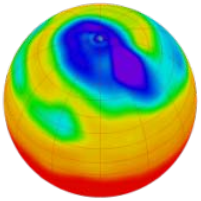
National Centre for Atmospheric Science
NATURAL ENVIRONMENT RESEARCH COUNCIL



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

Object-Oriented Programming (OOP)

Thanks to all contributors:

Alison Pamment, Sam Pepler, Ag Stephens, Stephen Pascoe, Kevin Marsh, Anabelle Guillory, Graham Parton, Esther Conway, Eduardo Damasio Da Costa, Wendy Garland, Alan Iwi and Matt Pritchard.

National Centre for Atmospheric Science
NATURAL ENVIRONMENT RESEARCH COUNCIL

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

Let’s see how OOP is useful in everyday Python:

```
>>> s = "some silly string"
>>> s.upper()
'SOME SILLY STRING'
>>> s.find("t")
12
>>> s.replace("silly", "sensible").title()
'Some Sensible String'
```

And you can actually interrogate this object s to find out their methods:

```
>>> dir(s)
['__add__', '__class__', '__contains__',
'__delattr__', ..., '__str__', '__subclasshook__',
'_formatter_field_name_split', '_formatter_parser',
'capitalize', 'center', 'count', 'decode', 'encode',
'endswith', 'expandtabs', 'find', 'format', 'index',
'isalnum', 'isalpha', 'isdigit', 'islower', 'isspace',
'istitle', 'isupper', 'join', 'ljust', 'lower',
'lstrip', 'partition', 'replace', 'rfind', 'rindex',
'rjust', 'rpartition', 'rsplit', 'rstrip', 'split',
'splitlines', 'startswith', 'strip', 'swapcase',
'title', 'translate', 'upper', 'zfill']
```

And you can find out which class *s* is an instance of:

```
>>> type(s)
<type 'str'>
```

You can build your own **class** for your own domain:

```
class FileAnalyser(object):
    "A class above the rest"
    def __init__(self, path):
        items = open(path).read().split()
        self.data = []
        for item in items:
            self.data.append(float(item))

    def max(self):
        return max(self.data)

    def mean(self):
        return sum(self.data) / len(self.data)
```

Then create an **instance** of your **class** and use it:

```
$ cat some_data.txt   # Inside the data file...
1000 750 500 250 0

$ python
>>> da = DataAnalyser("some_data.txt")
>>> da.max()
1000.0
>>> da.mean()
500.0
```

You can make use of help() on your own class:

```
>>> help(FileAnalyser)
Help on class FileAnalyser in module __main__:

class FileAnalyser(__builtin__.object)
 |  A class above the rest
 |
 |  Methods defined here:
 |
 |  __init__(self, path)
 |
 |  max(self)
 |
 |  mean(self)
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors defined here:
 |
```