Tuples

- fields of different types
- fixed number of fields

- fields listed in parentheses

Tuple Example

- album title, year, label

```
In [1]: ('Revolver', 1966, 'Parlophone')
Out[1]: ('Revolver', 1966, 'Parlophone')
```

Accessing Fields

- indexed like lists

Accessing Fields Example

```
In [2]: album = ('Revolver', 1966, 'Parlophone')

In [3]: album[0]
Out[3]: 'Revolver'

In [4]: year = album[1]

In [5]: year
Out[5]: 1966
```

Tuple Immutability

- tuples are immutable

```
In [6]: album[2] = 'EMI'
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent
<ipython-input-6-7e280d6480e5> in <module>()
----> 1 album[2] = 'EMI'

TypeError: 'tuple' object does not support item assignment
```

Dictionaries

- mapping from keys to values
- key-value pairs listed in curly braces

Dictionary Example

```
In [7]: {'title': 'Revolver', 'year': 1966, 'label': 'Parlophone
Out[7]: {'label': 'Parlophone', 'title': 'Revolver', 'year': 196
```

Accessing Items

- indexing by key gives value
- non-existing key: `KeyError`

Accessing Item Example

```
In [8]: album = {'title': 'Revolver', 'year': 1966, 'label': 'Pa

In [9]: album['year']
Out[9]: 1966

In [10]: album['minutes']
---------------------------------------------------------------
KeyError                                  Traceback (most recent
<ipython-input-10-2fa58735413a> in <module>()
----> 1 album['minutes']

KeyError: 'minutes'
```

Updating Dictionaries

- dictionaries are mutable
- value for a key can be updated

```
In [11]: album['label'] = 'EMI'

In [12]: album
Out[12]: {'label': 'EMI', 'title': 'Revolver', 'year': 1966}
```

Adding Keys

- new keys can be added

```
In [13]: album['minutes'] = 35

In [14]: album
Out[14]: {'label': 'EMI', 'minutes': 35, 'title': 'Revolver', 'y
```

Deleting Keys

- existing keys can be deleted

```
In [15]: del album['label']

In [16]: album
Out[16]: {'minutes': 35, 'title': 'Revolver', 'year': 1966}
```

Classes

- combine data with operations on the data
- encapsulation

- data: attributes
- operations: methods

Objects

- objects are instances of a class
- creating an object: instantiation

- syntax: similar to a function call

```
ClassName(PARAMETERS)
```

Class Example

- `datetime` is a class

```
In [17]: from datetime import datetime

In [18]: datetime(2000, 12, 31, 23, 59)
Out[18]: datetime.datetime(2000, 12, 31, 23, 59)
```

Object Example

- epoch is an instance of `datetime`

```
In [19]: epoch = datetime(1970, 1, 1, 0, 0)

In [20]: epoch
Out[20]: datetime.datetime(1970, 1, 1, 0, 0)
```

Accessing Attributes

- using the dot notation

```
OBJECT.attribute
OBJECT.method(PARAMETERS)
```

Accessing Attributes Example

```
In [21]: epoch.year
Out[21]: 1970

In [22]: epoch.hour
Out[22]: 0

In [23]: epoch.ctime()
Out[23]: 'Thu Jan  1 00:00:00 1970'
```

Classes vs Functions

```
In [24]: current_time = datetime.now()

In [25]: current_time.ctime()
Out[25]: 'Fri Dec 15 14:55:14 2017'
```

```
In [26]: from time import localtime, asctime

In [27]: current_time = localtime()

In [28]: asctime(current_time)
Out[28]: 'Fri Dec 15 14:55:14 2017'
```

String Operations

- checking for properties
- converting case
- stripping
- splitting
- replacing

String Properties

```
In [29]: title = 'Jabberwocky'

In [30]: title.isalpha()
Out[30]: True

In [31]: title.isdigit()
Out[31]: False
```

Case Conversion

```
In [32]: title.lower()
Out[32]: 'jabberwocky'

In [33]: title.upper()
Out[33]: 'JABBERWOCKY'
```

String Properties

```
In [34]: title = '   Jabberwocky  '

In [35]: title.lstrip()
Out[35]: 'Jabberwocky  '

In [36]: title.rstrip()
Out[36]: '   Jabberwocky'

In [37]: title.strip()
Out[37]: 'Jabberwocky'
```

String Splitting

```
In [38]: title = 'Monty Python and The Holy Grail'

In [39]: title.split()
Out[39]: ['Monty', 'Python', 'and', 'The', 'Holy', 'Grail']

In [40]: title.split(' and ')
Out[40]: ['Monty Python', 'The Holy Grail']
```

Substring Replacement

```
In [41]: title = "Monty Python's The Meaning of Life"

In [42]: title.replace('The Meaning of Life', 'Flying Circus')
Out[42]: "Monty Python's Flying Circus"
```