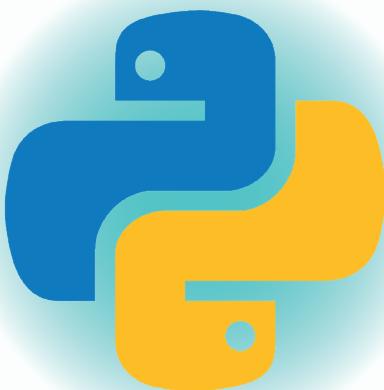


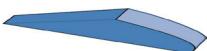
Посвящается Янне, Джулии, Катрин и Даниэль



Содержание

| | |
|-----------------------|-----------|
| Введение | 13 |
|-----------------------|-----------|

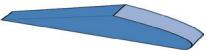
1



| | |
|--|-----------|
| Твои первые шаги в программировании | 19 |
|--|-----------|

| | |
|-------------------------------------|----|
| Начало работы с Python..... | 19 |
| Числа и текст | 23 |
| Рабочая среда IDLE | 27 |
| Работа с PY-файлами | 30 |
| Эксперименты с исходным кодом..... | 34 |
| Выход из Python | 37 |
| Подведение итогов | 37 |
| Несколько контрольных вопросов..... | 38 |
| ...а задач пока нет! | 38 |

2



| | |
|-----------------------------------|-----------|
| Условные конструкции | 39 |
|-----------------------------------|-----------|

| | |
|--------------------------|----|
| Конструкция if | 39 |
| Конструкция if else..... | 44 |
| Простые вычисления | 48 |

| | |
|---------------------------------------|----|
| Вкратце о числах | 51 |
| Работа с командами try и except | 55 |
| Подведение итогов | 57 |
| Несколько вопросов..... | 58 |
| ...и задач | 58 |

Сравнение и повторение 59

| | |
|------------------------------------|----|
| Оценки..... | 59 |
| Небольшая игра на угадывание | 64 |
| Компьютер считает попытки | 69 |
| Шлифуем игру | 72 |
| Подведение итогов | 74 |
| Несколько вопросов..... | 75 |
| ...и задач | 75 |

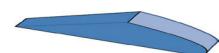
Азартная игра 76

| | |
|---------------------------|----|
| Игра наудачу | 76 |
| Конструкция for | 79 |
| На пути к миллиону | 82 |
| Выиграть в лотерею? | 87 |
| Управление строками..... | 90 |
| Подведение итогов | 92 |
| Несколько вопросов | 93 |
| ...и задач | 93 |

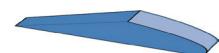
Функции 94

| | |
|--|-----|
| Python учится | 94 |
| Локальные или глобальные переменные? | 98 |
| Параметры..... | 100 |
| Обмен значений..... | 104 |
| Сортировка чисел..... | 108 |
| Подведение итогов | 112 |
| Несколько вопросов..... | 112 |
| ...и задач | 112 |

3



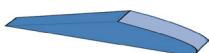
4



5



6



| | |
|--------------------------------|-----|
| Классы и модули | 113 |
| Потомки | 113 |
| self и __init__ | 117 |
| Наследование | 120 |
| Модули программы | 125 |
| Приватный или публичный? | 130 |
| Подведение итогов | 134 |
| Несколько вопросов | 134 |
| ...а задач нет | 134 |

7



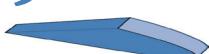
| | |
|-------------------------------------|-----|
| Введение в tkinter | 135 |
| Создаем окно | 135 |
| Что же происходит? | 139 |
| Разметка интерфейса программы | 142 |
| Диалоговые окна и заголовки | 145 |
| А теперь с классами | 146 |
| Подведение итогов | 148 |
| Несколько вопросов | 148 |
| ...и одна задача | 148 |

8



| | |
|-------------------------------------|-----|
| Библиотека компонентов | 149 |
| Череда кнопок | 149 |
| Кнопки и ответы | 152 |
| Списки выбора | 155 |
| О переключателях | 157 |
| ...и флагжках | 160 |
| Декорирование приложения | 164 |
| Подведение итогов | 167 |
| Несколько вопросов | 168 |
| ...и задач | 168 |

9



| | |
|--|-----|
| Домашний психолог | 169 |
| Пошаговая разработка программы-психолога | 169 |
| Приступим к диагностике? | 173 |

| | |
|--------------------------|-----|
| Работа с файлами..... | 177 |
| Все вместе..... | 181 |
| Журнал диагностики | 183 |
| Подведение итогов | 185 |
| Несколько вопросов..... | 185 |
| ...и задач | 185 |

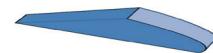
10



| | |
|--|-----|
| Меню и диалоговые окна | 186 |
| Меню для программы-психолога | 186 |
| Два диалоговых окна | 190 |
| Полный исходный код | 192 |
| Контекстные меню и всплывающие окна..... | 194 |
| Используем сочетания клавиш | 198 |
| Подведение итогов | 200 |
| Несколько вопросов..... | 200 |
| ...и нет задач..... | 200 |

| | |
|-------------------------------|-----|
| Графика в Python | 201 |
| Точки и координаты | 201 |
| Первое изображение..... | 204 |
| Добавим цвета..... | 208 |
| Углы и круги | 209 |
| Эксперименты с текстом | 212 |
| Звездное небо..... | 213 |
| Сам себе художник..... | 215 |
| Черепашья графика | 217 |
| Подведение итогов | 221 |
| Несколько вопросов..... | 221 |
| ...и задач | 222 |

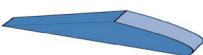
11



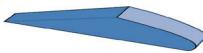
| | |
|--------------------------------------|-----|
| Создание анимации | 223 |
| Начнем с круга | 223 |
| Загружаем на холст изображение | 227 |
| Коллекция изображений | 230 |

12

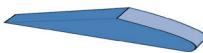


13

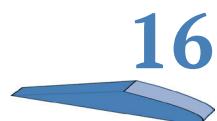
| | |
|--------------------------------|-----|
| Класс Player | 232 |
| Все работает?..... | 236 |
| Повороты | 238 |
| Исчезновение и появление | 240 |
| Подведение итогов | 243 |
| Вопрос..... | 243 |
| ...и несколько задач | 244 |

14

| | |
|-----------------------------------|-----|
| Как раздавить жука | 268 |
| Разбираемся с мышью | 268 |
| Никуда без математики..... | 270 |
| Собираем все вместе..... | 275 |
| Свободное ползание | 277 |
| Тапком по виртуальному жуку | 281 |
| Управление классами | 283 |
| Подведение итогов | 286 |
| Несколько вопросов..... | 287 |
| ...и задача | 287 |

15

| | |
|--|-----|
| Уклониться или проиграть | 288 |
| Новый персонаж | 288 |
| Стоять, приседать и подпрыгивать | 292 |
| Класс Thing | 296 |



| | |
|---------------------------------|-----|
| Учим персонажа уклоняться | 299 |
| Основная программа | 303 |
| Подведение итогов | 304 |
| Нет вопросов... | 304 |
| ...и одна задача..... | 305 |

| | |
|--------------------------------------|-----|
| Пора развлекаться..... | 306 |
| Игровой счет..... | 306 |
| Класс Game | 310 |
| Ошибка при сборке программы | 313 |
| Очки при попадании | 316 |
| Подведение итогов и заключение | 320 |

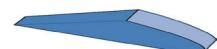
| | |
|---|-----|
| Установка Python | 321 |
| Установка библиотеки Pygame, вариант 1..... | 325 |
| Установка библиотеки Pygame, вариант 2..... | 329 |
| Файлы примеров..... | 331 |

| | |
|---|-----|
| Решение распространенных проблем | 332 |
|---|-----|

| | |
|--|-----|
| Ответы на вопросы и задачи..... | 333 |
|--|-----|

| | |
|----------------------------------|-----|
| Предметный указатель..... | 340 |
|----------------------------------|-----|

A

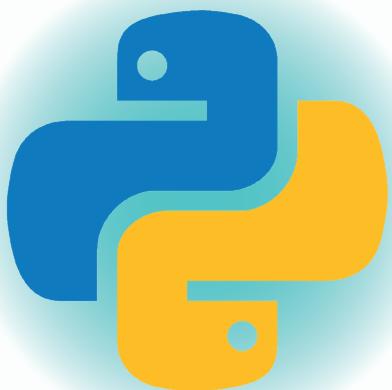


B



C



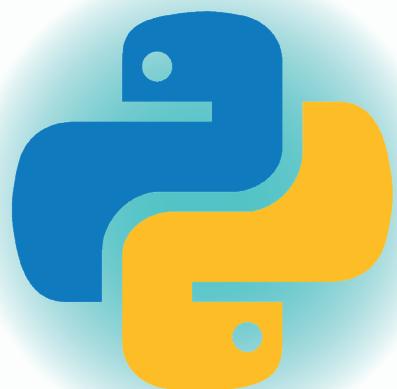


В своё время, для того чтобы изучить основы Python, я потратил очень много времени на поиск необходимой информации в интернете. Книг тогда было достаточно, но они были перегружены огромным количеством информации, которая только путала новичка.

Эта книга представляет собой руководство средней сложности для изучения языка программирования Python.

Она очень хорошо подойдёт для детей, изучающих Python с наставником, и для самостоятельного освоения данного языка, а также для составления методических пособий в организациях дополнительного образования.

Дмитрий Филиппов,
преподаватель курса по Python
в международной школе программирования
для детей «CODDY»



Введение

Python – это язык программирования, который был разработан в начале 1990-х годов. Название этого языка программирования произошло от имени английской комедийной труппы под названием «Монти Пайтон», которая в 70-е годы была довольно популярна и принимала участие в кинофильмах, таких как «Жизнь Брайана».

Хотя Python во многом похож на другие языки программирования, он стал очень популярен, так как именно на этом языке довольно легко научиться программировать. Эта книга посвящена третьей версии языка программирования Python, на сегодня последней (и универсальной).

Что такое программирование?

Программирование – это когда ты записываешь то, что должен делать компьютер. Самое замечательное, что ты сам можешь решить, что конкретно ему нужно делать. Если ты запустишь свою программу, компьютер выполнит то, что ты только что придумал и записал.

Конечно, твой компьютер не сможет убрать твою комнату и не принесет тебе чашечку какао в постель. Но как только ты освоишь программирование, то сразу же сможешь позволить компьютеру, так сказать, танцевать под свою дудку. Однако во время программирования часто бывает так, что компьютер не хочет выполнять то, что хочешь от него ты. Обычно такое поведение вызывает ошибку в программе. Проблема также может быть где-то еще в компьютере или

операционной системе. Проблема ошибок заключается в том, что им очень уж нравится прятаться, так что их поиск заставляет некоторых программистов попросту отчаяваться. Я очень надеюсь, что тебе все-таки захочется научиться программировать. Тогда все, что тебе нужно, – это подходящая среда разработки, и ты уже готов к работе!

Что такое среда разработки?

Чтобы создать программу, сначала нужно что-то куда-то ввести. Это как письмо или рассказ, который ты пишешь. Текстовая программа для этого может быть очень простой, поскольку не нужно выбирать разные шрифты или вставлять картинки. Такие программы принято называть *текстовыми редакторами*.

Если программа набрана в виде текста в редакторе, то компьютер не сможет просто прочитать этот документ и выполнить программу. Текстовая версия (*код*) программы должна быть переведена так, чтобы компьютер понял, что от него требуется в конечном итоге. Но поскольку он говорит на совершенно ином языке, чем ты или твои друзья, то придется использовать специальный переводчик.

Ты программируешь на понятном тебе языке, а специальная программа переводит то, что ты написал, в команды, понятные для компьютера. Такую программу называют *компилятором* или *интерпретатором*.

Для языка Python существуют интерпретаторы для нескольких операционных систем. Твой компьютер может работать под управлением операционной системы Windows, Linux или macOS. Независимо от этого одна и та же программа, написанная на языке программирования Python, будет работать (возможно, с небольшими изменениями) на любом компьютере.

Наконец, программы должны быть проверены, пересмотрены, усовершенствованы, доработаны и собраны. А еще есть вспомогательные инструменты. Все это становится целой системой – *средой разработки*.

Почему Python?

К сожалению, ты не сможешь программировать так, как захочется. Язык программирования должен быть структурирован таким образом, чтобы как можно больше людей могли понять его по всему миру.

В мире большинство людей может, по крайней мере, прочитать несколько слов по-английски, и поэтому почти каждый язык программирования состоит из английских слов. Есть также проекты языков программирования на других языках, но в большинстве случаев команды звучат настолько непривычно, что ты, скорее всего, все же вернешься к английскому. На самом деле не имеет значения, какой конкретно язык программирования ты используешь. Конечно, лучше всего тот, которому легко научиться. Как ты уже понял, в этой книге ты научишься создавать программы с помощью языка программирования Python, который сейчас очень популярен. (Если ты захочешь познакомиться с другими языками программирования C++ и Java.)

Стать хорошим программистом может быть очень трудно. Нередко человек теряет желание, потому что у него просто не работают написанные им программы. Программа делает нечто совершенно другое, а он сам не может найти ошибку и спрашивает себя: зачем я учусь программировать, если в мире уже и так достаточно программ?

Хорошие программисты всегда легко находят работу, а потребность в качественном программном обеспечении будет только расти. Программисты на языке Python, безусловно, востребованы в трудовой сфере. И, честно сказать, хорошие программисты действительно получают хорошую заработную плату. Это не просто твоя первая попытка, ведь, может быть, тебе действительно станет интересно узнать, как программировать на Python.

Среда разработки

Тебе не нужно переживать о том, где взять среду разработки на языке Python, так как это достаточно просто. Ты получишь ее бесплатно в составе пакета Python (называемого IDLE (Integrated Development and Learning Environment) – интегрированная среда разработки и обучения на языке Python). Мы будем пользоваться ею в этой книге постоянно.

Скачать полный пакет ты сможешь на сайте www.python.org. Необязательно устанавливать самую последнюю версию. В этой книге мы будем работать с версией Python 3.1.

О чем эта книга?

В этой книге ты:

- получишь базовые знания о языке Python;

- узнаешь об объектно-ориентированном программировании;
- научишься работать с компонентами библиотеки `tkinter` (это строительные блоки, с помощью которых ты сэкономишь время в процессе программирования);
- узнаешь о возможностях Python при работе с графикой;
- обучишься основам работы с игровыми модулями `pygame`;
- узнаешь, как разработать собственную игру.

В приложениях содержится дополнительная информация и справочные сведения, которые помогут разобраться в настройках и способах устранения ошибок.

Как читать эту книгу

Эта книга состоит из большого количества текста с иллюстрациями. И я, конечно, старался написать ее так, чтобы она была понятной. Чтобы сделать информацию в ней еще более понятной для тебя, я добавил некоторые символы. Вот что они обозначают:

Практические шаги

- Если ты видишь такой символ, знай: тебе нужно выполнить задание в данном абзаце текста. Это приблизит тебя к новой цели в программировании.

В целом ты будешь быстрее усваивать материал, когда будешь вводить код программы или станешь изменять его самостоятельно. Но у тебя не всегда будет желание этим заниматься. Поэтому все примеры данной книги доступны для бесплатного скачивания по адресу dmkpress.com.

Скачав файлы примеров, ты сможешь найти проект по имени (например, `→ project1.py`). Поэтому если ты не захочешь самостоятельно создавать проект, то всегда сможешь открыть соответствующий файл.

Задачи

В конце каждой главы ты найдешь несколько вопросов и задач. Эти упражнения не всегда просты, но они помогают освоить программирование. Решения задач можно найти в папке с файлами примеров. Ты сможешь просмотреть их в текстовом редакторе. Или распечатать на принтере и сложить стопкой рядом с компьютером.

Решение проблем

Случается, что ты не знаешь, как что-то сделать, или, возможно, просто что-то забыл. И тогда вся работа кажется невыполнимой и сложной. И ты спрашиваешь себя: что же мне делать? С помощью этого значка ты сможешь довольно легко найти решение или восстановить что-то в памяти. При необходимости также можешь заглянуть в последнюю часть этой книги, там расположено приложение Б, в котором приведены советы и другая справочная информация.



Важные примечания

Время от времени ты будешь встречать в книге такой восклицательный знак. Он указывает на текст с очень важной информацией.



Экспертное мнение

Если ты видишь такой значок, «Bay!», этот текст содержит дополнительную информацию по теме.



Что тебе нужно для этой книги

Среда разработки Python устанавливается в каталог на твой выбор, например в папку C:\Python. Там ты также сможешь разместить свои проекты Python, только чуть позже. Примеры программ в этой книге доступны для загрузки с главной страницы сайта издательства русского издания этой книги dmkpress.com.

Там же ты найдешь ответы на вопросы и решения задач (все они находятся в папке с примерами).

Операционная система

Большинство компьютеров сегодня работает под управлением операционной системы Windows. Для работы с примерами из этой книги тебе понадобится версия Windows 7 или 10. (Среда Python также доступна и для macOS и Linux.)

Носители информации

Тебе понадобится USB-накопитель (т. н. «флешка») или SD-карта, если ты захочешь сохранить свои программы на диск. На внешнем хранилище твоя работа всегда будет в безопас-

ности. При необходимости попроси своих родителей или преподавателя купить тебе такой носитель информации.

Примечания для преподавателей

Эта книга также может использоваться как учебный материал на уроках информатики в школе. Конечно, каждый учитель устанавливает свои собственные приоритеты в обучении детей программированию. Если вы уже используете другой учебник в своей работе, то сможете использовать это издание в качестве источника информации в дополнение к существующему учебнику. Эта книга начинает обучение «с нуля», так сказать, является прямым вхождением в язык программирования Python, без необходимости наличия каких-либо навыков программирования.

Важным направлением в этой книге является объектно-ориентированное программирование (ООП). Наиболее важные концепции (инкапсуляция, наследование и полиморфизм) обсуждаются в книге достаточно подробно. Еще одним направлением является игровое программирование. В проектах используются все основные элементы словаря Python, а также наиболее важные графические компоненты библиотеки `tkinter`. Вы встретите большое количество задач по программированию.

Хранение файлов проектов

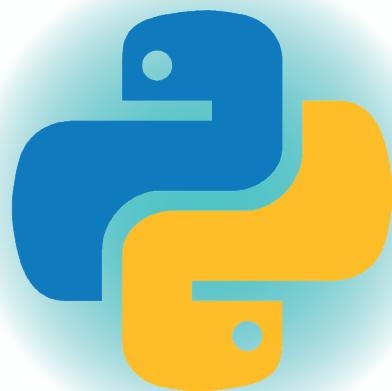
На уроках информатики каждому ученику необходимо иметь собственный внешний носитель для хранения всех заданий и собственноручно написанных программ. Таким образом, жесткому диску школьного компьютера не придется накапливать горы лишнего «мусора». Кроме того, собственный носитель данных служит для их защиты: ученик не потеряет свои файлы и с удобством сможет ими управлять.

Постоянное сохранение прогресса

Отличная идея – сохранять файлы программы в процессе работы примерно каждые десять минут. Все знают, что компьютеры любят «выходить из строя» именно тогда, когда файл не был сохранен.

1

Твои первые шаги в программировании



В этой главе разговор пойдет о том, как после установки и запуска Python сделать свои первые шаги. Я расскажу тебе, как настроить рабочее окружение так, чтобы твоя первая, написанная на языке программирования Python программа появилась на свет.

В этой главе ты узнаешь:

- как запустить Python;
- как использовать инструкции для вывода и ввода данных;
- что такое переменные;
- что такое строковый тип данных;
- как использовать среду разработки IDLE;
- как создавать и сохранять программы;
- как завершать работу с Python.

Начало работы с Python

Прежде чем ты начнешь программировать, давай настроим Python.

1

Настройка предполагает запуск установленной программы. Подробные сведения об установке ищи в приложении А. Тебе понадобится помочь взрослых, если ты не знаешь, как работать с такой программой самостоятельно. Один из способов запуска Python такой:

- Открой папку, в которую ты установил Python, например C:\Program Files\Python или C:\Python (рис. 1.1).

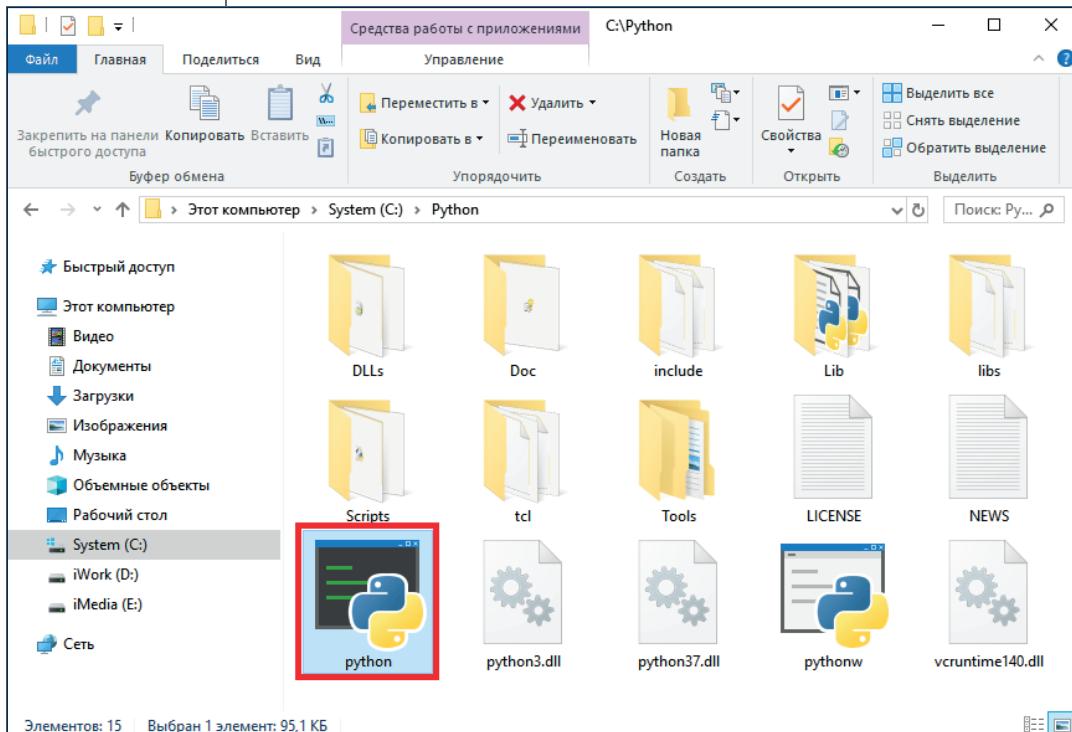


Рис. 1.1. Содержимое папки с установленным Python

- Найди среди значков файл с именем *python.exe*. Дважды щелкни мышью по значку с этим именем.

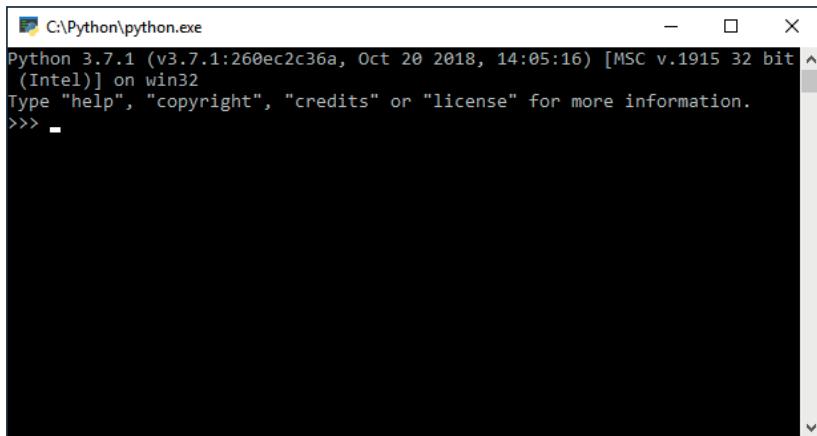
Для удобства ты можешь создать ярлык на рабочем столе:

- ❖ щелкни правой кнопкой мыши по значку *python.exe*;
- ❖ в контекстном меню выбери пункт **Отправить** ⇒ **Рабочий стол (Создать ярлык)** (*Send To* ⇒ *Desktop (Create shortcut)*);
- ❖ не обязательно ставить стандартное имя для ярлыка на файл *python.exe*. Ты можешь использовать любое подходящее слово.

Теперь можешь дважды щелкнуть мышью по новому значку и запустить Python.



Что же произойдет после запуска? Появится окно оболочки командной строки, показанное на рис. 1.2.



The screenshot shows a standard Windows command-line interface window. The title bar reads "C:\Python\python.exe". The main area displays the following text:
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:05:16) [MSC v.1915 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> -

Рис. 1.2. Запущенный Python

Первая строка сообщает о текущей версии Python, а ниже указано несколько справочных команд и три угловые скобки (>>>).

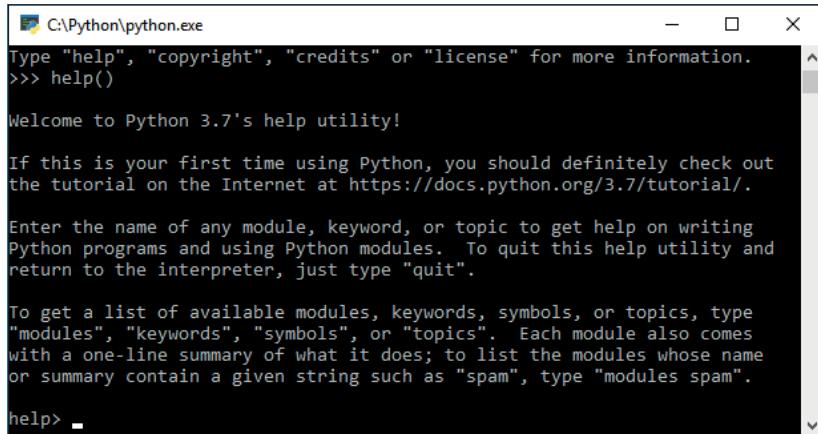
Эти три символа называются «приглашением». Это действительно своего рода приглашение, потому что ты сможешь ввести команду после них (и тебе придется это сделать, если ты хочешь продолжить работу).



- Давай попробуем поработать с Python с помощью команды `help`. Введи это слово. Обрати внимание, что помимо самого слова тебе нужно набрать две круглые скобки следом.
- Введи команду `help()` и нажми клавишу **Enter**.

На экране появится много текста (рис. 1.3).

1



```
C:\Python\python.exe
Type "help", "copyright", "credits" or "license" for more information.
>>> help()

Welcome to Python 3.7's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the Internet at https://docs.python.org/3.7/tutorial/.

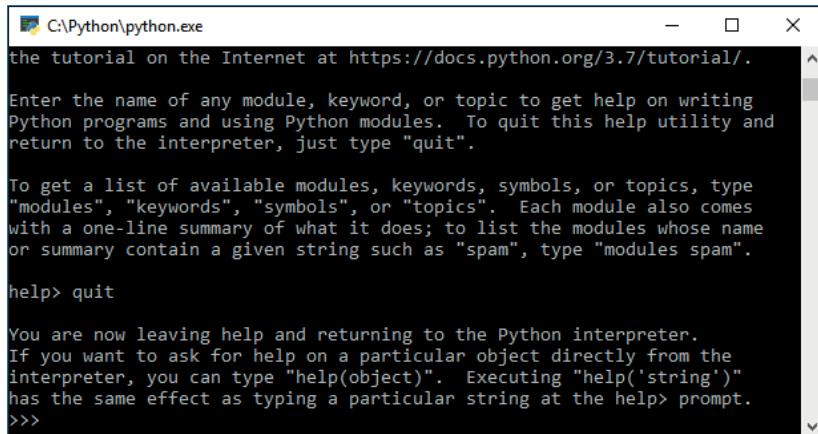
Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules. To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics". Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".
help> -
```

Рис. 1.3. Результат выполнения команды `help()`

Теперь в оболочке командной строки указано слово `help>`, которое служит подсказкой. Ты можешь указать следом слово (на английском языке), и если оно есть в словаре Python, ты увидишь небольшое объяснение.

- Чтобы вернуться к приглашению, введи команду `quit()` (рис. 1.4).



```
C:\Python\python.exe
the tutorial on the Internet at https://docs.python.org/3.7/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules. To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics". Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".
help> quit

You are now leaving help and returning to the Python interpreter.
If you want to ask for help on a particular object directly from the
interpreter, you can type "help(object)". Executing "help('string')"
has the same effect as typing a particular string at the help> prompt.
>>>
```

Рис. 1.4. Ввод команды `quit()`

И вот мы снова находимся в **интерпретаторе** языка Python.

Что же такое *интерпретатор* (другими словами, переводчик)? Во-первых, ты должен знать, что то, что ты вводишь в качестве команды, совершенно непонятно для компьютера. Как правило, он не может выполнить такую команду.

Интерпретатор *переводит* командную строку на *машинный язык*, который компьютер понимает, чтобы он мог выполнить данную тобой команду. Для программы, которая может состоять из многих-многих строк, интерпретатор переводит каждую строку по очереди и выполняет ее.

Кроме того, существуют *компиляторы*, которые переводят всю программу на машинный язык. Программа выполняется только после того, как скомпилирована и проверена, так что без ошибок может быть выполнена компьютером. В этой книге мы используем интерпретатор, но есть также и компиляторы для языка Python.



Числа и текст

Теперь попробуем что-нибудь ввести.

- Введи: `1+2+3`, а затем нажми клавишу **Enter** (рис. 1.5).

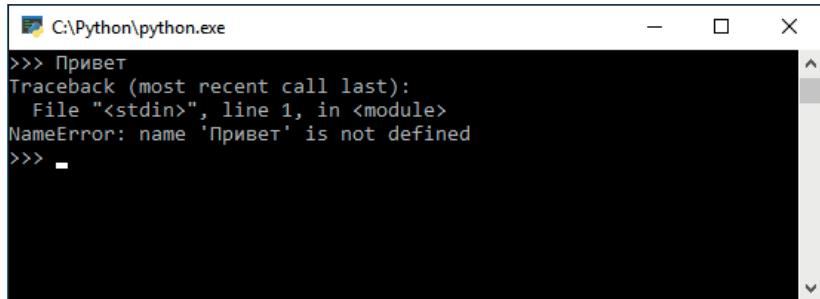
A screenshot of a Windows terminal window titled 'C:\Python\python.exe'. The window shows the Python interpreter prompt '=>'. The user has typed the expression '1+2+3' and pressed Enter. The result '6' is displayed on the next line. The window has standard window controls (minimize, maximize, close) at the top right.

Рис. 1.5. Выполнение операции сложения

Результат решения этой маленькой математической задачи появился на экране.

- Выполни еще несколько задач, используя операции вычитания (-), умножения (*) и деления (/). Как калькулятор интерпретатор Python работает отлично, но мы же с тобой хотим гораздо большего, не так ли?
- Давай попробуем небольшое приветствие: напечатай Привет и нажми клавишу ↵ **Enter** (рис. 1.6).

1

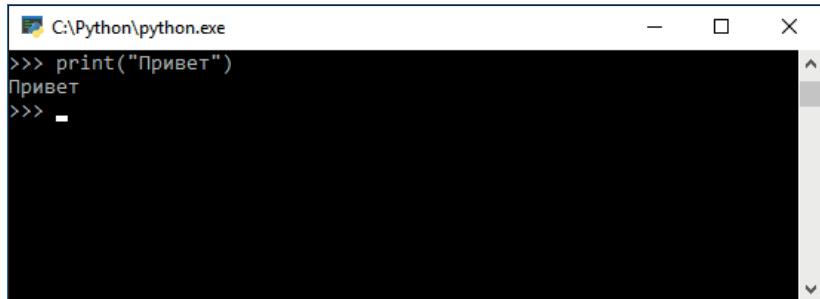


```
C:\Python\python.exe
>>> Привет
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'Привет' is not defined
>>> -
```

Рис. 1.6. Ввод слова привел к ошибке

Что случилось? Тут явно что-то не работает. А я хотел, чтобы компьютер сказал мне (т. е. написал) дружеское «Привет!».

- Введи строку `print("Привет")` и нажми клавишу **Enter** (рис. 1.7):

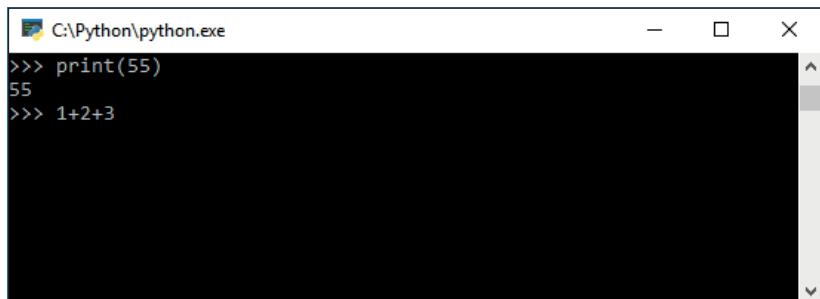


```
C:\Python\python.exe
>>> print("Привет")
Привет
>>> -
```

Рис. 1.7. Отображение приветствия

Работает! Здесь имя команды `print()` означает отображение текста, вывод на экран. В круглых скобках после слова `print` ты вводишь именно то, что хочешь отобразить на экране. Это называется *параметром*.

Конечно, такое же возможно и с числами (рис. 1.8):



```
C:\Python\python.exe
>>> print(55)
55
>>> 1+2+3
```

Рис. 1.8. Отображение чисел

Конец ознакомительного фрагмента.
Приобрести книгу можно
в интернет-магазине
«Электронный универс»
e-Univers.ru