Первые приложения. Базовый синтаксис и “Hello, World!”

Язык программирования python считается одним из наиболее простых в изучении языков программирования. Его использование позволяет использовать как функциональный, процедурный, так и объектно-ориентированные подходы к программированию. В языке поддерживаются высокоуровневые типы данных наряду с возможностью реализацией объектного подхода, что делает язык мощным инструментом для создания различных кроссплатформенных приложений.

Для того, чтобы начать создавать первые приложения на языке python, изучим базовые конструкции синтаксиса, необходимые для написания любых приложений.

Перед началом обучение введем условные обозначения:

>> - таким символом обозначается код, вводимый в IDE (Integrated development environment, интегрированная среда разработки, например, Spyder, Jupiter, Visual studio и тд).

ПРИВЕТ! - белые буквы на черном фоне обозначают результат вывода в консоль (специальная утилита, обеспечивающая поддержку прямой связи пользователя с ОС)

В рамках данного урока будут рассмотрены такие конструкции, как:

- Наиболее часто используемые типы данных (строки и числа)
- Функция print (вывод в консоль)
- Переменные
- Операции числами
- Комментарии
- Виды ошибок

В рамках данных работ будут написаны первые приложения на изучаемом языке Python.

Наиболее часто используемые типы данных

Строки

Строка – это текстовое сообщение, которое возможно обрабатывать с использованием встроенных инструментов языков программирования. В python строки могут быть заданы несколькими способами:

- Они могут быть заключены в одинарные кавычки ('…')
- Они могут быть заключены в двойные кавычки («…»)

Оба способа абсолютно идентичны и можно использовать любой из них. Однако помните, что в начале и конце вводимой строки должны быть одинаковые кавычки!

Строкам соответствует тип данных String.

Числа

Числа в python задаются стандартно, как и во всех других языках программирования. Числа бывают двух типов: целые и дробные (или числа с плавающей запятой). Тип данных, соответствующий целым числам – integer (int). Данный тип содержит как отрицательные, так и положительные целые числа (например, -1, -2, 2, 3) и 0. При записи такого числа не допустимо использовать мантиссу (дробную часть). Такие числа используются при подсчете предметов, нумерации строк и т.д.

Числа с плавающей запятой, напротив, существуют для записи дробных чисел (чисел с плавающей запятой). Для этих целей наиболее часто в python используется тип данных float. Для задач, требующих большей точности измерений и вычислений используется тип данных double.

Числа с плавающей запятой используются для измерения относительных величин, статистических показателей и т.д. Например для расчета среднего роста учеников класса.

Функция print

Для общения с компьютером необходим инструмент, который бы выводил сообщения о своем текущем состоянии или операциях, которые он выполняет. С этой целью в python встроена функция print. Данная функция выводит строковые сообщения. Для задания строки, которую необходимо вывести, в скобках пишется текст сообщения в форме строки python (в кавычках):

```
>>print(“Прикоснись к прекрасному, обними программиста!”)
```

В данном фрагменте кода необходимо вывести в консоль выдержку из книги. Результатом исполнения данного кода будет следующее сообщение:

```
Прикоснись к прекрасному, обними программиста!
```

Переменные

Выше было рассмотрено, как можно ввести конкретное значение в python среду и вывести в консоль. Такие значения называются **литералами**. Например,

```
Прикоснись к прекрасному, обними программиста!

2
```

В данном случае были просто заданы строка и число, однако, они нигде не сохранены. Для возможности повторного использования этих значений необходимо выделить место в памяти компьютера, где будут хранится эти данные для дальнейшего использования. Для обращения к этим значениям выделяется некая область памяти компьютера, которой должно быть присвоено имя.

Для этих целей в языках программирования существуют переменные. Для задания переменных необходимо ввести имя переменной, знак равенства (=) и значение, которое туда необходимо поместить.

Например:

```
>>message_string = "Приветик"
>># выведет "Приветик"
>>print(message_string)
```

В данном случае в переменной message_string сохранена строка, которая выводится в консоль посредством обращения к переменной.

На самом деле, В Python переменные – это символическое имя, которое является ссылкой или указателем на объект (область памяти).

Итак, если есть запись:

>>a=50

Это означает, что в определенном адресе ячейки записано значение 50, и мы можем получить доступ к данному числу с использованием имени a, то есть a ссылается на значение 50:

Рассмотрим другой случай. Например,

>>a=50

>>b=a

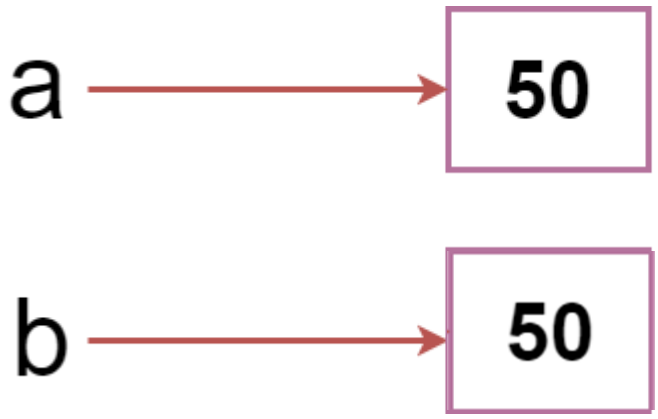
В этом случае, и a, и b ссылаются на одну и ту же область память (на объект – число 50). То есть:

Но, если мы запишем так:

>>a=50

>>b=50

В этом случае, a и b будут ссылаться на разные объекты.



Названия переменных

Название переменных не может содержать пробелов или символов кроме подчеркивания (_). Имена переменных не могут начинаться с чисел, но у них могут быть числа после первого символа (например, cool_variable_5 приемлемо). Имя переменной не может содержать специальные символы (!, @, #,%, ^, &, *). Имена переменных не должны совпадать с зарезервированными словами python, например, не допустимо называть переменные print, type, def и т.д.. Так же необходимо запомнить, что python чувствителен к регистру, то есть My_var и my_var – это разные переменные!

При назначении имени переменной необходимо подбирать имя, отражающее ее назначение. Например, для переменной, содержащей возраст, подойдет имя age. Python поддерживает кодировку utf-8, в связи с чем имена могут содержать кириллические символы и это не вызовет ошибок. Однако, по стандартам чистого кода, кириллицу в названиях объектов в программе не рекомендуется.

При присвоении имени переменной, содержащей несколько слов, допустимо использовать несколько общепринятых подходов:

- Camel Case – каждое слово или сокращение в середине начинается с заглавной буквы, пробелов нет. Например – nameOfStudent, valueOfVaraible и т. д.
- Pascal Case – это то же самое, что и Camel Case, но здесь первое слово заглавное. Например – NameOfStudent и т. д.
- Snake Case – слова разделяются подчеркиванием. Например – name_of_student и т. д.

Значения в переменных можно модифицировать в процессе выполнения программы, что объясняет происхождение названия данной конструкции языка программирования.

```
>># Приветствие
>>message_string = "Приветик"
>>print(message_string)
>># Обновляем значение переменной
>>message_string = "Hasta la vista"
>>print(message_string)
```

В данном примере в начале в переменную message_string введен текст приветствия, который затем выведен в консоль. После этого значение переменной заменено на прощание, а затем также выведено в консоль.

Задание

1. Создадим приложения, с которого начинается изучение всех классических языков программирования. На первом этапе выведите в консоль приветствие Hello, world!
2. Создайте переменную greeting и задайте ей значение Hello, world!
3. Выведите значение переменной greeting. Поздравляем! Вы создали свое первое приложение в python

Форматирование строк при выводе в консоль

Иногда, необходимо вывести какую-то строку одновременно со значением переменной. Например, есть переменная:

>>name = ‘Андрей’

Мы хотим вывести строку в консоль

Привет, Андрей!

В данном случае значение ‘Андрей’ содержится в переменной name, которую мы хотим вывести внутри строки. Для решения данной проблемы возможно 2 подхода:

1. Сложение строк (конкатенация)

Строки, как и числа, возможно складывать. Необходимо иметь в виду, что строку можно сложить только со строкой! Для сложения строк используется оператор +.

>> name = ‘Андрей’

>>print(‘Привет, ‘+name+’!’)

Данная команда позволит нам вывести желаемый результат.

2. Форматирование строк.

Для вывода переменной в строке можно использовать ключевое слово format. Для вывода желаемой нами строки будет использована следующая команда:

>> name = ‘Андрей’

>>print(‘Привет, {}!‘.format(name))

Операции числами

В рамках языка python можно выполнять арифметические действия над числами такие, как сложение, вычитание, деление и умножение (+, -, /, *).

```
>>print(573 - 74 + 1)
>>print(25 * 2)
>>print(10 / 5)
```

При выполнении последней операции в качестве результата будет получено число с плавающей запятой, так как деление преобразует целый тип данных integer в тип данных float (числа с плавающей запятой).

В результате деления на ноль будет выдана ошибка ZeroDivisionError.

Комментарии

Одним из наиболее полезных инструментов python являются комментарии. Они позволяют сделать код более читабельным и понятным. Для создания комментария необходимо ввести знак #. Все, что будет введено в строке после знака #, будет распознано, как комментарий, и компилятор будет игнорировать данные в этой строке при исполнении программы.

Комментарии могут:

• Обеспечить контекст, позволяющий понять, почему программа написана именно таким образом (например, для чего вводится дополнительная переменная и т.д.):

```
>># Эта переменная содержит информацию по количеству просмотров курса
>>viewers_count = 0
```

• Помочь другим программистам прочитать Ваш код:

```
>># Данная переменная содержит информацию о сотрудниках
>>Personal_count = 10
```

• Игнорировать часть кода при отладке приложения, чтобы найти, в какой строке происходит ошибка

```
>># value_for_test = 0
    >>New_value_for_test = 1
```

Ошибки

В языках программирования существует возможность нахождения ошибок кода, а также их разъяснения.

Python называет эти ошибки «Errors» и указывает на местоположение ошибки символом ^.

Существует 2 вида возможных ошибок в python:

- `SyntaxError –` означает, что что-то не так в тексте самой программы (синтаксисе) — например, несоответствующая пунктуация, которая не должна быть при данной операции. Также недостающая круглая скобка может привести к такого рода ошибке.
- `NameError –` происходит, когда интерпретатор Python не может распознать слово. Например переменную, которой не существует.

Задание

1. Необходимо расчитать, сколько плитки понадобится для проведения ремонта в комнате! Создать переменные length и width и присвоить в них значения 8 и 10 соответственно.
2. Оказалось, что произошла ошибка в расчетах и в длину необходимо 20 плиток.. Какое количество плитки понадобится в этом случае?

Самостоятельно! Рассчитать площадь прямоугольника со сторонами 23, 13.