**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

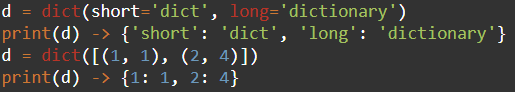
**Способ 1.** При помощи фигурных скобок



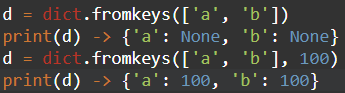
**Способ 2.** При помощи фигурных скобок и перечисление пары вида **ключ: значение**



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

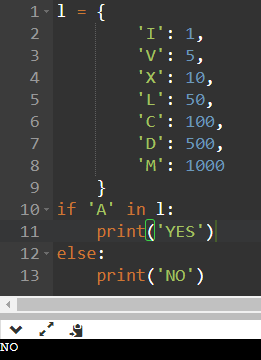


**Способ 4.** С помощью метода fromkeys. На вход метод получает два параметра: массив ключей и массив значений. Если не передать массив значений, то ключам будут ставиться значения None



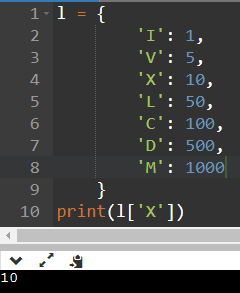
**Работа с элементами словаря**

**Проверка наличия ключа в словаре**

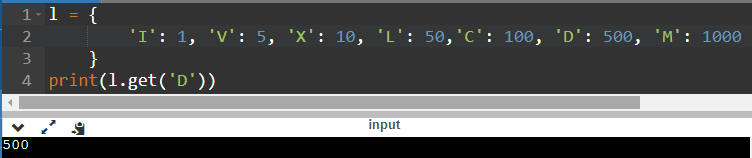


**Вывод элемента словаря**

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



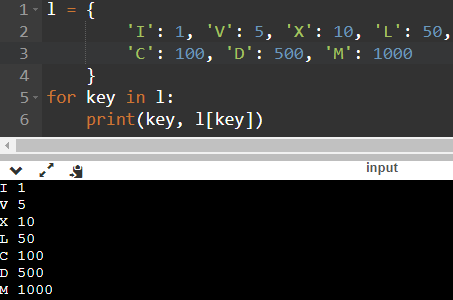
**Способ 2.** При помощи метода **get**



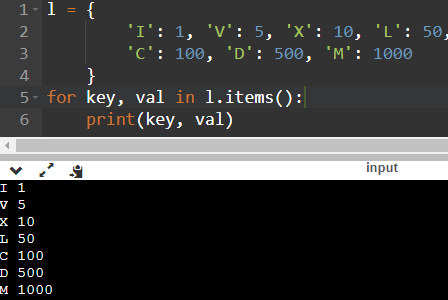
**Примечание:** если требуемый ключ отсутствует, то метод get вернёт значение None.

**Перебор словаря при помощи циклов**

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



**Способ 2.** Перебор ключей и значений



**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:**

pythonpypy

**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 …модельk’

**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}

Перевёрнутый 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.