CITY
UNIVERSITY OF LONDON
EST 1894

STEM Digital Academy

School of Science & Technology

www.city.ac.uk

Dictionaries

Programming and Algorithms

Lecture by

Dr Daniil Osudin

```
n = 3
for i in range(1,n+1):
    print("Hello World!")
```

```
Hello World!
Hello World!
Hello World!
```

What will we Cover?

- Introduction to the dictionary data structure
- Storing data in dictionaries

What is a Dictionary?

- Collection of data where each value has an associative key
- Rather than the index, data is accessed via a key
  - Keys can be of int , float or string type
  - Numeric keys can be non-sequential
  - All keys must be unique, but the values can be the same
  - values can be numeric, strings, Boolean, list, set, tuple or another dictionary

What is the Purpose of Dictionaries?

Useful when working with large data files, sending and receiving data

Examples:

- User records on a website
  - Key: username
  - Value: account information
- Maps
  - Key: coordinates
  - Value: objects located there

Creating a Dictionary

- Defined inside curly brackets { } with colons : separating a key from its value
- Commas separate the key-value pairs

```
# dictionary with 3 key-value pairs
d = {1: "b", 2: "m", 3: "c"}

#dictionary with one pair
singer = {1 : "Sam Smith"}

#empty dictionary
empty_dictionary = {}
```

Dictionary from a Sequence

- `dict()` – is used to convert a sequence of pairs into a dictionary
- Works on lists and tuples

Example	Result
`d1 = dict(((1, "a"), (2, "b"), (3, "c")))`	`{1: 'a', 2: 'b', 3: 'c'}`
`d2 = dict([[1, "a"], [2, "b"], [3, "c"]])`	`{1: 'a', 2: 'b', 3: 'c'}`
`d3 = dict([(1, "a"), (2, "b"), (3, "c")])`	`{1: 'a', 2: 'b', 3: 'c'}`

Accessing Values in Dictionaries

- To access a value, use its associative key
  - Use the key inside square brackets - [<key>]
- Alternatively, use the `get()` method

Example	Result
`d4 = {1: 'a', 2: 'b', 3: 'c'}` `print(d4[1])`	`a`
`d5 = {'a': 1, 'b': 2, 'c': 3}` `d5.get('c')`	`3`

Modifying Values in Dictionaries

- To modify a value, use its associative key with new value to be assigned

Example	Result
`d6 = {1: True, 2: False, 3: True}` `d6[1] = False`	`{1: False, 2: False, 3: True}`
`d7 = {'a': 1, 'b': 2, 'c': 3}` `d7['a'] = 5`	`{'a': 5, 'b': 2, 'c': 3}`

Removing Entries

- `pop()` – used to remove a value with the specified key, returns the value of the key being removed
- `del <dictionary name> [<key>]` – used to remove a value with the specified key, does not return anything

Example	Result
`d8 = {1: True, 2: False, 3: True}` `d8.pop(1)`	`True` `{2: False, 3: True}`
`d9 = {'a': 1, 'b': 2, 'c': 3}` `del d9['b']`	`{'a': 5, 'c': 3}`

Try It Yourself

Enter and run the following statements in the python environment:

```
d = {"movie": "Forrest Gump", "artist": "Tom Hanks", "year": 1994}
print(d["artist"])
print(d.get("year"))
d["artist"] = "Robin Wright"
print(d)
d.pop("artist")
print(d)
```