Описание языка Python

Язык программирования Python (пайтон) был создан Гвидо ван Россумом (Guido van Rossum) в начале 90-х.

Python – высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью. Python является интерпретируемым, объектно-ориентированным языком программирования. Он прост, гибок, выразителен и содержит небольшое число ключевых слов.

Далее приводится описание ***некоторых*** конструкций языка с помощью РБНФ.

Управляющие конструкции

if <условие1>**:** <оператор1> [**elif** <условие2>**:** <оператор2>]* [**else:** <оператор3>]	Оператор "если". Часть в квадратных скобках является необязательной. Следующий за скобками символ "*" означает, что заключенная в скобки часть может быть записана неоднократно одна за другой. При истинности <условия1> будет выполнен <оператор1> и проигнорированы ветки elif и else. В противном случае, если истинно <условие2>, то выполняется <оператор2>, ветка else игнорируется. Иначе выполняется <оператор3>.

while <условие>**:** <оператор1> [**else:** <оператор2>]	Цикл "пока". <Оператор1> будет выполняться все время, пока истинно <условие>. При нормальном завершении цикла, т.е. без применения **break**, выполнится <оператор2>.

for <переменная> **in** <список>**:** <оператор1> [**else:** <оператор2>]	Цикл "для". <Переменная> пробегает все элементы <списка> и для каждого текущего значения <переменной> выполняется <оператор1>. При нормальном завершении цикла, т.е. без применения **break**, выполнится <оператор2>.

return [<результат>]	Осуществляет возврат из функции или метода класса, возвращая значение <результат>.

Объявление функций

def <имя_функции> **(** [<список_параметров>] **):** <тело_функции>	Здесь <тело_функции> - последовательность операторов, выровненных по тексту правее слова "def". <список_параметров> в самом общем виде выглядит так: [< id > [,< id >]*] [< id >=< v > [,< id >=< v >]*] [, *< id >] Здесь < id > - идентификатор переменной; < v > - некое значение. Параметры < id > за которыми следует "=" получают значения < v > по умолчанию. Если список заканчивается строкой " *< id > ", то id присваивается тьюпл (tuple) из всех оставшихся аргументов, переданных функции.

Объявление классов

class <имя_класса> [(<предок1> [,<предок2>]*)]**:** <тело_класса>	Здесь <тело_класса> может содержать присваивания переменным (эти переменные становятся атрибутами, т.е. полями класса) и определения функций (являющихся методами класса). Первым аргументом метода всегда является экземпляр класса который вызывает данный метод (или, говоря иначе, к которому применяется метод). По соглашению, этот аргумент называется "self". Специальный метод __init__() вызывается автоматически при создании экземпляра класса.

Операторы для всех типов последовательностей (списки, тьюплы, строки)

len (s)	возвращает длину s.
min (s), max (s)	наименьший и наибольший элементы s соответственно.
x in s	истина (**1**), если s включает в себя элемент равный x, иначе - ложь (**0**).
x not in s	ложь, если s включает x, иначе истина.
s + t	слияние s и t.
s * n , n * s	n копий s, слитых вместе (например, ‘*’ * 5 - это строка ‘*****’).
s[i]	i-тый элемент s, где i отсчитывается с 0.

Файловые объекты

f.close ()	закрыть файл.
f.read ([size])	читает < size > байт из файла и возвращает в виде строки. Если < size > отсутствует, то читает до конца файла.
f.readline ()	читает целиком одну строку из файла.
f.write (str)	записывает строку < str > в файл.

Другие элементы языка и встроенные функции

=	присваивание.
print [< c1 > [,< c2 >]* [,]]	выводит значения < c1 >, < c2 > в стандартный вывод. Ставит пробел между аргументами. Если запятая в конце перечня аргументов отсутствует, то осуществляет переход на новую строку.
abs (x)	возвращает абсолютное значение x.
chr (i)	возвращает односимвольную строку с ASCII кодом i.
cmp (x, y)	возвращает отрицательное, ноль, или положительное значение, если, соответственно, x <, ==, или > чем y.
float (x)	возвращает вещественное значение равное числу x.
hex (x)	возвращает строку, содержащую шестнадцатеричное представление числа x.
input (<строка>**)**	выводит <строку>, считывает и возвращает значение со стандартного ввода.
oct (x)	возвращает строку, содержащую представление числа x.
open (<имя файла>**,** <режим>=‘r’**)**	возвращает файловый объект, открытый для чтения. <режим> = ‘w’ - открытие для записи.
ord (c)	возвращает ASCII код символа (строки длины 1) c.
str (<объект>**)**	возвращает строковое представление <объекта>.
type (<объект>**)**	возвращает тип объекта. Например: if type(x) == type(‘’): print ‘ это строка ’

Импортирование модулей

import <модуль1> [**,** <модуль2>]*	подключает внешние модули.
from <модуль> **import ***	импортирует все имена из <модуля>, за исключением начинающихся символом "_".