

## Практическая работа № 14. Словари.

**Словари в Python** - неупорядоченные коллекции произвольных объектов с доступом по ключу. Их иногда ещё называют ассоциативными массивами или хеш-таблицами. Также можно сказать, что **словари** – это структура данных, позволяющая идентифицировать ее элементы не по числовому индексу, а по произвольному. Ещё словари называют ассоциативными массивами. Соответствующая структура данных в языке Питон называется **dict**.

### Создание словарей

Чтобы работать со словарями, их нужно создать. Сделать это можно несколькими способами

**Способ 1.** При помощи фигурных скобок

```
d = {} # в данном случае создаётся пустой словарь
```

**Способ 2.** При помощи фигурных скобок и перечисление пары вида **ключ: значение**

```
d = {'dict': 1, 'dictionary': 2} # создаётся словарь с двумя ключами
```

**Способ 3.** С помощью функции **dict**. Если данной функции не передавать никаких параметров, то создаётся пустой словарь

```
d = dict(short='dict', long='dictionary')
print(d) -> {'short': 'dict', 'long': 'dictionary'}
d = dict([(1, 1), (2, 4)])
print(d) -> {1: 1, 2: 4}
```

**Способ 4.** С помощью метода **fromkeys**. На вход метод получает два параметра: массив ключей и массив значений. Если не передать массив значений, то ключам будут ставиться значения **None**

```
d = dict.fromkeys(['a', 'b'])
print(d) -> {'a': None, 'b': None}
d = dict.fromkeys(['a', 'b'], 100)
print(d) -> {'a': 100, 'b': 100}
```

### Работа с элементами словаря

**Проверка наличия ключа в словаре**

```
1 l = {
2     'I': 1,
3     'V': 5,
4     'X': 10,
5     'L': 50,
6     'C': 100,
7     'D': 500,
8     'M': 1000
9 }
10 if 'A' in l:
11     print('YES')
12 else:
13     print('NO')
```

NO

## Вывод элемента словаря

**Способ 1.** На прямую обратиться по ключу

```
1 l = {
2     'I': 1,
3     'V': 5,
4     'X': 10,
5     'L': 50,
6     'C': 100,
7     'D': 500,
8     'M': 1000
9 }
10 print(l['X'])
```

10

**Способ 2.** При помощи метода `get`

```
1 l = {
2     'I': 1, 'V': 5, 'X': 10, 'L': 50, 'C': 100, 'D': 500, 'M': 1000
3 }
4 print(l.get('D'))
```

input

500

**Примечание:** если требуемый ключ отсутствует, то метод `get` вернёт значение `None`.

## Перебор словаря при помощи циклов

**Способ 1.** Перебор ключей и вывод значений

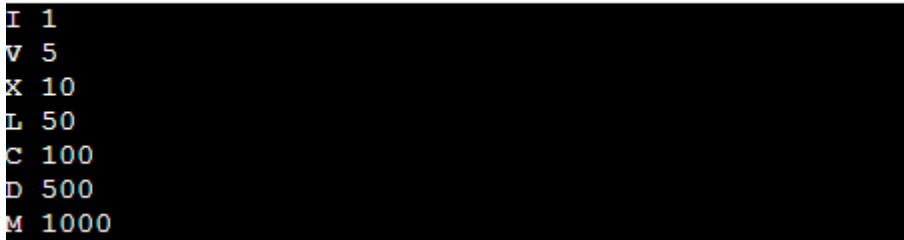
```
1 l = {
2     'I': 1, 'V': 5, 'X': 10, 'L': 50,
3     'C': 100, 'D': 500, 'M': 1000
4 }
5 for key in l:
6     print(key, l[key])
```

input

I 1  
V 5  
X 10  
L 50  
C 100  
D 500  
M 1000

## Способ 2. Перебор ключей и значений

```
1 l = {
2     'I': 1, 'V': 5, 'X': 10, 'L': 50,
3     'C': 100, 'D': 500, 'M': 1000
4 }
5 for key, val in l.items():
6     print(key, val)
```



## Методы словарей

- **dict.clear()** - очищает словарь.
- **dict.copy()** - возвращает копию словаря.
- **dict.fromkeys(seq[, value])** - создает словарь с ключами из **seq** и значением **value** (по умолчанию None).
- **dict.get(key[, default])** - возвращает значение ключа, но если его нет, не бросает исключение, а возвращает **default** (по умолчанию None).
- **dict.items()** - возвращает пары (ключ, значение).
- **dict.keys()** - возвращает ключи в словаре.
- **dict.pop(key[, default])** - удаляет ключ и возвращает значение. Если ключа нет, возвращает default (по умолчанию бросает исключение).
- **dict.popitem()** - удаляет и возвращает пару (ключ, значение). Если словарь пуст, бросает исключение `KeyError`. Помните, что словари неупорядочены.
- **dict.setdefault(key[, default])** - возвращает значение ключа, но если его нет, не бросает исключение, а создает ключ со значением **default** (по умолчанию None).
- **dict.update([other])** - обновляет словарь, добавляя пары (ключ, значение) из **other**. Существующие ключи перезаписываются. Возвращает None (не новый словарь!).
- **dict.values()** - возвращает значения в словаре.

## Задачи

### №1

В единственной строке записан текст. Для каждого слова из данного текста подсчитайте, сколько раз оно встречалось в этом тексте ранее.

#### Пример:

**Входные данные:** one two one tho three

**Выходные данные:**

one 2

two 1

tho 1

three 1

### № 2

Вам дан словарь, состоящий из пар слов. Каждое слово является синонимом к парному ему слову. Все слова в словаре различны. Для слова из словаря, записанного в последней строке, определите его синоним.

#### Пример:

**Входные данные:**

3

Hello Hi

Bye Goodbye

List Array

Goodbye

**Выходные данные:**

Bye

### №3

Дан текст: в первой строке задано число строк, далее идут сами строки. Выведите слово, которое в этом тексте встречается чаще всего. Если таких слов несколько, выведите то, которое меньше в лексикографическом порядке.

#### Пример:

**Входные данные:**

2

taia ikm ikm ikm taia taia taia

ikm ikm ikm

**Выходные данные:**

Ikm

## №4

Даны два списка одинаковой длины. Необходимо создать из них словарь таким образом, чтобы элементы первого списка были ключами, а элементы второго — соответственно значениями нашего словаря.

### Пример:

**Входные данные:**

```
['x', 'y', 'z']
```

```
[1, 2, 3]
```

**Выходные данные**

```
{ 'x': 1, 'y': 2, 'z': 3 }
```

## №5

Создайте словарь из строки, вводимой с клавиатуры, следующим образом: в качестве ключей возьмите буквы строки, а значениями пусть будут числа, соответствующие количеству вхождений данной буквы в строку.

### Пример:

**Входные данные:**

```
pythonпуру
```

**Выходные данные:**

```
{ 'p': 3, 'y': 3, 't': 1, 'h': 1, 'o': 1, 'n': 1 }
```

## №6

Напишите программу, которая создаёт словарь, значениями которого являются массивы, содержащие модели автомобилей, а ключами словаря будут марки автомобилей. На вход программа получает число  $n$  — количество автомобильных марок, затем с клавиатуры вводятся  $n$  строк следующего формата: 'марка модель1 модель2 ...модель $k$ '

### Пример:

**Входные данные:**

```
2
```

```
Renault logan duster arkana
```

```
Lada vesta xray granta priora kalina
```

**Выходные данные:**

```
{ 'Renault': ['logan', 'duster', 'arkana'],
```

```
  'Lada': ['vesta', 'xray', 'granta', 'priora', 'kalina'] }
```

## №7

Напишите функцию, которая принимает один словарь, и возвращает другой, в котором ключами являются значения из первого словаря, а значениями — соответствующие им ключи. Создайте словарь, передайте его в функцию. Выведите на экран исходный и "перевернутый" словари.

### Пример:

Входные данные:

{ 'x': 1, 'y': 2, 'z': 3 }

Выходные данные

Исходный список: { 'x': 1, 'y': 2, 'z': 3 }

Перевернутый список { 1: 'x', 2: 'y', 3: 'z' }

## №8\*

Создайте словарь, связав его с переменной **school**, и наполните данными, которые бы отражали количество учащихся в разных классах (1а, 1б, 2б, 6а, 7в и т. п.). Внесите изменения в словарь согласно следующему: а) в одном из классов изменилось количество учащихся, б) в школе появился новый класс, с) в школе был расформирован (удален) другой класс. Вычислите общее количество учащихся в школе.