

# Словари в Python

Занятие 1

# Что такое словари

**Словари** – изменяемые коллекции элементов с произвольными индексами – ключами. Если в списках элементы индексируются целыми числами, начиная с 0, то в словарях — любыми ключами, в том числе в виде строк.

# Создание словаря

Чтобы создать словарь, нужно перечислить его элементы, пары ключ—значение, через запятую в фигурных скобках, как и элементы множества. Первым указывается ключ, после двоеточия — значение, доступное в словаре по этому ключу.

```
languages = {'Python': 'Гвидо ван Россум',  
            'C#': 'Андерс Хейлсберг',  
            'Java': 'Джеймс Гослинг',  
            'C++': 'Бьёрн Страуструп' }
```

# Обращение к элементу словаря

Извлечь значение элемента словаря можно обратившись к нему по его ключу. Чтобы получить значение по заданному ключу, как и в списках, используем квадратные скобки.

```
languages = {'Python': 'Гвидо ван Россум',  
            'C#': 'Андерс Хейлсберг',  
            'Java': 'Джеймс Гослинг',  
            'C++': 'Бьёрн Страуструп' }  
print('Создателем языка C# является', languages['C#'])
```

# Создание словаря с помощью функции dict()

Для создания словаря можно использовать функцию dict().

Приведенный ниже код:

```
info = dict(name = 'Timur', age = 28, job = 'Teacher')
```

создает словарь с тремя элементами, ключами которого служат строки 'name', 'age', 'job' , а значениями: 'Timur', 28, 'Teacher'

# Создание словаря на основании списков и кортежей

Создавать словари можно на основе списков кортежей или кортежей списков. Первый элемент списка или кортежа станет ключом, второй — значением.

```
info_list = [('name', 'Timur'), ('age', 28), ('job', 'Teacher')] # список кортежей  
info_dict = dict(info_list) # создаем словарь на основе списка кортежей
```

```
info_tuple = (['name', 'Timur'], ['age', 28], ['job', 'Teacher']) # кортеж  
СПИСКОВ
```

```
info_dict = dict(info_tuple) # создаем словарь на основе кортежа  
СПИСКОВ
```

# Создание словаря на основании списков и кортежей

Если необходимо создать словарь, каждому ключу которого соответствует одно и то же значение, можно воспользоваться методом `fromkeys()`.

```
dict1 = dict.fromkeys(['name', 'age', 'job'], 'Missed information')
```

Если методу `fromkeys()` не передавать второй параметр, то по умолчанию присваивается значение `None()`.

```
dict1 = dict.fromkeys(['name', 'age', 'job'])
```

# Пустой словарь

**Пустой словарь** можно создать двумя способами:

- с помощью пустых фигурных скобок;
- с помощью функции `dict()`

```
dict1 = {}  
dict2 = dict()  
print(dict1)  
print(dict2)  
print(type(dict1))  
print(type(dict2))
```



# Для чего используют словари

Словари удобно использовать для хранения различных сущностей. Например, если нужно работать с информацией о человеке, то можно хранить все необходимые сведения, включающие такие разные сущности как "возраст", "профессия", "название города", "адрес электронной почты" в одном словаре `info` и легко обращаться к его элементам по ключам:

```
info = {'name': 'Timur',  
       'age': 28,  
       'job': 'Teacher',  
       'city': 'Moscow',  
       'email': 'timyr-guev@yandex.ru'}  
print(info['name'])  
print(info['email'])
```

# Словарь на основе 2-х списков

Создать словарь на основании двух списков (кортежей) можно с помощью встроенной функции-упаковщика `zip()`

```
keys = ['name', 'age', 'job']  
values = ['Timur', 28, 'Teacher']  
info = dict(zip(keys, values))  
print(info)
```

Проверка

# Словари (тип данных dict) являются

- Неизменяемыми
- Изменяемыми

Элемент в словаре имеет две части.  
Как они называются?

- Индекс
- Элемент
- Ключ
- Значение
- Строка

Предположим, что пара значений 'name':  
'Вася' является элементом некоторого  
словаря. Что служит ключом? И что -  
значением?

- 'name' – ключ
- 'name' – значение
- 'Вася' – ключ
- 'Вася' - значение

# Особенности словарей

# Ключи должны быть уникальными

Словарь не может иметь два и более значений по одному и тому же ключу. Если при создании словаря (в литеральной форме) указать дважды один и тот же ключ, будет использовано последнее из указанных значений.

```
info = {'name': 'Ruslan',  
       'age': 28,  
       'name': 'Timur'}  
print(info['name'])
```



# Ключи должны быть неизменяемым ТИПОМ ДАННЫХ

Ключом словаря могут быть данные любого неизменяемого типа:

- число;
- строка;
- булево значение;
- кортеж;
- замороженное множество (frozenset);
- ...

Ключ словаря не может относиться к изменяемому типу данных:

- список;
- множество;
- словарь;
- ...

# Значения могут относиться к любому типу данных, их тип данных произведен

Нет никаких ограничений для значений, хранящихся в словарях. Значения в словарях могут принадлежать к произвольному типу данных и повторяться для разных ключей многократно.

Какая часть элемента словаря ключ:  
значение должна быть неизменяемой?

- Ключ
- Значение

# Основы работы со словарями

# Функция len()

**Длиной словаря** называется количество его элементов. Для определения длины словаря используют встроенную функцию len() (от слова length – длина).

```
fruits = {'Apple': 70, 'Grape': 100, 'Banana': 80}
```

```
capitals = {'Россия': 'Москва', 'Франция': 'Париж'}
```

```
print(len(fruits))
```

```
print(len(capitals))
```

# Оператор принадлежности in

Оператор `in` позволяет проверить, содержит ли словарь заданный **ключ**.

```
capitals = {'Россия': 'Москва', 'Франция': 'Париж', 'Чехия': 'Прага'}
```

```
if 'Франция' in capitals:
```

```
    print('Столица Франции - это', capitals['Франция'])
```

Можно использовать оператор `in` вместе с логическим оператором `not`.

Не забывайте, что при обращении по **несуществующему ключу**, возникнет ошибка во время выполнения программы.

# Встроенный функции `sum()`, `min()`, `max()`

Встроенная функция `sum()` принимает в качестве аргумента **словарь с числовыми ключами** и вычисляет сумму его ключей.

Для корректной работы функции `sum()` ключами словаря должны быть именно числа.

Встроенные функции `min()` и `max()` принимают в качестве аргумента словарь и находят минимальный и максимальный ключ соответственно, при этом ключ может принадлежать к любому типу данных, для которого возможны операции порядка (`<`, `<=`, `>`, `>=`) (числа, строки, и т.д.).

# Сравнение словарей

Словари можно сравнивать между собой. Равные словари имеют одинаковое количество элементов и содержат равные элементы (ключ: значение)ю Для сравнения словарей используются операторы == и !=



# Примечания

- Обращение по индексу и срезы **недоступны** для словарей.
- Операция конкатенации + и умножения на число \* **недоступны** для словарей.