Глава 1

ВВЕДЕНИЕ В ПРОГРАММИРОВАНИЕ НА PYTHON

Что такое Python

Python — это универсальный язык программирования высокого уровня, который легко изучать и читать и на котором просто писать. В 1991 году Гвидо ван Россум впервые опубликовал исходный код Python, и с тех пор он стал одним из самых популярных языков программирования в мире.

Python обладает рядом особенностей, которые делают его отличным выбором как для начинающих, так и для опытных программистов. Одна из ключевых особенностей Python — читаемость. Код на Python легко читается и понимается, что облегчает его изучение и сопровождение. Python также имеет простой и интуитивно понятный синтаксис, это означает, что разработчики могут писать код быстрее и с меньшим количеством ошибок.

Еще одна причина популярности языка — его универсальность. Python можно использовать для широкого спектра приложений, включая веб-разработку, научные вычисления, искусственный интеллект и анализ данных. Python также имеет большое и активное сообщество разработчиков, которые вносят свой вклад в создание различных библиотек и инструментов с открытым исходным кодом.

Python — интерпретируемый язык, а это означает, что его интерпретатор выполняет код напрямую, без необходимости компиляции. Это делает код Python легким для тестирования и отладки, а также позволяет разработчикам быстро повторять код и экспериментировать с ним.

Python — отличный выбор для всех, кто хочет научиться программированию. Его простота, читабельность и универсальность делают его отличным языком для начинающих, а мощные возможности и активное сообщество делают его ценным инструментом для опытных разработчиков.

История Python

История языка программирования Python началась в конце 1980-х. Гвидо ван Россум — голландский ученый-компьютерщик, который в то время работал в научно-исследовательском институте Centrum Wiskunde & Informatica в Амстердаме, начал работу над Python в качестве хобби. Он намеревался создать язык, который было бы легко читать и писать.

Первая версия Python, 0.9.0, была выпущена в феврале 1991 года. Это был простой язык с базовыми типами данных, управляющими структурами и функциями, но он быстро завоевал популярность среди небольшого сообщества разработчиков, которые его использовали.

В 1994 году была выпущена версия Python 1.0, где появились новые возможности: инструменты функционального программирования и система модулей. Эта версия Python также включала систему Python Enhancement Proposals (PEPs), которая позволяла разработчикам предлагать и обсуждать изменения в языке.

Python 2.0 был выпущен в 2000 году, в него были включены сборка мусора и поддержка Unicode. За Python 2.0 последовало несколько других релизов в серии 2.x, включая Python 2.7, который был выпущен в 2010 году и широко используется по сей день.

В 2008 году началась разработка Python 3.0, целью которой было устранение некоторых недостатков и несоответствий в Python 2.x. Python 3.0 был выпущен в 2008 году и внес в язык несколько серьезных изменений, включая новую функцию `print`, улучшенную поддержку Unicode и упрощенный синтаксис для определения классов.

Сегодня Python — один из самых популярных языков программирования в мире с большим и активным сообществом разработчиков, которые вносят свой вклад в создание различных библиотек и инструментов с открытым исходным кодом.

Установка Python и среды разработки

Для начала нужно установить Python на свой компьютер. Это бесплатный язык с открытым исходным кодом, его можно загрузить с официального сайта Python по адресу www.python.org.

Есть две основные версии Python: Python 2.x и Python 3.x. Хотя Python 2.x все еще используется в некоторых старых приложениях, для новых проектов рекомендую использовать Python 3.x, поскольку это последняя версия языка, имеющая несколько важных улучшений по сравнению с Python 2.x.

Вы можете загрузить программу установки с официального сайта и следовать инструкциям по установке. В процессе установки можно выбрать глобальную

установку Python на своем компьютере или локальную в определенном каталоге.

Помимо самого Python вам может понадобиться установить среду разработки, которая поможет писать код Python и управлять им. Среда разработки — это приложение, которое предоставляет инструменты для написания, тестирования и отладки кода. Некоторые популярные среды разработки для Python включают PyCharm, Visual Studio Code и Sublime Text.

При выборе среды разработки следует учитывать ее простоту использования, поддержку библиотек и инструментов Python и совместимость с вашей операционной системой. Многие среды разработки предлагают автодополнение кода, подсветку синтаксиса и инструменты отладки, которые помогут писать код более эффективно и с меньшим количеством ошибок.

После установки Python и среды разработки можно приступать к написанию кода на Python! В следующем разделе рассмотрим основы написания и выполнения программ на Python.

Интерпретатор Python и REPL (Read-Eval-Print Loop)

Одна из уникальных особенностей Python — его интерактивный режим, который позволяет выполнять код и сразу же видеть результаты. Это возможно благодаря интерпретатору Python, который представляет собой программу, читающую и выполняющую код Python.

Когда вы устанавливаете Python на свой компьютер, он включает интерактивный интерпретатор цикл чтения-вывода-печати (REPL). REPL позволяет вводить код по одной строке за раз и сразу же видеть результаты. Чтобы запустить Python REPL, откройте терминал или командную строку и введите `python`.

Запустите интерпретатор Python, введите код и нажмите Enter, чтобы посмотреть результаты. Например, чтобы вывести на экране «Привет, мой новый друг», наберите такой код:

```
>>> print("Привет, мой новый друг!")
Привет, мой новый друг!
```

`>>>` — это подсказка, которую интерпретатор Python выводит на экран, чтобы показать, что он готов к вводу кода. Интерпретатор оценит код и выведет результат на следующей строке.

Помимо интерактивного режима, можно записать код в файл и выполнить его из командной строки. Для этого создайте новый текстовый файл и сохраните его с расширением `.py` (например, `myprogram.py`). Затем запустите программу, набрав `python myprogram.py` в командной строке.

Интерпретатор Python и REPL — это мощные инструменты для написания и тестирования кода Python. Используя интерактивный режим, можно быстро экспериментировать с различными фрагментами кода и сразу же видеть результаты. В следующем разделе рассмотрим основы написания кода в файле и его запуск из командной строки.

Ваша первая программа на Python

Теперь, когда у вас установлен Python и есть базовое понимание того, как использовать интерпретатор, напишем нашу первую программу на Python!

В Python программа — это набор операторов, которые выполняются по порядку. Оператор — это строка кода, которая выполняет определенное действие, например выводит сообщение на экране или вычисляет значение.

Откройте текстовый редактор (например, Notepad или Sublime Text) и создайте новый файл. Сохраните файл с расширением `.py`, которое указывает, что это программа на Python. Например, `program.py`.

Затем введите следующий код:

```
print("Я готов, хозяин!")
```

Код говорит Python вывести на экран сообщение «Я готов, хозяин!». Функция `print` — это встроенная функция в Python, позволяющая выводить текст на экран.

После того как вы ввели код, сохраните файл и выйдите из текстового редактора. Теперь откройте окно командной строки или терминала и перейдите в каталог, где сохранили файл. Введите `python program.py` и нажмите Enter.

Python выполнит код в файле и выведет на экран сообщение «Я готов, хозяин!». Поздравляю, вы только что написали и запустили свою первую программу на Python!

Конечно, это только начало того, что можно сделать с помощью Python. В следующем разделе рассмотрим некоторые основные концепции программирования в Python, включая переменные, типы данных и операторы.

Синтаксис и основные концепции программирования

Рассмотрим основные концепции программирования в Python, включая синтаксис, переменные, типы данных и операторы.

Синтаксис относится к правилам и рекомендациям по написанию кода Python. В этом языке утверждения обычно пишутся на отдельных строках, а для обозначения блоков кода используются отступы.

Например, следующий код создает переменную и присваивает ей значение:

```
x = 33
```

Знак равенства (=) используется для присвоения значения переменной. Переменные предназначены для хранения значений, которые могут быть использованы в программе позже.

Python поддерживает несколько типов данных, включая целые числа, числа с плавающей точкой, строки и булевы значения. Целочисленные данные — это целые числа (например, 1, 2, 3), числа с плавающей точкой — это десятичные числа (например, 9.99, 3.1415), а строки — это последовательности символов (например, «робот», «хозяин»). Булевы значения — `True` или `False`.

Операторы используются для выполнения операций над переменными и значениями. Python поддерживает несколько операторов, включая арифметические операторы (+, –, *, /), операторы сравнения (==, !=, <, >) и логические операторы (and, or, not). В следующем коде для выполнения вычислений используются арифметические операторы:

```
x = 30
y = 3
z = x + y
print(z) # Вывод: 33
```

В этом примере знак «плюс» (+) используется для сложения значений `x` и `y`, а результат сохраняется в переменной `z`. Затем функция `print` используется для вывода значения `z` на экран.

Изучив синтаксис, переменные, типы данных и операторы, вы сможете писать простые программы на Python, выполняющие полезные задачи. В следующем разделе рассмотрим управляющие структуры, которые позволяют управлять работой программы на основе условий и циклов.

Запуск программ на Python

После написания программы на Python ее нужно запустить, чтобы увидеть результаты. В этом разделе рассмотрим способы запуска программ.

Самый простой способ — использовать интерпретатор Python, о чем говорилось в предыдущем разделе. Откройте окно командной строки или терминала, перейдите в каталог, где сохранена программа на Python, и введите `python program.py`

(замените `program.py` именем файла вашей программы). Интерпретатор Python выполнит код, содержащийся в файле, и выведет результат.

Другой способ — использовать интегрированную среду разработки (IDE), например PyCharm или Visual Studio Code. IDE предоставляет более совершенную среду для написания, отладки и тестирования кода. Чтобы запустить программу в IDE, откройте файл программы и воспользуйтесь командой IDE `run` или `execute`.

Если вы пишете программу для веб-приложения или сервера, может понадобиться запустить программу с помощью веб-сервера, например Apache или Nginx. В этом случае для обработки запросов и ответов от веб-сервера обычно используется веб-фреймворк, например Django или Flask.

Независимо от того, как вы решите запустить свою программу, важно тщательно протестировать ее и убедиться, что она работает так, как ожидается. Это включает в себя тестирование пограничных состояний, обработку ошибок и исключений, а также проверку входных и выходных данных.

Основные методы отладки

Отладка — важная часть программирования, поскольку позволяет находить и исправлять ошибки в коде. В этом разделе рассмотрим некоторые основные методы отладки, которые можно использовать для выявления и устранения проблем в программах на Python.

Первый шаг в отладке — понять суть проблемы. Часто это включает в себя изучение выходных данных программы и поиск ошибок или неожиданного поведения. Можно использовать ведение журналов и операторы `print`, чтобы вывести информацию о состоянии программы в разных местах кода.

Следующий шаг после определения проблемы — изолировать ее причину. Для этого часто используется процесс исключения, чтобы определить части кода, вызывающие проблему. Можно использовать точки останова и пошаговую отладку, чтобы построчно изучить код и увидеть, как изменяются переменные и значения.

Помимо изучения кода, можно использовать сообщения об ошибках и трассировку стека, чтобы точно определить местоположение проблемы. Сообщения об ошибках предоставляют информацию о типе и месте ошибки, а трассировка стека показывает последовательность вызовов функций, которые привели к ошибке.

После выявления причины проблемы ее нужно устранить. Для этого вносят изменения в код и снова тестируют программу, чтобы убедиться, что проблема решена. Можно использовать инструменты автоматизированного тестирования

и код-ревью, чтобы убедиться, что изменения не приведут к появлению новых проблем.

Стиль кода Python и лучшие практики

Стиль кода и лучшие практики важны для написания читабельного, удобного и эффективного кода Python. В этом разделе рассмотрим ключевые принципы стиля и лучшие практики.

Во-первых, при написании кода на Python важно придерживаться последовательного стиля. PEP 8 — официальное руководство по стилю Python — содержит рекомендации по оформлению кода, отступам, соглашениям об именовании и т. д. Придерживаясь единого стиля, вы можете писать простой в чтении и понимании код.

Важно писать модульный и переиспользуемый код. Это предполагает разбиение кода на небольшие, независимые функции или модули, которые можно использовать в разных частях программы. Так вы облегчите тестирование и отладку кода, а также снизите риск ошибок.

Используйте описательные имена переменных и комментарии для пояснения кода. Это облегчает другим разработчикам понимание вашего кода и снижает риск путаницы или ошибок.

Тщательно тестируйте код, чтобы убедиться, что все работает так, как ожидается. Это включает использование инструментов автоматизированного тестирования и написание юнит-тестов для проверки отдельных функций или модулей.

Наконец, следите за последними возможностями Python и лучшими практиками. Сообщество Python постоянно развивается, регулярно выпускаются новые функции и инструменты. Оставаясь в курсе последних событий, вы сможете писать более эффективный и действенный код.

Ресурсы для изучения Python

Python — популярный и широко используемый язык программирования, есть множество ресурсов для его изучения и освоения. Рассмотрим некоторые из лучших ресурсов для изучения Python: от онлайн-туториалов и курсов до книг и документации.

Онлайн-туториалы и курсы — отличный способ начать изучение Python. Они обеспечивают структурированную среду обучения и часто включают интерактивные примеры и упражнения. Популярные онлайн-ресурсы для изучения Python — Codecademy, Coursera, Udemy и edX.

Книги — это тоже отличный вариант, поскольку они дают всестороннее и глубокое представление о языке и его возможностях. Вот некоторые популярные книги для изучения Python: *Python Crash Course*[1] Эрика Мэтиза, *Python for Everybody* Чарльза Северанса и *Learning Python* Марка Лутца.

Документация — еще один важный ресурс для изучения, поскольку в ней содержится подробная информация о языке и его возможностях. Официальная документация Python доступна на сайте docs.python.org, и это всеобъемлющий ресурс для разработчиков Python всех уровней.

Наконец, сообщество Python — это ценный ресурс для освоения языка. Есть множество групп пользователей Python и онлайн-форумов, где разработчики общаются друг с другом, делятся знаниями и опытом, а также получают помощь по конкретным вопросам.

[1] *Мэтиз Э.* Изучаем Python. Программирование игр, визуализация данных, веб-приложения. — СПб.: Питер, 2016.