

Dictionaries

Prof.Dr. Bahadır AKTUĞ
BME362 Introduction To Python

**Compiled from sources given in the references.*

Dictionaries

- ▶ One of the most powerful programming tools provided by Python must be the "dictionaries".
- ▶ Dictionaries are mutable types like the lists.
- ▶ Dictionaries can be expanded or can be shrunk.
- ▶ Dictionaries can contain lists as the lists can contain dictionaries.
- ▶ The most distinct features of dictionaries:
 - ▶ They are not sequential data types.
 - ▶ The elements of the dictionaries are accessed through "keys" not "indices".
 - ▶ Dictionaries are, in general, the application of "associative arrays" in programming.
 - ▶ Such data types have a (key, value) structure.
 - ▶ Dictionary concept can be named differently in other languages (hash table etc.)
 - ▶ The key has to be unique while values can be repeated as many as wanted

Dictionaries

```
▶ cities = {"Ankara": "06", "Adana": "01", "Samsun": "55"}
```

```
▶ grades = {"Ahmet KILIÇ": 76, "Veli Demir": 64, "Kazım Gök": 87}
```

```
>>> cities = {"Ankara": "06", "Adana": "01", "Samsun": "55"}
```

```
>>> cities["Ankara"]
```

```
'06'
```

```
>>> cities[0]
```



Not a sequential data type!
Cannot be accessed by the
index!

```
Traceback (most recent call last):
```

```
File "<stdin>", line 1, in <module>
```

```
KeyError: 0
```

```
>>> cities["Bursa"] = 16
```



A new element can be added
with a (key/value) pair

```
>>> print(cities)
```

```
{'Ankara': '06', 'Samsun': '55', 'Bursa': 16, 'Adana': '01'}
```

Dictionaries

```
>>> cities["Kocaeli"]
```



Undefined key!

Traceback (most recent call last):

File "<stdin>", line 1, in <module>

KeyError: 'K

```
>>> districts = {}
```



Defining an empty dictionary!

```
>>> print(districts)
```

```
{}
```

```
>>> dict = {"kırmızı":"red","yeşil":"green","mavi":"blue"}
```

```
>>> dict["kırmızı"]
```

```
'red'
```

```
>>> dict["mavi"]
```

```
'blue'
```


Dictionaries

```
>>> dict= {[1,2,3]:"EEE105"}
```

```
Traceback (most recent call last):
```

```
File "<stdin>", line 1, in <module>
```

mutable types cannot
be "keys"!



```
TypeError: unhashable type: 'list'
```

```
>>> dict = {(1,2,3):"EEE105"}
```

tuples can be keys



```
>>> dict[(1,2,3)]
```

```
'EEE105'
```

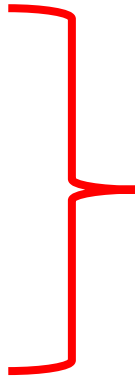
```
>>> dict = {1:2,3:5,9:10}
```

```
>>> dict = {"1":2, "3":5, "9":10}
```

```
>>> dict = {1:2, 3:5, 9:10}
```

```
>>> dict = {1:"2", 3:" 5" 9:"10"}
```

None of them corresponds
to the same dictionary
definiton!



Dictionaries

Some command that can be used with dictionaries:

Command/ Operator	Description
<code>len(d)</code>	Total number of elements (key, value pairs)
<code>del d[k]</code>	Deletes the element with a key named "k" and the associated value
<code>k in d</code>	checks whether there is a key named "k" in the dictionary "d" and returns True if there is
<code>k not in d</code>	checks whether there is a key named "k" in the dictionary "d" and returns True if there is not

Dictionaries

pop():

Since the "dictionaries" are not sequential unlike "lists", "pop" commands required an extra parameter.

It returns the value of the given key and deletes the key/value pair.

```
>>> capitals= {"Avusturya":"Viyana", "Almanya":"Berlin",  
"Hollanda":"Amsterdam"}
```

```
>>> capital = capitals.pop ("Avusturya")
```

```
>>> print(capital)
```

```
'Viyana'
```

```
>>> print(capitals)
```

```
{'Almanya': 'Berlin', 'Hollanda': 'Amsterdam'}
```

Dictionaries

popitem():

- ▶ This command does not take an input as opposed to "pop" command and applies the "pop" operation on a random element of the dictionary.
- ▶ Another difference is that "popitem()" returns the key/value pair instead of value as done with "pop" command.
- ▶ Thus, it returns key/value pair and deletes them from the dictionary.

```
>>> capitals= {"Avusturya":"Viyana", "Almanya":"Berlin",  
"Hollanda":"Amsterdam"}
```

```
>>> capitals.popitem()
```

```
('Almanya', 'Berlin')
```

```
>>> print(capitals)
```

```
{'Hollanda': 'Amsterdam', 'Avusturya': 'Viyana'}
```


Dictionaries

Working with non-existing keys:

When a non-existing key is used, an error message is returned.

```
>>> locations = {"Toronto" : "Ontario", "Vancouver":"British Columbia"}
```

```
>>> locations["Ottawa"]
```

Traceback (most recent call last):

```
File "<stdin>", line 1, in <module>
```

```
KeyError: 'Ottawa'
```

If you cannot make sure of the existence of a key, there are two workarounds:
(Why do we need workarounds here?)

1.

```
>>> if "Ottawa" in locations: print(locations["Ottawa"])
```

2.

```
>>> locations.get("Ottawa")
```

Dictionaries

Copying dictionaries:

Shallow copy is possible for the dictionaries too.

```
>>> words = { "cat":"Katze", "house":"Haus"}
```

```
>>> w = words ()
```

```
>>> words["cat"]="chat"
```

```
>>> print(w)
```

```
{'house': 'Haus', 'cat': 'Katze'}
```

```
>>> print(words)
```

```
{'house': 'Haus', 'cat': 'chat'}
```

Dictionaries

Deleting the content of a dictionary:

Note the the content is cleared not the dictionary itself!

```
>>> words = {"cat": "Katze", "house": "Haus"}
>>> words.clear()
>>> print(words)
{}
```

Updating Dictionaries (update)

```
>>> og1= {"Ali ÇELİK":{"EEE105","EEE106"},"Zeynep TAŞ": {"EEE106","EEE109"}}
>>> og2= {"Ali ÇELİK":{"EEE110","EEE109"},"Zeynep TAŞ":
{"EEE111","EEE110"},"Can Demir":{"EEE106"}}
>>> og1.update(og2)
>>> og1
{'Zeynep TAŞ': {'EEE111', 'EEE110'}, 'Can Demir': {'EEE106'}, 'Ali ÇELİK': {'EEE110', 'EEE109'}}
```

Dictionaries

Loops over the dictionary keys:

```
>>> d = {"a":123, "b":34, "c":304, "d":99}
>>> for key in d:
...     print(key)
...
b
c
a
d
```

```
>>> for key in d.keys():
...     print(key)
...
b
c
a
d
```

Loops over the dictionary key/value pairs:

```
>>> d = {"a":123, "b":34, "c":304, "d":99}
>>> for k in d.items():
...     print(k)
...
('b', 34)
('c', 304)
('a', 123)
('d', 99)
```

```
>>> for k in d.values():
...     print(k)
...
34
304
123
99
```

Dictionaries

Converting a dictionary to a list:

```
>>> d = {"a":123,"b":547,"c":878}
```

```
>>> list(d)
```

```
['b', 'c', 'a']
```

```
>>> list(d.items())
```

```
[('b', 547), ('c', 878), ('a', 123)]
```

```
>>> list(d.keys())
```

```
['b', 'c', 'a']
```

```
>>> list(d.values())
```

```
[547, 878, 123]
```

Dictionaries

Converting a dictionary to a list or vice versa:

```
>>> cities = ["Bursa", "Kayseri", "Gaziantep", "Konya", "Urfa"]
>>> meals = ["İskender", "Mantı", "Lahmacun", "Etli Ekmek", "Kebap"]
>>> list(zip(cities, meals))
[('Bursa', 'İskender'), ('Kayseri', 'Mantı'), ('Gaziantep', 'Lahmacun'),
 ('Konya', 'Etli Ekmek'), ('Urfa', 'Kebap')]
>>> dict(list(zip(cities, meals)))
{'Gaziantep': 'Lahmacun', 'Urfa': 'Kebap', 'Kayseri': 'Mantı', 'Bursa':
 'İskender', 'Konya': 'Etli Ekmek'}
```

OR

```
>>> dict(zip(cities, meals))
{'Gaziantep': 'Lahmacun', 'Urfa': 'Kebap', 'Kayseri': 'Mantı', 'Bursa':
 'İskender', 'Konya': 'Etli Ekmek'}
```

► References

- 1 Wentworth, P., Elkner, J., Downey, A.B., Meyers, C. (2014). *How to Think Like a Computer Scientist: Learning with Python (3rd edition)*.
- 2 Pilgrim, M. (2014). *Dive into Python 3 by*. Free online version: DiveIntoPython3.org ISBN: 978-1430224150.
- 3 Summerfield, M. (2014) *Programming in Python 3 2nd ed (PIP3)* :- Addison Wesley ISBN: 0-321-68056-1.
- 4 Summerfield, M. (2014) *Programming in Python 3 2nd ed (PIP3)* :- Addison Wesley ISBN: 0-321-68056-1.
- 5 Jones E, Oliphant E, Peterson P, et al. *SciPy: Open Source Scientific Tools for Python, 2001-*, <http://www.scipy.org/>.
- 6 Millman, K.J., Aivazis, M. (2011). *Python for Scientists and Engineers, Computing in Science & Engineering*, 13, 9-12.
- 7 John D. Hunter (2007). *Matplotlib: A 2D Graphics Environment, Computing in Science & Engineering*, 9, 90-95.
- 8 Travis E. Oliphant (2007). *Python for Scientific Computing, Computing in Science & Engineering*, 9, 10-20.
- 9 Goodrich, M.T., Tamassia, R., Goldwasser, M.H. (2013). *Data Structures and Algorithms in Python*, Wiley.
- 10 <http://www.diveintopython.net/>
- 11 <https://docs.python.org/3/tutorial/>
- 12 <http://www.python-course.eu>
- 13 <https://developers.google.com/edu/python/>
- 14 <http://learnpythonthehardway.org/book/>