Ex 1

Python provides several global functions that allow you to manage attributes on an object

Ex 2

You can also add and remove attributes on an object directly.

Ex 3

Every class provides metadata via the following built-in attributes
- You can also get metadata about an object too

There are various "special" methods you can implement in your Python classes
- These methods allow your class objects to take advantage of standard Python idioms

It's good practice to implement these methods where relevant
- Python programmers will recognise these methods immediately
- Makes your classes easier to maintain

Constructors and Destructors

Constructor
- __init__(self, otherArgs)

Destructor
- __del__(self)

Ex 4

Stringify Methods

Return a machine-readable representation of an object
- __repr__(self)

Return a human-readable rep… object
- __str__(self)

Ex 4

Operator Methods

There are a large number of method that represent standard operators, including:
- __eq__(self, other)
- __ne__(self, other)
- Etc…

Ex 4

Inheritance is a very important part of object-oriented development
- Allows you to define a new class based on an existing class
- You just specify how the new class differs from the existing class

Terminology:
- For the "existing class": Base class, superclass, parent class
- For the "new class": Derived class, subclass, child class

Potential benefits of inheritance:
- Improved OO model
- Faster development
- Smaller code base

Ex 5

Python supports multiple inheritance

Client code can access public members in the subclass or in any superclass

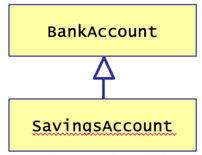
Ex6

Also see these additional demos, which describe "method resolution order" (MRO)

Ex7

The subclass inherits everything from the superclass (except constructors)
- You can define additional variables and methods
- You can override existing methods from the superclass
- You typically have to define constructors too

We'll see how to implement the following simple hierarchy:



Note:
- BankAccount defines common state and behaviour that is relevant for all kinds of account
- SavingsAccount "is a kind of" BankAccount that earns interest

We might define additional subclasses in the future…
- E.g. CurrentAccount, a kind of BankAccount that has cheques

To define a subclass, use the following syntax
- Note that a Python class can inherit from multiple superclasses
- We'll discuss multiple inheritance later in this chapter

Accounting

```
class Subclass(Superclass1, Superclass2, …) :

    # Additional attributes and methods …

    # Constructor(s) …

    # Overrides for superclass methods, if necessary …
```

The subclass inherits everything from the superclass
- (Except for constructors)
- The subclass can define additional members if it needs to …

Accounting

The subclass can override superclass instance methods
- To provide a different (or supplementary) implementation
- No obligation :)

An override can call the original superclass method, to leverage existing functionality
- Call super().methodName(params)