



PyQB

Monga

Dictionaries

Sets

Comprehensions

Types,
docstrings,
doctests

Files

Programming in Python¹

Mattia Monga

Dip. di Informatica
Università degli Studi di Milano, Italia

mattia.monga@unimi.it

Academic year 2025/26, I semester

¹ © ⓘ 2025 M. Monga. Creative Commons Attribuzione — Condividi allo stesso modo 4.0 Internazionale. <http://creativecommons.org/licenses/by-sa/4.0/deed.it>



PyQB

Monga

Dictionaries

Sets

Comprehensions

Types,
docstrings,
doctests

Files

Lecture VIII: Other Composite Objects

State of the homework



PyQB

Monga

Dictionaries

Sets

Comprehensions

Types,
docstrings,
doctests

Files

- 1 Click on the link and accept the assignment: this will create your own git repository on GitHub with the homework
- 2 Clone the repository on your machine (the easiest way is to use the GitHub Desktop app)
- 3 Work on the solution in the file `exercise.py`
- 4 Commit your work (again with GitHub Desktop it is easy)
- 5 Push (again with GitHub Desktop it is easy) the solution on GitHub so I can comment on it
- 6 Read my comments
- 7 If the solution can be improved, go back to step 3



Dictionaries

A composite type `dict` that implements a `mapping` between immutable `keys` and `values`.

```
d = {'key': 'foo', 3: 'bar'}
```

```
print(d['key']) # 'foo'
print(d[3])     # 'bar'
print(d[2])     # error!
```

Notation is similar to lists/tuples, but `dicts` are not sequences indexed by numbers, you must use only the existing keys (`d.keys()`).

```
if x in d.keys():
    print(d[x])
```

A sequence of values can be obtained with `d.values`. A sequence of 2-tuples (key, value) with `d.items()`.

PyQB

Monga

Dictionaries

Sets

Comprehensions

Types,
docstrings,
doctests

Files



PyQB

Monga

Dictionaries

Sets

Comprehensions

Types,
docstrings,
doctests

Files

A **set** is a composite object with no duplicate (non mutable) elements. Common set operations are possible.

- Set literals: `{1,2,3}` **set**()
- `{1,2,3}.union({3,5,6})`
`{1,2,3}.intersection({3,5,6})`



Comprehensions

Comprehensions are a concise way to create lists, sets, maps... It resembles the mathematical notation used for sets

$$A = \{a^2 | a \in \mathbb{N}\}.$$

```
squares = [x**2 for x in range(10)]
```

equivalent to:

```
squares = []
```

```
for x in range(10):  
    squares.append(x**2)
```

filtering is possible

```
odds = [x for x in range(100) if x % 2 != 0]
```

with a set

```
s = {x for x in range(50+1) if x % 5 == 0}
```

with a dict

```
d = {x: x**2 for x in range(10)}
```

PyQB

Monga

Dictionaries

Sets

Comprehensions

Types,
docstrings,
doctests

Files

Make a program readable



PyQB

Monga

Dictionaries

Sets

Comprehensions

Types,
docstrings,
doctests

Files

You never write a program only for a machine! You, others, tools will *read* the program for different purposes. Every minute spent in making a program more understandable pays off hours saved later.

- **Type hinting** makes clear what a function needs to work properly, and what it produces
- **Documentation** helps understanding without the need to read implementation details
- **Examples of use** make easy to remember how to use a function and can be used for verification

Example



PyQB

Monga

Dictionaries

Sets

Comprehensions

Types,
docstrings,
doctests

Files

```
Num = int | float
```

```
def cube(x: Num) -> Num:  
    """Return the cube of x.
```

```
>>> cube(-3)  
-27
```

```
>>> abs(cube(0.2) - 0.008) < 10e-5  
True  
"""
```

```
return x * x * x
```

Examples can be tested by:

```
python -m doctest filename.py.
```



A **file** is an abstraction the operating system uses to preserve data among the execution of programs. Data must be accessed **sequentially**. (Italian reading people might enjoy **this**)

- We need commands to ask to the OS to give access to a file (**open**).
- It is easy to read or write data **sequentially**, otherwise you need special commands (**seek**) to move the file “cursor”
- The number of open files is limited ($\approx$ thousands), thus it is better to **close** files when they are not in use

Files contain bits (normally considered by group of bytes, 8 bits), the interpretation (“format”) is given by the programs which manipulate them. However, “lines of printable characters” (**plain text**) is a rather universal/predefined interpretation, normally the easiest to program.

PyQB

Monga

Dictionaries

Sets

Comprehensions

Types,
docstrings,
doctests

Files



File read access

```
f = open('filename.txt', 'r') # read only

# iterating on a file reads (all) the lines
for i in f:
    print(i)

# End of file already reached, result is ''
f.readline()

f.close()

# File closed, error!
f.readline()
```

To avoid remembering to close explicitly, Python provides the **context manager** syntax.

```
with open('filename.txt', 'r') as f:
    for i in f:
        print(i)
```

PyQB

Monga

Dictionaries

Sets

Comprehensions

Types,
docstrings,
doctests

Files