National Centre for Atmospheric Science
NATURAL ENVIRONMENT RESEARCH COUNCIL

Centre for Environmental Data Archival
SCIENCE AND TECHNOLOGY FACILITIES COUNCIL
NATURAL ENVIRONMENT RESEARCH COUNCIL

Dictionaries

Extracted from material by:

Use a *dictionary*

An unordered collection of key/value pairs

Like set elements, keys are:

Use a *dictionary*

An unordered collection of key/value pairs

Like set elements, keys are:

- Immutable

Use a *dictionary*

An unordered collection of key/value pairs

Like set elements, keys are:

- Immutable

- Unique

Use a *dictionary*

An unordered collection of key/value pairs

Like set elements, keys are:

- Immutable

- Unique

- Not stored in any particular order

Use a *dictionary*

An unordered collection of key/value pairs

Like set elements, keys are:

- Immutable

- Unique

- Not stored in any particular order

No restrictions on values

Use a *dictionary*

An unordered collection of key/value pairs

Like set elements, keys are:

- Immutable

- Unique

- Not stored in any particular order

No restrictions on values

- Don't have to be immutable or unique

Create a dictionary by putting key:value pairs in {}

Create a dictionary by putting key:value pairs in {}

```
>>> birthdays = {'Newton' : 1642, 'Darwin' : 1809}
```

Create a dictionary by putting key:value pairs in {}

```
>>> birthdays = {'Newton' : 1642, 'Darwin' : 1809}
```

Retrieve values by putting key in []

Create a dictionary by putting key:value pairs in {}

```
>>> birthdays = {'Newton' : 1642, 'Darwin' : 1809}
```

Retrieve values by putting key in []

Just like indexing strings and lists

Create a dictionary by putting key:value pairs in {}

```
>>> birthdays = {'Newton' : 1642, 'Darwin' : 1809}
```

Retrieve values by putting key in []

Just like indexing strings and lists

```
>>> print birthdays['Newton']
1642
```

Create a dictionary by putting key:value pairs in {}

```
>>> birthdays = {'Newton' : 1642, 'Darwin' : 1809}
```

Retrieve values by putting key in []

Just like indexing strings and lists

```
>>> print birthdays['Newton']
1642
```

Just like using a phonebook or dictionary

Add another value by assigning to it

Add another value by assigning to it

```
>>> birthdays['Turing'] = 1612     # that's not right
```

Add another value by assigning to it

```
>>> birthdays['Turing'] = 1612     # that's not right
```

Overwrite value by assigning to it as well

Add another value by assigning to it

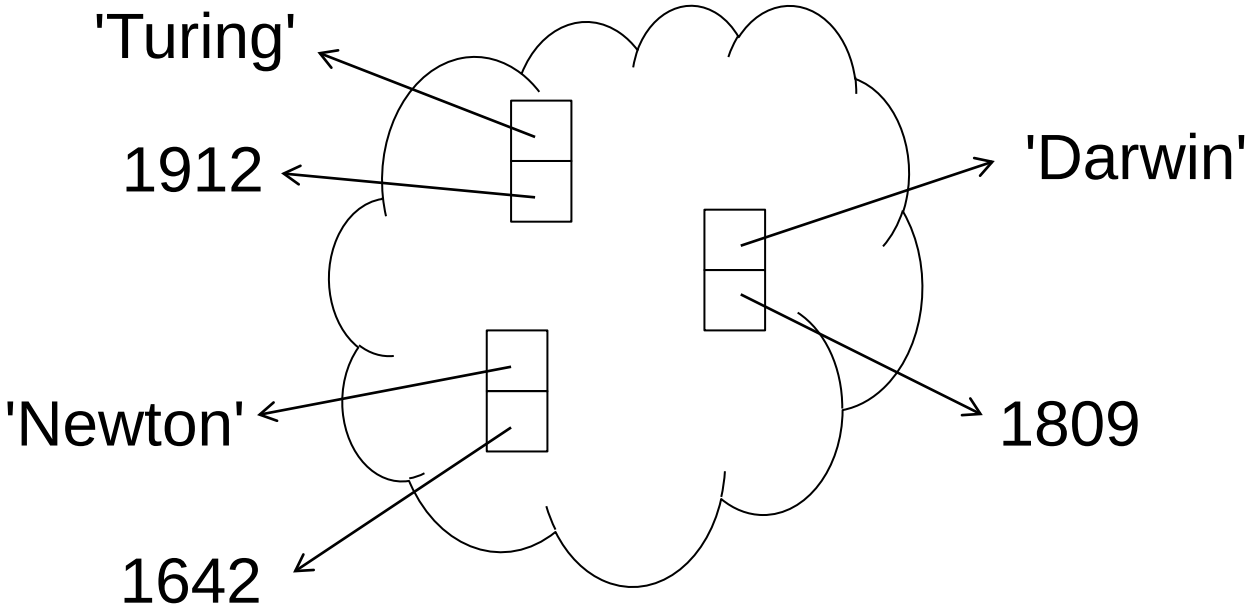
```
>>> birthdays['Turing'] = 1612     # that's not right
```

Overwrite value by assigning to it as well

```
>>> birthdays['Turing'] = 1912
>>> print birthdays
{'Turing' : 1912, 'Newton' : 1642, 'Darwin' : 1809}
```

Note: entries are *not* in any particular order

Note: entries are *not* in any particular order



Key must be in dictionary *before* use

Key must be in dictionary *before* use

```
>>> birthdays['Nightingale']
```

KeyError: 'Nightingale'

Key must be in dictionary *before* use

```
>>> birthdays['Nightingale']
```

KeyError: 'Nightingale'

Test whether key is present using in

Key must be in dictionary *before* use

```
>>> birthdays['Nightingale']
KeyError: 'Nightingale'
```

Test whether key is present using in

```
>>> 'Nightingale' in birthdays
False
>>> 'Darwin' in birthdays
True
```

Use for to loop over keys

Use for to loop over keys

Unlike lists, where for loops over values

Use for to loop over keys

Unlike lists, where for loops over values

```
>>> for name in birthdays:
...   print name, birthdays[name]
```

Turing 1912

Newton 1642

Darwin 1809