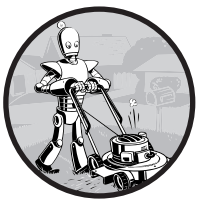
ВВЕДЕНИЕ



“За каких-то пару часов ты сделал то, на что у нас троих ушло бы целых три дня”. В начале 2000-х годов мой сосед по комнате в колледже работал в магазине электроники. Время от времени они получали электронные таблицы с ценниками своих конкурентов, включающие тысячи наименований. Распечатка одной таблицы представляла собой толстую стопку бумажных листов. Обработкой данных занимались три сотрудника магазина. Они сравнивали цены, указанные в таблице, с ценами в своем магазине и отмечали тот товар, который конкуренты продавали по более низкой цене. На эту работу у них уходило примерно два дня.

“А знаете что? Если вы дадите мне исходный файл таблицы, то я напишу программу, которая выполнит всю работу вместо вас”, — сказал мой товарищ, увидев, как они копошатся среди груды разбросанных на полу и сложенных в стопки бумажных листов.

Через пару часов у него была готова небольшая программа, которая читала данные о ценах конкурентов из файла, находила для каждого продукта аналог в базе данных магазина и отмечала весь товар конкурентов, цена которого была ниже. Здесь уместно сказать, что мой товарищ был всего лишь начинающим программистом, и бо́льшую часть этого времени он потратил на просмотр нужных разделов в книге по программированию. Собственно выполнение программы заняло всего лишь несколько секунд, что дало возможность моему товарищу и его коллегам по работе насладиться в тот день удлиненным обеденным перерывом.

Этот пример наглядно демонстрирует всю мощь программирования. Компьютер подобен армейскому ножу, который можно использовать в самых разных ситуациях. Многие люди часами работают за клавиатурой, вводя данные для выполнения повторяющихся задач, и даже не догадываются, что их компьютер, если снабдить его соответствующими инструкциями, способен выполнить ту же работу за считанные секунды.

Для кого предназначена эта книга

В наши дни трудно найти сферу человеческой деятельности, в которой вообще не используется программное обеспечение (ПО). Почти каждый из нас общается в социальных сетях, телефоны многих из нас — это по сути компьютеры, подключенные к Интернету, а бо́льшая часть офисного персонала для выполнения своих функциональных обязанностей нуждается в компьютерной технике. Как следствие, это привело к необычайно высокому спросу на специалистов, способных писать программный код.

Бесчисленные книги по программированию, интерактивные онлайновые руководства, практические семинары для разработчиков — все это направлено на превращение амбициозных новичков в специалистов программной индустрии, заработная плата которых выражается шестизначными числами.

Эта книга предназначена не для них. Она предназначена для всех остальных.

Прочтение одной только этой книги не сможет сделать из вас разработчика-профессионала, точно так же как пяти уроков игры на гитаре вряд ли будет достаточно для того, чтобы стать рок-звездой. Но если вы офисный работник, администратор, преподаватель или вообще один из тех, кто использует компьютер для работы или развлечения, то изучения основ программирования в том объеме, который предлагается в данной книге, вам хватит для автоматизации следующих простых задач:

- перемещение и переименование тысяч файлов и их сортировка по папкам;
- заполнение онлайновых форм без ввода данных вручную;
- загрузка файлов или копирование текста с веб-сайта при его обновлении;
- вывод компьютером заранее подготовленных уведомлений;
- обновление и форматирование электронных таблиц Excel;
- проверка электронной почты и отправка заранее подготовленных ответных писем.

Все эти задачи просты, но отнимают у человека массу времени. Кроме того, зачастую они настолько тривиальны или узкоспециальны, что подыскать какую-то готовую программу для их выполнения не удается. Вооружившись даже минимальными знаниями в области программирования, вы сможете заставить свой компьютер выполнять эти задачи вместо вас.

Исходные предположения

Эта книга — не справочник, а руководство для начинающих. Используемый в ней стиль программирования иногда идет вразрез с принципами наилучшей практики (например, в некоторых программах используются глобальные переменные), но это компромиссное решение, позволяющее сделать код более легким для изучения. Книга предназначена для тех, кому будет достаточно научиться писать простой одноразовый код, поэтому стилю оформления программ и приданию им элегантного вида не уделяется особого внимания. Такие понятия “продвинутого” программирования, как “объектно-ориентированный подход”, “списковые включения” и “генераторы”, не рассматриваются, дабы не усложнять излагаемый материал.

Возможно, опытные программисты легко укажут в книге те места, где код следовало бы изменить, чтобы сделать его более эффективным, но в этой книге нас в основном заботит создание работоспособных программ с минимальными усилиями.

Что такое программирование

В телевизионных шоу и фильмах часто показывают программистов, пристально вглядывающихся в потоки загадочных нулей и единиц, бегущих по экрану, но современное программирование далеко не столь таинственно. *Программирование* — всего лишь снабжение компьютера инструкциями, приказывающими ему что-либо сделать. Это может быть оперирование числами, изменение текста, поиск информации в файлах или обмен данными с другими компьютерами через Интернет.

Во всех программах в качестве строительных блоков используются элементарные инструкции. Вот как выглядят некоторые из наиболее распространенных инструкций такого рода, если перевести их на обычный человеческий язык.

> **“Сделай это; затем сделай то”.**
> **“Если данное условие удовлетворяется, выполни такое-то действие; в противном случае выполни такое-то действие”.**
> **“Выполни это действие такое-то количество раз”.**
> **“Продолжай выполнять эти действия до тех пор, пока удовлетворяется данное условие”.**

Эти строительные блоки можно комбинировать для реализации более сложных решений. В качестве примера ниже приведены инструкции (так называемый *исходный код*) простой программы на языке Python. Программное обеспечение последовательно выполняет каждую строку кода, начиная с первой (при этом некоторые строки выполняются лишь при соблюдении определенных условий, иначе выполняется другая строка), пока не будет достигнут конец программы.

```
❶ passwordFile = open('SecretPasswordFile.txt')
❷ secretPassword = passwordFile.read()
❸ print('Введите пароль.')
  typedPassword = input()
❹ if typedPassword == secretPassword:
❺    print('Доступ разрешен.')
❻    if typedPassword == '12345':
❼        print('Рекомендуем установить более сложный пароль!')
  else:
❽    print('В доступе отказано.')
```

Даже если вы ничего не смыслите в программировании, вы все равно сможете сделать разумные предположения относительно того, что делает этот код, просто читая его. Сначала открывается файл *SecretPasswordFile.txt* ❶, из которого считывается секретный пароль ❷. Затем пользователю предлагается ввести свой пароль (с помощью клавиатуры) ❸. Далее оба пароля сравниваются между собой ❹, и в случае их совпадения программа выводит на экран текст `Доступ разрешен` ❺. Далее программа проверяет, является ли введенный пароль числом `12345` ❻, и, если это так, подсказывает, что такой вариант выбора пароля не является оптимальным ❼. В случае несовпадения паролей программа выводит на экран сообщение `В доступе отказано` ❽.

Что означает название Python

Название *Python* относится как к языку программирования Python (с его собственным синтаксисом, определяющим правила написания корректного кода), так и к интерпретатору Python — программе, предназначенной для чтения исходного кода (написанного на языке Python) и выполнения его инструкций. Различные версии интерпретатора Python, ориентированные на платформы Linux, OS X и Windows, доступны для бесплатной загрузки по адресу `http://python.org`.

Своим названием язык Python обязан вовсе не одноименному виду пресмыкающихся (питон), а комедийной группе из Великобритании "Monty Python"[1], работавшей в жанре сюрреалистического юмора. Программистов на Python шутливо называют *питонистами* (Pythonistas), а многочисленные руководства и документация по языку Python пестрят ссылками как на группу "Monty Python", так и на рептилию.

Программисту вовсе не обязательно в совершенстве знать математику

По моим наблюдениям, наибольшую озабоченность у тех, кто собирается учиться программированию, вызывает то, что, по их мнению, для этого надо хорошо знать математику. На самом деле большинству программистов не нужно знать ничего кроме элементарной арифметики. В этом смысле хорошему программисту понадобится не намного больший объем математических знаний по сравнению с тем, который требуется для решения головоломок судоку.

[1] Читается как "Монти Пайтон". Грамматически правильное звучание названия языка Python — также [пай-тон], однако среди программистов принято произносить его как [пи-тон]. — *Примеч. ред.*

Суть головоломки судоку заключается в заполнении цифрами от 1 до 9 каждого из внутренних квадратов размером 3×3, расположенных на игровом поле размером 9×9, таким образом, чтобы ни одна строка и ни один столбец большого квадрата не содержали повторяющихся цифр. Для нахождения решения необходимо использовать дедуктивный логический метод, исходя из заданной начальной конфигурации цифр. Например, поскольку в головоломке, показанной на рис. 1, цифра 5 находится в левом верхнем углу игрового поля, она не может появиться ни в верхней строке, ни в крайнем слева столбце, ни в левом верхнем квадрате размером 3×3. Последовательное применение подобной логики к строкам, столбцам и внутренним квадратам будет предоставлять вам подсказки, позволяющие заполнять пустые клетки головоломки.

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Рис. 1. Головоломка судоку (слева) и ее решение (справа). Несмотря на то что в головоломке используются числа, никаких особых математических знаний для нахождения решения не требуется (изображения предоставлены компанией Wikimedia Commons)

Из того факта, что в головоломке судоку используются числа, вовсе не следует, что для нахождения решения необходимо быть хорошим математиком. То же самое справедливо и в отношении программирования. Как и в судоку, написание программ предусматривает разбиение задачи на ряд отдельных этапов. Аналогичным образом при *отладке программ* (процесс обнаружения и предотвращения возможности возникновения ошибок) вы кропотливо анализируете результаты интересующих вас действий, выполненных программой, пытаясь выявить причину ошибки. И, как это характерно для любого другого вида деятельности, чем больше вы программируете, тем лучше у вас это будет получаться.

Программирование — творческий вид деятельности

Программирование — это вид творчества, несколько напоминающий возведение з*а*мков из элементов LEGO. Сначала вы формулируете для себя основные идеи относительно того, что собой должна представлять будущая программа и какие строительные элементы имеются в вашем распоряжении. После этого вы приступаете к построению программы. Завершив построение программы, вы наводите окончательный порядок в своем коде, аналогично тому как по окончании строительства з*а*мка принялись бы за уборку его территории.

Различие между программированием и другими творческими видами деятельности заключается в том, что все необходимые исходные материалы находятся в вашем компьютере, и вам не нужно дополнительно закупать какие-либо холсты, краску, пленку, нитки, блоки LEGO или электронные компоненты. Написав программу, вы можете легко поделиться ею через Интернет с целым миром. И даже если в процессе программирования вы будете допускать неизбежные ошибки, это занятие доставит вам массу удовольствия.

Структура книги

В части I книги рассмотрены основы программирования на языке Python, тогда как часть II посвящена различным задачам, решение которых можно автоматизировать. Каждая глава части II включает программные проекты, с которыми вам предстоит работать. Ниже приведено краткое описание глав.

Часть I. Основы программирования на языке Python

- **Глава 1. Основные понятия языка Python.** В этой главе рассматриваются выражения — базовый тип инструкций Python, а также описывается использование интерактивной программной оболочки Python для экспериментирования с кодом.
- **Глава 2. Поток управления.** В этой главе речь идет о том, как заставить программу принимать решения, касающиеся последовательности выполнения инструкций, чтобы код мог самостоятельно реагировать на возникновение различных условий.
- **Глава 3. Функции.** В этой главе показано, как использовать собственные функции для разбиения кода на отдельные части, с которыми проще работать.
- **Глава 4. Списки.** Вводится понятие *списка*, одного из встроенных типов данных Python, и рассказывается о том, как использовать списки для организации данных.

- **Глава 5. Словари и структурирование данных.** Вводится понятие другого встроенного типа данных Python, *словаря*, и демонстрируются более совершенные способы организации данных.
- **Глава 6. Манипулирование строками.** Описываются методы работы с текстовыми данными (которые в языке Python принято называть *строками*).

Часть II. Автоматизация задач

- **Глава 7. Поиск по шаблону с помощью регулярных выражений.** Обсуждаются приемы обработки строк и способы поиска образцов текста, соответствующих заданному шаблону, с помощью регулярных выражений.
- **Глава 8. Чтение и запись файлов.** В этой главе речь идет о том, как организовать в программе чтение данных из текстовых файлов и сохранить информацию на жестком диске.
- **Глава 9. Управление файлами.** Рассматриваются автоматизированные способы копирования, перемещения, переименования и удаления файлов, позволяющие выполнять данные операции быстрее, чем это можно сделать вручную.
- **Глава 10. Отладка.** Рассматриваются средства Python, предназначенные для обнаружения и устранения логических ошибок.
- **Глава 11. Автоматический сбор данных в Интернете.** Показано, как писать программы, выполняющие автоматическую загрузку веб-страниц и их синтаксический анализ с целью извлечения полезной информации. Для этого процесса часто используют термин *веб-скрапинг*. Соответствующие программы называют *интернет-ботами*.
- **Глава 12. Работа с электронными таблицами Excel.** Рассматриваются методы манипулирования электронными таблицами Excel, ориентированные на обработку данных без их чтения человеком. Такая возможность чрезвычайно полезна в тех случаях, когда количество обрабатываемых документов исчисляется сотнями и даже тысячами.
- **Глава 13. Работа с документами в форматах PDF и Word.** Рассматриваются программные методы чтения документов, подготовленных в форматах PDF и Word.
- **Глава 14. Работа с CSV-файлами и данными в формате JSON.** Рассматриваются методы манипулирования CSV-файлами и JSON-данными.
- **Глава 15. Обработка значений даты и времени, планировщик заданий и запуск программ.** Рассказывается о способах обработки информации, связанной с датой и временем, и выполнении задач по расписанию. Также показано, как запускать программы, написанные на языках, отличных от Python, из Python-программ.

- **Глава 16. Отправка сообщений электронной почты и текстовых сообщений.** Обсуждается написание программ, осуществляющих автоматическую рассылку сообщений электронной почты и текстовых сообщений.
- **Глава 17. Работа с изображениями.** Рассматриваются способы программного манипулирования изображениями, сохраненными в различных форматах, таких как JPEG или PNG.
- **Глава 18. Управление клавиатурой и мышью с помощью средств автоматизации графического интерфейса пользователя.** Речь идет о возможностях управления клавиатурой и мышью путем программной эмуляции щелчков мышью и нажатий клавиш.

Исходный код примеров доступен в виде архивного файла *Automate_the_Boring_Stuff_onlinematerials.zip* на сайте `https://www.nostarch.com/automatestuff/`. Там же содержатся другие полезные ресурсы с примерами, предлагаемые автором в дополнение к данной книге, с указанием глав, к которым они относятся, а также публикуется информация об ошибках и опечатках, обнаруженных в книге.

Загрузка и установка Python

Версии Python для Windows, OS X и Ubuntu доступны для бесплатной загрузки по адресу `http://python.org/downloads/`. Если вы загрузите текущую версию для своей системы, то все примеры программ, приведенные в книге, должны будут работать.

Предупреждение

Обязательно загрузите версию Python 3 (например, 3.4.5). Все примеры программ в книге написаны с использованием Python 3, и если вы попытаетесь запускать их в версии Python 2, то они могут выполняться неправильно или вообще не выполняться.

На указанной странице загрузки для каждой операционной системы предлагаются отдельные установщики, рассчитанные на 64- и 32-разрядные версии, поэтому предварительно определитесь, какой именно установщик вам нужен. Если вы приобрели компьютер в 2007 году или позже, то, скорее всего, на нем установлена 64-разрядная операционная система. В противном случае можно полагать, что вы пользуетесь 32-разрядной версией, но лучше убедиться в этом непосредственно, выполнив следующие действия.

- Если вы используете Windows, выберите пункты меню Пуск⇨Панель управления⇨Система и проверьте, какая система указана в качестве значения параметра Тип системы — 64- или 32-разрядная.
- Если вы используете OS X, перейдите в меню Apple, выберите пункты меню About This Mac⇨More Info⇨System Report⇨Hardware, а затем проверьте значение поля Processor Name. Если в этом поле отображается текст "Core Solo" или "Intel Core Duo", то у вас 32-разрядный компьютер. Если же в поле отображается какой-либо другой текст (включая "Intel Core 2 Duo"), то у вас 64-разрядный компьютер.
- Если вы используете Ubuntu Linux, откройте терминал и выполните команду **`uname -m`**. Ответ `i686` означает 32-разрядный компьютер, ответ `x86_64` — 64-разрядный.

Для Windows загрузите установщик Python (файл с расширением `.msi`) и дважды щелкните на нем. Чтобы установить Python, следуйте инструкциям, которые установщик отображает на экране.

1. Выберите вариант Install for All Users, а затем щелкните на кнопке Next.
2. Выполните установку в папку *C:\Python34*, щелкнув на кнопке Next.
3. Вновь щелкните на кнопке Next, чтобы пропустить раздел Customize Python.

Для MAC OS X загрузите файл с расширением *.dmg*, соответствующий вашей версии OS X, и дважды щелкните на нем. Чтобы установить Python, следуйте инструкциям, которые установщик отображает на экране.

1. Когда в новом окне откроется пакет DMG, дважды щелкните на файле *Python.mpkg*. Возможно, вам придется ввести пароль администратора.
2. Щелкните на кнопке Continue для прохождения раздела Welcome, а затем на кнопке Agree для принятия условий лицензии.
3. Выделите имя жесткого диска, на который выполняется установка, и щелкните на кнопке Install.

В случае использования Ubuntu можете установить Python из окна терминала, выполнив следующие действия.

1. Откройте окно Terminal.
2. Введите команду **`sudo apt-get install python3`**.
3. Введите команду **`sudo apt-get install idle3`**.
4. Введите команду **`sudo apt-get install python3-pip`**.

Запуск IDLE

Если *интерпретатор Python* — это программное обеспечение, предназначенное для выполнения программ на Python, то *интерактивная среда разработки (IDLE)* — это программное обеспечение, с помощью которого можно вводить текст программ примерно так же, как это делается с помощью текстового процессора. Приступим к запуску IDLE.

- Если ваш компьютер работает под управлением операционной системы Windows 7 или более новой версии Windows, щелкните на кнопке Пуск в левом нижнем углу экрана, введите **`IDLE`** в строке поиска и выберите в раскрывшемся меню пункт IDLE (Python GUI).
- Если ваш компьютер работает под управлением операционной системы Windows XP, щелкните на кнопке Пуск и выберите последовательно пункты меню Programs⇨Python 3.4⇨IDLE (Python GUI).
- Если ваш компьютер работает под управлением операционной системы MAC OS X, откройте окно Finder, выберите последовательно Applications и Python 3.4, а затем щелкните на значке `IDLE`.
- Если ваш компьютер работает под управлением операционной системы Ubuntu, выберите Applications⇨Accessories⇨Terminal, а затем введите команду **`idle3`**. (Вы также можете щелкнуть на кнопке Applications в верхней части экрана, выбрать раздел Programming и щелкнуть на пункте IDLE 3.)

Интерактивная оболочка

Независимо от того, в какой операционной системе вы работаете, окно IDLE при его первом открытии будет в основном пустым, не считая текста, который будет выглядеть примерно так.

```
Python 3.4.0 (v3.4.0:04f714765c13, Mar 16 2014, 19:25:23)
[MSC v.1600 64 bit (AMD64)] on win32Type "copyright", "credits"
or "license()" for more information.
>>>
```

Это окно называется *интерактивной оболочкой*. Оболочка — это программа, которая позволяет вводить инструкции в компьютер во многом аналогично тому, как это делается в окне терминала или командной строки на компьютерах OS X и Windows соответственно. Команды, которые вы вводите в интерактивной оболочке, выполняются интерпретатором Python. Компьютер читает введенные инструкции (команды) и немедленно выполняет их.

Чтобы увидеть, как это работает на практике, введите в интерактивной оболочке сразу же за приглашением к вводу (>>>) следующую инструкцию:

```
>>> print('Hello world!')
```

После того как вы введете эту инструкцию и нажмете клавишу <Enter>, интерактивная оболочка должна отреагировать на это выводом следующей строки:

```
>>> print('Hello world!')
Hello world!
```

Как получить справку

Самостоятельно находить решения проблем, возникающих в процессе программирования, гораздо легче, чем вы думаете. Чтобы убедить вас в этом, давайте намеренно вызовем ошибку при попытке выполнить инструкцию. Введите в интерактивной оболочке инструкцию `'42' + 3`. Вам необязательно знать сейчас, что она означает, но результат должен выглядеть так.

```
   >>> '42' + 3
❶  Traceback (most recent call last):
     File "<pyshell#0>", line 1, in <module>
       '42' + 3
❷  TypeError: Can't convert 'int' object to str implicitly
   >>>
```

Появление здесь сообщения об ошибке ❷ обусловлено тем, что смысл введенной вами инструкции остался непонятным для Python. В той части сообщения, которая касается текущего стека вызовов (`Traceback`) ❶, отображаются конкретная инструкция и номер строки, в которой Python столкнулся с проблемой. Если сообщение об ошибке ни о чем вам не говорит, выполните поиск в Интернете по точному тексту сообщения. Введите текст **`"TypeError: Can't convert 'int' object to str implicitly"`** (включая кавычки) в своем поисковике, и вы увидите тысячи ссылок, по которым можно узнать о том, что означает данное сообщение и что породило ошибку (рис. 2).

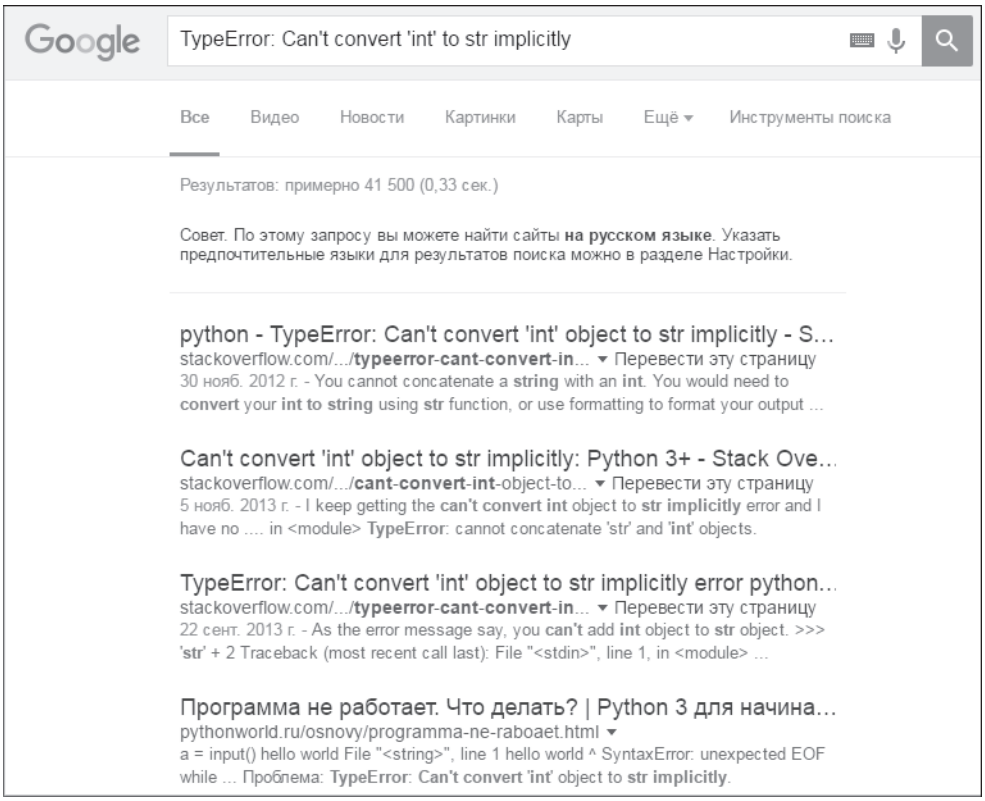


Рис. 2. Получение дополнительной информации о природе ошибки путем поиска в Google с использованием текста сообщения об ошибке в качестве ключа

Часто вы будете сталкиваться с тем, что у кого-то уже возникал тот же вопрос, что и у вас, и на этот вопрос уже был дан ответ. В программировании никому не дано знать абсолютно все, поэтому свыкнитесь с мыслью о том, что поиск ответов на различные вопросы технического характера — неотъемлемая часть ежедневной деятельности программиста-разработчика.

Правильно формулируйте вопросы, ответы на которые ищете

Если онлайновый поиск не позволил получить ответ на интересующий вас вопрос, то поспрашивайте людей на таких форумах, как Stack Overflow (`http://stackoverflow.com`), или посетите учебный раздел сайта Reddit (`http://reddit.com/r/learnprogramming`). Однако имейте в виду, что при обращении за помощью очень важно правильно формулировать свои вопросы. Обязательно посетите разделы Frequently Asked Questions (часто

задаваемые вопросы) этих сайтов, где вы сможете ознакомиться с формулировками вопросов, которые послужат вам образцом для подражания.

Задавая вопросы, касающиеся программирования, старайтесь придерживаться следующих рекомендаций.

- Объясните, что именно вы *пытаетесь* сделать, а не только то, что вы *делали*. Это позволит тому, кто хочет вам помочь, определить, находитесь вы на верном или на неверном пути.
- Укажите, когда именно возникает ошибка: сразу после запуска программы или после того, как вы выполняете определенные действия.
- Скопируйте и вставьте *полный* текст сообщения об ошибке и ваш код в буферное хранилище по адресу `http://pastebin.com/` или `http://gist.github.com/`.

  Указанные веб-сайты упрощают обмен большими объемами кода через Интернет без риска потерять форматирование текста. Затем можете переслать URL-адрес размещенного в буферном хранилище кода нужному человеку по электронной почте или опубликовать его на форуме. Чтобы увидеть, как это работает, просмотрите код, который я разместил в хранилищах по следующим адресам: `http://pastebin.com/SzP2DbFx/` и `https://gist.github.com/asweigart/6912168/`.

- Объясните, какие меры вы предпринимали для разрешения возникшей проблемы. Тем самым вы покажете, что уже приложили усилия со своей стороны, стараясь самостоятельно выяснить причину неполадок.
- Укажите версию Python, которую используете. (Между интерпретаторами, входящими в состав версий Python 2 и 3, имеются важные различия.) Также укажите используемую вами операционную систему и ее версию.
- Если ошибка появилась после того, как вы внесли изменения в код, детально опишите, какие именно изменения были вами внесены.
- Расскажите, воспроизводится ли ошибка всякий раз, когда вы выполняете программу, или она возникает лишь после того, как вы совершаете определенные действия. В последнем случае опишите, в чем именно заключаются эти действия.

Кроме того, строго соблюдайте правила сетевого этикета. Например, размещая на форуме свои вопросы, не набирайте весь текст прописными буквами, пытаясь сделать его более заметным, и не выдвигайте необоснованных требований к людям, которые пытаются вам помочь.

Резюме

Для большинства людей компьютер — это всего лишь полезное устройство, а не инструмент. Вместе с тем, научившись программировать, вы получите доступ к одному из наиболее мощных инструментов в современном мире, работа с которым доставит вам, кроме всего прочего, огромное удовольствие. Программирование вовсе не сродни нейрохирургии — оно предоставляет новичкам великолепную возможность экспериментировать и при этом не бояться, что допущенные ошибки могут быть чреваты катастрофическими последствиями.

Мне нравится помогать людям открывать для себя Python. Я пишу руководства по программированию на своем блоге по адресу `http://inventwithpython.com/blog/`, и вы можете связаться со мной и задать вопросы, отправив сообщение электронной почты по адресу `al@inventwithpython.com`.

Эта книга лишь помогает преодолеть начальный барьер в изучении программировании, поэтому вы не всегда найдете в ней ответы на все свои вопросы. Не забывайте о том, что умение правильно формулировать вопросы и знание того, как находить ответы на них, окажут вам неоценимую помощь в вашем путешествии в мир программирования.

Итак, приступим!