

Engineering programming

Python basics - 2

Thibault Thétier - thibault.thetier@vub.be

Tom Godden - tgodden@vub.be

1ste semester 2021

TUPLES

Tuples are **fixed-length**, **ordered** and **unchangeable** sequences of python objects.

- ▶ `T = 10, 20, 30, 40, 50`
- ▶ `T = (10, 20, 30, 40, 50)`
- ▶ `T[1]`
- ▶ `T[1]=21`
- ▶ `nst_tup = (4, 5, 6), (7, 8)`
- ▶ `tuple([4, 0, 2])`
- ▶ `tup = tuple('string')`

- ▶ `tup + (1, [2,3])`
- ▶ `tup = (4, 5, 6)`
- ▶ `a, b, c = tup`
- ▶ `tup = 4, 5, (6, 7)`
- ▶ `a, b, (c, d) = tup`
- ▶ `values = 1, 2, 3, 4, 5`
- ▶ `a, b, *rest = values`

ZIP

zip “pairs” up the elements of a number of lists, tuples, or other sequences to create a list of tuples:

- ▶ `seq1 = ['foo', 'bar', 'baz']`
- ▶ `seq2 = ['one', 'two', 'three']`
- ▶ `zipped = zip(seq1, seq2)`
- ▶ `list(zipped)`
- ▶ `[('foo', 'one'), ('bar', 'two'), ('baz', 'three')]`

DICTIONARIES

Dict is likely the most important built-in Python data structure. A dictionary is an **unordered**, **changeable** and **indexed** collection. Dictionaries have keys and values.

```
dict1 = dict()
```

```
dict1 = {}
```

```
1 car = {  
2     "brand": "Ford",  
3     "model": "Mustang",  
4     "year": 1964  
5 }
```

Receive value:

```
model = car["model"]
```

Change value:

```
car["year"] = 2018
```

Insert elements:

```
car["doors"] = 2
```

OPERATIONS ON DICTS

- ▶ `d1 = {'a' : 'some value', 'b' : [1, 2, 3, 4]}`
- ▶ `d1[5] = 'some value'`
- ▶ `d1['dummy'] = 'another value'`
- ▶ `del d1[5]`

- ▶ `ret = d1.pop('dummy')`
- ▶ `list(d1.keys())`
- ▶ `list(d1.values())`
- ▶ `d1.update({'b' : 'foo', 'c' : 12})`

LOOPS WITH DICTIONARIES

You can loop over a dictionary in different ways:

```
1 for key in car:  
2     print(key)
```

```
1 for key in car:  
2     print(car[key])
```

```
1 mapping = {}  
2 for key, value in zip(key_list, value_list):  
3     \quad mapping[key] = value  
4 mapping = dict(zip(key_list, value_list))
```

```
1 for value in car.values():  
2     print(value)
```

```
1 for key, value in car.items():  
2     print(key, value)
```

SETS

Sets are collections **without indexation** and **without duplicates**.
You can think of sets as dicts, but with keys only.

```
S = set([10, 20, 30])
```

```
S = {10, 20, 30}
```

► Add elements:

`S.add(15)` Add a single element

`S.update([12,25,40])` Add multiple elements

`S.update([1,2,3],{4,5,6})` Merging list and set

► Remove elements:

`S.discard(40)` if the element does not exist, there is no change

`S.remove(3)` gives an error if element does not exist