновом режиме, так и в виде устанавливаемой версии для Windows.

Приложения и библиотеки

Как обсуждалось ранее, книга также не является руководством по специфическим *приложениям* вроде программирования для веб-сети, построения графических пользовательских интерфейсов и разработки систем. Опосредованно это включает библиотеки и инструменты, применяемые при работе над приложениями; хотя в ней представлены некоторые *стандартные библиотеки* и инструменты, в том числе `timeit`, `shelve`, `pickle`, `struct`, `json`, `pdb`, `os`, `urllib`, `re`, `xml`, `random`, *PyDoc* и *IDLE*, они не входят в основные границы ее охвата. Если вы ищете более полное раскрытие таких тем и уже хорошо знаете Python, тогда я рекомендую среди прочих почитать книгу *Programming Python* (`http://www.oreilly.com/catalog/9780596158101`), но сначала вы обязаны получить устойчивое представление об основном языке. В инженерной области, подобной разработке программного обеспечения, нужно сначала научиться ходить, а затем уже бегать.

Это не краткая история для спешащих людей

По объему книги легко понять, что она не экономит на деталях: книга представляет *полный язык Python*, а не краткий обзор упрощенного подмножества языковых средств. Попутно в ней также раскрываются *программные принципы*, которые важны для написания хорошего кода Python. Как упоминалось ранее, это книга для изучения на протяжении многих недель или месяцев, предназначенная для передачи уровня квалификации, который был бы получен в результате прослушивания полноценного курса лекций по Python.

Так тоже сделано намеренно. Конечно, многим читателям книги не требуются навыки полномасштабной разработки программного обеспечения, а некоторые могут осваивать Python постепенно. Вместе с тем, поскольку в коде, который вы будете встречать, может использоваться *любая* часть языка, ни одна часть не будет в полном смысле необязательной для большинства программистов. Кроме того, даже те, кто пишет сценарии от случая к случаю, и те, кто занимается Python в качестве хобби, должны знать базовые принципы разработки программного обеспечения, чтобы кодировать правильно и надлежащим образом применять готовые инструменты.

Эта книга призвана удовлетворять обе указанные потребности, т.е. язык и принципы, достаточно глубоко, чтобы быть полезной. Однако в конечном итоге обнаружится, что более развитые инструменты Python, такие как поддержка объектно-ориентированного и функционального программирования, относительно легко освоить, справившись с их предварительными условиями — и вы справитесь, если будете прорабатывать книгу по одной главе за раз.

Изложение последовательно до той степени, до которой позволяет Python

Говоря о *порядке чтения*, в этом издании также предпринята попытка свести к минимуму *ссылки вперед*, но изменения, внесенные в Python 3.X, в некоторых случаях делают их неизбежными (на самом деле иногда Python 3.X похоже предполагает, что вы уже знаете Python, хотя вы только его изучаете!).

Вот лишь несколько типичных примеров.

- Вывод, сортировки, строковый метод `format` и ряд вызовов `dict` полагаются на *ключевые* аргументы.
- Списки и проверки словарных ключей, а также вызовы `list`, используемые многими инструментами, заключают в себе концепции *итерации*.
- Применение `exec` для выполнения кода теперь предполагает наличие знаний *файловых объектов* и интерфейсов.
- Написание кода для обработки новых *исключений* требует знания *классов* и основ объектно-ориентированного программирования.
- И так далее — даже элементарное наследование поднимает более сложные темы, такие как *метаклассы* и *дескрипторы*.

Язык Python по-прежнему лучше всего изучать как прогрессию от простого к сложному, и последовательное чтение здесь все еще наиболее благоразумно. Тем не менее, некоторые темы могут требовать перепрыгивания и произвольного поиска. Чтобы свести это к минимуму, в книге будут указываться ссылки вперед, когда они случаются, но их влияние максимально смягчено.

Но если у вас мало времени. Хотя при изучении Python важна глубина, некоторые читатели могут располагать лишь ограниченным временем. Если вы заинтересованы в том, чтобы начать с быстрого тура по Python, то я рекомендую прочитать главы 1, 4, 10 и 28 (и возможно 26) — краткий обзор, который наверняка вызовет интерес к более полной истории, изложенной в остальных главах книги и необходимой большинству читателей. Книга намеренно *разделена на уровни*, чтобы облегчить освоение материала — с введениями, за которыми следуют детали, так что вы можете начать с обзоров и со временем погружаться глубже. Вам не обязательно читать эту книгу всю сразу, но принятый в ней постепенный подход призван помочь в конечном итоге разобраться с ее материалами.

Программы в книге

В целом я стремился в книге придерживаться независимого подхода в отношении версий Python и платформ. Книга задумана так, чтобы быть полезной пользователям всех версий Python. И все же из-за того, что Python с течением времени меняется, а платформы отличаются с практической точки зрения, имеет смысл описать специфические системы, которые вы увидите в действии в большинстве примеров.

Версии Python

Все примеры программ основаны на версиях Python *3.3/3.7 и 2.7*. Кроме того, многие примеры запускаются под управлением выпусков, предшествующих Python 3.X/2.X, и в ходе изложения даются примечания об истории изменения языка для пользователей более старых версий Python.

Однако поскольку книга сконцентрирована на ядре языка, вы можете быть уверены в том, что большинство приведенных здесь деталей не особенно сильно изменится в *будущих* выпусках Python, как отмечалось ранее. Большая часть книги применима также к *ранним* версиям Python кроме случаев, когда это не так; естественно, если вы попытаетесь использовать расширения, добавленные после выхода версии, с которой работаете, то успеха не добьетесь. В качестве эмпирического правила запомните: при наличии возможности модернизации наилучшей версией Python будет самая последняя.

Так как внимание в книге сосредоточено на ядре языка, большинство материала применимо к *Jython* и *IronPython* (реализациям языка Python, основанным на Java и .NET), а также к другим реализациям Python, таким как *Stackless* и *PyPy* (рассматриваются в главе 2). Упомянутые альтернативы отличаются в основном деталями использования, но не языком.

Платформы

Рассмотренные в книге примеры запускались на компьютере с *Windows 7 и 8*, хотя переносимость Python снижает важность данного момента особенно с учетом того, что книга ориентирована на основы. Вы заметите несколько особенностей, связанных с Windows, включая окна командной строки, экранные снимки, указания по установке и появившийся в версии Python 3.3 запускающий модуль Windows, но это отражает лишь тот факт, что большинство новичков, скорее всего, будут начинать с платформы Windows, и может быть благополучно проигнорировано пользователями других операционных систем.

Я также предоставляю несколько деталей запуска для других платформ вроде Linux, такие как применение строки `#!`, но в главе 3 и в приложении Б второго тома вы увидите, что запускающий модуль Windows, введенный в версии Python 3.3, даже это делает более переносимым приемом.

Загрузка кода примеров для книги

Исходный код примеров, рассмотренных в книге, а также решений предложенных упражнений, доступен для загрузки в форме файла ZIP по ссылке `http://oreil.ly/LearningPython-5E` и на веб-сайте издательства.

Разумеется, примеры лучше прорабатывать одновременно с чтением книги, к тому же вам понадобятся базовые знания по запуску программ на Python. Подробности начального запуска приведены в главе 3.

Использование кода, сопровождающего книгу

Код в моих книгах по Python предназначен для обучающих целей, и я буду рад, если он поможет читателям в таком качестве. Само издательство O'Reilly придерживается официальной политики относительно повторного использования кода примеров, рассмотренных в книге, которая сформулирована ниже.

Настоящая книга призвана помочь вам выполнять свою работу. Обычно если в книге предлагается пример кода, то вы можете применять его в собственных программах и документации. Вы не обязаны обращаться к нам за разрешением, если только не используете значительную долю кода. Скажем, написание программы, в которой задействовано несколько фрагментов кода из этой книги, разрешения не требует. Для продажи или распространения компакт-диска с примерами из книг O'Reilly разрешение обязательно. Ответ на вопрос путем цитирования данной книги и ссылки на пример кода разрешения не требует. Для встраивания значительного объема примеров кода, рассмотренных в этой книге, в документацию по вашему продукту разрешение обязательно.

Мы высоко ценим указание авторства, хотя и не требуем этого. Установление авторства обычно включает название книги, фамилию и имя автора, издательство и номер ISBN. Например: "Learning Python, Fifth Edition, by Mark Lutz. Copyright 2013 Mark Lutz, 978-1-4493-5573-9".

Если вам кажется, что способ использования вами примеров кода выходит за законные рамки или упомянутые выше разрешения, тогда свяжитесь с нами по следующему адресу электронной почты: `permissions@oreilly.com`.

Соглашения, используемые в этой книге

В книге приняты следующие типографские соглашения.

Курсив

Применяется для новых терминов.

`Моноширинный`

Используется для программного кода, содержимого файлов и вывода команд, а также для обозначения модулей, методов, операторов и системных команд.

`Моноширинный полужирный`

Применяется в разделах кода для обозначения текста или команд, которые должны быть набраны пользователем, и временами для выделения порций кода.

`Моноширинный курсив`

Используется для заменяемых фрагментов и ряда комментариев в коде.

Здесь приводится совет, указание или общее замечание, связанное с близлежащим текстом.

Здесь приводится предупреждение или предостережение, относящееся к близлежащему тексту.

Вы также будете периодически сталкиваться с *врезками* и *сносками*, которые часто необязательны для чтения, но дополнительно проясняют ситуацию по обсуждаемой теме. Во врезках, посвященных применению, таких как врезка "Что потребует внимания: срезы" в главе 7, нередко приводятся примеры сценариев использования для исследуемых средств.

Ждем ваших отзывов!

Вы, читатель этой книги, и есть главный ее критик. Мы ценим ваше мнение и хотим знать, что было сделано нами правильно, что можно было сделать лучше и что еще вы хотели бы увидеть изданным нами. Нам интересны любые ваши замечания в наш адрес.

Мы ждем ваших комментариев и надеемся на них. Вы можете прислать нам бумажное или электронное письмо либо просто посетить наш веб-сайт и оставить свои замечания там. Одним словом, любым удобным для вас способом дайте нам знать, нравится ли вам эта книга, а также выскажите свое мнение о том, как сделать наши книги более интересными для вас.

Отправляя письмо или сообщение, не забудьте указать название книги и ее авторов, а также свой обратный адрес. Мы внимательно ознакомимся с вашим мнением и обязательно учтем его при отборе и подготовке к изданию новых книг.

Наши электронные адреса:

E-mail: `info@dialektika.com`
WWW: `http://www.dialektika.com`

Благодарности

Так как в 2013 году я пишу уже пятое издание этой книги, мне трудно удержаться от взгляда в прошлое. Я использовал и продвигал Python в течение 21 года, писал книги о нем 18 лет и обучал группы вживую на протяжении 16 лет. Несмотря на пройденное время, я по-прежнему постоянно поражаюсь успехам, которых достиг Python — в областях, которые большинство из нас не могли себе даже представить в начале 1990-х годов. Поэтому, рискуя походить на безнадежно эгоцентричного автора, я надеюсь, что вы простите мне несколько заключительных слов касательно истории и благодарностей.

Предыстория

Моя собственная история, связанная с Python, предшествовала Python 1.0 и веб-сети (и восходит к тому времени, когда установка подразумевала извлечение вложений из сообщений электронной почты, их объединение, раскодирование и упование на то, что все это каким-то образом заработает). Когда я впервые открыл для себя Python как несостоявшийся разработчик программного обеспечения на C++ в 1992 году, то понятия не имел о том, каким образом он повлияет на следующие два десятилетия моей жизни. Через два года после выхода первого издания книги *Programming Python*, посвященной Python 1.3, в 1995 году я начал путешествовать по стране и миру, обучая Python новичков и экспертов. Завершив написание первого издания книги *Learning Python* в 1999 году, я стал независимым инструктором и писателем книг по Python, отчасти благодаря феноменальному росту его популярности.

И вот результат. К настоящему времени я написал 13 книг по Python (5 изданий этой книги и по 4 издания двух других), которых по моим данным было продано около 400 000 экземпляров (по состоянию на середину 2019 года 14 книг и 675 000 проданных экземпляров — *прим.пер.*). Я обучал языку Python более 15 лет, провел 260 обучающих курсов в США, Европе, Канаде и Мексике, а также выпустил примерно 4 000 студентов. Помимо движения к несбыточной мечте стать пассажиром, часто летающим самолетами, обучающие курсы помогли мне улучшить как эту, так и другие мои книги по Python. Обучение оттачивало книги и наоборот, результатом чего стало то, что мои книги близко отражают происходящее в группах студентов и могут служить для них жизнеспособной альтернативой.

Что касается самого языка Python, то в последние годы он вошел в число от 5 до 10 наиболее широко применяемых языков программирования в мире (по разным источникам и в разное время). Состояние Python будет исследоваться в первой главе книги, поэтому остаток истории я продолжу там.

Благодарности Python

Поскольку процесс обучения учит инструкторов обучать, эта книга многим обязана моим *обучающим курсам*. Я хотел бы поблагодарить всех *студентов*, посещавших мои курсы в течение последних 16 лет. Наряду с изменениями в самом Python ваши отзывы сыграли важную роль в приведении в порядок материала книги; нет ничего более поучительного, чем воочию наблюдать за тем, как 4 000 человек повторяют одни и те же ошибки начинающих! Изменения, внесенные в последние издания данной книги, обязаны своим появлением в первую очередь недавним курсам, хотя каждый курс, проводимый с 1997 года, в определенной степени помог улучшить книгу. Я хотел бы поблагодарить тех, кто выделял помещения для проведения курсов в Дублине,

Мехико, Барселоне, Лондоне, Эдмонтоне и Пуэрто-Рико; такой опыт был одним из самых значительных достижений в моей карьере.

Из-за того, что процесс написания учит инструкторов писать, книга также многим обязана своей *читательской аудитории*. Я хочу выразить благодарность бесчисленным *читателям*, которые в течение последних 18 лет находили время, чтобы помочь советами, как в онлайновом режиме, так и лично. Ваши отзывы были жизненно важны для развития этой книги и стали значимым фактором ее успеха — преимущества, которые присущи миру открытого кода. Комментарии читателей включали в себя весь спектр от "Вам следовало бы вообще запретить писать книги" до "Премного благодарен вам за написание этой книги"; если в таких вещах и возможен консенсус, то вероятно он находится где-то посередине, хотя я перефразирую Толкина: книга все еще слишком коротка.

Я также хотел бы выразить признательность всем, кто принимал участие в *производстве* данной книги. Всем, кто помогал сделать книгу надежным продуктом на многие годы — научным и художественным редакторам, маркетологам, техническим рецензентам и многим другим. Самому издательству O'Reilly за то, что мне был предоставлен шанс принять участие в 13 проектах по написанию книг; это было весело (и лишь немного напоминало фильм "День сурка").

Дополнительная благодарность всему *сообществу Python*; подобно большинству систем с открытым кодом Python является продуктом многих невоспетых усилий. Для меня было честью наблюдать, как Python вырос из младенца в семье сценарных языков до широко используемого инструмента, который в определенном виде развернут почти в каждой организации, занимающейся разработкой программного обеспечения. Оставив в стороне формальные расхождения, это был захватывающий процесс, заслуживающий того, чтобы стать его частью.

Я также хочу поблагодарить своего первоначального редактора в O'Reilly, покойного Фрэнка Уиллисона. Эта книга в значительной степени была идеей Фрэнка. Он оказал глубокое влияние и на мою карьеру, и на успех языка Python, когда он был в новинку — наследие, о котором я вспоминаю каждый раз, когда возникает искушение неправильно употребить слово "только".

Личные благодарности

И напоследок несколько личных благодарностей. Покойному Карлу Сагану за то, что вдохновил 18-летнего парня из Винконсина. Моей матери за мужество. Моим близким родственникам за поиск истины. Книге *The Shallows* за крайне необходимое предупреждение.

Моему сыну Майклу и дочерям Саманте и Роксане за то, что вы такие, какие есть. Я не уверен, что заметил, когда вы выросли, но горжусь тем, как вы это сделали, и стремлюсь увидеть, куда жизнь поведет вас дальше.

Моей жене Вере за терпение, стойкость, диетическую колу и претцели. Я рад, что наконец нашел тебя. Я не знаю, что нас ждет в ближайшие 50 лет, но я надеюсь провести их все вместе с тобой.

Марк Лутц, весна 2013 года