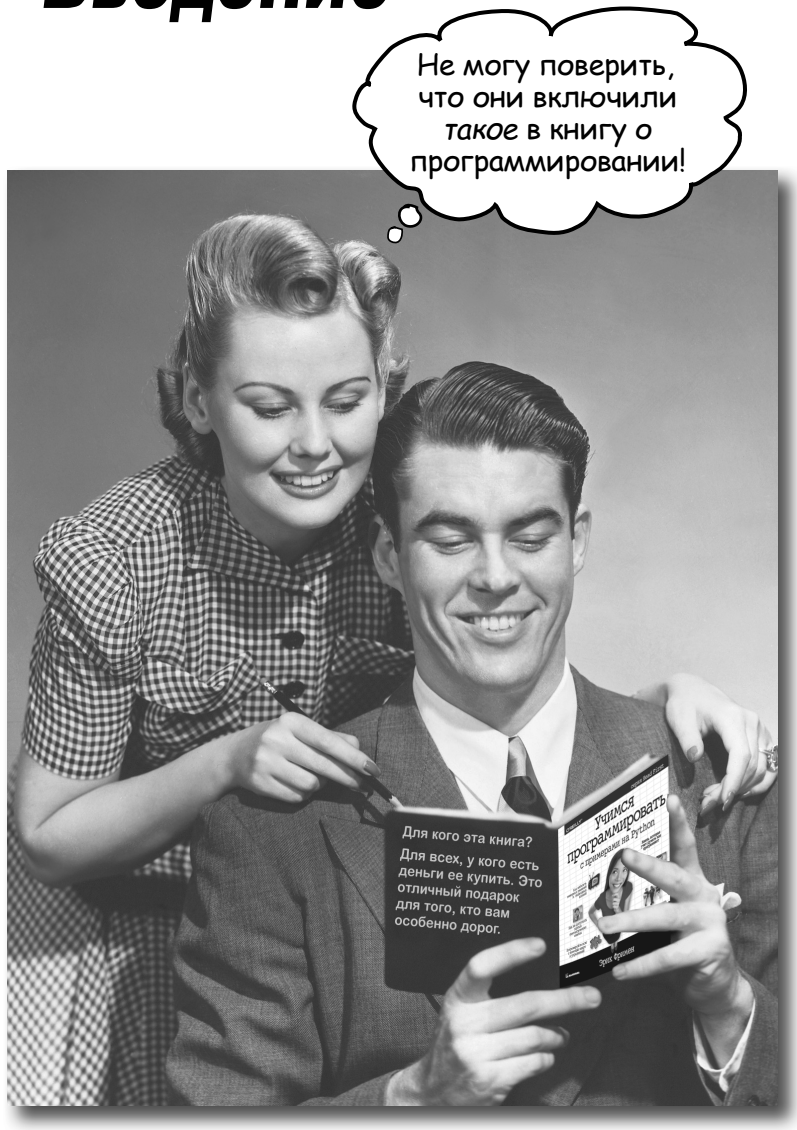
Введение



Сейчас мы ответим на самый главный вопрос:
"Зачем они включили ТАКОЕ в книгу о программировании?"

Для кого эта книга

Если вы ответите “да” на все нижеперечисленные вопросы...

1. Вы хотите научиться **писать**, **понимать** и **редактировать** программы?

2. Вы предпочитаете **неформальное общение**, а не **скучные лекции**?

...то эта книга для вас.

(Замечание от сбытовиков: это книга для всех, у кого есть деньги ее купить.)

Это НЕ справочник и не руководство по синтаксису Python. (Для этого есть Google, правильно?) Это книга для тех, кто хочет научиться программировать.

Кому эта книга не подойдет

Если вы ответите “да” на любой из нижеперечисленных вопросов...

1. **Вы абсолютный новичок в компьютерах?**

   Если вы не знаете, как работает компьютер, как управлять файлами и папками, как устанавливать приложения и набирать тексты, то лучше сначала освоить компьютер.

2. Вы опытный программист, который ищет **справочник**?

3. Вы **боитесь нестандартных решений**? Вам проще остаться дома, чем пойти в ресторан в кедах? Вы считаете, что юмору не место в книге по программированию?

...то эта книга не для вас.

Мы знаем, о чем вы думаете

“Это точно не комикс?”

“Что это за странные рисунки?”

“А что, так можно чему-то научиться?”

И мы знаем, о чем думает ваш мозг

Мозг всегда ищет новизну. Он напряжен, насторожен и *ждет* чего-то необычного. Таким его создала природа, и это помогает нам выживать.

В современном мире у нас не так много шансов попасть в лапы тигру, но мозг все равно контролирует ситуацию. Мало ли что...

А как мозг реагирует на обыденные, повседневные, рутинные ситуации? Всячески *игнорирует* их, чтобы они не мешали ему решать *основную* задачу — фиксировать только то, что *важно*. Он не утруждает себя запоминанием скучной информации, создавая для нее фильтр “это явно не важно”.

Но как мозг определяет, что важно? Представьте, что вы вышли погулять в парк, и тут — бац! — из кустов выпрыгивает тигр. Что происходит в этот момент в вашей голове и вашем теле?

Вся жизнь проносится перед глазами... Всплеск адреналина... Запускается цепь *химических реакций*.

Именно они подсказывают мозгу...

Вот что важно! Думай быстрее!

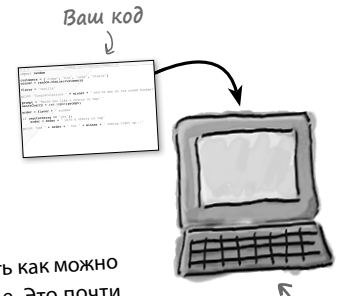
А теперь представим, что вы сидите дома или в библиотеке. Тепло, комфортно, тигры не тревожат. Вы занимаетесь учебой, готовитесь к экзамену или пытаетесь справиться с заданием, на которое начальник отвел вам неделю, максимум полторы.

Тут-то и возникает проблема: ваш мозг исправно пытается вам помочь. Разве можно тратить ценные ресурсы памяти на запоминание столь очевидно несущественной информации? Лучше ведь использовать память для хранения чего-то действительно важного. Вот тигры — это важно. Лесные пожары — тоже важно. Идти или не идти в ресторан в кедах — ну куда уж важнее?!

Нельзя сказать мозгу: “Эй, приятель, спасибо, конечно, за заботу, но я тут книгу хочу почитать! Ты не смотри, что она скучная и не вызывает всплеска адреналина. Просто потрудись все запомнить”.

Мы считаем читателей книги учениками

Как выучить что-либо? Сначала нужно понять, а затем постараться не забыть. Дело вовсе не в зубрежке, когда мы забиваем голову фактами. Последние исследования в области когнитивной науки, нейробиологии и педагогической психологии показали, что обучение не сводится к запоминанию текста на страницах учебника. Мозг включается по-другому.

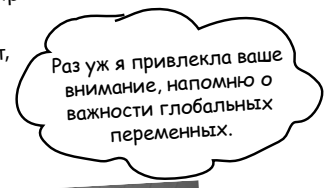


Ключевые принципы обучения в книгах серии Head First

Визуальный подход. Изображения запоминаются лучше, чем текст, существенно повышая эффективность обучения (до 89% согласно современным исследованиям). Они также делают книгу более понятной.

Выноски и подписи к рисункам. Релевантный текст лучше располагать как можно ближе к иллюстрациям, а не где-то внизу или на следующей странице. Это почти вдвое улучшает способность ученика решать соответствующие задачи.

Разговорный стиль и импровизированные диалоги. Согласно последним исследованиям ученики на 40% лучше усваивают информацию, когда автор обращается напрямую к читателю от первого лица, используя разговорный стиль вместо формальных описаний и рассказывая истории вместо того, чтобы читать лекции. Язык должен быть простым, не стоит все воспринимать слишком серьезно. К кому вы больше прислушаетесь: к другу, популярно объясняющему, как все работает, или лектору, читающему конспект перед большой аудиторией?



Развитие навыков мышления. Если вы не научитесь включать мозги, то знаний в голове не прибавится. Читатель должен быть мотивирован, пытлив и вовлечен в процесс обучения, ему должно быть интересно решать задачи, находить ответы и приобретать новые знания. Для этого нужны головоломки, упражнения и каверзные вопросы — все то, что заставляет работать оба полушария мозга.

Привлечение (и удержание) внимания читателя. Все мы сталкивались с ситуацией, когда книгу хочется прочитать, но после первой же страницы мы начинаем засыпать от скуки. Мозг реагирует на все, что нестандартно, непривычно, странно, притягательно, неожиданно. Изучение сложных технических тем не обязательно должно быть скучным. Мозг будет быстрее учиться в нетипичной обстановке.

Эмоциональная вовлеченность. Мы знаем, что способность к запоминанию сильно зависит от эмоционального состояния. Вы помните то, что для вас важно, то, что вызывает *переживания*. Нет, речь не идет о душераздирающих историях про собак и их преданность хозяевам. Мы говорим о таких эмоциях, как удивление, любопытство и смех, неожиданных реакциях ("Что за ерунда?") и возгласах "Ух ты!", когда удается справиться с головоломкой или найти решение задачи, которую все вокруг считали слишком сложной.

Метапознание: осознание знаний

Если вы хотите учиться, причем учиться быстро, получая фундаментальные знания, то следует уделять внимание тому, как мы концентрируем внимание на главном. Необходимо думать о том, как мы думаем, и изучать то, как мы учимся.

Никто из нас не изучал метакогнитивные процессы и теорию обучения, когда учился в школе. Учителя *ожидали*, что мы будем учиться, но не объясняли нам, как правильно учить предметы.

Раз вы держите в руках эту книгу, значит, хотите научиться программировать, но наверняка не заинтересованы в том, чтобы тратить на это все свободное время. Вы стремитесь *запомнить* прочитанное и применить полученные знания на практике. Для этого вы должны *понять* то, что прочитали. Чтобы извлечь максимум из книги и вообще из любого обучающего процесса, необходимо настроить свой мозг на обучение.

Трудность состоит в том, чтобы убедить мозг рассматривать изучаемый материал как *особо важный*. Жизненно важный. Такой же важный, как тигр, выпрыгивающий из кустов. В противном случае вам придется постоянно сражаться за то, чтобы не дать мозгу выкинуть из головы новую информацию как нестоящую запоминания.

Как же убедить мозг в том, что программирование не менее важно, чем тигр в кустах?

Есть медленный, утомительный путь и быстрый, более эффективный. Медленный заключается в регулярном повторении. Нам по силам изучить и запомнить даже самую скучную тему, если постоянно ее повторять. В какой-то момент мозг подумает: “Не вижу тут ничего важного, но он повторяет это *снова и снова*, наверное, для него это все-таки важно”.

Быстрый способ заключается в ***повышении активности мозга*** и сочетании разных ее видов. Доказано, что все принципы, перечисленные на предыдущей странице, помогают мозгу запоминать информацию, повышая его активность. Согласно исследованиям включение поясняющего текста в иллюстрации (вместо того чтобы располагать его в пределах страницы) заставляет мозг искать связь между словами и рисунками, вовлекая в процесс больше нейронов. А чем больше активных нейронов, тем выше шансы, что мозг посчитает данную информацию стоящей внимания и запоминания.

Разговорный стиль изложения полезен, поскольку люди становятся более внимательными, когда чувствуют, что вовлечены в диалог и должны следить за ходом мысли. И что самое поразительное, мозг совершенно не заботит тот факт, что “разговор” происходит между вами и книгой. С другой стороны, когда стиль изложения сухой и формальный, мозг воспринимает это так, будто вы на лекции в аудитории. Значит, можно пересесть на заднюю парту и поспать.

Впрочем, наглядные иллюстрации и разговорный стиль — это далеко не все.

Вот что сделали мы

Мы задействовали множество ***иллюстраций***, поскольку мозгу легче воспринимать образы, чем текст. Не зря говорят, что одна картинка стоит тысячи слов. Точнее, не 1000, а 1024. Мы старались сочетать текст с изображениями, так как известно, что мозг работает эффективнее, когда текст включен в иллюстрацию, к которой относится, а не расположен отдельно от нее.

Мы нередко применяли ***избыточность***, высказывая одну и ту же мысль по-разному и разными способами. Это повышает шансы на то, что информация в конечном итоге будет запечатлена в разных участках вашего мозга.

Учимся понимать Python

Мы старались подавать концепции и рисунки в ***неожиданной*** манере, ведь мозг восприимчив ко всему новому. Кроме того, мы стремились создавать определенный ***эмоциональный фон***, поскольку мозг чутко реагирует на биохимию эмоций. Нам легче запоминать то, что вызывает *сопереживание*, даже если это всего лишь ***юмор***, ***удивление*** или ***заинтересованность***.

Мы использовали ***разговорный стиль*** изложения от первого лица, так как мозг более сосредоточен во время беседы, чем когда вы просто слушаете лекцию. Это происходит даже в процессе *чтения* книги.

Мы включили в книгу более 120 ***упражнений***, поскольку мозгу легче обучаться и запоминать информацию, когда вы что-то ***делаете***, а не просто *читаете* текст. Все задачи непростые, но решаемые — такой формат наиболее привычен.

САМОЕ ГЛАВНОЕ

Мы применяли ***разные методики обучения***: кому-то нравятся пошаговые инструкции, кто-то предпочитает сначала понять картину в целом, а кому-то достаточно увидеть пример кода. Но независимо от предпочтений читатели только выиграют, если научатся анализировать одни и те же задачи под разными углами.

Кроссворды

Мы постарались задействовать ***оба полушария*** вашего мозга, ведь чем большая часть мозга вовлечена в процесс обучения, тем выше вероятность усвоения и запоминания информации и тем дольше сохраняется концентрация внимания. Зачастую, когда одно из полушарий работает, другое отдыхает. Это позволяет повысить продуктивность обучения в течение более длительного времени.

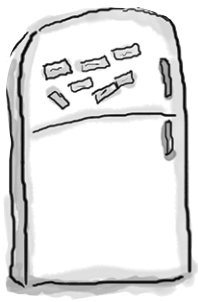
Они будут учиться с нами

Мы добавили в книгу ***диалоги*** и упражнения, позволяющие взглянуть на проблему с ***разных точек зрения***. Наш мозг начинает глубже вникать в проблему, когда его заставляют давать оценки и составлять суждения.

Мы предложили читателям ряд ***открытых задач*** в виде упражнений и ***вопросов***, на которые не всегда можно дать однозначный ответ. Наш мозг настроен на обучение и запоминание, когда ему приходится *работать* над решением. Сами посудите: вы не сможете стать атлетом, наблюдая за тем, как тренируются другие. Но мы постарались сделать так, чтобы, упорно работая, вы продвигались к намеченной цели и чтобы ***ни один дендрит вашего мозга не был задействован зря*** в процессе рассмотрения сложных примеров или анализа насыщенного техническими терминами текста.

Мы создали ***персонажей***. Они встретятся вам в диалогах, примерах, иллюстрациях — в общем, по всей книге. Мозг уделяет больше внимания живым существам, чем неодушевленным предметам или отвлеченным понятиям.

Мы придерживались принципа ***80/20*** (известного как *принцип Парето*). Мы полагаем, что, раз уж вы решили стать программистом, то наверняка захотите прочитать и другие книги. Так что мы не пытались рассказать вам *все*. Только то, что вам *действительно* нужно.



Вот что можете сделать вы, чтобы настроить свой мозг на обучение

Итак, мы свое дело сделали. Остальное зависит от вас. Приведенные ниже советы — это лишь начальные ориентиры. Прислушайтесь к своему внутреннему голосу и постарайтесь понять, что вам подходит, а что — нет. Не бойтесь искать новые пути!

Можете вырезать и приклеить на дверцу холодильника

1 Не торопитесь. Чем больше вы поймете, тем меньше придется зубрить.

Не нужно просто *читать*. Делайте паузы и анализируйте прочитанное. Если вам задают вопрос, не спешите подсматривать ответ. Представьте, будто кто-то из друзей спрашивает вашего совета. Чем глубже вы пытаетесь вникнуть в суть вопроса, тем выше шансы понять и запомнить ответ.

2 Выполняйте упражнения и делайте заметки.

Упражнения придумали мы, но выполнять их вам. И помните: они в книге не для того, чтобы вы просто прочитали ответ. ***Возьмите карандаш*** и запишите свое решение. Доказано, что физическая активность в процессе обучения улучшает усвоение информации.

3 Читайте врезки “Не бойтесь задавать вопросы”.

Все они важны. В данной книге это ***основной материал***! Их нельзя пропускать.

4 Не читайте на ночь другие книги. Не нагружайте мозг.

Основная часть обучения (та, что связана с запоминанием) происходит *после* того, как вы закрываете книгу. Мозгу требуется время на то, чтобы все переварить. Если в этот промежуток времени вы нагрузите его еще чем-то, часть прочитанного будет успешно забыта.

5 Пейте воду. Много воды.

Мозг на 90% состоит из воды. Обезвоживание (которое может случиться еще до того, как вы почувствуете жажду) снижает когнитивные способности человека.

6 Обсуждайте прочитанное. Думайте вслух.

Речь стимулирует активность различных участков мозга. Если вы пытаетесь понять и запомнить прочитанное, проговаривайте материал вслух. А еще лучше, попробуйте объяснить его кому-то другому. Это ускорит обучение, и к тому же так можно открыть для себя малозаметные детали, на которые вы не обратили внимания в процессе чтения книги.

7 Прислушивайтесь к внутреннему голосу.

Следите за тем, чтобы не перегружать мозг. Если чувствуете, что теряете нить изложения или начинаете забывать прочитанное, значит, пора сделать перерыв. После определенного момента в голову уже ничего не лезет, сколько ни запихивай. Только хуже сделаете.

8 *Дайте волю эмоциям!*

Мозг должен понимать, что подаваемая ему информация *важна для вас*. Придумывайте истории персонажей. Делайте шутливые подписи к иллюстрациям. Лучше поморщиться от плохой шутки, чем не испытать никаких эмоций.

9 *Пишите программы.*

Примените полученные знания в новом проекте или вернитесь к старому проекту и переработайте его. Постарайтесь получить какой-то опыт помимо упражнений и примеров из книги. Все, что вам нужно, — карандаш и задача, которую можно решить средствами программирования.

10 *Старайтесь высыпаться.*

Чтобы научиться программировать, необходимо создать множество новых связей между нейронами мозга. Это происходит во время сна. Так что утро вечера мудренее!

Это важно

Перед вами обучающая книга, а не справочник. Мы намеренно выкинули все, что может помешать изучению тех или иных тем. И если вы впервые взяли книгу в руки, то рекомендуем читать с самого начала, поскольку в книге предполагается, что вы знакомы с предыдущим материалом.

Мы хотим, чтобы вы научились думать как программист.

Кто-то считает программирование компьютерной наукой, но откроем маленький секрет: это вообще не наука, да и связь с компьютерами условна (они связаны не сильнее, чем астрономия с телескопами). Это в большей степени способ мышления, называемый в наши дни *вычислительным мышлением*. Научившись мыслить подобным образом, вы сможете применять имеющиеся знания к любым задачам, системам и языкам программирования.

В книге мы используем язык Python.

Невозможно научиться водить автомобиль, не сев за руль. Точно так же невозможно обучиться вычислительному мышлению, не освоив язык программирования. Без практики такая теория бесполезна. Поэтому в книге мы остановили свой выбор на чрезвычайно популярном языке программирования Python. Его достоинства будут подробно описаны в главе 1, а пока вам достаточно знать, что он прекрасно подойдет как начинающим, так и профессиональным программистам.

Мы не собираемся рассматривать все аспекты языка Python.

Мы даже не пытаемся этого делать, т.к. Python — слишком обширная тема. Это было бы уместно для справочника, но наша книга — не справочник, и мы не ставим перед собой цель рассказать все что только можно о Python. Наша настоящая цель — обучить вас основам программирования и вычислительного мышления, чтобы в будущем, когда вам захочется прочитать еще какую-то книгу по *любому из языков программирования*, вы понимали, о чем идет речь.

Не важно, какая у вас система: Mac, Windows или Linux.

В книге рассматривается Python — кросс-платформенный язык программирования, поддерживаемый в любой операционной системе. Большинство снимков экрана в книге было сделано в Mac, но в Windows и Linux они будут выглядеть почти так же.

Мы стараемся придерживаться практики написания хорошо структурированного и понятного кода.

Вам хочется писать программы, которые будут понятны и вам, и другим людям; программы, которые продолжат работать даже с выходом в следующем году очередной версии Python. В книге мы научим вас, как с самого начала писать понятный, четко структурированный код, которым можно гордиться и который не стыдно показать друзьям. Единственное, что отличает его от профессионального кода, — *аннотации* (а-ля рукописные), поясняющие работу программ. Мы считаем, что они намного лучше подходят для обучающей книги, чем традиционные комментарии в коде (если вы не понимаете, о чем идет речь, то скоро все поймете, буквально через пару глав). Но не переживайте: мы заодно научим вас правильно документировать код и покажем примеры составления документации к программам. Как бы там ни было, важно уметь писать программы самым простым способом, чтобы можно было как можно быстрее решить поставленную задачу и заняться чем-то более приятным.

Аннотации выглядят так

Программирование — серьезная профессия. Придется много и упорно работать.

Программисты мыслят иначе, смотрят на мир по-другому. Иногда подход к программированию будет казаться вам вполне логичным, а иногда — чересчур абстрактным или слишком запутанным. Некоторые концепции программирования требуют длительного обдумывания. Понадобится время, прежде чем вы все поймете. Главное — помните: мы стараемся все подавать в максимально дружественной манере. Просто уделите этому достаточно времени, чтобы все улеглось в голове, и не стесняйтесь перечитывать непонятные разделы, если почувствуете такую необходимость.

Упражнения обязательны.

Все приведенные в книге упражнения и задания *обязательны* для выполнения. Они являются неотъемлемой частью процесса обучения. Одни упражнения тренируют память, другие закрепляют понимание материала, третьи помогают применить полученные знания на практике. Если пропустить их, то часть материала останется непонятой (и рано или поздно вы будете сбиты с толку). Единственное, что можно смело пропустить, — это кроссворды, хотя и они помогают мозгу усвоить новую терминологию.

Повторение — мать учения.

Характерной особенностью книги является то, что мы пытаемся научить вас по-настоящему понимать программирование. Мы хотим, чтобы, прочитав книгу, вы помнили, чему научились. В большинстве справочных руководств не ставится цель добиться запоминания и усвоения материала, но наша книга — обучающая, поэтому время от времени некоторые концепции будут упоминаться повторно.

Краткость — сестра таланта.

Читатели часто пишут нам о том, как это утомительно — продираться сквозь 200 строк кода примера ради тех двух строчек, которые нужно изучить. Большинство примеров в книге подано максимально компактно, чтобы все было просто и понятно. Не ждите от примеров чудес — они составлены в целях обучения и не всегда полнофункциональны. Но мы постарались сделать их забавными и необычными, чтобы их было интересно показать друзьям или родственникам.

Не ко всем упражнениям и заданиям есть ответы.

Некоторые из заданий, приведенных во врезках “Сила мысли”, не имеют однозначного решения, а некоторые составлены таким образом, что вы сами должны решить, является ли ваш ответ правильным. В некоторых упражнениях есть подсказки, которые направят вас в нужную сторону.

Скачайте файлы примеров.

Все рассматриваемые в книге примеры доступны по следующему адресу:

`http://wickedlysmart.com/hflearntocode`

Кроме того, файлы примеров доступны также на сайте издательства “Диалектика”:

`http://www.williamspublishing.com/Books/978-5-907144-98-9.html`

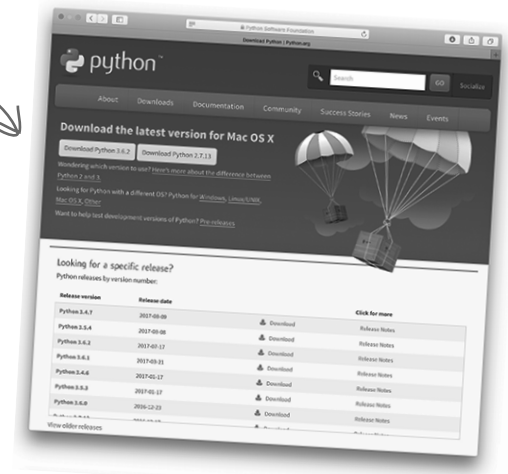
У нас нет горячей линии, но все примеры и рабочие файлы можно скачать на сайте `http://wickedlysmart.com/hflearntocode`

Необходимо установить Python

Скорее всего, на вашем компьютере либо не установлен Python, либо установлен, но более старой версии. В книге мы используем Python 3, который на момент написания книги имел версию 3.6. Соответственно, вам понадобится установить версию 3.6 или более позднюю. Вот как это сделать.

- **В macOS** откройте браузер и введите в адресной строке следующее:

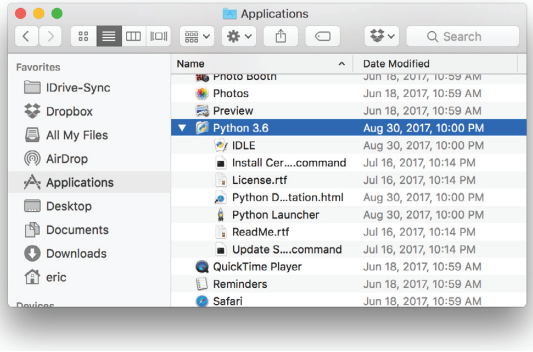
  `https://www.python.org/downloads`



Появится страница, где содержится ссылка для загрузки Python в macOS. Если не видите ее, откройте меню Downloads.

1. Выберите релиз Python 3.x (где x — номер новейшей версии языка). Не загружайте Python 2.7.
2. После скачивания инсталлятора откройте установочный пакет в папке загрузки и следуйте появляющимся на экране инструкциям.
3. По окончании установки перейдите в папку Applications, в которой вы найдете папку Python 3.x. Чтобы протестировать дистрибутив, выполните двойной щелчок на приложении IDLE в этой папке.

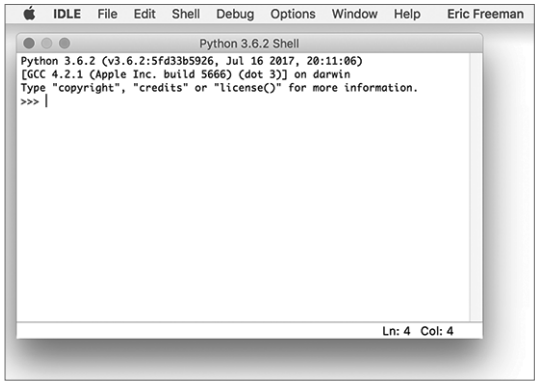
Учтите, что для установки Python нужно обладать правами администратора. Если вы регулярно устанавливаете программы, то беспокоиться не о чем. В противном случае попросите администратора помочь вам



Приложение IDLE находится в папке Python 3.x, которая сама находится в папке Applications. О том, что такое IDLE, мы поговорим в главе 1

Желательно закрепить значок IDLE на панели задач, поскольку мы будем много работать с данным приложением в книге

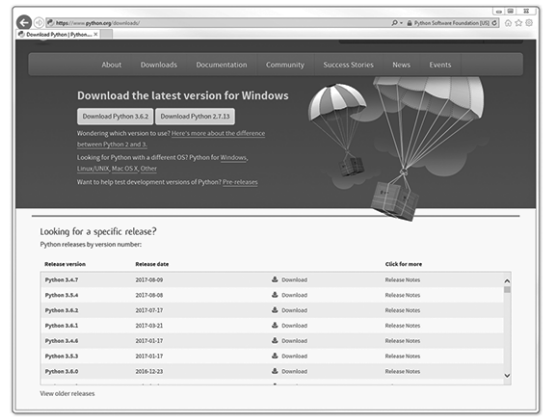
4. После запуска IDLE на экране появится окно, подобное показанному ниже. Если окна нет, проверьте, не было ли сообщений об ошибках в ходе установки.



Чтобы выйти из приложения, воспользуйтесь командой IDLE > Закрыть IDLE

- **В Windows** откройте браузер и введите в адресной строке следующее:

  `https://www.python.org/downloads`



1. Выберите релиз Python 3.x (где x — номер новейшей версии языка). Не загружайте Python 2.7.
2. Запустите инсталлятор либо сохраните его на диске, а затем дважды щелкните на нем для запуска.
3. Когда на экране появится окно инсталлятора, обязательно установите флажок Add Python 3.x to PATH внизу, а затем выберите вариант Install Now.

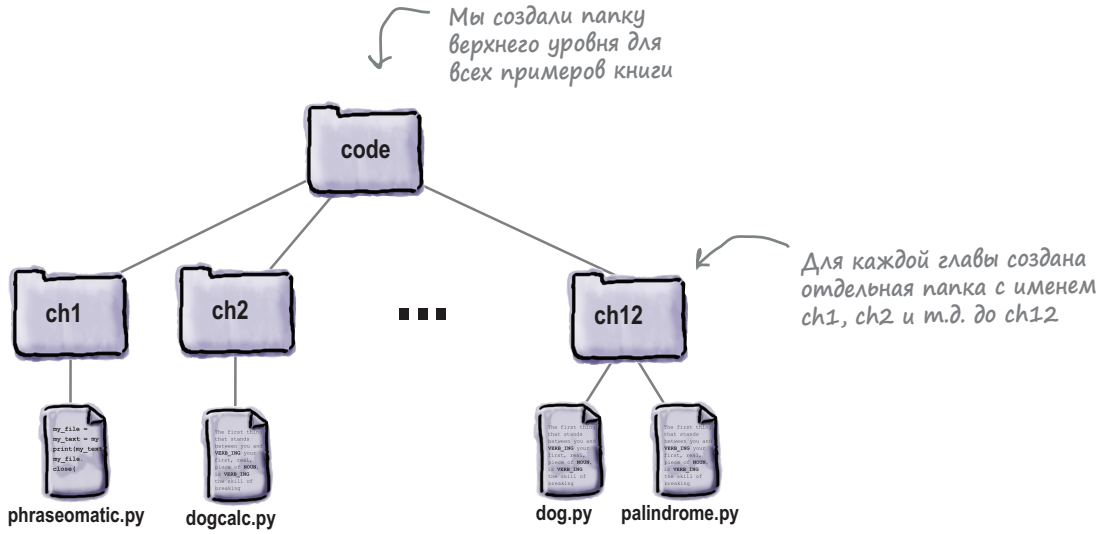
4. По окончании установки в меню Пуск > Все программы должно появиться подменю Python 3.x (где x — номер установленной версии языка). В нем содержатся команды запуска Python, справочной службы и IDLE — редактора кода, с которым мы будем работать в книге.
5. Чтобы протестировать дистрибутив, выберите пункт IDLE. На экране должно появиться окно, подобное показанному ниже. Если окна нет, проверьте, не было ли сообщений об ошибках в ходе установки.

Чтобы выйти из приложения, воспользуйтесь командой IDLE > Закрыть IDLE

Примечание для пользователей Linux: за вас мы совершенно спокойны, ведь вы всегда знаете, что делать. Просто скачайте соответствующий дистрибутив с сайта `python.org`.

Как работать с исходными кодами

Исходный код — это любой файл, создаваемый вами в процессе изучения книги. Мы рекомендуем сохранять файлы в отдельных папках, каждая из которых соответствует конкретной главе. В книге предполагается, что для каждой главы создана своя папка с исходными кодами.



Файлы всех рассматриваемых в книге примеров можно скачать по следующему адресу:

```
http://wickedlysmart.com/hflearntocode
```

На этой веб-странице вы найдете ссылку для скачивания архива примеров. В архиве содержатся авторские версии программ, которые вам предстоит написать, а также вспомогательные файлы данных и применяемые изображения.

Мы рассчитываем, что вы будете набирать все создаваемые программы самостоятельно, шаг за шагом, по мере чтения книги. Это помогает развивать мышечную память, формируя навыки программирования, и способствует лучшему усвоению материала. Но если вы вдруг столкнетесь с проблемами, причина которых не ясна, вы всегда сможете сравнить свой код с нашим, чтобы понять, где была допущена ошибка.

Кроме того, файлы примеров доступны также на сайте издательства "Диалектика":

`http://www.williamspublishing.com/Books/978-5-907144-98-9.html`

Несколько примеров в этом архиве (главы 6 и 7) подверглись незначительным модификациям, чтобы программы могли работать так, как описано в книге.

Благодарности*

Элизабет Робсон Джош Шарфман Дейвид Пауэрс Пол Барри

В первую очередь хочу сердечно поблагодарить моих технических рецензентов. **Элизабет Робсон** внимательно просмотрела черновик книги, и ни одна смысловая ошибка не укрылась от ее зоркого глаза. **Джош Шарфман** был нашим главным рецензентом, давшим массу полезных советов и рекомендаций по каждой главе. **Дейвид Пауэрс** в своем привычном стиле тщательно проверил все главы на предмет технических ошибок. Ветеран серии Head First **Пол Барри** взял на себя важную миссию по проверке листингов Python. Кроме того, неоценимый вклад в проект внесла команда рецензентов (все они перечислены на следующей странице), которые проверили каждую страницу книги.

Огромная благодарность моим редакторам **Джеффу Блейелу**, **Дон Сканафелт** и **Меган Бланшетт**. Меган серьезно помогла мне на начальном этапе, Дон контролировала ход работы, а Джефф довел проект до отправки в типографию.

Отдельное спасибо всему коллективу издательства O'Reilly, в особенности **Сьюзан Конант**, **Рейчел Румелиотис** и **Мелани Ярбро**. Благодарю **Джейми Бертон** из компании WickedlySmart за помощь в подготовке книги, включая сбор отзывов от читателей и управление форумом рецензентов. И как всегда, спасибо **Берту Бейтсу** и **Кэти Сиерра** за вдохновение, увлекательные дискуссии и помощь в решении возникающих проблем. Кроме того, хочется поблагодарить **Кори Доктороу** за поддержку и за то, что предоставил свою книгу в качестве материала для главы 7.

В завершение хочу упомянуть тех людей, которые, сами того не ведая, послужили для меня источником вдохновения: это профессор **Даниэль П. Фридман**, **Нейтан Берджи**, а также разработчики из **Raspberry Pi Foundation** и **Socratica**.

Джефф Блейел

Дон Сканафелт

Меган Бланшетт

Джейми Бертон

Берт Бейтс

Кэти Сиерра

* Столь большой список благодарностей объясняется тем, что мы решили проверить теорию, согласно которой каждый, кто упомянут на странице благодарностей, купит хотя бы один экземпляр книги, а возможно и больше, ведь у него есть родственники, члены семьи, друзья, которым захочется иметь свой экземпляр. Так что если хотите быть упомянутым в нашей следующей книге и у вас большая семья, пишите, не стесняйтесь.

Команда рецензентов

Рецензированием книги занималась целая команда потрясающих людей самой разной квалификации — от **новичков** до **экспертов** — и самых разных профессий, включая **архитектора**, **стоматолога**, **учителя младших классов**, **агента по недвижимости** и **преподавателя информатики в школе**. Причем все они были из разных уголков земного шара: от **Албании** до **Австралии**, от **Кении** до **Косово**, от **Нидерландов** и **Нигерии** до **Новой Зеландии**.

Рецензенты вычитали каждую из более чем 600 страниц книги, выполнили все упражнения и протестировали каждую строку кода, высказав свои замечания. Они сформировали настоящую команду, помогая друг другу в прояснении непонятных моментов и перепроверяя найденные ошибки.

Каждый из рецензентов внес важный вклад в книгу и помог сделать ее лучше.

Спасибо вам всем!


Кристал Гилмор Родригес


Ридван Бунджаку


Трой Уэлч


Хадсон Рид


Марк ван дер Линден


Митч Джонсон


Крис Талент


Андреа Тостон


Крис Григгз


Дейвид Опаранти


Джонни Ривера


Дейвид Киноти


Майкл Пек


Мауро Кейзер


Бенджамин И. Холл


Альфред Дж. Спеллер


Тайрон Андрич


Деннис Фицджеральд


Абдул Рахман Шаик