# Python

примеры использования синтаксиса

|  |  |  |
| --- | --- | --- |
| **Функция** | **Описание** | **Пример** |
| list**.index**( needle ) | Возвращает индекс первого встретившегося элемента списка. Если не находит, возникает ошибка | animals = ["aardvark", "badger", "duck", "emu", "fennec fox"]  duck\_index = animals.index("duck") #возвращает 2 |
| my\_dict**.items()** | Выводит массив кортежей словаря. | my\_dict = {  "Surname":"Leony",  "Language":"Python"  }  print my\_dict.items()  # [('Surname', 'Leony'), ('Language', 'Python')] |
| My\_dict**.keys()** | Возвращает массив ключей словаря | ['Surname', 'Language'] |
| My\_dict**.values()** | Возвращает массив значений словаря | ['Leony', 'Python'] |
| list**.insert** ( index, ‘elem’) | Вставить элемент в список на указанный индекс. Текущие элементы списка сдвигаются вправо. Список изменяется. | # Your code here  animals.insert(1, "panther")  #возвращает: animals = ["aardvark", "panther", "badger", "duck", "emu", "fennec fox"] |
| list**.count**(item) | Возвращает число раз, которые число встречается в списке | list = [1,2,1,1,3]  print list.count(1)  #3 |
| list**.sort()** | Сортирует список по возрастанию. Список изменяется | list = [5, 3, 4]  list.sort()  print list #возвращает [3,4,5] |
| **for** elem **in** [1,2,3,4] | Циклом проходим по элементам списка | my\_list = [1,9,3,8,5,7]  for number in my\_list:  # Your code here  print number\*2 |
| **for** key **in** {'math' : 'intellect',  'history' : 'memory'} | Циклом проходим по ключам словаря | dict = {'math' : 'intellect',  'history' : 'memory'}  for key in dict:  print dict[key] |
| **del** dict['key'] | Удаляет элемент словаря по *ключу*. | dict = {'math' : 'intellect',  'history' : 'memory'}  del dict['math']  print dict  # возвращает {'history' : 'memory'} |
| list**.remove**( 'value' ) | Удаляет элемент списка по значению | list = ['a','b','c']  list.remove('a')  print list  # ['b','c'] |
| **round**( value ) | Округляет число до целого | print round(11.6)  # 12 |
| **sum**(iterable[, start]) | Суммирует числа | print sum([1,2,3])  #6 |
| " ".**join**(list) | Соединяет элементы списка указанным символом | list = ['a','b','c']  print "-".join(list)  #a-b-c |
| random.**randint**(x, y) | Генерирует случайное число от х до y | import random  print random.randint(0,6)  #6 |
| while / else | else выполняется, если цикл успешно заканчивает все итерации или если цикл ни разу не был запущен.  else не будет выполняться, если из while вышли при помощи оператора break | count = 0  while count < 3:  num = randrange(1, 7)  print num  count += 1  if num == 5:  print "Sorry, you lose!"  break  else:  print "You win!" |
| for / else | else выполняется, если цикл успешно заканчивает все итерации.  else не будет выполняться, если из for вышли при помощи оператора break | fruits = ['banana', 'apple', 'orange', 'tomato', 'pear', 'grape']  for f in fruits:  if f == 'tomato':  print 'A tomato is not a fruit!'  break  print 'a ', f  else:  print 'A fine selection of fruits!'  #Выведет:  You have...  a banana  a apple  a orange  A tomato is not a fruit! |
|  |  |  |
| for *index*, *value* in **enumerate**(list) | В списке дает доступ к индексу | choices = ['pizza', 'pasta', 'salad', 'nachos']  for index, item in **enumerate**(choices):  print index+1, item  #Выведет:  1 pizza  2 pasta  3 salad  4 nachos |
| for a, b in **zip**(list\_a, list\_b) | Позволяет просканировать два списка. Сканирование заканчивается, когда закончится самый короткий список | list\_a = [1,2,3,5,6]  list\_b = [5,4,5,5]  for a, b in **zip**(list\_a, list\_b):  if a > b:  print a  else:  print b  #Выведет:  5  4  5  5 |
| **isinstance**( -123, ( int, long ) ) | Показывает, является ли переменная целым числом | print isinstance( -123, ( int, long ) )  #Вернет true |
| math**.floor**(num) | Округляем в меньшую сторону до целого | import math  v = 8.8  print math.floor(v)  # 8.0 |
| Округляем в меньшую сторону до сотых | num = 8.3566666  Print math.floor(num\*100)/100  #8.35 |
| math**.ceil**(num) | Округляем в большую сторону до целых | import math  v = 8.8  print math.ceil(v)  # 9.0 |
|  | Округляем в меньшую сторону до сотых | num = 8.3566666  Print math.ceil(num\*100)/100  #8.36 |
| 'string'**[::-1]** | Это расширенный слайс [begin:end:step] | >>> 'hello world'[::-1] 'dlrow olleh' |
| print ''.join(**reversed**('Nikita')) | Строчка наоборот. |  |
|  |  |  |
| for i in **reversed**(xrange(1,10,2)): | Выводит список в обратном порядке | for i in **reversed**(xrange(1,10,2)):  print i  #Вывод:  9  7  5  3  1 |
| str**.find**(needle) | Найти подстроку в строке | x = "Hello World" x.find('World') #6  x.find('Aloha'); #-1 |
| str**.replace**(old, new[, max]) | Заменяет старую строку на новую.  max – наибольшее число замен, начиная с начала. Если опущен, замена идет везде. | print "I like you!".replace('like', 'love')  # I love you! |
| **\_\_name\_\_** | Если равен \_\_main\_\_ , значит этот скрипт – точка входа программы | if \_\_name\_\_ == '\_\_main\_\_':  pass |
| \_\_repr\_\_(self) | Репрезентация объекта класса | class Point3D(object):  def \_\_init\_\_(self, x, y, z):  self.x = x  self.y = y  self.z = z    def \_\_repr\_\_(self):  return "(%s, %s, %s)" % (self.x, self.y, self.z)    myPoint = Point3D(1, 2, 3)  print myPoint |

class Animal(object):

"""Makes cute animals."""

is\_alive = True

health = 'good'

def \_\_init\_\_(self, name, age):

self.name = name

self.age = age

def description(self):

print self.name, self.age

# Add your method here!

hippo = Animal('Monkey',21)

class ShoppingCart(object):

"""Creates shopping cart objects

for users of our fine website."""

items\_in\_cart = {}

def \_\_init\_\_(self, customer\_name):

self.customer\_name = customer\_name

def add\_item(self, product, price):

"""Add product to the cart."""

self.product = product

self.price = price

if not product in self.items\_in\_cart:

self.items\_in\_cart[product] = price

print product + " added."

else:

print product + " is already in the cart."

def remove\_item(self, product):

self.product = product

if product in self.items\_in\_cart:

del self.items\_in\_cart[product]

print product + " removed."

else:

print product + " is not in the cart."

my\_cart = ShoppingCart("Nikita")

my\_cart.add\_item("Book", 1000)

# Наследование в классах

Объект второго класса имеет метод, несмотря на то, что он не определен в нем.

class Customer(object):

"""Produces objects that represent customers."""

def \_\_init\_\_(self, customer\_id):

self.customer\_id = customer\_id

def display\_cart(self):

print "I'm a string that stands in for the contents of your shopping cart!"

class ReturningCustomer(Customer):

"""For customers of the repeat variety."""

def \_\_init\_\_(self, customer\_id):

self.customer\_id = customer\_id

def display\_order\_history(self):

print "I'm a string that stands in for your order history!"

monty\_python = ReturningCustomer("ID: 12345")

monty\_python.display\_cart()

monty\_python.display\_order\_history()

# Вызов метода родительского класса из дочернего

class DerivedClass(Base):

def some\_method(self):

**super**(DerivedClass, self).methodOfMainClass()

# Создание массива в одну строчку

evens\_to\_50 = [i for i in range(51) if i % 2 == 0]

print evens\_to\_50

[0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50]

even\_squares = [x\*\*2 for x in range(1,11) if (x\*\*2)%2 == 0]

print even\_squares

[4, 16, 36, 64, 100]

cubes\_by\_four = [x\*\*3 for x in range(1,11) if (x\*\*3)%4 == 0]

print cubes\_by\_four

[8, 64, 216, 512, 1000]