**Словари (dict)**

Массив ставит в соответствие каждому числу (индексу, ключу) значение.

В ассоциативном массиве ключом может быть все что угодно (целые, действительные числа, строки).

Элементы в словаре неупорядочены.

**Пустой словарь**

d = {}

d = dict()

**Создание словарей**

Имя={ключ1:значение1, ключ2:значение2,…}

Month = {"January":1, "February":2, " March":3, "April":4, "May":5}

**Операции cо словарями**

**1. Получение значения по ключу**

d[key]

Обращение к элементу словаря выглядит как обращение к элементу списка, только вместо целочисленного индекса используется ключ. В качестве ключа можно указать выражение — Python вычислит его значение, прежде чем обратится к искомому элементу.

Если элемента с ключом key нет, то возвращается ошибкаl

x = d.get(key)

x = d.get(key, val)

Если элемента с ключом key нет, то возвращается значение val

Важная особенность словаря — его динамичность. Мы можем добавлять новые элементы, изменять их или удалять. Изменяются элементы точно так же, как в списках, только вместо целочисленного индекса в квадратных скобках указывается ключ:

**2. Добавление элемента к словарю**

Элементу нужно присвоить новое значение.

Month["September"] = 9

**3. Удаление элемента из словаря**

Удаление элемента

**del** d[key]

**del** Month["September"]

Удаляет элемент c индексом key и возвращает информацию о значении удаленного элемента

x = d.**pop**(key)

x = Month.**pop**("September") #x=9

Если элемента в словаре нет, то будет ошибка

Удаляет элемент c индексом key и возвращает информацию о значении удаленного элемента. Если элемента нет, то возвращается значение val

x = d.pop(key, val)

x = Month.pop("September", -1)

**4. Количество элементов в словаре**

**len**(D)

**5. Есть ли элемент в словаре**

Оператор **in** позволяет проверить, есть ли определённый ключ в словаре:

**if** key **in** d:

Проверить, что ключа нет, можно с помощью похожего оператора **not in**:

**6. Перебор элементов словаря по ключам**

**for** el **in** d:

 el ключ

 d[el] значение элемента

d.**keys**() – список ключей словаря

d.**keys**() – список ключей словаря

d.**values**() – список значений словаря

Если вы хотите перебрать элементы словаря так, чтобы в переменной key оказывался ключ, а в value — соответствующее ему значение, то это можно сделать с помощью метода .items() и цикла **for**.

**for** key, val **in** d.**items**():

Словарь это аналог телефонной книги. Ключ – фамилия, значение – телефон. Вы можете по фамилии найти телефон. Обратная операция по значению (номеру телефона) найти фамилию очень сложная. Ее сделать быстро нельзя, нужен перебор всех пар (ключ, значение)

**Задача**. Нам дан длинный список целых чисел numbers, причём некоторые числа встречаются в нём несколько раз. Узнать, сколько раз встречается каждое из чисел.

numbers = [1, 10, 1, 6, 4, 10, 4, 2, 2, 1, 10, 1]

counts = {}

**for** number **in** numbers:

 **if** number **not in** counts:

 counts[number] = 1

 **else:**

 counts[number] += 1

Можно написать проще

numbers = [1, 10, 1, 6, 4, 10, 4, 2, 2, 1, 10, 1]

counts = {}

**for** number **in** numbers:

 counts[number] = counts.**get**(number, 0) + 1

**Когда нужно использовать словарь?**

Словари нужно использовать в следующих случаях:

* Подсчет числа каких-то объектов. В этом случае нужно завести словарь, в котором ключами являются объекты, а значениями — их количество.
* Хранение каких-либо данных, связанных с объектом. Ключи — объекты, значения — связанные с ними данные. Например, если нужно по названию месяца определить его порядковый номер, то это можно сделать при помощи словаря Num["January"] = 1; Num["February"] = 2; ….
* Установка соответствия между объектами (например, “родитель—потомок”). Ключ — объект, значение — соответствующий ему объект.
* Если нужен обычный массив, но при этом максимальное значение индекса элемента очень велико, но при этом будут использоваться не все возможные индексы (так называемый “разреженный массив”), то можно использовать ассоциативный массив для экономии памяти.