Использование словарей

Данное занятие посвящено следующим вопросам:

- Использование ключа, чтобы получить значение из словаря
- Проверка наличия ключей
- Перебор ключей и значений в словарях

Получить значение по ключу

Если у вас есть словарь, вы можете получить доступ к его значениям, указав ключ. Например, представим, что у нас есть словарь, в котором высота зданий отображается в метрах:

>>building_heights = {"Burj Khalifa": 828, "Shanghai Tower": 632, "Abraj Al Bait": 601, "Ping An": 599, "Lotte World Tower": 554.5, "One World Trade": 541.3}

Затем мы можем получить доступ к данным в нем следующим образом:

```
>>> building_heights["Burj Khalifa"]
828
```

Задание

1. Мы предоставили словарь, в котором элементы стихий соотносятся со знаками зодиака. Выведите в консоль список знаков зодиака, связанных со стихией «земля».

zodiac_elements = {"water": ["Cancer", "Scorpio", "Pisces"], "fire": ["Aries", "Leo", "Sagittarius "], "earth": ["Taurus", "Virgo", "Capricorn"], "air":["Gemini", "Libra", "Aquarius"]}

2. Выведите в консоль список знаков стихии огня.

Недействительные ключи

Допустим, у нас есть словарь высот зданий из последнего упражнения:

>>building_heights = {"Burj Khalifa": 828, "Shanghai Tower": 632, "Abraj Al Bait": 601, "Ping An": 599, "Lotte World Tower": 554.5, "One World Trade": 541.3}

Что, если бы мы хотели узнать высоту Landmark 81 в Хошимине? Мы могли бы попробовать:

>>print(building_heights["Landmark 81"])

Но «Landmark 81» не существует в качестве ключа в словаре building_heights! Таким образом, это приведет к ошибке KeyError:

```
KeyError: 'Landmark 81'
```

Один из способов избежать этой ошибки - сначала проверить, существует ли ключ в словаре:

```
>>key_to_check = "Landmark 81"
>>if key_to_check in building_heights:
>>      print(building_heights["Landmark 81"])
```

Это не приведет к ошибке, потому что key_to_check в building_heights вернет False, и поэтому мы никогда не пытаемся получить доступ к ключу.

Задание

1. Запустите код. Он должен выдать KeyError! «energy» не существует как один из элементов.
2. Добавьте ключ «energy» к zodiac_elements. Он должен соответствовать значению «Not a Zodiac element». Запустите код. Устранено ли это KeyError?

zodiac_elements = {"water": ["Cancer", "Scorpio", "Pisces"], "fire": ["Aries", "Leo", "Sagittarius "], "earth": ["Taurus", "Virgo", "Capricorn"], "air":["Gemini", "Libra", "Aquarius"]}

Try/Except для получения ключа

Мы увидели, что можем избежать KeyErrors, сначала проверив, есть ли ключ в словаре. Другой метод, который мы могли бы использовать, — это try / except:

```
key_to_check = "Landmark 81"
try:
  print(building_heights[key_to_check])
except KeyError:
  print("That key doesn't exist!")
```

Когда мы пытаемся получить доступ к несуществующему ключу, программа перейдет в блок except и выведет «Этот ключ не существует!».

Задание

1. Используйте блок try/except чтобы попытаться вывести уровень кофеина в матче. Если возникла ошибка KeyError, выведите «Неизвестный уровень кофеина».

caffeine_level = {"espresso": 64, "chai": 40, "decaf": 0, "drip": 120}

2. Над блоком try добавьте в словарь «matcha» со значением 30

Безопасно получить ключ

В последнем упражнении мы видели, что можно добавить в словарь пару ключ: значение, чтобы избежать KeyError. Это решение неустойчиво. Мы не можем предсказать каждый ключ, который может вызвать пользователь, и добавить все эти значения-заполнители в наш словарь!

В словарях есть метод .get () для поиска значения вместо my_dict [key], которую мы использовали ранее. Если ключ, который вы пытаетесь использовать в методе .get (), не существует, по умолчанию он вернет None:

```
>>building_heights = {"Burj Khalifa": 828, "Shanghai Tower": 632, "Abraj Al Bait": 601, "Ping An": 599, "Lotte World Tower": 554.5, "One World Trade": 541.3}
>>#this line will return 632:
>>building_heights.get("Shanghai Tower")
>>#this line will return None:
>>building_heights.get("My House")
```

Возможно указать значение, которое будет возвращено, если ключ не существует. Например, мы можем вернуть высоту здания, равную 0, если желаемое здание отсутствует в словаре:

```
>>> building_heights.get('Shanghai Tower', 0)
632
>>> building_heights.get('Mt Olympus', 0)
0
>>> building_heights.get('Kilimanjaro', 'No Value')
'No Value'
```

Задание

1. Используйте .get (), чтобы получить значение идентификатора пользователя teraCoder, со значением по умолчанию 100000, если пользователь не существует. Сохраните его в переменной tc_id. Выведите tc_id в консоль.
2. Используйте .get (), чтобы получить значение идентификатора пользователя superStackSmash, со значением по умолчанию 100000, если пользователь не существует. Сохраните его в переменной stack_id. Вывести stack_id в консоль.

```
user_ids = {"teraCoder": 100019, "pythonGuy": 182921, "samTheJavaMaam": 123112, "lyleLoop": 102931, "keysmithKeith": 129384}
```

Удалить ключ

Иногда мы хотим получить ключ и удалить его из словаря. Представьте, что мы проводим розыгрыш, и у нас есть словарь, сопоставляющий номера билетов с призами:

```
>>raffle = {223842: "Teddy Bear", 872921: "Concert Tickets", 320291: "Gift Basket", 412123: "Necklace", 298787: "Pasta Maker"}
```

Когда мы получаем номер билета, мы хотим вернуть приз, а также удалить эту пару из словаря, поскольку приз уже был отдан. Для этого мы можем использовать метод .pop (). Как и в случае с .get (), мы можем предоставить значение по умолчанию, если ключ не существует в словаре:

```
>>> raffle.pop(320291, "No Prize")
"Gift Basket"
>>> raffle
```

```
{223842: "Teddy Bear", 872921: "Concert Tickets", 412123: "Necklace", 298787: "Pasta Maker"}
>>> raffle.pop(100000, "No Prize")
"No Prize"
>>> raffle
{223842: "Teddy Bear", 872921: "Concert Tickets", 412123: "Necklace", 298787: "Pasta Maker"}
>>> raffle.pop(872921, "No Prize")
"Concert Tickets"
>>> raffle
{223842: "Teddy Bear", 412123: "Necklace", 298787: "Pasta Maker"}
```

Получить все ключи

Иногда нам нужно оперировать всеми ключами в словаре. Например, если у нас есть словарь учащихся математического класса и их оценок:

```
>>test_scores = {"Grace":[80, 72, 90], "Jeffrey":[88, 68, 81], "Sylvia":[80, 82, 84], "Pedro":[98, 96, 95], "Martin":[78, 80, 78], "Dina":[64, 60, 75]}
```

Мы хотим получить список учащихся в классе без учета их оценок. Мы можем сделать это с помощью встроенной функции list ():

```
>>> list(test_scores)
["Grace", "Jeffrey", "Sylvia", "Pedro", "Martin", "Dina"]
```

В словарях также есть метод .keys (), который возвращает объект dict_keys. Объект dict_keys — это объект представления, который обеспечивает просмотр текущего состояния словаря без возможности изменения пользователем чего-либо. Объект dict_keys, возвращаемый функцией .keys (), представляет собой набор ключей в словаре. Вы не можете добавлять или удалять элементы из объекта dict_keys, но его можно использовать вместо списка для итерации:

```
>>for student in test_scores.keys():
>>        print(student)
```

выведет:

```
Grace
Jeffrey
Sylvia
Pedro
Martin
Dina
```

Задание

1. Создайте переменную с именем users и назначьте ей объект dict_keys для всех ключей словаря user_ids.

user_ids = {"teraCoder": 100019, "pythonGuy": 182921, "samTheJavaMaam": 123112, "lyleLoop": 102931, "keysmithKeith": 129384}

num_exercises = {"functions": 10, "syntax": 13, "control flow": 15, "loops": 22, "lists": 19, "classes": 18, "dictionaries": 18}

2. Создайте переменную с именем classes и назначьте ее объектом dict_keys для всех ключей словаря num_exercises.
3. Вывести users в консоль.
4. Вывести lessons на консоли.

Получить все значения

В словарях есть метод .values (), который возвращает объект dict_values (точно так же, как объект dict_keys, но для значений!) Со всеми значениями в словаре. Его можно использовать вместо списка для итерации:

>>test_scores = {"Grace":[80, 72, 90], "Jeffrey":[88, 68, 81], "Sylvia":[80, 82, 84], "Pedro":[98, 96, 95], "Martin":[78, 80, 78], "Dina":[64, 60, 75]}

```
>>for score_list in test_scores.values():
>>        print(score_list)
```

Выдаст:

```
[80, 72, 90]
[88, 68, 81]
[80, 82, 84]
[98, 96, 95]
[78, 80, 78]
[64, 60, 75]
```

Нет встроенной функции для получения всех значений в виде списка, но если вы действительно хотите, вы можете использовать:

>>list(test_scores.values())

Задание

1. Создайте переменную с именем total_exercises и установите ее равной 0.
2. Просмотрите значения в списке num_exercises и добавьте каждое значение в переменную total_exercises.
3. Выведите в консоль переменную total_exercises.

>>num_exercises = {"functions": 10, "syntax": 13, "control flow": 15, "loops": 22, "lists": 19, "classes": 18, "dictionaries": 18}

Получить все ключи и значения

Вы можете получить как ключи, так и значения с помощью метода .items (). Подобно .keys () и .values (), он возвращает объект dict_list. Каждый элемент dict_list, возвращаемый функцией .items (), представляет собой кортеж, состоящий из:

(key, value)

поэтому для итерации вы можете использовать этот синтаксис:

```
>>biggest_brands = {"Apple": 184, "Google": 141.7, "Microsoft": 80, "Coca-Cola": 69.7, "Amazon": 64.8}


>>for company, value in biggest_brands.items():
>>         print(company + " has a value of " + str(value) + " billion dollars. ")
```

В результате будет выведено::

```
Apple has a value of 184 billion dollars.
Google has a value of 141.7 billion dollars.
Microsoft has a value of 80 billion dollars.
Coca-Cola has a value of 69.7 billion dollars.
Amazon has a value of 64.8 billion dollars.
```

Заключение

В этом уроке вы узнали, как просматривать словари и получать доступ к ключам и значениям разными способами. В частности, вы узнали, как:

- Использовать ключ, чтобы получить значение из словаря
- Проверить наличие ключей
- Удалить пару ключ: значение из словаря
- Перебирать ключи и значения в словарях

Задание

1. Мы предоставили колоду карт Таро. Вы собираетесь вытащить из колоды три карты своего прошлого, настоящего и будущего.

tarot = { 1: "The Magician", 2: "The High Priestess", 3: "The Empress", 4: "The Emperor", 5: "The Hierophant", 6: "The Lovers", 7: "The Chariot", 8: "Strength", 9: "The Hermit", 10: "Wheel of Fortune", 11: "Justice", 12: "The Hanged Man", 13: "Death", 14: "Temperance", 15: "The Devil", 16: "The Tower", 17: "The Star", 18: "The Moon", 19: "The Sun", 20: "Judgement", 21: "The World", 22: "The Fool"}

Создайте пустой словарь под названием spread.

2. Первая карта, которую вы берете, — это карта 13. Извлеките значение, присвоенное ключу 13, из словаря Таро и назначьте его как значение к ключу «прошлое».
3. Вторая карта, которую вы берете, — это карта 22. Извлеките значение, присвоенное ключу 22, из словаря Таро и назначьте его как значение ключу «настоящееe».
4. Третья карта, которую вы берете, — это карта 10. Извлеките значение, присвоенное ключу 10, из словаря Таро и назначьте его как значение к ключу «будущее»