

PyQB

Monga

Dictionaries

Sets

Comprehension

Programming in Python¹

Mattia Monga

Dip. di Informatica Università degli Studi di Milano, Italia mattia.monga@unimi.it

Academic year 2024/25, I semester

¹ ⊕⊕ 2024 M. Monga. Creative Commons Attribuzione — Condividi allo stesso modo 4.0 Internazionale. http://creativecommons.org/licenses/by-tsa/4.0/deed it → (Ξ→ Ξ →) < ...



PyQB

Monga

Dictionaries

Sets

Comprehension

Lecture VII: Other Composite Objects

<ロト 4 団 ト 4 臣 ト 4 臣 ト 王 · の 4 で 50

State of the homework

Students Solved by One Triangle 16 8

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



Monga Dictionaries Sets

Dictionaries

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

}

d = {'key': 'foo	o',	3: 'bar	. 1
<pre>print(d['key'])</pre>	#	'foo'	
<pre>print(d[3])</pre>	#	'bar'	
<pre>print(d[2])</pre>	#	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().



Monga

Dictionaries

Sets

Comprehension



PyQB

Monga

Dictionaries

Sets

Comprehension

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 \text{ for } x \text{ in range}(100) \text{ if } x \% 2 != 0]
# with a set
s = {x \text{ for } x \text{ in range}(50+1) \text{ if } x \% 5 == 0}
# with a dict
d = \{x: x + 2 \text{ for } x \text{ in range}(10)\} \in \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R}
```



Comprehensions