СОДЕРЖАНИЕ

ПРЕДИСЛОВИЕ

СТРАННАЯ ИСТОРИЯ ЭТОЙ КНИГИ

В январе 1999 года я готовился преподавать вводный курс программирования на языке Java. Я уже делал это трижды и теперь находился в замешательстве. Слишком много студентов не тянули курс, и даже среди преуспевающих общий уровень оставлял желать лучшего.

Как я обратил внимание, одной из причин были книги. Огромные, с излишним количеством ненужных подробностей о языке Java и явным недостатком простых уроков по программированию. И все студенты попадали в одну и ту же ловушку: бодрый старт, плавный прогресс, а вблизи пятой главы ловушка захлопывалась. Студенты получали слишком много нового материала и слишком быстро, и остаток семестра приходилось по кусочкам собирать знания воедино.

За две недели до начала занятий я решил написать собственную книгу. Вот что я хотел:

— сделать ее лаконичной. Лучше студенты прочтут десять страниц, чем не прочтут пятьдесят;
— быть осторожным с терминами. Как можно меньше использовать профессиональный жаргон и давать определение каждому термину при первом вхождении;
— увеличивать сложность постепенно. Чтобы избежать «ловушек», я взял самые трудные темы и разбил их на серии маленьких шагов;
— я сфокусировался на практике, а не на теории программирования. Я описал необходимый минимум знаний о языке Java и опустил все остальное.

Требовалось привлекательное название, поэтому по воле случая моя книга была названа «Думай как компьютерный ученый».

Первая версия книги получилась далеко не шедевром, но стала эффективной. Студенты читали и, главное, понимали, так что я мог уделять

время сложным темам, интересному материалу и (самое главное) позволить студентам практиковаться.

Я выпустил книгу под лицензией GNU Free Documentation License, которая позволяет читателям бесплатно копировать, изменять и распространять материалы из книги.

То, что произошло дальше, — удивительная история. Джефф Элкнер, учитель одной из школ в Вирджинии, взял мою книгу и адаптировал ее под язык программирования Python. Он прислал мне копию своего детища, и я приобрел необычный опыт: я изучал Python в процессе чтения собственной книги. Я опубликовал первое издание адаптации под Python в 2001 году в издательстве Green Tea Press.

В 2003 году я начал преподавать в колледже имени Франклина У. Олина, и темой первых моих уроков стал язык Python. Контраст с Java был поразительным. Студенты меньше страдали, больше учились, работали над более интересными проектами и в целом получали гораздо больше удовольствия.

С тех пор я продолжал развивать книгу, исправляя ошибки, улучшая некоторые примеры и добавляя материал, в частности упражнения.

Результатом стала эта книга, теперь с менее пафосным названием — «Основы Python». Перечислю некоторые из изменений.

— Я добавил раздел об отладке в конец каждой главы. В этих разделах представлены общие методы поиска и предотвращения ошибок, а также предупреждения о подводных камнях Python.
— Я добавил дополнительные упражнения, начиная с коротких задач для закрепления материала и заканчивая несколькими крупными проектами. Большинство упражнений содержат ссылки на мои варианты решений.
— Я добавил серию случаев из практики — объемные примеры с упражнениями, решениями и обсуждением.
— Я подробнее поговорил о способе разработки программ и основных шаблонов проектирования.
— Я добавил приложения об отладке и анализе алгоритмов.

Второе издание книги содержит следующие изменения.

— Текст книги и код всех примеров были обновлены до версии Python 3.
— Я добавил несколько новых разделов и разместил больше информации в интернете, чтобы помочь новичкам запустить Python в браузере, поэтому устанавливать Python не придется, пока вы сами не захотите.

— Для модуля turtle из главы 4 я переключился с собственного графического «черепашьего» пакета Swampy на стандартный модуль Python, turtle, простой в установке и более мощный.
— Я добавил новую главу под названием «Синтаксический сахар». В ней описаны дополнительные возможности Python, не строго необходимые, но иногда очень полезные.

Надеюсь, вам понравится работать с этой книгой, и она поможет вам научиться программировать и мыслить как программист, хотя бы немного.

Аллен Б. Дауни
Инженерно-технический колледж
имени Франклина У. Олина

УСЛОВНЫЕ ОБОЗНАЧЕНИЯ

В этой книге используются следующие обозначения.

Курсивный шрифт
Им оформлены новые термины, имена и расширения файлов.

Полужирный шрифт
Указывает на термины, определенные в словаре терминов, а также полужирным шрифтом выделены URL-адреса и адреса электронной почты.

`Моноширинный шрифт`
Используется для оформления листингов программ, а также для выделения в основном тексте фрагментов кода программ, таких как имена переменных или функций, базы данных, типы данных, переменных сред, инструкции и зарезервированные слова.

`Полужирный моноширинный шрифт`
Обозначает команды или текст, который должен быть введен пользователем.

`Курсивный моноширинный шрифт`
Так оформлен текст, который читатель должен заменить собственными значениями или значениями, определенными контекстом.

ИСПОЛЬЗОВАНИЕ ПРИМЕРОВ КОДА

Дополнительный материал (примеры кода, упражнения и так далее) доступны для загрузки по адресу **http://thinkpython2.com/code**.

БЛАГОДАРНОСТИ

Большое спасибо Джеффу Элкнеру, который адаптировал мою книгу про язык Java под Python, запустил этот проект и познакомил меня с ныне любимым моим языком.

Также спасибо Крису Мейерсу за то, что написал несколько разделов в книгу «Основы Python».

Спасибо организации Free Software Foundation за разработку лицензии GNU Free Documentation License, которая сделала возможной мое сотрудничество с Джеффом и Крисом, и Creative Commons за лицензию, которую я использую сейчас.

Спасибо редакторам из издательства Lulu, которые работали над книгой How to Think Like a Computer Scientist «Основы Python».

Спасибо редакторам из издательства O'Reilly Media, которые трудились над книгой Think Python «Основы Python».

Спасибо всем студентам, которые работали с предыдущими изданиями этой книги, и всем читателям (перечисленным ниже), которые прислали исправления и предложения.

СПИСОК УЧАСТНИКОВ ПРОЕКТА

За последние несколько лет свыше сотни проницательных и вдумчивых читателей прислали свои предложения и исправления. Их вклад и энтузиазм очень способствовали развитию проекта.

Предложения или исправления отправляйте на адрес **feedback@thinkpython.com**. Если ваши отзывы помогут, я добавлю вас в список участников (если, конечно, вы не попросите об обратном).

Если вы напишете хотя бы фрагмент предложения, в котором обнаружили ошибку, вы упростите мне поиск. Номера страниц и разделов тоже можно указывать, но это не так удобно. Спасибо!

— Ллойд Хью Аллен прислал исправление в раздел 8.4.
— Ивон Булянн прислал исправление семантической ошибки в главе 5.

— Фред Бреммер представил исправление в раздел 2.1.
— Джона Коэн написал Perl-скрипт для преобразования исходного кода LaTeX этой книги в красивый HTML-код.
— Майкл Конлон предложил исправить грамматическую ошибку в главе 2 и помог со стилистикой главы 1, а также инициировал обсуждение технических аспектов работы интерпретаторов.
— Бенуа Жирар исправил смешную ошибку в разделе 5.6.
— Кортни Глисон и Кэтрин Смит написали скрипт *horsebet.py*, который использовался в качестве примера в предыдущем издании книги. Их программа теперь опубликована на сайте книги.
— Ли Харр представил больше исправлений, чем я могу тут перечислить, посему его можно указать в качестве одного из главных научных редакторов.
— Джеймс Кейлин — очень внимательный студент. Он предоставил многочисленные исправления.
— Дэвид Кершоу исправил нерабочую функцию `catTwice()` в разделе 3.10.
— Эдди Лам прислал многочисленные исправления в главы 1, 2 и 3. Он также исправил код `make`-файла, чтобы тот создавал индекс при первом запуске, и помог нам настроить систему управления версиями.
— Ман Ён Ли прислал правку кода в примере в разделе 2.4.
— Дэвид Майо отметил, что слово «неосознанно» в главе 1 необходимо изменить на «подсознательно».
— Крис Макалун прислал несколько исправлений в разделы 3.9 и 3.10.
— Мэтью Дж. Моэлтер, опытный редактор, прислал многочисленные исправления и улучшения книги.
— Саймон Дикон Монтфорд сообщил об отсутствии определения функции и нескольких опечатках в главе 3. Он также нашел ошибки в функции инкремента в главе 13.
— Джон Оузтс исправил определение «возвращаемого значения» в главе 3.
— Кевин Паркс прислал ценные комментарии и маркетинговое предложение о продвижении книги.
— Дэвид Пул сообщил об опечатке в словаре терминов главы 1, а также написал добрые слова о проекте.
— Майкл Шмитт отправил исправление к главе о файлах и исключениях.
— Робин Шоу указал на ошибку в разделе 13.1, где функция `printTime()` использовалась в примере до определения.

— Пол Слай обнаружил ошибки в главе 7 и Perl-скрипте Джоны Коэна, который генерирует HTML из LaTeX.
— Крейг Т. Снидал протестировал книгу в Университете Дрю. Он внес несколько ценных предложений и исправлений.
— Ян Томас и его ученики используют книгу на курсах по программированию. Они первыми проверили главы второй половины книги и внесли многочисленные исправления и предложения.
— Кит Верхейден прислал изменения для главы 3.
— Питер Уинстанли сообщил нам о назойливой ошибке в главе 3.
— Крис Вробель исправил код в главе, посвященной вводу-выводу и исключениям.
— Моше Задка внес неоценимый вклад. Он не только написал черновик главы о словарях, но и взял на себя руководство проектом на ранних этапах.
— Кристоф Цвершке прислал несколько исправлений и педагогических предложений и объяснил разницу между немецкими словами *gleich* и *selbe*.
— Джеймс Майер заметил множество орфографических и типографских ошибок, в том числе две в этом списке.
— Хайден Макафи обнаружил потенциально противоречивое несоответствие между двумя примерами.
— Анхель Арнал из международной команды переводчиков, работающих над испанской версией книги, нашел несколько ошибок в английской версии.
— Таухидул Хок и Лекс Бережный создали иллюстрации к главе 1 и помогли усовершенствовать многие другие рисунки.
— Д-р Микеле Альзетта обнаружил ошибку в главе 8 и прислал несколько интересных педагогических комментариев и предложений для алгоритмов Фибоначчи и карточной игры Old Maid.
— Энди Митчелл обнаружил опечатку в главе 1 и проблему с примером из главы 2.
— Калин Харви предложил разъяснение к главе 7 и нашел несколько опечаток.
— Кристофер П. Смит нашел несколько опечаток и помог нам обновить книгу под Python версии 2.2.
— Дэвид Хатчинс нашел опечатку в предисловии.

— Грегор Лингл преподает Python в средней школе в Вене, Австрия. Он работал над переводом книги на немецкий язык и обнаружил пару ошибок в главе 5.
— Джули Питерс нашла опечатку в предисловии.
— Флорин Оприна предложил улучшение метода `makeTime()`, исправление метода `printTime()` и нашел забавную опечатку.
— Д. Ж. Верб предложил разъяснение к главе 3.
— Кен обнаружил несколько ошибок в главах 8, 9 и 11.
— Иво Вевер нашел опечатку в главе 5 и предложил разъяснение к главе 3.
— Кертис Янко предложил разъяснение к главе 2.
— Бен Логан прислал информацию о многих опечатках и проблемах при переводе книги в формат HTML.
— Джейсон Армстронг увидел, что пропущено слово в главе 2.
— Луи Кордье заметил в главе 16 место, где код не соответствовал тексту.
— Брайан Кейн предложил несколько пояснений к главам 2 и 3.
— Роб Блэк прислал множество исправлений, включая некоторые изменения для Python 2.2.
— Жан-Филипп Рей, сотрудник компании Ecole Centrale Paris, прислал несколько исправлений, включая некоторые обновления для Python 2.2, и другие значимые улучшения.
— Джейсон Мэдер из Университета Джорджа Вашингтона внес ряд полезных предложений и исправлений.
— Ян Гундтофте-Брюн напомнил нам, какой артикль нужно использовать со словом ошибка (*an* error).
— Абель Давид и Алексис Динно напомнили нам, что множественное число от matrix — это matrices, а не matrixes. Эта ошибка была в книге годами, и два читателя с одинаковыми инициалами сообщили об этом в один и тот же день. Удивительно.
— Чарльз Тейер призвал нас избавиться от точек с запятой, которые мы ставили в конце некоторых строк кода, и помог разобраться с путаницей в использовании терминов «аргумент» и «параметр».
— Роджер Сперберг указал на извращенную логику в главе 3.
— Сэм Булл указал на запутанный абзац в главе 2.
— Эндрю Ченг указал на два случая использования переменных до их определения.

— С. Кори Капел обнаружил пропущенное слово и опечатку в главе 4.
— Алессандра помогла разобраться с «черепашкой».
— Вим Шампань нашел смысловую ошибку в примере со словарями.
— Дуглас Райт указал на проблему с целочисленным делением в функции `arc()`.
— Джаред Спиндор нашел ненужный текст.
— Лин Пэйхэн прислал несколько очень полезных предложений.
— Рэй Хагтведт прислал информацию о двух ошибках и одной не совсем ошибке.
— Торстен Хюбш указал на несоответствие в модуле Swampy.
— Инга Петухова исправила код примера в главе 14.
— Арне Бабенхаузерхайде прислал несколько полезных исправлений.
— Марк Э. Касида здорово помог избавиться от повторов.
— Скотт Тайлер доработал некоторые упущения. А еще прислал кучу исправлений.
— Гордон Шепард прислал несколько исправлений, каждое в отдельном письме.
— Эндрю Тернер заметил ошибку в главе 8.
— Адам Хобарт исправил проблему с целочисленным делением в функции `arc()`.
— Дэрил Хаммонд и Сара Циммерман указали, что я слишком рано начал объяснять модуль math.pi. А еще Сара заметила опечатку.
— Джордж Сасс обнаружил ошибку в разделе «Отладка».
— Брайан Бингхэм предложил упражнение 11.5.
— Лиа Энгельберт-Фентон обнаружила, что я использовал слово `tuple` в качестве имени переменной вопреки моему собственному совету. А также нашла кучу опечаток и случаев обращения ранее определения.
— Джо Фанке заметил опечатку.
— Чао Чао Чен обнаружил несоответствие в примере с Фибоначчи.
— Джефф Пейн разъяснил разницу между словами *space* и *spam*.
— Либо Пенти нашел опечатку.
— Грегг Линд и Эбигейл Хейтхофф предложили упражнение 14.3.
— Макс Хайлперин прислал ряд исправлений и предложений. Макс — один из авторов экстраординарной книги Concrete Abstractions (Course Technology, 1998), которую вы можете прочитать, когда закроете эту.
— Чотипат Порнавалай обнаружил ошибку в сообщении об ошибке.

— Станислав Антол прислал список очень полезных предложений.
— Эрик Пашман прислал ряд исправлений для глав 4–11.
— Мигель Азеведо нашел несколько опечаток.
— Цзяньхуа Лю прислал длиннющий список исправлений.
— Ник Кинг нашел пропущенное слово.
— Мартин Зютер прислал длинный список предложений.
— Адам Циммерман обнаружил несоответствие в моем экземпляре «экземпляра» и несколько других ошибок.
— Ратнакар Тивари добавил сноску, объясняющую, что такое вырожденные треугольники.
— Анураг Гоэль предложил другое решение для функции `is_abecedarian()` и некоторые дополнительные исправления. И он знает, как правильно пишется имя Джейн Остин!
— Келли Кратцер заметил одну опечатку.
— Марк Гриффитс указал на запутанный пример в главе 3.
— Ройдан Онги обнаружил ошибку в моей реализации метода Ньютона.
— Патрик Воловец помог мне с проблемой в HTML-версии.
— Марк Чонофски рассказал мне о новом ключевом слове в Python 3.
— Рассел Коулман помог мне с геометрией.
— Вэй Хуан заметил несколько опечаток.
— Карен Барбер обнаружила самую древнюю опечатку в книге.
— Нам Нгуен нашел опечатку и указал, что я использовал шаблон Decorator, но не упомянул об этом.
— Стефан Морен прислал несколько исправлений и предложений.
— Пол Ступ исправил опечатку в функции `uses_only()`.
— Эрик Броннер указал на путаницу в обсуждении порядка операций.
— Александрос Гезерлис своими приложениями установил новый стандарт их количества и качества. Мы очень признательны!
— Серый Томас знает, где право, а где лево.
— Джованни Эскобар Соса прислал длинный список исправлений и предложений.
— Аликс Этьен исправила один из URL-адресов.
— Куанг Он нашел опечатку.
— Даниэль Нилсон исправил ошибку в порядке операций.
— Уилл Макгиннис отметил, что функция `polyline()` была определена по-разному в двух местах.
— Сваруп Саху заметил пропущенную точку с запятой.

— Фрэнк Хекер указал на неясности в упражнении и некоторые неработающие ссылки.
— Анимеш Б помог мне сделать запутанный пример более понятным.
— Мартин Касперсен обнаружил две ошибки округления.
— Грегор Ульм прислал несколько исправлений и предложений.
— Димитриос Циригкас предложил мне уточнить упражнение.
— Карлос Тафур прислал список исправлений и предложений.
— Мартин Нордслеттен обнаружил ошибку в решении.
— Ларс О.Д. Кристенсен нашел неработающую ссылку.
— Виктор Симеоне нашел опечатку.
— Свен Хоукстер отметил, что имя переменной `input` совпадает с именем встроенной функции.
— Вьет Ле нашел опечатку.
— Стивен Грегори указал на проблему с функцией `cmp()` в Python 3.
— Мэтью Шульц дал мне знать о неработающей ссылке.
— Локеш Кумар Макани сообщил о неработающих ссылках и некоторых изменениях в сообщениях об ошибках.
— Ишвар Бхат исправил мое утверждение о последней теореме Ферма.
— Брайан Макги предложил разъяснение.
— Андреа Занелла перевела книгу на итальянский язык и попутно внесла ряд исправлений.
— Огромное спасибо Мелиссе Льюис и Лучано Рамальо за прекрасные комментарии и предложения по поводу второго издания.
— Спасибо Гарри Персивалю из компании PythonAnywhere за его помощь, позволившую людям запускать Python в браузере.
— Ксавье Ван Обель внес несколько полезных исправлений во второе издание.

ГЛАВА 1

ПУТЬ РАЗРАБОТКИ

Цель этой книги — научить вас мыслить как настоящий программист. Этот способ сочетает в себе особенности мышления математика, инженера и ученого. Как математики компьютерные специалисты используют формальные языки для выражения идей (в частности, вычислений). Как инженеры они что-то проектируют, собирают отдельные компоненты в системы и оценивают компромиссы между альтернативами. Как ученые они наблюдают за поведением сложных систем, формируют гипотезы и тестируют прогнозы.

Единственный самый важный навык для разработчика — умение **находить решение задачи**. Для этого он должен сформулировать задачу, подойти творчески к поиску решения, а затем точно и ясно его реализовать. Как видите, обучение программированию — это прекрасная возможность попрактиковаться в решении задач. Вот почему эта глава называется «Путь разработки».

С одной стороны, вы будете учиться программировать, что само по себе полезный навык. С другой — вы будете использовать программирование как средство для достижения цели. По мере того как мы будем продвигаться дальше, вы поймете, о чем я.

ЧТО ТАКОЕ ПРОГРАММА?

Программа — это последовательность инструкций, в которых указано, как выполнять вычисления. Вычисления могут быть математическими, такими как решение системы уравнений или поиск корней многочлена, но это также могут быть символические вычисления, например поиск и замена текста в документе, или что-то графическое, например обработка изображения или воспроизведение видеоролика.

Детали реализации выглядят по-разному на разных языках, но несколько основных инструкций универсальны для любого языка:

— *ввод данных (input):*
  Получение данных с клавиатуры, из файла, по сети или с другого устройства.
— *вывод данных (output):*
  Отображение данных на экране, сохранение их в файл, отправка по сети и так далее.
— *математические операции (math):*
  Выполнение основных математических операций, таких как сложение и умножение.
— *условное выполнение (conditional execution):*
  Проверка определенных условий и выполнение соответствующего кода.
— *повторение (repetition):*
  Выполнение некоторого действия несколько раз, часто с некоторыми изменениями.

Верьте или нет, но это все, что нужно знать. Каждая программа, которую вы когда-либо использовали, независимо от ее сложности, состоит из таких инструкций. Таким образом, вы можете представить программирование как процесс разбиения большой и сложной задачи на всё более мелкие подзадачи, пока подзадачи не станут достаточно простыми, чтобы их можно было сформулировать с помощью одной из этих инструкций.

ЗАПУСК PYTHON

Работа с Python начинается с установки Python и связанного программного обеспечения на компьютер. Если вы знакомы с вашей операционной системой и особенно если вы знакомы с интерфейсом командной строки, у вас не должно возникнуть проблем. Но новичкам сложновато изучать системное администрирование и программирование одновременно.

Чтобы облегчить задачу, я рекомендую запустить Python в браузере. Позже, когда вы освоитесь, я предложу вам установить Python на компьютер.

Существует несколько веб-сайтов для запуска Python. Если у вас уже есть любимый, можете смело использовать его. В противном случае я рекомендую

PythonAnywhere. Подробные инструкции по началу работы приведены на странице **http://tinyurl.com/thinkpython2e**.

Существует две версии языка, Python 2 и Python 3. Они очень похожи, поэтому, если вы изучите одну из них, то легко сможете использовать и другую. На самом деле есть только несколько отличий, с которыми вы столкнетесь как новичок. Эта книга написана под Python 3, но я добавил несколько примечаний, касающихся и Python 2.

Интерпретатор (interpreter) Python — это программа, которая анализирует, обрабатывает и выполняет код Python. В зависимости от установленной операционной системы вы можете запустить интерпретатор, щелкнув мышью по значку или набрав слово python в командной строке. В случае успешного запуска вы должны увидеть примерно следующий результат:

```
Python 3.4.0 (default, Jun 19 2015, 14:20:21)
[GCC 4.8.2] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Первые три строки содержат информацию об интерпретаторе и операционной системе, в которой он запущен, поэтому у вас сведения могут отличаться. Но вы должны проверить, что номер версии (в этом примере — `3.4.0`) начинается с 3, что указывает на то, что вы используете Python 3. Если номер версии начинается с 2, вы работаете (как вы уже догадались) с Python 2.

Последняя строка представляет собой **приглашение (prompt)**, которое указывает, что интерпретатор ожидает ввод команды от пользователя. Если вы наберете в строке 1 + 1 и нажмете клавишу **Enter**, интерпретатор отобразит результат:

```
>>> 1 + 1
2
```

Теперь вы готовы начать учиться. С этого момента я предполагаю, что вы знаете, как использовать интерпретатор Python.

ПЕРВАЯ ПРОГРАММА

Традиционно первая программа, которую пишут на любом новом языке программирования называется Hello, World! Все, что она делает, это отображает слова Hello, World! (то есть «Привет, мир!»). На языке Python программа выглядит так:

```
>>> print('Привет, мир!')
```

Это пример **инструкции печати**, хотя на самом деле она ничего не печатает на бумаге. Она отображает результат на экране. В этом случае результатом будут следующие слова:

```
Привет, мир!
```

Кавычки в программе отмечают начало и конец отображаемого текста; они не видны в выводе.

Скобки `()` указывают, что `print` — это функция. Мы рассмотрим функции в главе 3.

В Python 2 инструкция печати немного отличается; это не функция, поэтому скобки не используются.

```
>>> print 'Привет, мир!'
```

Это различие будет иметь смысл далее, но для начала достаточно просто об этом помнить.

АРИФМЕТИЧЕСКИЕ ОПЕРАТОРЫ

После простенькой программы Hello, World наш следующий шаг — арифметика. В языке Python есть **операторы (operators)**, которые выглядят как специальные символы, представляющие вычисления, такие как сложение и умножение.

Операторы `+`, `-` и `*` выполняют сложение, вычитание и умножение, соответственно, как показано в следующих примерах:

```
>>> 40 + 2
42
>>> 43 - 1
42
>>> 6 * 7
42
```

Оператор `/` выполняет деление:

```
>>> 84 / 2
42.0
```

Вы можете спросить, почему результат `42.0`, а не `42`. Я объясню это в следующем разделе.