MLClass. "Прикладной анализ данных".

Курс "Инструментарий Data Science".

Преподаватель ФКН НИУ ВШЭ Кашницкий Юрий

Урок 4. Структуры данных II

Часть 1. Словари

Словари позволяют харанить связанную информацию. Пример: храним имя и возраст человека в одной структуре.

Что такое словарь?

Словарь - это способ хранения информации, которая каким-то образом связана. Словари хранят информацию в виде пары *ключ - значение*, то есть, каждая часть информации соединена как минимум с одним значением.

Словари не хранят информацию в каком-то определенном порядке, поэтому не факт, что вы получите информацию в том порядке, в котором вы ее добавили.

Общий синтаксис

В общем виде словарь в Python выглядит так:

In [ ]:

```
dictionary_name = {key_1: value_1, key_2: value_2, key_3: value_3}
```

Поскольку пара *ключ-значение* может занимать довольно большое место на экране удобнее представлять словари в форме:

In [ ]:

```
dictionary_name = {key_1: value_1,
                   key_2: value_2,
                   key_3: value_3,
                   }
```

Такой способ записи намного более нагляден.

Пример

Простой пример использования словаря:

In [2]:

```
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }
```

Для обращения к элементу достаточно указать имя словаря и значения ключа в квадратных скобках:

In [1]:

```
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

print("\nWord: %s" % 'list')
print("Meaning: %s" % python_words['list'])

print("\nWord: %s" % 'dictionary')
print("Meaning: %s" % python_words['dictionary'])

print("\nWord: %s" % 'function')
print("Meaning: %s" % python_words['function'])
```

```
Word: list
Meaning: A collection of values that are not connected, but hav
e an order.

Word: dictionary
Meaning: A collection of key-value pairs.

Word: function
Meaning: A named set of instructions that defines a set of acti
ons in Python.
```

Вариант, который мы использовали в предыдущий раз, выглядит ужасно, поэтому для более компактной версии воспользуемся циклом:

In [2]:

```
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

# Print out the items in the dictionary.
for word, meaning in python_words.items():
    print("\nWord: %s" % word)
    print("Meaning: %s" % meaning)
```

```
Word: function
Meaning: A named set of instructions that defines a set of acti
ons in Python.

Word: list
Meaning: A collection of values that are not connected, but hav
e an order.

Word: dictionary
Meaning: A collection of key-value pairs.
```

3 строки вместо 6 выглядят намного лучше. То есть если у вас 100 элементов в словаре, напечатаете вы его с использованием тех-же 3-х строк.

Общий синтаксис для использования цикла со словарем:

In [ ]:

```
for key_name, value_name in dictionary_name.items():
    print(key_name) # The key is stored in whatever you called the first va
riable.
    print(value_name) # The value associated with that key is stored in you
r second variable.
```

Стандартные операции со словарями

Есть ряд стандартных операций: добавление, изменение и удаление

Добавление новой пары

Для добавления новой пары необходимо к имени словаря в квадратных скобках вписать новый ключ и присвоить величину, с которой хотите связать ключ. Начнем с пустого словаря.

In [3]:

```
# Create an empty dictionary.
python_words = {}

# Fill the dictionary, pair by pair.
python_words['list'] ='A collection of values that are not connected, but h
ave an order.'
python_words['dictionary'] = 'A collection of key-value pairs.'
python_words['function'] = 'A named set of instructions that defines a set
of actions in Python.'

# Print out the items in the dictionary.
for word, meaning in python_words.items():
    print("\nWord: %s" % word)
    print("Meaning: %s" % meaning)
```

```
Word: function
Meaning: A named set of instructions that defines a set of acti
ons in Python.

Word: list
Meaning: A collection of values that are not connected, but hav
e an order.

Word: dictionary
Meaning: A collection of key-value pairs.
```

Изменение значения

В какой-то момент приходится изменять значения в словаре. Изменение значения очень похоже на изменени элемента списка. Имя словаря[ключ] и присваиваем новое значение

In [4]:

```
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

print('dictionary: ' + python_words['dictionary'])

# Clarify one of the meanings.
python_words['dictionary'] = 'A collection of key-value pairs. Each key can
be used to access its corresponding value.'

print('\ndictionary: ' + python_words['dictionary'])
```

```
dictionary: A collection of key-value pairs.

dictionary: A collection of key-value pairs. Each key can be us
ed to access its corresponding value.
```

Удаление

Для удаления можно использовать `del`. как и для списков.

In [5]:

```
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

# Show the current set of words and meanings.
print("\n\nThese are the Python words I know:")
for word, meaning in python_words.items():
    print("\nWord: %s" % word)
    print("Meaning: %s" % meaning)

# Remove the word 'list' and its meaning.
del python_words['list']

# Show the current set of words and meanings.
print("\n\nThese are the Python words I know:")
for word, meaning in python_words.items():
    print("\nWord: %s" % word)
    print("Meaning: %s" % meaning)
```

```
These are the Python words I know:

Word: function
Meaning: A named set of instructions that defines a set of acti
ons in Python.

Word: list
Meaning: A collection of values that are not connected, but hav
e an order.

Word: dictionary
Meaning: A collection of key-value pairs.


These are the Python words I know:

Word: function
Meaning: A named set of instructions that defines a set of acti
ons in Python.

Word: dictionary
Meaning: A collection of key-value pairs.
```

Если мы будем продолжать работать с этим кодом, лучше сделать вывод значений словаря в виде функции. Посмотрим как это выглядит:

In [3]:

```
def show_words_meanings(python_words):
    # This function takes in a dictionary of python words and meanings,
    #  and prints out each word with its meaning.
    print("\n\nThese are the Python words I know:")
    for word, meaning in python_words.items():
        print("\nWord: %s" % word)
        print("Meaning: %s" % meaning)


python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

show_words_meanings(python_words)

# Remove the word 'list' and its meaning.
del python_words['list']

show_words_meanings(python_words)
```

```
These are the Python words I know:

Word: function
Meaning: A named set of instructions that defines a set of acti
ons in Python.

Word: list
Meaning: A collection of values that are not connected, but hav
e an order.

Word: dictionary
Meaning: A collection of key-value pairs.


These are the Python words I know:

Word: function
Meaning: A named set of instructions that defines a set of acti
ons in Python.

Word: dictionary
Meaning: A collection of key-value pairs.
```

Давайте поменяем наш пример с учетом написанной функции:

In [6]:

```
def show_words_meanings(python_words):
    # This function takes in a dictionary of python words and meanings,
    #  and prints out each word with its meaning.
    print("\n\nThese are the Python words I know:")
    for word, meaning in python_words.items():
        print("\n%s: %s" % (word, meaning))


python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

show_words_meanings(python_words)

# Remove the word 'list' and its meaning.
del python_words['list']

show_words_meanings(python_words)
```

```
These are the Python words I know:

function: A named set of instructions that defines a set of act
ions in Python.

list: A collection of values that are not connected, but have a
n order.

dictionary: A collection of key-value pairs.


These are the Python words I know:

function: A named set of instructions that defines a set of act
ions in Python.

dictionary: A collection of key-value pairs.
```

Изменяем ключи

Изменение ключа делается следующим способом:

- Создать новый ключ и скопировать значение к новому ключу.
- Удалить старый ключ, который удалит и значение.

Посмотрим как это выглядит на примере.

In [37]:

```
# We have a spelling mistake!
python_words = {'lisst': 'A collection of values that are not connected, bu
t have an order.'}

# Create a new, correct key, and connect it to the old value.
#  Then delete the old key.
python_words['list'] = python_words['lisst']
del python_words['lisst']

# Print the dictionary, to show that the key has changed.
print(python_words)
```

```
{'list': 'A collection of values that are not connected, but ha
ve an order.'}
```

Цикл по словарю

После добавления значени часто приходится что-то искать или выводить добавленные в словарь значения. Эти действия можно сделать несколькими способами:

- Цикл через все пары;
- Цикл по ключам;
- Цикл по значениям.

Цикл через все пары

Такой цикл уже был показан в предыдущих примерах. В общем формате цикл выглядит так:

In [7]:

```
my_dict = {'key_1': 'value_1',
    'key_2': 'value_2',
    'key_3': 'value_3',
    }

for key, value in my_dict.items():
    print('\nKey: %s' % key)
    print('Value: %s' % value)
```

```
Key: key_1
Value: value_1

Key: key_3
Value: value_3

Key: key_2
Value: value_2
```

Это работает потому, что используется метод `.items()`, который вытаскивает пары в виде кортежей:

In [10]:

```
my_dict = {'key_1': 'value_1',
    'key_2': 'value_2',
    'key_3': 'value_3',
    }

print(my_dict.items())
```

```
[('key_1', 'value_1'), ('key_3', 'value_3'), ('key_2', 'valu
e_2')]
```

Синтаксис `for key, value in my_dict.items():` выполняет проходку по списку tuples, и достает первый и второй элемент для нас.

In [11]:

```
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

for word, meaning in python_words.items():
    print("\nWord: %s" % word)
    print("Meaning: %s" % meaning)
```

```
Word: function
Meaning: A named set of instructions that defines a set of acti
ons in Python.

Word: list
Meaning: A collection of values that are not connected, but hav
e an order.

Word: dictionary
Meaning: A collection of key-value pairs.
```

Цикл по ключам

Python предоставляет инструмент для прохождения только по ключам словаря:

In [13]:

```
my_dict = {'key_1': 'value_1',
    'key_2': 'value_2',
    'key_3': 'value_3',
    }

for key in my_dict.keys():
    print('Key: %s' % key)
```

```
Key: key_1
Key: key_3
Key: key_2
```

Это стандартное поведение при проходе по словарю. Поэтому можно убрать `.keys()` и получить абсолютно такой же эффект:

In [14]:

```
###highlight=[7]
my_dict = {'key_1': 'value_1',
    'key_2': 'value_2',
    'key_3': 'value_3',
    }

for key in my_dict:
    print('Key: %s' % key)
```

```
Key: key_1
Key: key_3
Key: key_2
```

Единственное преимущество в использовании `.keys()` - читаемость кода. В любом случае всякий, кто знаком с Python, поймет что происходит. Далее мы будем писать наши циклы без`.keys()`, если нам понадобится такой эффект.

Можно также достать любое необходимое значение по ключу во время прохода циклом, используя стандартный способ доступа:

In [17]:

```
my_dict = {'key_1': 'value_1',
    'key_2': 'value_2',
    'key_3': 'value_3',
    }

for key in my_dict:
    print('Key: %s' % key)
    if key == 'key_2':
        print("The value for key_2 is %s." % my_dict[key])
```

```
Key: key_1
Key: key_3
Key: key_2
  The value for key_2 is value_2.
```

Давайте применим этот способ к нашему словарю:

In [20]:

```
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

# Show the words that are currently in the dictionary.
print("The following Python words have been defined:")
for word in python_words:
    print("- %s" % word)
```

```
The following Python words have been defined:
- function
- list
- dictionary
```

Можно добавить немного интерактива и спросить пользователя, какое слово ему интересно, и выдать ему значение:

In [8]:

```
###highlight=[12,13,14]
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

# Show the words that are currently in the dictionary.
print("The following Python words have been defined:")
for word in python_words:
    print("- %s" % word)

# Allow the user to choose a word, and then display the meaning for that wo
rd.
requested_word = raw_input("\nWhat word would you like to learn about? ")
print("\n%s: %s" % (requested_word, python_words[requested_word]))
```

```
The following Python words have been defined:
- function
- list
- dictionary

What word would you like to learn about? list

list: A collection of values that are not connected, but have a
n order.
```

Можно использовать цикл while, что позволит пользователю смотреть информацию по многим ключам:

In [4]:

```
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

# Show the words that are currently in the dictionary.
print("The following Python words have been defined:")
for word in python_words:
    print("- %s" % word)

requested_word = ''
while requested_word != 'quit':
    # Allow the user to choose a word, and then display the meaning for tha
t word.
    requested_word = raw_input("\nWhat word would you like to learn about?
(or 'quit') ")
    if requested_word in python_words.keys():
        print("\n  %s: %s" % (requested_word, python_words[requested_wor
d]))
    else:
        # Handle misspellings, and words not yet stored.
        print("\n  Sorry, I don't know that word.")
```

```
The following Python words have been defined:
- function
- list
- dictionary

What word would you like to learn about? (or 'quit') list

  list: A collection of values that are not connected, but have
an order.

What word would you like to learn about? (or 'quit') dictionary

  dictionary: A collection of key-value pairs.

What word would you like to learn about? (or 'quit') quit

  Sorry, I don't know that word.
```

Добавим конструкцию `elif`, чтобы `quit` не воспринималось как запрашиваемый ключ словаря:

In [6]:

```
###highlight=[16,17,18,19,20,21,22,23,24]
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

# Show the words that are currently in the dictionary.
print("The following Python words have been defined:")
for word in python_words:
    print("- %s" % word)

requested_word = ''
while requested_word != 'quit':
    # Allow the user to choose a word, and then display the meaning for tha
t word.
    requested_word = raw_input("\nWhat word would you like to learn about?
(or 'quit') ")
    if requested_word in python_words.keys():
        # This is a word we know, so show the meaning.
        print("\n  %s: %s" % (requested_word, python_words[requested_wor
d]))
    elif requested_word != 'quit':
        # This is not in python_words, and it's not 'quit'.
        print("\n  Sorry, I don't know that word.")
    else:
        # The word is quit.
        print "\n  Bye!"
```

```
The following Python words have been defined:
- function
- list
- dictionary

What word would you like to learn about? (or 'quit') function

  function: A named set of instructions that defines a set of a
ctions in Python.

What word would you like to learn about? (or 'quit') dictionary

  dictionary: A collection of key-value pairs.

What word would you like to learn about? (or 'quit') list

  list: A collection of values that are not connected, but have
an order.

What word would you like to learn about? (or 'quit') class

  Sorry, I don't know that word.

What word would you like to learn about? (or 'quit') quit

  Bye!
```

Цикл по всем значениям

Python также позволяет проходить по значениям словаря:

In [15]:

```
my_dict = {'key_1': 'value_1',
    'key_2': 'value_2',
    'key_3': 'value_3',
    }

for value in my_dict.values():
    print('Value: %s' % value)
```

```
Value: value_1
Value: value_3
Value: value_2
```

Используя предыдущий пример, можно сделать небольшой опросник, пользователь должен угадать слово по его описанию:

In [16]:

```
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

for meaning in python_words.values():
    print("Meaning: %s" % meaning)
```

```
Meaning: A named set of instructions that defines a set of acti
ons in Python.
Meaning: A collection of values that are not connected, but hav
e an order.
Meaning: A collection of key-value pairs.
```

Давайте спросим пользователя:

In [2]:

```
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

# Print each meaning, one at a time, and ask the user
#  what word they think it is.
for meaning in python_words.values():
    print("\nMeaning: %s" % meaning)

    guessed_word = raw_input("What word do you think this is? ")

    # The guess is correct if the guessed word's meaning matches the curren
t meaning.
    if python_words[guessed_word] == meaning:
        print("You got it!")
    else:
        print("Sorry, that's just not the right word.")
```

```
Meaning: A named set of instructions that defines a set of acti
ons in Python.
What word do you think this is? function
You got it!

Meaning: A collection of values that are not connected, but hav
e an order.
What word do you think this is? function
Sorry, that's just not the right word.

Meaning: A collection of key-value pairs.
What word do you think this is? dictionary
You got it!
```

И конечно несколько раз тоже можно спрашивать:

In [20]:

```
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

# Print each meaning, one at a time, and ask the user
#  what word they think it is.
for meaning in python_words.values():
    print("\nMeaning: %s" % meaning)

    # Assume the guess is not correct; keep guessing until correct.
    correct = False
    while not correct:
        guessed_word = input("\nWhat word do you think this is? ")

        # The guess is correct if the guessed word's meaning matches the cu
rrent meaning.
        if python_words[guessed_word] == meaning:
            print("You got it!")
            correct = True
        else:
            print("Sorry, that's just not the right word.")
```

```
Meaning: A named set of instructions that defines a set of acti
ons in Python.

What word do you think this is? function
You got it!

Meaning: A collection of values that are not connected, but hav
e an order.

What word do you think this is? dictionary
Sorry, that's just not the right word.

What word do you think this is? list
You got it!

Meaning: A collection of key-value pairs.

What word do you think this is? dictionary
You got it!
```

Последнее, можно показать из чего пользователь должен выбрать:

In [8]:

```
###highlight=[7,8,9,10,11,12,23,24,25]
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

def show_words(python_words):
    # A simple function to show the words in the dictionary.
    display_message = ""
    for word in python_words.keys():
        display_message += word + '  '
    print display_message

# Print each meaning, one at a time, and ask the user
#  what word they think it is.
for meaning in python_words.values():
    print("\n%s" % meaning)

    # Assume the guess is not correct; keep guessing until correct.
    correct = False
    while not correct:

        print("\nWhat word do you think this is?")
        show_words(python_words)
        guessed_word = raw_input("- ")

        # The guess is correct if the guessed word's meaning matches the cu
rrent meaning.
        if python_words[guessed_word] == meaning:
            print("You got it!")
            correct = True
        else:
            print("Sorry, that's just not the right word.")
```

```
A named set of instructions that defines a set of actions in Py
thon.

What word do you think this is?
function  list  dictionary
- function
You got it!

A collection of values that are not connected, but have an orde
r.

What word do you think this is?
function  list  dictionary
- dictionary
Sorry, that's just not the right word.

What word do you think this is?
function  list  dictionary
- list
You got it!

A collection of key-value pairs.

What word do you think this is?
function  list  dictionary
- dictionary
You got it!
```

Немного порядка

Проблемой словарей является то, что связанные значения не отсортированы в каком-либо порядке. Когда вы достаете ключи или значение, вы никогда не знаете, в каком порядке их получите. Пройтись по значениям довольно просто, но давайте сделаем сортировку

Посмотрим, что если просто использовать *dictionary.keys()*:

In [2]:

```
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

for word in python_words.keys():
    print(word)
```

```
function
list
dictionary
```

Значения никак не отсортированы. Можно использовать *sorted()* чтобы отсортировать то, что мы будем получать:

In [3]:

```
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

for word in sorted(python_words.keys()):
    print(word)
```

```
dictionary
function
list
```

Такой подход можно использовать и для ключей, и для значений. Пример, когда слова и их значения могут быть выведены в алфавитном порядке:

In [8]:

```
python_words = {'list': 'A collection of values that are not connected, but
have an order.',
                'dictionary': 'A collection of key-value pairs.',
                'function': 'A named set of instructions that defines a set
of actions in Python.',
                }

for word in sorted(python_words.keys()):
    print("%s: %s" % (word.title(), python_words[word]))
```

```
Dictionary: A collection of key-value pairs.
Function: A named set of instructions that defines a set of act
ions in Python.
List: A collection of values that are not connected, but have a
n order.
```

Данный подход не затрагивает содержание словаря. Но если вы хотите использовать отсортированные словари, существует структра данных, которая реализовывает такой подход - OrderedDict (http://docs.python.org/3.3/library/collections.html#ordereddict-objects).

Nesting

Nesting - одна из самых мощных концепций в Python. В данном случае она позволяет завернуть один словарь внутрь другого. Мы рассмотрим два примера: списки внутри словаря и словарь внутри словаря. Таким способом можно расширять имеющиеся структуры данных и полностью пересмотреть подход к организации данных.

Список в словаре

Словарь объединяет 2 части информации (ключ и значение). Это могут быть части информации любого вида и любой структуры. В качестве ключей оставим строковые переменные, но в качестве значения будет список.

В первом примере мы будем хранить имя человека и последовательность его любимых чисел. И мы по очереди пройдем по каждому человеку и его списку.

In [20]:

```
# This program stores people's favorite numbers, and displays them.
favorite_numbers = {'eric': [3, 11, 19, 23, 42],
                    'ever': [2, 4, 5],
                    'willie': [5, 35, 120],
                    }

# Display each person's favorite numbers.
print("Eric's favorite numbers are:")
print(favorite_numbers['eric'])

print("\nEver's favorite numbers are:")
print(favorite_numbers['ever'])

print("\nWillie's favorite numbers are:")
print(favorite_numbers['willie'])
```

```
Eric's favorite numbers are:
[3, 11, 19, 23, 42]

Ever's favorite numbers are:
[2, 4, 5]

Willie's favorite numbers are:
[5, 35, 120]
```

Заменим циклом по ключам:

In [7]:

```
# This program stores people's favorite numbers, and displays them.
favorite_numbers = {'eric': [3, 11, 19, 23, 42],
                    'ever': [2, 4, 5],
                    'willie': [5, 35, 120],
                    }

# Display each person's favorite numbers.
for name in favorite_numbers:
    print("\n%s's favorite numbers are:" % name.title())
    print(favorite_numbers[name])
```

```
Willie's favorite numbers are:
[5, 35, 120]

Ever's favorite numbers are:
[2, 4, 5]

Eric's favorite numbers are:
[3, 11, 19, 23, 42]
```

Структура вполне понятная, очевидно, что к чему привязанно в данном случае.

Тем не менее это плохая идея - пробовать использовать сырой вывод для структуры с такой связностью. Лучше использовать еще один цикл.

In [1]:

```
# This program stores people's favorite numbers, and displays them.
favorite_numbers = {'eric': [3, 11, 19, 23, 42],
                    'ever': [2, 4, 5],
                    'willie': [5, 35, 120],
                    }

# Display each person's favorite numbers.
for name in favorite_numbers:
    print("\n%s's favorite numbers are:" % name.title())

    # Each value is itself a list, so let's put that list in a variable.
    current_favorite_numbers = favorite_numbers[name]
    for favorite_number in current_favorite_numbers:
        print(favorite_number)
```

```
Willie's favorite numbers are:
5
35
120

Eric's favorite numbers are:
3
11
19
23
42

Ever's favorite numbers are:
2
4
5
```

Словарь в словаре

Для демонстрации будем использовать следующий пример, возьмем словарь животных и добавим описание этому животному. Ключами будут имена/клички животных. Значение будет содержать: вид животного, хозяина и информацию о том, была ли проведена вакцинация.

In [24]:

```
# This program stores information about pets. For each pet,
#   we store the kind of animal, the owner's name, and
#   the breed.
pets = {'willie': {'kind': 'dog', 'owner': 'eric', 'vaccinated': True},
        'walter': {'kind': 'cockroach', 'owner': 'eric', 'vaccinated': Fals
e},
        'peso': {'kind': 'dog', 'owner': 'chloe', 'vaccinated': True},
        }

# Let's show all the information for each pet.
print("Here is what I know about Willie:")
print("kind: " + pets['willie']['kind'])
print("owner: " + pets['willie']['owner'])
print("vaccinated: " + str(pets['willie']['vaccinated']))

print("\nHere is what I know about Walter:")
print("kind: " + pets['walter']['kind'])
print("owner: " + pets['walter']['owner'])
print("vaccinated: " + str(pets['walter']['vaccinated']))

print("\nHere is what I know about Peso:")
print("kind: " + pets['peso']['kind'])
print("owner: " + pets['peso']['owner'])
print("vaccinated: " + str(pets['peso']['vaccinated']))
```

```
Here is what I know about Willie:
kind: dog
owner: eric
vaccinated: True

Here is what I know about Walter:
kind: cockroach
owner: eric
vaccinated: False

Here is what I know about Peso:
kind: dog
owner: chloe
vaccinated: True
```

Довольно громоздкое описание, но, тем не менее, дает четкое понимание того, что происходит внутри структуры и как организованы данные.

Давайте сделаем более компактный вариант:

In [12]:

```
###highlight=[10,11,12,13,14,15]
# This program stores information about pets. For each pet,
#   we store the kind of animal, the owner's name, and
#   the breed.
pets = {'willie': {'kind': 'dog', 'owner': 'eric', 'vaccinated': True},
        'walter': {'kind': 'cockroach', 'owner': 'eric', 'vaccinated': Fals
e},
        'peso': {'kind': 'dog', 'owner': 'chloe', 'vaccinated': True},
        }

# Let's show all the information for each pet.
for pet_name, pet_information in pets.items():
    print("\nHere is what I know about %s:" % pet_name.title())
    print("kind: " + pet_information['kind'])
    print("owner: " + pet_information['owner'])
    print("vaccinated: " + str(pet_information['vaccinated']))
```

```
Here is what I know about Peso:
kind: dog
owner: chloe
vaccinated: True

Here is what I know about Willie:
kind: dog
owner: eric
vaccinated: True

Here is what I know about Walter:
kind: cockroach
owner: eric
vaccinated: False
```

Запись компактная, но если мы добавим информацию к исходному словарю, возникнут некоторые проблемы. Поэтому давайте дадим каждому словарю по циклу:

In [13]:

```
# This program stores information about pets. For each pet,
#   we store the kind of animal, the owner's name, and
#   the breed.
pets = {'willie': {'kind': 'dog', 'owner': 'eric', 'vaccinated': True},
        'walter': {'kind': 'cockroach', 'owner': 'eric', 'vaccinated': Fals
e},
        'peso': {'kind': 'dog', 'owner': 'chloe', 'vaccinated': True},
        }

# Let's show all the information for each pet.
for pet_name, pet_information in pets.items():
    print("\nHere is what I know about %s:" % pet_name.title())
    # Each animal's dictionary is in 'information'
    for key in pet_information:
        print(key + ": " + str(pet_information[key]))
```

```
Here is what I know about Peso:
owner: chloe
kind: dog
vaccinated: True

Here is what I know about Willie:
owner: eric
kind: dog
vaccinated: True

Here is what I know about Walter:
owner: eric
kind: cockroach
vaccinated: False
```

Такая структура в начале может выглядить сложно, давайте поясним.

- Первый цикл идет по ключам - имена животных
- Каждое имя используется чтобы 'распаковать' словарь с описанием для этого имени.
- Внутренний цикл идет по ключам описания, и вытаскивает все ключи, которые сохранены во вложенном словаре.
- Мы печатаем ключ, который показывает нам, что мы собираемся увидеть, и значения ключа.
- Мы так жеможем улучшить вывод:
  - Имя владельца можно печатать с большой буквы
  - Печатать 'yes' или 'no', вместо True и False.

Давайте выведем финальную версию:

In [14]:

```
# This program stores information about pets. For each pet,
#   we store the kind of animal, the owner's name, and
#   the breed.
pets = {'willie': {'kind': 'dog', 'owner': 'eric', 'vaccinated': True},
        'walter': {'kind': 'cockroach', 'owner': 'eric', 'vaccinated': Fals
e},
        'peso': {'kind': 'dog', 'owner': 'chloe', 'vaccinated': True},
        }

# Let's show all the information for each pet.
for pet_name, pet_information in pets.items():
    print("\nHere is what I know about %s:" % pet_name.title())
    # Each animal's dictionary is in pet_information
    for key in pet_information:
        if key == 'owner':
            # Capitalize the owner's name.
            print(key + ": " + pet_information[key].title())
        elif key == 'vaccinated':
            # Print 'yes' for True, and 'no' for False.
            vaccinated = pet_information['vaccinated']
            if vaccinated:
                print 'vaccinated: yes'
            else:
                print 'vaccinated: no'
        else:
            # No special formatting needed for this key.
            print(key + ": " + pet_information[key])
```

```
Here is what I know about Peso:
owner: Chloe
kind: dog
vaccinated: yes

Here is what I know about Willie:
owner: Eric
kind: dog
vaccinated: yes

Here is what I know about Walter:
owner: Eric
kind: cockroach
vaccinated: no
```

Важное замечание о nesting

Вложения на 1-2 уровня могут быть очень удобны, но это становится плохо читаемым, если слишком увлекаться таким подходом. Если у вас большая глубина вложений - задумайтесь, возможно, вы используете неправильный способ хранения информации. Для очень сложных разветвленных типов существуют классы и базы данных, которые позволяют сохранять сложную связанную информацию и взаимодействовать с ней.

Чаще всего вы частично извлекаете информацию из базы данных и упаковываете её в словарь, нежели используете струтуры глубокой вложенности.

Полезные ссылки

- Детальный анализ словаря Python как структуры данных на Хабрахабре (http://habrahabr.ru/post/247843/)
- Краткое введение в списки, кортежи, словари и файлы на Хабрахабре (http://habrahabr.ru/post/30092/)