**Lambda-функций**

В Python лямбда-выражение позволяет создавать анонимные функции - функции, которые не привязаны к имени.

В анонимной функции:

* может содержаться только одно выражение
* могут передаваться сколько угодно аргументов

Стандартная функция:

In [1]: def sum\_arg(a, b): return a + b

In [2]: sum\_arg(1,2)

Out[2]: 3

Аналогичная анонимная функция, или лямбда-функция:

In [3]: sum\_arg = lambda a, b: a + b

In [4]: sum\_arg(1,2)

Out[4]: 3

Обратите внимание, что в определении лямбда-функции нет оператора return, так как в этой функции может быть только одно выражение, которое всегда возвращает значение и завершает работу функции.

Лямбда-функцию удобно использовать в выражениях, где требуется написать небольшую функцию для обработки данных.

Например, в функции sorted лямбда-выражение можно использовать для указания ключа для сортировки:

In [5]: list\_of\_tuples = [('IT\_VLAN', 320),

...: ('Mngmt\_VLAN', 99),

...: ('User\_VLAN', 1010),

...: ('DB\_VLAN', 11)]

In [6]: sorted(list\_of\_tuples, key=lambda x: x[1])

Out[6]: [('DB\_VLAN', 11), ('Mngmt\_VLAN', 99), ('IT\_VLAN', 320), ('User\_VLAN', 1010)]

Также лямбда-функция пригодится в функциях map и filter.

Вместе с map удобно использовать лямбда-выражения:

In [5]: vlans = [100, 110, 150, 200, 201, 202]

In [6]: list(map(lambda x: 'vlan {}'.format(x), vlans))

Out[6]: ['vlan 100', 'vlan 110', 'vlan 150', 'vlan 200', 'vlan 201', 'vlan 202']

Из списка чисел оставить только нечетные:

In [3]: list(filter(lambda x: x%2, [10, 111, 102, 213, 314, 515]))

Out[3]: [111, 213, 515]

Аналогично, только четные:

In [4]: list(filter(lambda x: not x%2, [10, 111, 102, 213, 314, 515]))

Out[4]: [10, 102, 314]

Из списка слов оставить только те, у которых количество букв больше двух:

In [5]: list\_of\_words = ['one', 'two', 'list', '', 'dict']

In [6]: list(filter(lambda x: len(x) > 2, list\_of\_words))

Out[6]: ['one', 'two', 'list', 'dict']