Python - słowniki

Dawid Kosiński

Czym są słowniki?

- Ang. **dictionary** - skrót. **dict**
- Są to zbiór par odpowiadające zasadzie: **klucz -> wartość**
- Klucz jest unikalny w danym słowniku - nie może się powtarzać
- Wartość odpowiada konkretnemu kluczowi
- Kluczem może być tylko i wyłącznie obiekt hashowalny, np. integer, str
- Wartością może być obiekt **dowolnego** typu, np. integer, str, list lub nawet inny słownik (dict)
- Stworzyć słownik można na dwa główne sposoby:
    - za pomocą **klamer** (nawiasy sześcienne): my_dictionary = {'klucz': 'wartość'}
    - za pomocą klasy **dict**: my_dictionary: **dict**(klucz='wartość')

Po co używać słowników?

- słowniki są głównie używane w celu przechowywania wartości pod danym kluczem i wyciąganie tych wartości za pomocą tego klucza:
  - >>> temp = {‘test’: 123}
    >>> temp.get(‘test’)
    >>> 123
- słowniki umożliwiają łatwy dostęp do wartości - nie trzeba zapamiętywać kolejności tworzenia nowych par klucz -> wartość, wystarczy że znamy klucz.

Kolejność dodawania par do słownika

- od całkiem niedawna (wersji 3.7) kolejność jest zależna od kolejności dodawania par, np. jeśli wpierw dodałeś do słownika parę ‘test’ -> ‘wartość a następnie ‘first’ -> ‘value’ to słownik zapamiętał iż wpierw dodano klucz ‘test’ a następnie ‘first’.
- we wcześniejszych wersjach nie było to zapewnione, klucze mogły być zwrócone w losowej kolejności

Sposoby dobrania się do wartości

- za pomocą methody .get(...) --- o tym dalej w prezentacji
- za pomocą nawiasów kwadratowych, np.
    - >>> temp = {‘test’: 123}
      >>> temp[‘test’]
      >>> 123

Sposoby dodawania nowych par do słownika

- za pomocą metody .update(...) ---- o tym dalej w prezentacji
- za pomocą nawiasów kwadratowych i operacji przypisania:
    - >>> temp = {‘test’: 123}
      >>> temp[‘new’] = 400
      >>> print(temp)
      >>> {‘test’: 123, ‘new’: 400}

Widoki słownikowe (dict views)

- są to obiekty, które zwracają aktualny stan słowników
- widoki można przypisać do zmiennych i nie martwić się, że będą one odwzorowywać nieaktualny stan słownika
- dostępne widoki na słownikach:
  - .**keys**() - zwraca widok, który przedstawia aktualne klucze w słowniku
  - .**values**() - zwraca widok, który przedstawia aktualne wartości w słowniku
  - .**items**() - zwraca widok, który przedstawia pary (klucz, wartość)

Przydatne metody słowników

- **.get**(‘klucz’) **.get**(‘klucz’, ‘wartość domyślna’)
  - **get()** zwraca nam wartość przypisaną do danego klucza. **Drugi parametr** (ten po przecinku) to wartość zwrócona w przypadku **nie znalezienia** klucza w słowniku. Jeśli nie podamy wartości domyślnej to zwrócony zostanie obiekt **None**.
- **.update**(...)
  - aktualizuje słownik za pomocą innego zbioru. Parametr może być innym **słownikiem** lub **zbiorem** co spełnia zasady tworzenia słownika (przedstawione na pierwszym slajdzie). W przypadku gdy dany klucz **już istnieje** w słowniku to jego wartość zostaje **nadpisana (stara wartość zostaje zapomniana)**.

Przydatne metody słowników

- **.pop**(‘klucz’) **.pop**(‘klucz’, ‘wartość domyślna’)
  - zwraca **wartość** przypisaną pod danym kluczem **jednocześnie usuwając** tą parę ze słownika. **Drugi parametr** (ten po przecinku) to wartość zwrócona w przypadku **nie znalezienia** klucza w słowniku. Jeśli nie podamy wartości domyślnej i nie zostanie znaleziony klucz to wystąpi błąd: **KeyError**.

Instrukcje warunkowe

- ‘klucz’ **in** my_dictionary
  - sprawdza czy klucz ‘klucz’ występuje w słowniku my_dictionary
- ‘cos’ **in** dict_view
  - sprawdza czy wartość ‘klucz lub wartość’ jest w danym widoku słownikowym: kluczach, wartościach lub zbiorze par (klucz, wartość)
- ‘klucz’: **not in** my_dictionary:
  - sprawdza czy klucz ‘klucz’ nie występuje w słowniku my_dictionary (odwrotność pierwszego punktu)
- ‘cos’ **not in** dict_view:
  - sprawdza czy wartość ‘cos’ nie jest w danym widoku słownikowym: kluczach, wartościach lub zbiorze par (odwrotność drugiego punktu)

Operacje na słownikach

- **len**(my_dictionary)
  - zwraca wielkość słownika. Wielkość słownika jest definiowana za pomocą ilości kluczy w nim.
- **list**(my_dictionary)
  - zwraca listę dostępnych kluczy w słowniku, jest adekwatne do: **list(my_dictionary.keys())**
- **Operacje dodawania, odejmowania dwóch słowników są niedostępne**

Iterowanie po słownikach

- **for** key **in** my_dictionary:
  - Powyższa pętla będzie tylko ‘przechodzić’ po **kluczach** w słowniku my_dictionary
- **for** key **in** my_dictionary.keys():
  - Powyższa pętla jest podobna do wcześniejszej pętli - przechodzi po **kluczach** w danym słowniku. Lecz wykorzystuje do tego widok słownikowy
- **for** value **in** my_dictionary.values():
  - Powyższa pętla przechodzi po wartościach w słowniku
- **for** key, value **in** my_dictionary.items():
  - Powyższa pętla przechodzi parach (klucz, wartość). Dane pary zostaną zwrócone za pomocą widoku słownikowego
- **Kolejne elementy są zwracane w kolejności dodania do słownika!**