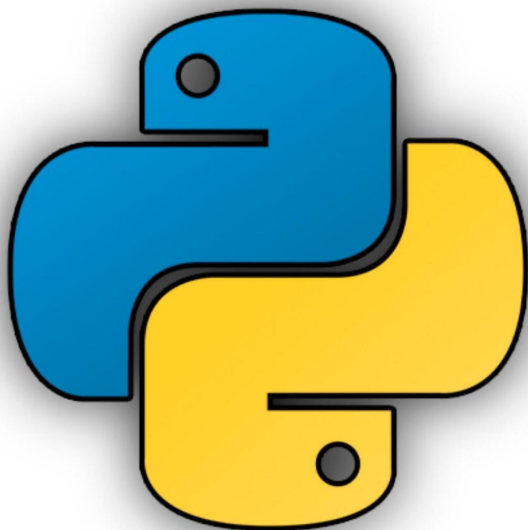


# **С нуля до** **Python-разработчика** **3 издание**



# **Фин Дерт**

# **С нуля до Python-разработчика**

*[http://www.litres.ru/pages/biblio\\_book/?art=68753790](http://www.litres.ru/pages/biblio_book/?art=68753790)*

*SelfPub; 2024*

## **Аннотация**

Книга об основах языка программирования Python, рассказанных простыми словами. Автор постарался упростить всю теорию, сделав её понятной для тех, кто только начинает знакомиться с программированием и ранее не имел опыта с другими языками. Книга рассчитана на новичков, которые не имеют никаких практических знаний или только начали изучать язык программирования Python

# Содержание

Предисловие	4
1. Редакторы Кода	5
2. Встроенные инструменты Python	7
3. Создание Python-файла	8
4. Вывод текста	9
*4.1. Полезные параметры print()	10
5. Числа, дроби и текст	12
6. Операции с числами	14
*6.1 Продвинутое математические операции	15
7. Переменные	16
Конец ознакомительного фрагмента.	19

# Фин Дерт

# С нуля до Python-разработчика

## Предисловие

Всем доброго дня! В этой книге я расскажу об **основах Python**, которых будет достаточно для начала изучения большинства технологий, при помощи которых вы сможете создать первый серьёзный проект. Постараюсь рассказывать информацию как можно проще, чтобы даже новички смогли без труда освоить Python. На момент написания книги актуальной версией Python является 3.11, но если выйдет новая версия, то структура языка не сильно изменится.

Хотелось бы отметить пару правил :

- *Повторение – мать учения (Овидия)*
- *Знания сокровища, но ключ к ним – практика (Томас Фуллер)*

**Удачи в обучении!**

# 1. Редакторы Кода

Начать обучение стоило бы с места, где можно писать код – редактора кода. Если вы любитель хардкора, то можно использовать блокнот, но давайте будем честны, кто будет так поступать? Так как редакторов кода очень много и выбрать из них подходящий под вас будет тяжело, то я покажу самые известные и проверенные опытом редакторы.

PyCharm – Самый популярный редактор для Python, выпущенный компанией JetBrains. Он имеет удобное рабочее пространство со встроенным полем для ввода команд в консоль и, что я считаю самым крутым – возможность запускать код не выходя из программы. Так же, что будет полезно для новичков – он подсвечивает вам ошибки и предлагает способы их исправления. Ниже показан пример кода в PyCharm.

```
print("Немного текста")  
print(Забыл кавычки)
```

VS Code – Универсальный редактор, который подходит для всех языков программирования. Для удобной работы с Python вам придётся установить дополнительные библиотеки.

ки. В общем – удобный редактор для тех, кто разрабатывает не только на Python, но и на других языках.

```
print("Some text")  
print(Guide from dertfin)
```

Sublime Text – Как и VS Code, является универсальным инструментом. Так же подходит для написания простого текста.

```
print("Немного текста")  
print(Забыл кавычки)
```

Надеюсь, что среди этих трёх вариантов вы нашли тот, который вам нравится. Желаю вам приятной установки!

## 2. Встроенные инструменты Python

Сам по-себе Python предоставляет некоторые инструменты, которые считаю важным рассмотреть.

IDLE – Встроенный в Python редактор. Подойдёт для тестирования небольших функций, но лучше рассмотреть редакторы из первой главы, так как в них поддерживается тёмная тема, которая будет приятней для ваших глаз.

```
print("Немного текста")  
print(Забыл кавычки)
```

Python-Консоль – Консоль, в которой можно выполнять команды, не прописывая их в файле. В начале обучения её можно будет использовать, но при создании кода в несколько строк у вас смогут возникнуть затруднения.

```
>>> print("Немного Текста")  
Немного Текста
```

## 3.Создание Python-файла

Создать Python-Файл очень просто. Всё, что вам нужно :

1. Создать *.txt* файл
2. Заменить *.txt* на *.py*

Готово!

Важно отметить, что в отличии от других языков программирования, файл с расширением *.py* может и редактироваться, и запускаться (*не требует компиляции*).

## 4. Вывод текста

Вывод текста – очень лёгкий процесс. Вам просто нужно написать `print(`""), а в кавычках указать нужный текст. Если в начале обучения вы выбрали редактор, не имеющий встроенный запуск кода, то при запуске программы она просто вылетит. Это нормально, так как Python-Консоль просто закрывается при выполнении кода. Для того чтобы исправить эту проблему, просто добавьте строчку `input()` в конце кода. Чуть-позже обсудим, что это такое.

В одном `print()` можно указать больше информации, просто разделив её запятыми:

```
print("Текст 1. ", "Текст 2")
```

```
>>> Текст 1. Текст 2
```

На данном этапе это может показаться бесполезным, но позже такой прием вам ещё не раз пригодится.

## \*4.1. Полезные параметры *print()*

У команды *print()* есть два полезных параметра – *sep* и *end*. Их использование не является обязательным, но в будущем они вам могут понадобиться в самых разных задачах.

Начнем с *sep*. Если вы использовали более, чем один текст(или другие данные, которые мы рассмотрим позже), то можете добавить *sep*, который скажет программе, чем разделить эту информацию.

Рассмотрим на примере. У нас есть список городов: Москва, Омск и Волгоград. Если мы просто напишем их по-отдельности, но не разделим вручную, что получим такой результат:

```
print("Москва", "Омск", "Волгоград")
```

```
>>> МоскваОмскВолгоград
```

Добавим в конце параметр *sep*, где укажем, что разделить названия городов нужно запятой и пробелом:

```
print("Москва", "Омск", "Волгоград", sep=", ")
```

```
>>> Москва, Омск, Волгоград
```

Так-то лучше, правда, как-то не хватает точки в конце. Тут нам и поможет второй параметр – *end*. Давайте добавим его и укажем в нем точку, чтобы наш текст приобрел грамотный вид:

```
print("Москва", "Омск", "Волгоград", sep=", ", end=".")
```

```
>>> Москва, Омск, Волгоград.
```

Красота!

## 5. Числа, дроби и текст

Только что мы полностью изучили команду *print()*. Вы наверняка заметили, что текст, находящийся внутри скобок мы написали в кавычках. Но почему же?

Вся информация в Python имеет свой тип данных, который по-разному указывается в коде. Давайте рассмотрим три самых простых типа данных :

**String** – текстовый тип данных. Может содержать в себе буквы, знаки и цифры. Ограждается одинарными или двойными кавычками с обеих сторон.

*Пример : "Текст"*

Как раз String мы и использовали в команде *print()*. Теперь перейдём к числам :

**Integer** (int) – Тип данных, который может содержать целое число.

*Пример : 3035*

**Float** – в переводе с Англ. Языка "float" будет "плавать", так как данный тип данных может хранить дробное число. Дробная часть отделяется „плавающей“ точкой.

*Пример : 3.14*

Как вы видите, численные типы данных не ограждаются кавычками. Можем сделать вывод, что для их вывода можно использовать просто *print(число)*. Например, *print(3010)* .

Рекомендую вам самостоятельно попрактиковаться в выводе разных типов данных, чтобы лучше закрепить материал.

## 6. Операции с числами

В прошлой теме мы изучили числа. Давайте „поиграем“ с ними. Сначала покажу вам стандартные математические действия в Python.

- Сложение (+)
- Вычитание (-)
- Умножение (\*)
- Деление (/)

Мы можем использовать эти действия прямо внутри функций, включая знакомый нам *print()*. Вот пару примеров их применения :

```
print(2.9524 + 1.41245)
```

```
print(D * 3.14) # Длина окружности, где D=Диаметр
```

Для лучшей читаемости кода можно ставить один пробел между числом и действием.

Как и в обычной математике, в Python работают скобки:

*Пример : print(3 \* (2 + 5))*

## **\*6.1 Продвинутое математические операции**

Помимо сложения, вычитания, умножения и деления в Python есть и другие операции:

- Возведение в степень(\*\*)
- Деление без остатка (//)
- Остаток от деления (%)

Давайте применим их на практике:

Возьмём примеры  $5 : 3 = 1$  (ост. 2) и  $5^3 = 125$

```
print(5 // 3)
```

```
>>> 1
```

```
print(5 % 3)
```

```
>>> 2
```

```
print(5**3)
```

```
>>> 125
```

## 7. Переменные

В 6-ом пункте был интересный пример вывода длинны окружности, где я обозначил диаметр буквой *D*. Указывать диаметр прямо внутри *print()* будет не очень удобно. Хотелось бы это „*D*” вывести куда-нибудь в отдельное место. Для таких случаев есть такая вещь, как переменная.

Переменная – это некая „коробочка“, куда можно положить информацию.

В нашей коробочке „*D*“ будет лежать какое-либо число.



Для того чтобы эту „коробочку“ создать, нам нужно на новой строке указать название переменной, знак „равно“ и значение.

*Пример :  $D = 20$*

Для использования переменной нужно просто написать её название. Если эту строку добавить перед `print(D * 3.14)`,

то результатом программы будет 62.8 , что является ответом на  $20 * 3.14$  .

# Конец ознакомительного фрагмента.

Текст предоставлен ООО «Литрес».

Прочитайте эту книгу целиком, [купив полную легальную версию](#) на Литрес.

Безопасно оплатить книгу можно банковской картой Visa, MasterCard, Maestro, со счета мобильного телефона, с платежного терминала, в салоне МТС или Связной, через PayPal, WebMoney, Яндекс.Деньги, QIWI Кошелек, бонусными картами или другим удобным Вам способом.