

Занятие 4

Словари

Что выведет код?

```
d = '4'
```

```
e = 'hi'
```

```
d, e = e, int(d)
```

```
print(d, e)
```

```
#####
```

```
lang = 'Python'
```

```
print(f"{'lang'} is the best!")
```

```
#####
```

```
print('All your need is love'.split())
```

```
print('All your need is love'.split(' '))
```

Задача 3-1

- Вводить в бесконечном цикле зарплаты сотрудников.
- Окончание ввода – ввод 0.
- После чего напечатать среднюю зарплату.

Задача 3-2

Дано целое число.

Сосчитать и напечатать, сколько в его записи нулей, единиц, двоек и т.д.

Например:

Ввод: 133244459

Вывод:

0 - 0

1 – 1

2 – 1

3 – 2

4 – 3

5 – 1

6 - 0

и т.д.

Задача 3-3

На вход подается предложение из нескольких слов.

Слова разделены пробелами.

Напечатайте первое самое длинное слово в этом предложении.

Более сложный вариант, напечатать все самые длинные слова,

если их несколько с наибольшей длиной

```
# print(len(max(['I', 'send', 'you', 'a', 'message'], key = len)))
```

Полезные функции строк

`ord('a')` – код символа 'a'

`chr(97)` – символ с кодом 97

`' 123 '.strip()` – с обеих сторон отбрасывает все пробелы

`lstrip()` – слева, `rstrip()` – справа

`str.strip([chars])`

`'www.example.com'.strip('cmowz.') # 'example'`

`str.startswith(prefix[, start[, end]])`

`'Hello world'.startswith('Hello')`

`str.endswith(suffix[, start[, end]])`

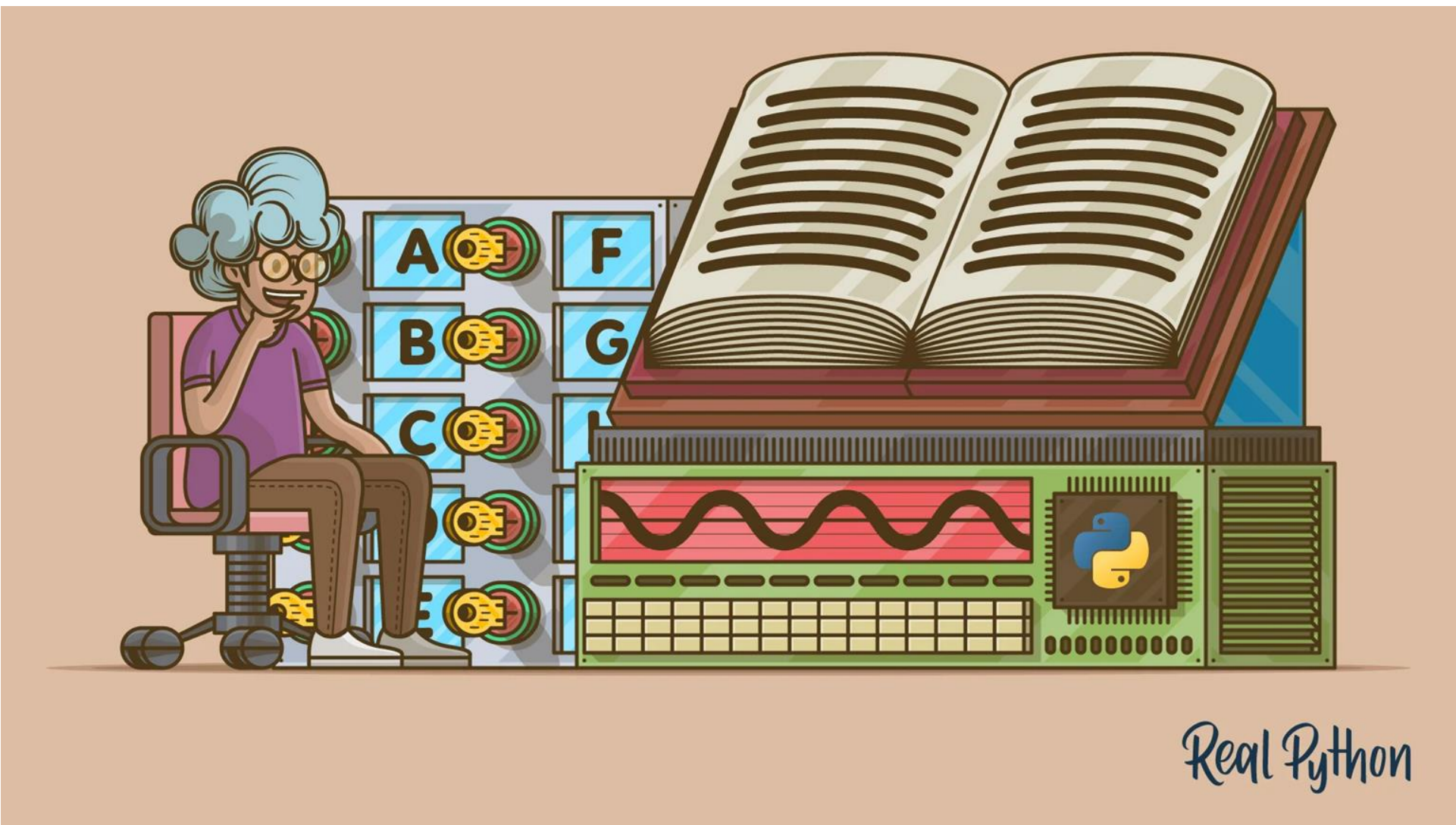
`'Hello world'.startswith('world')`

Коллекции

1. Строка (str) 'Hello world'
2. Список (list) [1, 100, 1, 'a', True]
3. Кортеж (tuple) (1, 100, 1, 'a', True)
4. **Словарь (dict) {1:1, 22:100, 123:1, 'a':'a', 5:True}**
5. Множество (set) {1, 100, 'a', True}

-

Словарь



Real Python

Определение словаря

```
dict = {k:v}
```

- Словарь задается парой **ключ:значение**, ключ – уникален!

- dic = {
- <key>: <value>,
- <key>: <value>,
- .
- .
- .
- <key>: <value>
- }

Пример 1:



```
person = {  
    'name': 'Маша',  
    'login': 'masha',  
    'age': 25,  
    'email': 'masha@yandex.ru',  
    'password': 'mmaasshhaa'  
}  
  
print(person)
```


Измените Машу на Мишу во всех элементах словаря, где это возможно

Например, person['name'] = 'Миша'

Напечатайте новый person

Доступ к элементу по ключу. Замена значения

dict = { k : v }



- >>> person['name']
- Маша
- # Замена значения
- >>> person['name'] = 'Даша'

Пример 2:

#Словарь, где ключи являются целыми числами.

```
dict_sample = {  
    1: 'mango',  
    2: 'coco'  
}
```

Напечатайте словарь в цикле:

```
for k in dict_sample:  
    print(k, dict_sample[k])
```

Список vs словарь

Можно ли определить по оператору `a[0] = 123` является `a` списком или словарем?

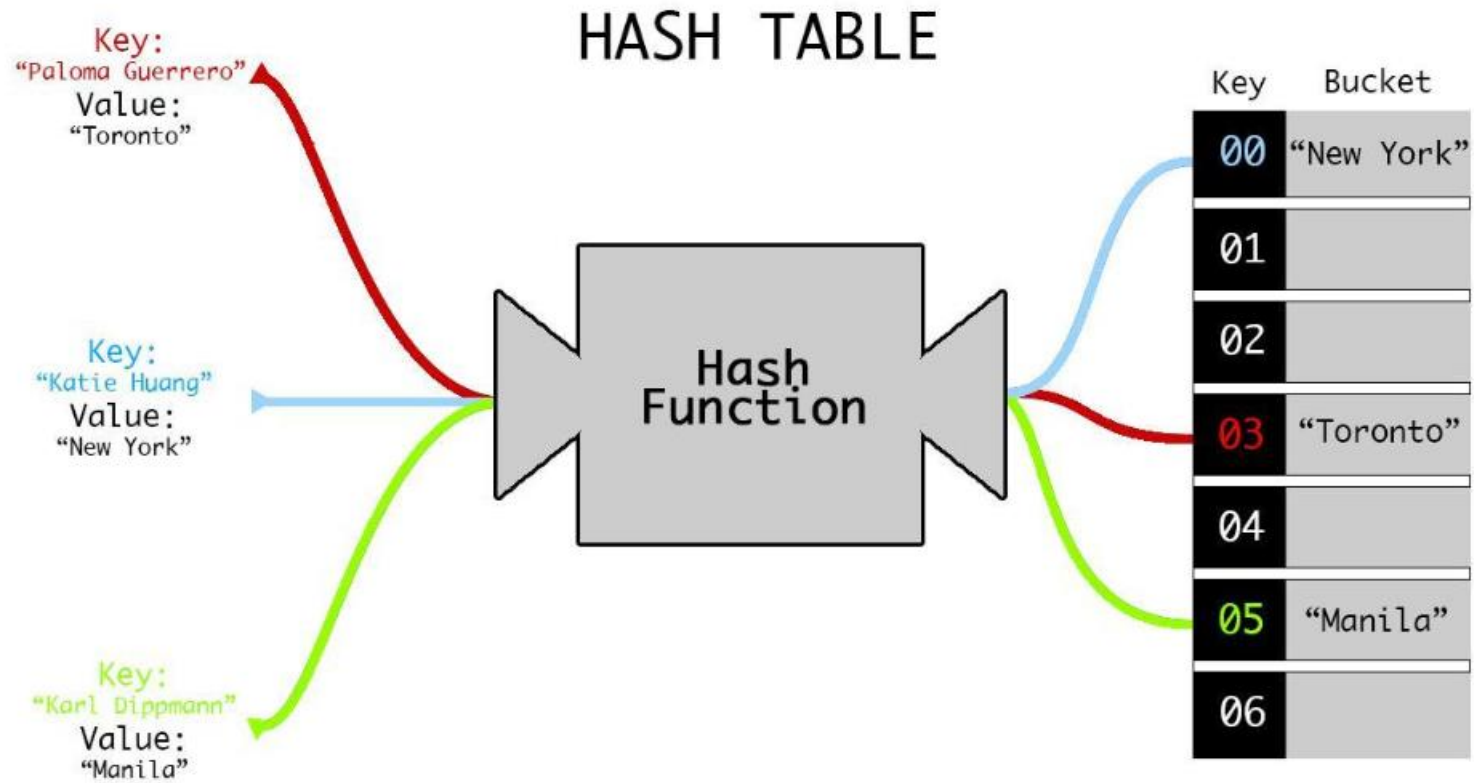
А как отличить?

Можно ли считать, что словарь – это развитие списка в каком-то смысле?

В списке индекс – это порядковый номер: от 0 до какого-то значения.

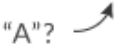
В словаре индекс – это уникальный идентификатор объекта, который преобразуется в некий искусственный «номер».

Хэш-таблицы – быстрый доступ



Проверка на наличие ключа

```
dict = { "A": v }
```



```
>>> 'name' in person  
True
```

```
if ('name' in person):  
    print('Ключ есть')  
else:  
    print('Ключа нет')
```

А что будет, если все-таки попытаться обратиться к ключу, которого нет?

```
d = {1:123}  
print(d[2])
```

Решение задачи про количество цифр в числе с помощью словаря и без count()

```
s = input()
```

```
dct = {}
```

```
for k in s:
```

```
    if k not in dct:
```

```
        dct[k] = 0
```

```
    dct[k] += 1
```

```
print(dct)
```

А как сделать, чтобы печатались по порядку?

Задание (по аналогии, без использования count())

Вводится строка букв, например: abracadabra

Составьте словарь, который для каждой буквы хранит количество ее вхождений в введенное слово.

Напечатайте словарь.

Пример 3:

```
dict_sample1 = {  
    True: 'mango',  
    False: 'coco'  
}
```

```
# Напечатайте этот словарь.
```

```
# Поменяйте порядок элементов:
```

```
dict_sample2 = {  
    False: 'coco'  
    True: 'mango',  
}
```

```
# Сравните их.
```

```
print(dict_sample1 == dict_sample2)
```

Пример 4:

```
# .. пойдём дальше
```

```
dict_sample = {  
    None: 'mango',  
    None: 'coco'  
}
```

```
# Напечатайте, что получилось. Почему?
```

Задание

Вводится число, например: 1231

Вывести строчку, например: “один два три один”

Подсказка: предварительно создайте словарь, где ключами являются цифры, а значениями являются слова их обозначающие.

Способы создания

```
{'name':'Маша','age': 16} # литеральным выражением
```

```
person = {} # пустой словарь
```

```
person['name'] = 'Маша' # динамическое присваивание по ключам
```

```
person['age'] = 16
```

```
dict(name='Маша', age=16) # через конструктор класса dict (позже узнаем детали)
```

```
letters = ['a', 'b', 'c', 'd']
```

```
pronans = ['эй', 'би', 'си', 'ди']
```

```
d = dict(zip(letters, pronans)) # используя функцию zip
```

```
# {'a':'эй', 'b':'би', 'c':'си', 'd':'ди'}
```

```
# Проверьте, что все эти способы работают
```

Задание

Создайте словарь: номер месяца -> количество дней в месяце.

После чего напишите программу, которая в бесконечном цикле вводит год и номер месяца и выводит количество дней в месяце.

Будем считать, что если год делится на 4 ($\text{year} \% 4 == 0$), то год високосный (в феврале будет 29 дней).

Выход из цикла: ввод двух нулей.

Добавление нового элемента

dict = { k: v, k2: v2 }



```
>>> person['surname'] = 'Медведева'
```

```
{  
    'name': 'Даша',  
    'login': 'masha',  
    'age': 25, 'email': 'masha@yandex.ru',  
    'password': 'fhei23jj~',  
    'surname': 'Медведева'  
}
```

Удаление элемента

dict = { k : v ,  }

- >>> del person['login']
- {
 - 'name': 'Даша',
 - 'age': 25,
 - 'email': 'masha@yandex.ru',
 - 'password': 'fhei23jj~',
 - 'surname': 'Медведьева'
- }

Длина словаря в Python

dict = { k : v , k2 : v2 }



len = 2

- Количество записей мы можем получить, воспользовавшись функцией len()
- >>> num_of_items = len(person)
- >>> print(num_of_items)
- >>> 5
-

Задание

Введите длинный текст с пробелами

Напечатайте наиболее часто встречающееся слово

Если таких слов несколько, то создайте список из этих слов и напечатайте его.

`get(key[, default])`

Метод `dict.get()` возвращает значение для ключа `key`, если ключ находится в словаре, если ключ отсутствует то вернет значение `default`.

Если значение `default` не задано и ключ `key` не найден, то метод вернет значение `None`.

Метод `dict.get()` никогда не вызывает исключение `KeyError`, как это происходит в операции получения значения словаря по ключу `dict[key]`.

```
x = {'one': 1, 'two': 2, 'three': 3, 'four': 4}
```

```
print(x.get('two', 0)) # 2
```

```
print(x.get('ten', 0)) # 0
```

```
print(x)
```

```
# Выполните этот код
```

`get` не меняет словарь!!!

Используем get для расчета количества цифр

```
s = input()
dct = {}
for k in s:
    dct[k] = dct.get(k, 0) + 1 # Тут и проверка, и присвоение 0.
print(dct)
```

Разберем, как это работает с помощью средств отладки в PyCharm

А что может стоять вместо 0 в операторе get? Что угодно – любое начальное значение для какого-то алгоритма.

Иногда пустой список [], тогда вместо + можно использовать .append

Иногда пустая строка "", если мы накапливаем символы.

Иногда может быть и пустой словарь {})).

Задание

Напишите программу, которая принимает на вход строку, и отслеживает, сколько раз каждый символ уже встречался.

Количество повторов добавляется к символам с помощью постфикса формата _n.

Пример ввода:

a a a b c a a d c d d

Пример вывода:

a_1 a_2 a_3 b_1 c_1 a_4 a_5 d_1 c_2 d_2 d_3

Задание

Ввод: 2 слова, разделенных пробелами.

Для ввода используем функцию `s = input().split()`

Определить, являются ли эти слова анаграммами (словами с одинаковым набором букв).

Если да, то `True`

Если нет, то `False`

(Примеры: АКВАРЕЛИСТ-КАВАЛЕРИСТ, АНТИМОНИЯ-АНТИНОМИЯ, АНАКОНДА-КАНОНАДА, ВЕРНОСТЬ-РЕВНОСТЬ, ВЛАДЕНИЕ-ДАВЛЕНИЕ, ЛЕПЕСТОК-ТЕЛЕСКОП)

Задание 4-1

- Напишите калькулятор (простой).
- На вход подается строка, например:
- $1 + 2$ или $5 - 3$ или $3 * 4$ или $10 / 2$.
- Вывод: сосчитать и напечатать результат операции.
- Гарантируется, что два операнда и операция есть в каждой строчке, и все они разделены пробелами.

Задача 4-2

- Вводим натуральное число n .
- Напечатайте спираль из чисел $1, 2, 3, \dots, n * n$
- Например для $n = 4$:

```
1  2  3  4
12 13 14 5
11 16 15 6
10 9  8  7
```

Можно использовать словарь с двумя индексами $d[x, y]$

Задача 4-3

Ввод: 2 предложения, содержащие пробелы, знаки препинания.

Определить, являются ли эти предложения анаграммами (т.е. имеют одинаковый набор букв).

Игнорируем пробелы, знаки препинания, цифры и т.д.

Вывод: Если да, то True, если нет, то False