КАЛУЖСКИЙ ФИЛИАЛ
ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО
ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ИМЕНИ Н.Э. БАУМАНА
(национальный исследовательский университет)»

Факультет «Информатика и управление»

Кафедра «Программное обеспечение ЭВМ, информационные технологии»

Высокоуровневое программирование

Лекция №11. «Функциональное программирование в Python»

Калуга - 2020

Map

- Принимает функцию и аргумент составного типа данных. Применяет переданную функцию к каждому элементу.
- Синтаксис:
  list(map(функция, последовательность))

```
1 lst = ['1', '2', '3', '4', '5']
2 int_lst = list(map(int, lst))
3 print(int_lst)
```

```
[1, 2, 3, 4, 5]
```

```
1 sqr_lst = list(map(lambda x: x**2, int_lst))
2 print(sqr_lst)
```

```
[1, 4, 9, 16, 25]
```

Filter

- Функция фильтрации составного типа данных по признаку передаваемой функции.
- Синтаксис:
  list(filter(функция, последовательность))

```
1 fltr_lst = list(filter(lambda x: x>5, sqr_lst))
2 print(fltr_lst)
```

```
[9, 16, 25]
```

Zip

- Объединяет переданные последовательности в кортежи.
- Синтаксис:
  list(zip(последовательность_1, последовательность_2))
- Прекращает выполнение как только достигнут конец минимального списка.

```
1  zip_lst1 = list(zip(lst, sqr_lst))
2  print(zip_lst1)
3  zip_lst2 = list(zip(lst, fltr_lst))
4  print(zip_lst2)
```

```
[('1', 1), ('2', 4), ('3', 9), ('4', 16), ('5', 25)]
[('1', 9), ('2', 16), ('3', 25)]
```

Reduce

- В Python 3-х не рекомендуется к использованию и вычеркнута из стандартной библиотеки из-за чего требуется подключать модуль:
  - **import functools**
- И использовать синтаксис: **functools.reduce()**
- Её может заменить цикл for.

```
1  import functools
2  items = [1, 2, 3, 4]
3  sum = functools.reduce(lambda x, y: x+y, items)
4  print(sum)
```

```
10
```

Совместное использование функций filter() внутри map()

- Когда вы совместно друг с другом используете функции, то сначала исполняются внутренние функции, а затем внешние функции обрабатывают результат выполнения внутренних функций.

```
1  c = map(lambda x: x+x,filter(lambda x: (x>=3), (1,2,3,4)))
2  print(list(c))
```

```
[6, 8]
```

Использование map() внутри filter()

```
1 c = filter(lambda x: (x>=3),map(lambda x:x+x, (1,2,3,4)))
2 print(list(c))
```

```
[4, 6, 8]
```

Использование map() и filter() внутри reduce():

```
1 d = functools.reduce(lambda x,y: x+y,_
2                      map(lambda x:x+x,filter(lambda x: (x>=3), (1,2,3,4))))
3 print(d)
```

```
14
```

Примеры

```
1  my_list = [66, 77, 88, 99]
2  print(" ".join(map(str, my_list)))
```

```
66 77 88 99
```

```
1  lst = ["Москва", "Рязань", "Смоленск", "Тверь", "Томск"]
2  b = map(len, lst)
3  a = list(b)
4  print(a)
```

```
[6, 6, 8, 5, 5]
```

```
1  b = list(map(lambda x: x[::-1], lst))
2  print(b)
```

```
['авксоМ', 'ьназяР', 'кснеломС', 'ьревТ', 'ксмоТ']
```

Примеры

```
1 b = map(lambda x: x.replace("a", "A"), lst)
2 print(b)
3 c = map(sorted, b)
4 print(list(c))
```

```
<map object at 0x056313B8>
[['A', 'M', 'в', 'к', 'о', 'с'], ['A', 'Р', 'з', 'н', 'ь', 'я'],
'ь'], ['Т', 'к', 'м', 'о', 'с']]
```

```
1 a = list(map(int, input().split()))
2 print(a)
```

```
1 2 3 4 5 6 7
[1, 2, 3, 4, 5, 6, 7]
```

```
1  b = list(filter(lambda x: x%2, a))
2  print(b)
```

```
[1, 3, 5, 7]
```

Примеры

```
1  lst = ("Москва", "Рязань1", "Смоленск", "Тверь2", "Томск")
2  b = filter(str.isalpha, lst)
3
4  for x in b:
5      print(x, end=" ")
```

```
Москва Смоленск Томск
```

Задача:

Посчитайте сумму квадратов всех двузначных чисел, делящихся на 9.
При решении задачи используйте комбинацию функций filter, map, sum.
Примечание: на 9 должно делиться исходное двузначное число, а не его квадрат.

Подключение модуля из стандартной библиотеки

```
1 import random
2 random.random()
```

```
0.04415253858714807
```

- После импортирования модуля его название становится переменной, через которую можно получить доступ к атрибутам модуля. Например, можно обратиться к константе e, расположенной в модуле math:

```
1 import math
2 math.e
```

```
2.718281828459045
```

Использование псевдонимов

```
1 import math as m
2 m.e
```

```
2.718281828459045
```

- Теперь доступ ко всем атрибутам модуля **math** осуществляется только с помощью переменной m, а переменной **math** в этой программе уже не будет (если, конечно, вы после этого не напишете **import math**, тогда модуль будет доступен как под именем **m**, так и под именем **math**)

Инструкция from

- Подключить определенные атрибуты модуля можно с помощью инструкции from. Она имеет несколько форматов:

```
from <Название модуля> import <Атрибут 1> [ as <Псевдоним 1> ], [<Атрибу
from <Название модуля> import *
```

- Первый формат позволяет подключить из модуля только указанные вами атрибуты. Для длинных имен также можно назначить псевдоним, указав его после ключевого слова as.

```
1 from random import random as rnd
2 rnd()
```

```
0.2142304392165213
```

Инструкция from

- Второй формат инструкции **from** позволяет подключить все (точнее, почти все) переменные из модуля.
- Следует заметить, что не все атрибуты будут импортированы. Если в модуле определена переменная **__all__** (список атрибутов, которые могут быть подключены), то будут подключены только атрибуты из этого списка. Если переменная **__all__** не определена, то будут подключены все атрибуты, не начинающиеся с нижнего подчёркивания. Кроме того, необходимо учитывать, что импортирование всех атрибутов из модуля может нарушить пространство имен главной программы, так как переменные, имеющие одинаковые имена, будут перезаписаны.

Задачи для самостоятельного решения

- Даны n предложений. Определите, сколько из них содержат хотя бы одну цифру.
- Дана строка s и символ k. Реализуйте функцию, рисующую рамку из символа k вокруг данной строки, например:

```
**************
*Текст в рамке*
**************
```

- Для введенного предложения выведите статистику *символ=количество*. Регистр букв не учитывается.
- Дата характеризуется тремя натуральными числами: день, месяц и год. Учитывая, что год может быть високосным, реализуйте две функции, которые определяют вчерашнюю и завтрашнюю дату.

Задачи для самостоятельного решения

- Напишите функцию, которая принимает неограниченное количество числовых аргументов и возвращает кортеж из двух списков:
  - отрицательных значений (отсортирован по убыванию);
  - неотрицательных значений (отсортирован по возрастанию).
- Составьте две функции для возведения числа в степень: один из вариантов реализуйте в рекурсивном стиле.
- Дано натуральное число. Напишите рекурсивные функции для определения:
  - суммы цифр числа;
  - количества цифр в числе.