Введение в словари

Словарь — это неупорядоченный набор пар ключ: значение.
Это дает нам возможность сопоставлять данные друг с другом для того, чтобы была возможность быстро находить значения, связанные друг с другом.
Предположим, мы хотим сохранить цены на различные товары, продаваемые в кафе:

- Тост с авокадо - 6 долларов
- Морковный сок - 5 долларов
- Черничный маффин стоит 2 доллара

В Python мы можем создать словарь под названием menu для хранения этих данных:

>>menu = {" Тост с авокадо ": 6, " Морковный сок ": 5, " Черничный маффин": 2}

Обратите внимание:

- Словарь начинается и заканчивается фигурными скобками {и}.
- Каждый элемент состоит из ключа («тост с авокадо») и значения (6).
- Каждая пара ключ: значение разделяется запятой.
- Считается хорошей практикой вставлять пробел после каждой запятой, но код все равно будет работать и без пробела.

Задание

1. Предположим, у нас есть словарь датчиков температуры в доме и текущие значения температуры. Мы только что добавили датчик в «кладовую», он показывает 22 градуса.

Добавьте эту пару в словарь:

sensors = {"living room": 21, "kitchen": 23, "bedroom": 20}

2. Скопируйте строчку кода ниже и удалите символ # перед определением словаря num_cameras, который представляет количество камер в каждой области вокруг дома.

Если вы запустите этот код, вы получите сообщение об ошибке:

SyntaxError: invalid syntax

Попробуйте найти и исправить синтаксическую ошибку, чтобы запустить этот код.

#num_cameras = {"backyard": 6, "garage": 2 "driveway" 1}

Создание словаря

В предыдущем упражнении мы видели словарь, в котором сопоставлены строки и числа (например, «тост с авокадо»: 6). Однако ключи могут быть и числами.

Например, если мы сопоставляем промежуточные итоги счетов в ресторане с общей суммой счетов после чаевых, словарь мог бы выглядеть так:

>>subtotal_to_total = {20: 24, 10: 12, 5: 6, 15: 18}

Значения могут быть любого типа. Мы можем использовать строку, число, список или даже другой словарь в качестве значения, связанного с ключом!

Например:

>>students_in_classes = {"software design": ["Aaron", "Delila", "Samson"], "cartography": ["Christopher", "Juan", "Marco"], "philosophy": ["Frederica", "Manuel"]}

Список ["Aaron", "Delila", "Samson"], который является значением ключа «software design», представляет учащихся этого класса.

Мы также можем смешивать и сопоставлять типы ключей и значений. Например:

>>person = {"name": "Shuri", "age": 18, "family": ["T'Chaka", "Ramonda"]}

Задание

1. Создайте словарь переводов, который сопоставляет следующие слова на английском языке с их определениями на синдарине (языке эльфов):

English	Sindarin
mountain	orod
bread	bass
friend	mellon
horse	roch

Недействительные ключи

У нас может быть список или словарь как значение элемента в словаре, но мы не можем использовать эти типы данных в качестве ключей словаря. Если мы попытаемся это сделать, мы получим TypeError.

Например:

>>powers = {[1, 2, 4, 8, 16]: 2, [1, 3, 9, 27, 81]: 3}

Этот код даст:

TypeError: unhashable type: 'list'

Слово «нехэшируемый» в этом контексте означает, что этот «список» — это объект, который можно изменить.

Словари в Python полагаются на то, что каждый ключ имеет хеш-значение, определенный идентификатор для ключа. Если ключ может измениться, это хеш-значение не будет надежным. Таким образом, ключи всегда должны быть неизменяемыми, хешируемыми типами данных, такими как числа или строки.

Пустой словарь

В словаре ничего не должно быть. Иногда нам нужно создать пустой словарь, когда мы планируем заполнить его позже на основе других входных данных.

Мы можем создать пустой словарь следующим образом:

>>empty_dict = {}

Добавить ключ

Чтобы добавить в словарь одну пару ключ: значение, мы можем использовать синтаксис:

dictionary[key] = value

Например, если бы у нас был словарь меню из первого упражнения:

>>menu = {"oatmeal": 3, "avocado toast": 6, "carrot juice": 5, "blueberry muffin": 2}

И мы хотели добавить новый товар, «чизкейк» за 8 долларов, мы могли бы использовать:

>>menu["cheesecake"] = 8

Теперь меню выглядит так:

>>{"oatmeal": 3, "avocado toast": 6, "carrot juice": 5, "blueberry muffin": 2, "cheesecake": 8}

Задание

1. Создайте пустой словарь с именем animals_in_zoo.
2. Гуляя по зоопарку, вы видите 8 зебр. Добавьте «зебры» в animals_in_zoo в качестве ключа со значением 8.
3. Добавьте «обезьяны» в animals_in_zoo в качестве ключа со значением 12.
4. Покидая зоопарк, вы опечалены тем, что не видели динозавров. Добавьте "динозавров" в animals_in_zoo в качестве ключа со значением 0.
5. Выведите в консоль animals_in_zoo.

Добавить несколько ключей

Если мы хотим добавить в словарь сразу несколько пар ключ: значение, мы можем использовать метод .update ().

Рассмотрим словарь с показаниями датчиков из предыдущего упражнения:

>>sensors = {"living room": 21, "kitchen": 23, "bedroom": 20}

Если бы мы хотели добавить 3 новые комнаты, мы могли бы использовать:

>>sensors.update({"pantry": 22, "guest room": 25, "patio": 34})

Это добавит все три элемента в словарь датчиков.

Теперь sensors выглядят так:

>>{"living room": 21, "kitchen": 23, "bedroom": 20, "pantry": 22, "guest room": 25, "patio": 34}

Задание

1. К словарю user_ids, представленному ниже, добавьте двух новых пользователей:

- theLooper, с id of 138475
- stringQueen, с id of 85739

>>ser_ids = {"teraCoder": 9018293, "proProgrammer": 119238}

2. Выведите в консоль результат

Перезаписать значения

Мы знаем, что можем добавить ключ, используя такой синтаксис:

>>menu["тост с авокадо "] = 7

Это создаст ключ «тост с авокадо» и установит значение, равное 7. Но что, если у нас уже есть запись «тост с авокадо» в словаре?

В этом случае наше присвоение значения перезапишет существующее значение, прикрепленное к ключу «тост с авокадо».

```
>>menu = {"oatmeal": 3, "avocado toast": 6, "carrot juice": 5, "blueberry muffin": 2}
>>menu["oatmeal"] = 5
>>print(menu)
```

Этоn код выдаст следующий результат:

```
{"oatmeal": 5, "avocado toast": 6, "carrot juice": 5, "blueberry muffin": 2}
```

Обратите внимание, что "oatmeal" поменялось на 5.

Задание

1. Добавьте к словарю oscar_winners ключ «"Supporting Actress"» и установите значение «Viola Davis».

oscar_winners = {"Best Picture": "La La Land", "Best Actor": "Casey Affleck", "Best Actress": "Emma Stone", "Animated Feature": "Zootopia"}

2. Не меняя определения словаря oscar_winners, измените значение, связанное с ключом «Best Picture», на «Moonlight».

Генерирование словаря

Допустим, у нас есть два списка, которые мы хотим объединить в словарь, например список студентов и список их роста в дюймах:

```
>>names = ['Jenny', 'Alexus', 'Sam', 'Grace']
>>heights = [61, 70, 67, 64]
```

Python позволяет создавать словарь, используя генерирование словаря, с таким синтаксисом:

```
>>students = {key:value for key, value in zip(names, heights)}
>>#students is now {'Jenny': 61, 'Alexus': 70, 'Sam': 67, 'Grace': 64}
```

Помните, что zip () объединяет два списка в итеративный тип, содержащий кортежи с элементами списка, соединенными вместе. Это генерирование словаря:

- Берет пару из итератора кортежей
- Называет элементы в паре ключом (тот, который изначально был из списка имен) и значение (тот, который был первоначально из списка роста студентов)
- Создает элемент "ключ: значение" в словаре студентов.
- Повторяет шаги 1-3 для всего итератора пар

Задание

1. У вас есть два списка, в которых представлены некоторые напитки, продаваемые в кафе, и миллиграммы кофеина в каждом. Сначала создайте переменную с именем zipped_drinks, которая является итеративным типом, включающим пары между списком напитков и списком кофеина.
2. Создайте словарь с именем drinks_to_caffeine, используя генерирование словаря, которое проходит через итератор zipped_drinks и превращает каждую пару кортежей в элемент ключ: значение.

```
drinks = ["espresso", "chai", "decaf", "drip"]
caffeine = [64, 40, 0, 120]
```

Заключение

На данный момент мы узнали:

- Как создать словарь
- Как добавить элементы в словарь
- Как обновить элементы в словаре
- Как использовать понимание списка для создания словаря из двух списков

Задание

1. Мы создаем музыкальный стриминговый сервис. Мы предоставили два списка, в которых представлены песни в библиотеке пользователя и количество воспроизведений каждой песни.

Используя генерирование словаря, создайте словарь под названием plays, который проходит через zip (песни, playcounts) и создает пару song: playcount для каждой песни в песнях и каждого playcount в playcounts.

songs = ["Like a Rolling Stone", "Satisfaction", "Imagine", "What's Going On", "Respect", "Goo
d Vibrations"]

playcounts = [78, 29, 44, 21, 89, 5]

2. Выведите plays.
3. После этого добавьте к plays новую запись. Запись должна быть для песни "Purple Haze", а количество воспроизведений - 1.
4. Этот пользователь заразился лихорадкой Ареты Франклин и прослушал «Respect» 5 раз.
5. Создайте словарь, называемый library, который имеет две пары ключ: значение:

«The Best Songs» со значением plays, (т.е. словарь, который вы создали использовать в качестве значения)

ключ "Sunday Feelings" со значением пустого словаря

6. Выведите library.