



# Оглавление

## Предисловие

8

## 1

## Приступаем к работе

9

Введение в язык Python . . . . .	10
Установка Python в операционной системе Windows . . . . .	12
Установка Python в операционной системе Linux. . . . .	14
Знакомство с интерпретатором . . . . .	16
Ваша первая программа . . . . .	18
Работа с переменными . . . . .	20
Получение введенных пользователем данных . . . . .	22
Исправление ошибок . . . . .	24
Заключение. . . . .	26

## 2

## Выполнение операций

27

Арифметические действия. . . . .	28
Присваивание значений . . . . .	30
Сравнение величин . . . . .	32
Оценочная логика . . . . .	34
Проверка условий . . . . .	36
Определение приоритетов . . . . .	38
Преобразование типов данных. . . . .	40
Манипуляции с битами . . . . .	42
Заключение. . . . .	44

## 3

## Конструирование инструкций

45

Списки . . . . .	46
Работа со списками. . . . .	48
Неизменяемые списки . . . . .	50
Элементы ассоциативного списка . . . . .	52
Ветвление с помощью условного оператора. . . . .	54
Цикл while. . . . .	56

Обход элементов в цикле. . . . .	58
Выход из цикла . . . . .	60
Заключение. . . . .	62

## 4 Определение функций 63

Область видимости переменных. . . . .	64
Подстановка аргументов . . . . .	66
Возвращение значений . . . . .	68
Использование обратного вызова. . . . .	70
Добавление заполнителей. . . . .	72
Генераторы в Python . . . . .	74
Обработка исключений . . . . .	76
Отладка с помощью инструкции assert. . . . .	78
Заключение. . . . .	80

## 5 Импорт модулей 81

Хранение функций . . . . .	82
Принадлежность имен функций . . . . .	84
Системные запросы . . . . .	86
Математические операции . . . . .	88
Вычисления с десятичными дробями. . . . .	90
Работа со временем . . . . .	92
Запуск таймера . . . . .	94
Шаблоны соответствий . . . . .	96
Заключение. . . . .	98

## 6 Строки и работа с файлами 99

Работа со строками. . . . .	100
Форматирование строк. . . . .	102
Модификация строк . . . . .	104
Преобразование строк . . . . .	106
Доступ к файлам. . . . .	108
Чтение и запись файлов. . . . .	110
Изменение текстового файла . . . . .	112
Консервация данных. . . . .	114
Заключение. . . . .	116

## 7 Объектное программирование 117

Инкапсуляция данных. . . . .	118
Создание экземпляров объектов . . . . .	120
Доступ к атрибутам класса. . . . .	122

Встроенные атрибуты . . . . .	124
Сборка мусора . . . . .	126
Наследование свойств . . . . .	128
Переопределение основных методов . . . . .	130
Реализация полиморфизма . . . . .	132
Заключение . . . . .	134

## 8      **Обработка запросов**      **135**

Отправка ответов . . . . .	136
Обработка данных . . . . .	138
Передача данных через формы . . . . .	140
Использование текстовых областей . . . . .	142
Установка флажков . . . . .	144
Установка переключателя в положение . . . . .	146
Элементы списка . . . . .	148
Выгрузка файлов . . . . .	150
Заключение . . . . .	152

## 9      **Разработка интерфейсов**      **153**

Запуск оконного интерфейса . . . . .	154
Работа с кнопками . . . . .	156
Вывод сообщений . . . . .	158
Прием данных от пользователя . . . . .	160
Выбор из списка . . . . .	162
Использование переключателей . . . . .	164
Флажки . . . . .	166
Добавление изображений . . . . .	168
Заключение . . . . .	170

## 10     **Разработка приложений**      **171**

Генерирование случайных чисел . . . . .	172
Планирование программы . . . . .	174
Построение интерфейса . . . . .	176
Определение постоянных величин . . . . .	178
Инициализация изменяемых значений . . . . .	179
Добавление рабочей функциональности . . . . .	180
Тестирование программы . . . . .	182
Компиляция программы . . . . .	184
Распространение приложения . . . . .	186
Заключение . . . . .	188

## **Предметный указатель**      **189**

# Предисловие

Создание этой книги лично для меня стало увлекательным путешествием в мир, раскрывающий возможности языка Python в современном процедурном и объектно ориентированном программировании, используемом для обеспечения функциональности при разработке онлайн-приложений. Примеры кода, представленные в этой книге, описывают, как за несколько простых шагов создавать программы на языке Python, а на скриншотах демонстрируются реальные результаты их работы. Я искренне надеюсь, что вам понравится открывать захватывающие возможности Python и вы получите при этом не меньше удовольствия, чем я во время написания этой книги.

Для того чтобы код, описанный в примерах, стал более наглядным, он отформатирован черным шрифтом, за исключением комментариев, выделенных серым шрифтом:

```
# Пишем традиционное приветствие  
  
greeting = 'Hello World!'  
  
print( greeting )
```

Кроме того, для идентификации исходных файлов, описываемых в пошаговых инструкциях, на полях рядом с каждым пунктом будут появляться значок и имя соответствующего файла:



script.py



page.html



image.gif

Для удобства файлы исходных кодов всех примеров, представленных в этой книге, помещены в один ZIP-архив. Вы можете получить его, выполнив следующие простые шаги.

1. Откройте браузер и загрузите архив по ссылке [http://eksmo.ru/Python\\_examples.zip](http://eksmo.ru/Python_examples.zip).
2. Извлеките из скачанного архива папки *MyScripts* и *MyProjects* в ваш домашний каталог (например, в C:\) а также скопируйте содержимое папки *htdocs* в каталог документов вашего веб-сервера.
3. Теперь вы можете, используя пошаговые инструкции, выполнять примеры с помощью интерпретатора Python и видеть результаты его работы.

# 1

## Приступаем к работе

*Добро пожаловать  
в увлекательный мир языка  
программирования Python.  
В этой главе показывается,  
как установить Python  
и создать вашу первую  
программу.*

- Введение в язык Python
- Установка Python в среде Windows
- Установка Python в среде Linux
- Знакомство с интерпретатором
- Ваша первая программа
- Работа с переменными
- Получение введенных пользователем данных
- Исправление ошибок
- Заключение



Будьте в курсе последних новостей проекта Python на сайте **python.org**.

#### Совет



Так называемое правило офсайда, которое используют некоторые языки программирования, выделяя блоки кода при помощи отступов, заимствовано из футбола.

# Введение в язык Python

Python является высокоуровневым («человекочитаемым») языком программирования, который для вывода результатов использует интерпретатор. Python содержит обширную стандартную библиотеку модулей протестированного кода, которые легко могут быть включены в ваши собственные программы.

Язык Python, разработанный Гвидо ван Россумом (Guido van Rossum) в конце восьмидесятых — начале девяностых годов в Национальном научно-исследовательском институте математики и компьютерных наук в Нидерландах, является производным от многих других языков, в том числе C, C++ и командной оболочки Unix. Сегодня Python поддерживается командой разработчиков ядра в институте, хотя Гвидо ван Россум по-прежнему играет важную роль в определении направления развития языка.

Читаемость кода, делающая язык Python особенно подходящим для новичков в программировании, — один из принципов философии Python, которую можно обобщить следующим образом.

- Красивое лучше, чем уродливое.
- Явное лучше, чем неявное.
- Простое лучше, чем сложное.
- Сложное лучше, чем запутанное.
- Читаемость имеет значение.

Поскольку Python ориентирован на читаемость кода, в нем часто используются ключевые слова на английском языке там, где другие языки программирования обычно используют знаки препинания. Особое его отличие состоит в том, что для группировки инструкций в блоке кода Python использует отступы, а не ключевые слова или знаки препинания. В языке Pascal, например, начало блоков обозначается ключевым словом `begin` и заканчивается ключевым словом `end`, в то время как программисты на C используют фигурные скобки для обозначения блоков кода. Очень часто такой подход группировки блоков отступами критикуется программистами, знакомыми с другими языками, но, несомненно, использование отступов в Python позволяет программам выглядеть менее нагроможденными.

Перечислим некоторые из важнейших отличительных особенностей языка Python, которые делают его привлекательным для начинающих программистов.

- **Python бесплатен** — это свободно распространяемое программное обеспечение с открытым исходным кодом.
- **Python легок в изучении** — он имеет простой синтаксис.
- **Python позволяет создавать легко читаемый код** — он не перегружен знаками препинания.
- **Python легок в обслуживании** — имеет модульную структуру.
- **Python располагает богатым «арсеналом»** — он предлагает большую стандартную библиотеку, которая легко интегрируется в ваши программы.
- **Python портируемый** — его можно запустить на обширном множестве различных платформ, и везде он будет иметь один и тот же интерфейс.
- **Python интерпретируемый** — компиляция не требуется.
- **Python является высокоуровневым языком** — он имеет статическое распределение памяти.
- **Python расширяемый** — позволяет добавлять низкоуровневые модули.
- **Python универсален** — поддерживает как процедурный, так и объектно ориентированный методы программирования.
- **Python гибок в использовании** — с его помощью можно создавать консольные программы, приложения графического интерфейса, а также сценарии для взаимодействия внешних программ с веб-серверами.

Как и любое другое программное обеспечение, Python продолжает развиваться, его новые версии выпускаются с определенной периодичностью. Объявлено, что версия 2.7 будет окончательной в ветке 2.x. Но ее поддержка будет продлена до 2020 года. Других больших релизов в данной ветке не ожидается.

Ветка версии 3.x находится в активной разработке и уже имеет несколько стабильных релизов. Это значит, что все последние улучшения стандартных библиотек, например, окажутся доступными только в версии Python 3.x. Описанные в нашей книге особенности языка будут относиться к версии 3.x.

#### На заметку



Название языка Python было дано в честь популярного британского комедийного шоу «Летающий цирк Монти Пайтона» (Monty Python's Flying Circus) — вы можете найти упоминание об этом в документации по языку.

#### Внимание



Python 3.x обратно несовместим с версией Python 2.7.





Установщики для операционной системы OS X 32-битной и 64-битной версий также доступны для загрузки на [python.org/download](https://python.org/download).

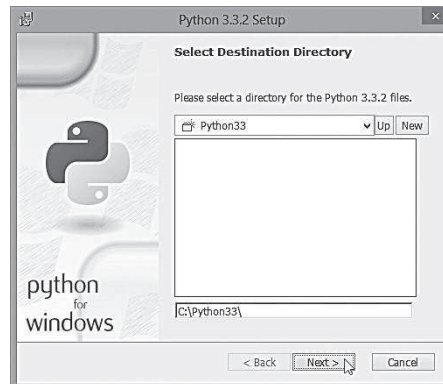
**Совет**

Поддержка установщика MSI включена для всех версий Windows и доступна для свободной загрузки на [microsoft.com/downloads](https://microsoft.com/downloads) — введите в строке поиска Windows Installer.

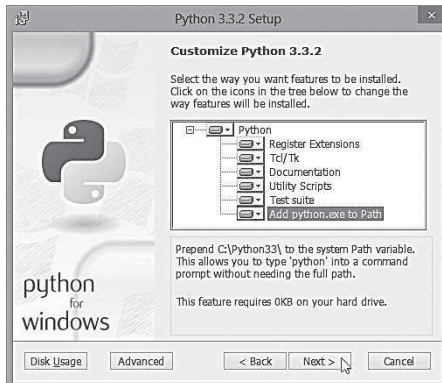
# Установка Python в операционной системе Windows

Перед тем как вы начнете программировать на языке Python, необходимо установить на ваш компьютер интерпретатор Python, а также стандартную библиотеку модулей кода, поставляемую вместе с ним. Все это можно свободно загрузить на странице [python.org/download](https://python.org/download). Для пользователей операционной системы Windows существуют две версии инсталлятора: для 32-битных и 64-битных систем.

1. Запустите веб-браузер, перейдите на страницу [python.org/download](https://python.org/download) и загрузите установщик, подходящий для вашей версии операционной системы — в нашем примере файл имеет имя *python-3.3.2.msi*.
2. После завершения загрузки запустите установщик, выберите режим установки для всех пользователей либо только для себя и нажмите кнопку **Next** (Далее) для продолжения.
3. Теперь подтвердите предлагаемое расположение установки, в название которого будет входить имя корневого диска, слово Python и номер версии — в данном примере установка произойдет в каталог *C:\Python33* для версии 3.3.2.



4. Нажмите кнопку **Next** (Далее) для продолжения и убедитесь, что выбран компонент **Add python.exe to Path** (Добавить путь в системную переменную Path).

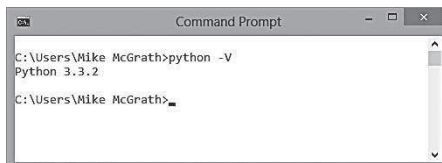


#### На заметку



Убедитесь, что все компоненты установки включены, как показано в нашем примере.

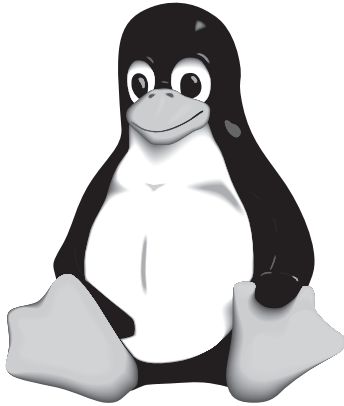
5. Нажмите **Next** (Далее), чтобы начать копирование файлов на ваш компьютер, а затем — **Finish** (Готово) для завершения процесса установки.
6. Чтобы убедиться, что Python теперь доступен, перезагрузите компьютер, запустите командную строку (*cmd.exe*) и наберите команду **python -V** — в ответ интерпретатор Python выдаст номер установленной версии.



#### Внимание



Буква **V** в команде должна быть указана обязательно прописной. Перед тем как продолжать работать с примерами в книге, убедитесь, что данная команда выдает необходимый номер установленной версии.



Обратитесь к документации по вашей операционной системе Linux для дальнейшей установки Python.

#### Внимание

Не удаляйте версию 2.7 из вашей системы, так как во многих случаях существуют зависимые от нее приложения, работоспособность которых может нарушиться.



# Установка Python в операционной системе Linux

В дистрибутивы Linux обычно включен Python — по умолчанию там используется версия 2.7. Для работы с веткой 3.x вам, очевидно, предстоит установить нужный релиз дополнительно.

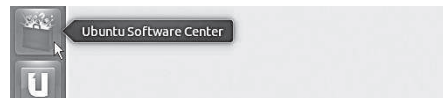
1. Запустите терминальное окно и наберите в точности, как указано, команду `python -V` для вывода информации об установленной версии по умолчанию.

```
mike@ubuntu: ~  
mike@ubuntu:~$ python -V  
Python 2.7.3  
mike@ubuntu:~$
```

2. Затем наберите в точности команду `python3 -V` для того, чтобы увидеть информацию об установленной версии ветки 3.x, если таковая имеется.

```
mike@ubuntu: ~  
mike@ubuntu:~$ python3 -V  
Python 3.2.3  
mike@ubuntu:~$
```

3. Теперь запустите на вашей Linux системе менеджер пакетов, чтобы посмотреть, какая из последних версий Python доступна для установки — например, на системах с Ubuntu вы можете использовать Центр приложений (Ubuntu Software Center).



4. Найдите в менеджере пакетов необходимое вам программное обеспечение, название которого содержит слово Python, чтобы посмотреть информацию, какие компоненты установлены или доступны для установки.



5. Наконец установите последнюю версию из ветки Python3.x — в данном случае это Python3.3.
6. Для проверки доступности последней версии Python на вашем компьютере запустите терминальное окно и наберите команду `python3.3 -V`.



#### Совет



Вы можете по желанию установить среду разработки IDLE для Python3.3, но это совсем не обязательно, так как все примеры в книге созданы при помощи обычного текстового редактора, такого, как Nano.

#### На заметку

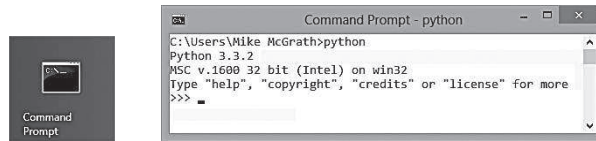


Теперь вы можете использовать команду `python3.3` для того, чтобы ваши программы отработывались интерпретатором именно этой версии.

# Знакомство с интерпретатором

Интерпретатор Python обрабатывает текстовый код вашей программы, а также имеет интерактивный режим, полезный для отладки и тестирования фрагментов кода. В интерактивный режим Python можно попасть несколькими способами:

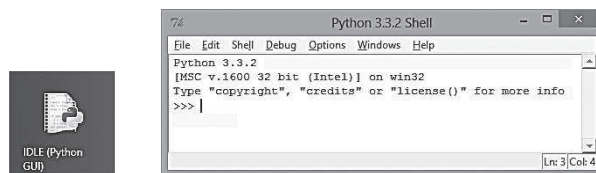
- из обычной командной строки — введите команду `python`, чтобы запустить начальную командную строку Python (символы `>>>`), в которой вы будете взаимодействовать с интерпретатором;



- из меню **Пуск** (Start) — выберите пункт **Python (command line)** — запустится окно, содержащее начальную командную строку интерпретатора Python с символами `>>>`;

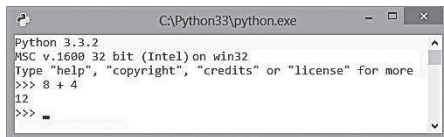


- из меню **Пуск** (Start) — выберите пункт **IDLE (Python GUI)**, чтобы запустить окно оболочки Python, содержащее командную строку с символами `>>>`.



Независимо от того, какой способ вы выбрали для входа в интерактивный режим, интерпретатор Python будет одинаково реагировать на команды, вводимые в его командной строке после знаков `>>>`. В этом режиме его можно использовать в качестве калькулятора.

1. Войдите в интерактивный режим Python, используя любой из вышеперечисленных методов, затем наберите простое выражение с операцией сложения и нажмите кнопку **Enter**. Интерпретатор в ответ выдаст вам сумму.



```
Python 3.3.2
MSC v.1600 32 bit (Intel) on win32
Type "help", "copyright", "credits" or "license" for more
>>> 8 + 4
12
>>>
```

Интерпретатор Python понимает любые арифметические выражения, поэтому можно использовать скобки для указания порядка вычисления — часть выражения, заключенная в скобки, будет вычисляться первой.

2. Затем в командной строке Python наберите выражение с тремя операндами без указания порядка вычисления.



```
>>> 3 * 8 + 4
28
>>>
```

3. Теперь в командной строке Python наберите то же самое выражение, но добавьте скобки, определяющие порядок вычисления.



```
>>> 3 * ( 8 + 4 )
36
>>>
```

#### Совет



Пробелы в выражениях игнорируются, поэтому выражение `8+4`, как показано здесь, можно записать с добавлением пробелов просто для красоты восприятия.

#### На заметку



Интерактивный режим используется в основном для тестирования и отладки фрагментов кода.

#### Внимание



IDLE расшифровывается как Integrated Development Environment — интегрированная среда разработки. Она имеет ограниченные функции и в данной книге не используется для демонстрации примеров.

**Внимание**

Не используйте текстовые процессоры для создания исходного кода программ, поскольку они добавляют дополнительное форматирование.



hello.py

**Совет**

Созданный каталог `C:\MyScripts` будет использоваться во всех примерах этой книги для Windows.

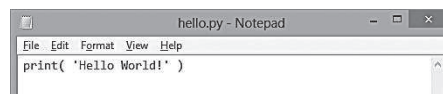
# Ваша первая программа

Кроме того, что интерактивный режим Python полезен в качестве простейшего калькулятора, его можно использовать для создания программ. Программа на языке Python — это обычный текстовый файл, созданный с помощью простого редактора, такого как Блокнот (Notepad), и сохраненный в файле с расширением `.py`. Запустить программу на Python можно, указав имя соответствующего файла после команды `python` в командной строке интерпретатора.

По традиции первая программа, которую создают при изучении языка программирования, просто выводит какое-либо сообщение с приветствием. На языке Python для этого используется функция `print()`, сообщение для вывода этой функции указывается в скобках. Это может быть строка символов, заключенная в кавычки. Кавычки могут быть как двойными (`"`), так и одинарными (`'`), но нельзя использовать одновременно и те, и другие.

1. На компьютере под управлением операционной системы Windows запустите простой текстовый редактор, такой как, например, Блокнот (Notepad).
2. Затем наберите следующую инструкцию в пустой строке редактора:  

```
print( 'Hello World!' )
```
3. Теперь создайте новый каталог `C:\MyScripts` и сохраните в нем файл под именем `hello.py`.



4. Теперь запустите окно командной строки, перейдите в только что созданный каталог и наберите команду `python hello.py` — вы увидите, как интерпретатор Python запустит вашу программу и выведет приветственное сообщение.

