# Урок 15.01. Основы словарей.

Сегодня у нас новая и довольно важная тема.

Мы знаем, что списки представляют собой набор пронумерованных элементов, и чтобы получить элемент списка необходимо обратиться по его номеру, или, что правильнее, по его индексу. Индекс однозначно определяет элемент, однако иногда не удобно определять данные по их номерам, и для этого давайте рассмотрим такой пример.

Задача «Телефонный справочник».

У нас есть несколько людей, у каждого из которых есть номер телефона. И нам надо написать программу, которая находит человека по имени и выводит его номер телефона. Сейчас мы можем написать такую программу в следующем виде:

phonebook\_list = [

 ['Ваня', 89123456789],

 ['Петя', 89987654321],

 ['Лена', 89014569731]

]

name = input('Введите имя контакта: ')

is\_exist = False

for i\_person in phonebook\_list:

 if i\_person[0].lower() == name.lower():

 print(f'Контакт: {i\_person[0]}, номер: {i\_person[1]}')

 is\_exist = True

 break

if not is\_exist:

 print('Контакт не найден.')

Если запустить, то всё работает. Однако тут есть не очевидная проблема – слишком много не эффективного кода. Для поиска одного значения необходимо перебирать весь список, и если нам нужно имя в конце, то программе потребуется перебрать весь список. Для маленьких списков это не проблема, а вот с увеличением длины списка, это становится всё бóльшей проблемой.

И это всё нужно переделать, но уже с небольшим дополнением к нашим знаниям. В python’е есть коллекция, которая позволяет хранить данным немного другим способом, и называется она словарем (или по англ. dictionary). Для того чтобы создать словарь нужно поставить фигурные скобки, дальше написать «ключ» словаря, и после двоеточия записать значение, которое будет хранится по нашему ключу. Вот как будет выглядеть наш телефонный справочник в виде словаря:

phonebook\_list = {

 'Ваня': 89123456789,

 'Петя': 89987654321,

 'Лена': 89014569731

}

Теперь посмотрим на словарь внимательно, и видим, что у нас образованы так сказать пары «имя»-«телефон». И если вывести весь словарь на экран, то увидим этот словарь только в одну строчку. Если нам нужно получить значение элемента, то нам нужно обратить аналогично, как и в списка по индексу, только роль индексов тут играют ключи. Если нам нужно получить номер телефона «Ваня», то нужно всего лишь написать print(phonebook\_list['Ваня']). С этим знанием попробуем переписать нашу программу:

phonebook\_list = {

 'Ваня': 89123456789,

 'Петя': 89987654321,

 'Лена': 89014569731

}

name = input('Введите имя контакта: ')

if name in phonebook\_list:

 print('Контакт: {name}, номер: {phone}'.format(

 name=name,

 phone=phonebook\_list[name]

 ))

else:

 print('Контакт {name} не найден.'.format(name=name))

Если запустить, то такая программа короче и работает быстрее предыдущей.

Стоит заметить, что строчка if name in phonebook\_list: определяет, есть ли значение переменной name в списке ключей словаря phonebook\_list.

И тут мы поговорим об особенностях словарей:

1. имя ключей должно быть уникальным, т.е. двух одинаковых ключей не будет;
2. ключом может быть либо строка, либо число, либо другой неизменяемый объект (immetable, о них мы еще поговорим на следующих занятиях);
3. значением словаря может выступать любой другой объект: число, строка, список, другой словарь и т.д.

Давайте еще немного поупражняемся в работе со словарями.

Задача «Школьники». Необходимо написать программу, в которой вводится фамилия, имя ученика, город проживания, номер школы, в которой он учится, и все его оценки. Всё вводится в одну строку через пробел, в том числе и оценки. Далее программа по этой информации составит словарь и выведет его на экран в удобном виде. Вот так будет выглядеть программа:

# Иванова Илья Москва 100 4 5 5 5 4 4

# Петров Роман Омск 10 3 4 5 5 5

# Григорьев Алексей Красноярск 113 4 4 5

def input\_data():

 data = input('Введите данные ученика через пробел\n'

 '(фамилия, имя, город проживания, номер школы и все оценки): ')

 pupil\_data = data.split()

 pupil = {

 'фамилия': pupil\_data[0],

 'имя': pupil\_data[1],

 'город': pupil\_data[2],

 'школа': pupil\_data[3],

 'оценки': [int(grade) for grade in pupil\_data[4:]]

 }

 return pupil

def print\_pupils(pupils\_list):

 for pupils\_key in pupils\_list:

 print(f'Фамилия: {pupils\_list[pupils\_key]["фамилия"]}')

 print(f'Имя: {pupils\_list[pupils\_key]["имя"]}')

 print(f'Город: {pupils\_list[pupils\_key]["город"]}')

 print(f'Школа: {pupils\_list[pupils\_key]["школа"]}')

 print(f'Оценки: {pupils\_list[pupils\_key]["оценки"]}')

pupils = dict()

number\_pupils = int(input('Введите сколько учеников будет: '))

for \_ in range(number\_pupils):

 pupil = input\_data()

 name = '{surname} {name}'.format(

 surname=pupil["фамилия"],

 name=pupil["имя"]

 )

 pupils[name] = pupil

print('Ученики в списке:')

print\_pupils(pupils)

Запустим и проверим программу. Всё работает.

Давайте немного расширим наши знания. Для того чтобы добавить нового ключа и значения нужно просто обратить по этому ключу и присвоить ему значение, например:

pupils['Сидоров Андрей'] = {

 'фамилия': 'Сидоров',

 'имя': 'Андрей',

 'город': 'Томск',

 'школа': '17',

 'оценки': [4, 5, 4, 3, 4]

}

А теперь переходите к заданиям