# Programming 1

Laszlo SZATHMARY
University of Debrecen
Faculty of Informatics

Lab #5
- set
- dictionary

(last update: 2022-02-03 [yyyy-mm-dd])

2021-2022, 2nd semester

# set

```
>>> basket = ['apple', 'ananas', 'banana', 'apple', 'orange', 'banana']
>>> fruits = set(basket)
>>> fruits
set(['orange', 'ananas', 'apple', 'banana'])
>>> type(fruits)
<type 'set'>
>>> li = list(fruits)
>>> li
['orange', 'ananas', 'apple', 'banana']
>>> type(li)
<type 'list'>
>>> sorted(li)
['ananas', 'apple', 'banana', 'orange']
>>> fruits
set(['orange', 'ananas', 'apple', 'banana'])
>>> 'kiwi' in fruits
False
>>> 'apple' in fruits
True
```

removing
duplicates

Is an element
in the set?

Exercise:
Consider the following elements: `[5, 2, 3, 5, 1, 4, -200, 5, 1, 3, 2, 2, 5]`.
Remove the duplicates, i.e. one element can be present in the result
at most once. The elements in the result should be sorted.     (set01)

```
>>> a = ['apple', 'banana', 'lemon']
>>> a = set(a)
>>> a
set(['lemon', 'apple', 'banana'])
>>> b = set()
>>> b.add('banana')
>>> b.add('orange')
>>> b
set(['orange', 'banana'])
>>> a.union(b)
set(['orange', 'lemon', 'apple', 'banana'])
>>> a.intersection(b)
set(['banana'])
>>> a.difference(b)
set(['lemon', 'apple'])
>>> a
set(['lemon', 'apple', 'banana'])
>>> a.remove('lemon')
>>> a
set(['apple', 'banana'])
```

empty set;
extending a set

classic
set operations
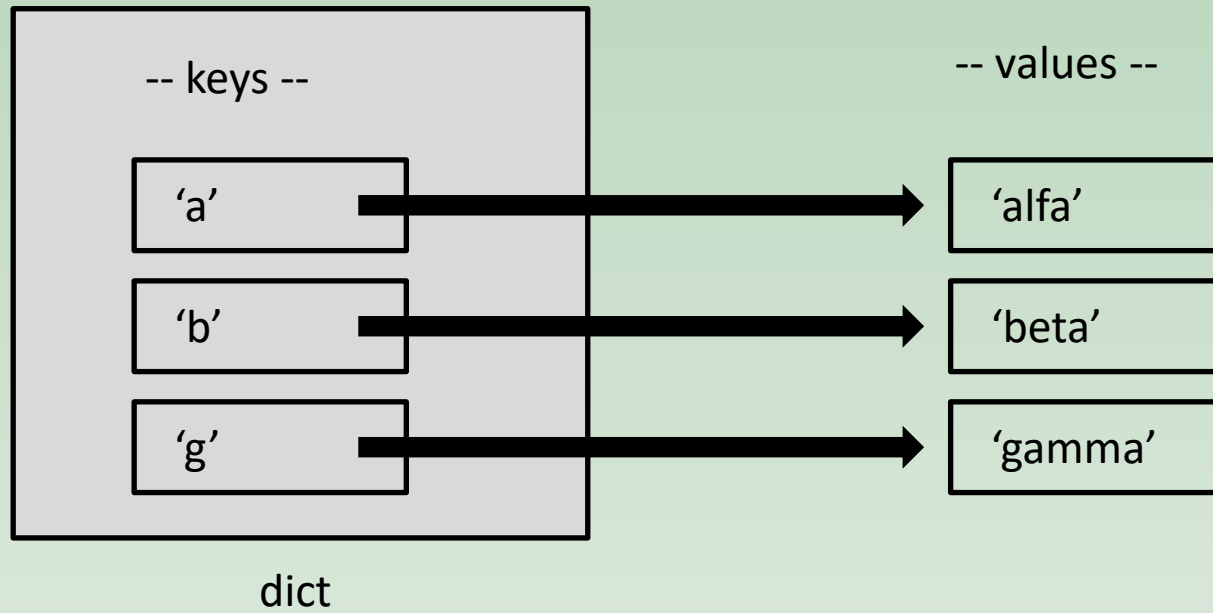
removing an element

# dictionary

```
 1   >>> d = { }
 2   >>> d['a'] = 'alfa'
 3   >>> d['b'] = 'beta'
 4   >>> d['g'] = 'gamma'
 5   >>> d
 6   {'a': 'alfa', 'b': 'beta', 'g': 'gamma'}
 7   >>> d['a']
 8   'alfa'
 9   >>> d['o']
10   Traceback (most recent call last):
11     File "<stdin>", line 1, in <module>
12   KeyError: 'o'
13   >>> d.get('o')
14   >>> d.get('a')
15   'alfa'
16   >>> 'a' in d
17   True
18   >>> 'o' in d
19   False
20   >>> d['a'] = 'ALFA'
21   >>> d['a']
22   'ALFA'
23   >>> d
24   {'a': 'ALFA', 'b': 'beta', 'g': 'gamma'}
```

empty dictionary
( or: `d = dict()` )

Is an element with the given key in the dictionary?

```
>>> d
{'b': 'beta', 'g': 'gamma', 'a': 'alfa'}
>>> d['o'] = 'omega'
>>> d
{'o': 'omega', 'b': 'beta', 'g': 'gamma', 'a': 'alfa'}
>>> d.keys()
dict_keys(['o', 'b', 'g', 'a'])
>>> type(d.keys())
<class 'dict_keys'>
>>> d.values()
dict_values(['omega', 'beta', 'gamma', 'alfa'])
>>> type(d.values())
<class 'dict_values'>
>>> for k in sorted(d.keys()):
...     print("key:", k, "=>", d[k])
...
...
key: a => alfa
key: b => beta
key: g => gamma
key: o => omega
>>> d.items()
dict_items([('o', 'omega'), ('b', 'beta'), ('g', 'gamma'), ('a', 'alfa')])
>>> type(d.items())
<class 'dict_items'>
>>> for k, v in d.items():
...     print(k, v)
...
...
o omega
b beta
g gamma
a alfa
>>> list(d.keys())
['o', 'b', 'g', 'a']
```

the order of the elements
is arbitrary

*iterators:*
d.keys()
d.values()
d.items()

using iterators
in a loop

Iterator to list.
The order is arbitrary.
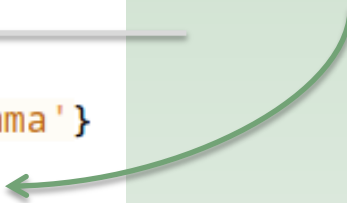
**HW:** dict1.py

6

# removing an element from a dictionary

```
1  >>> a = 6
2  >>> a
3  6
4  >>> del a
5  >>> a
6  Traceback (most recent call last):
7    File "<stdin>", line 1, in <module>
8  NameError: name 'a' is not defined
9  >>>
10 >>> li = range(5)
11 >>> li
12 [0, 1, 2, 3, 4]
13 >>> del li[-1]
14 >>> li
15 [0, 1, 2, 3]
16 >>>
17 >>> d
18 {'a': 'alfa', 'b': 'beta', 'o': 'omega', 'g': 'gamma'}
19 >>> del d['b']
20 >>> d
21 {'a': 'alfa', 'o': 'omega', 'g': 'gamma'}
```

# Exercises

1. [20120904a] removing duplicates (set)
2. [20120905a] dictionary #1
3. [20120921a] accent removal
4. [20130218b] certain characters
5. [20120818h] one hundred 50-digit long numbers (PE #13) [version **A** only]
6. [20120816a] 8 queens