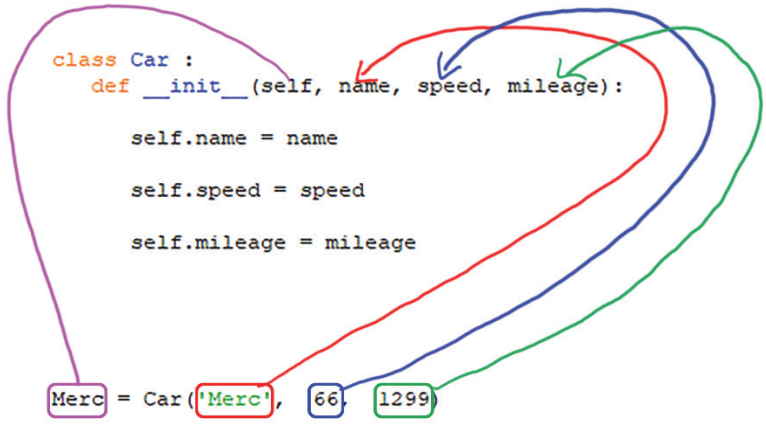
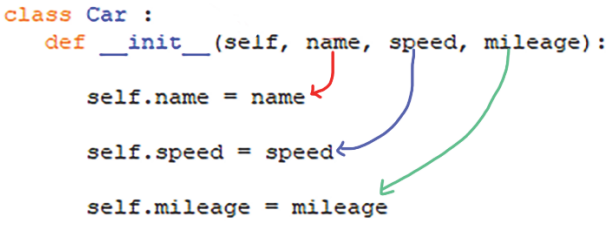
Let's take a look at what is happening. In the `__init__()` method, `self` refers to the instance of the object itself and is automatically passed. Here at the bottom of the diagram below, when we create an object (Merc) from the Car class, we pass some data `name, speed, mileage`.



Next we assign the attributes passed to the `__init__()` method to the attributes of the object.



We can use the object of a class to perform actions. For example:

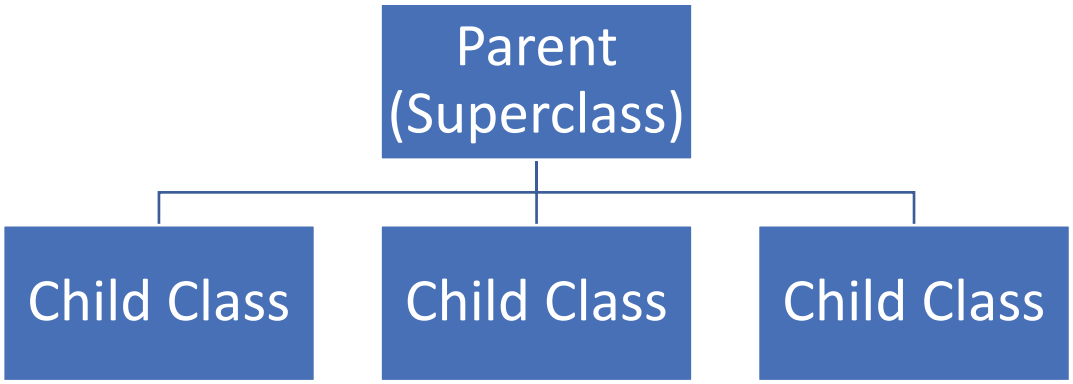
`Merc.`**`getName()`**

or

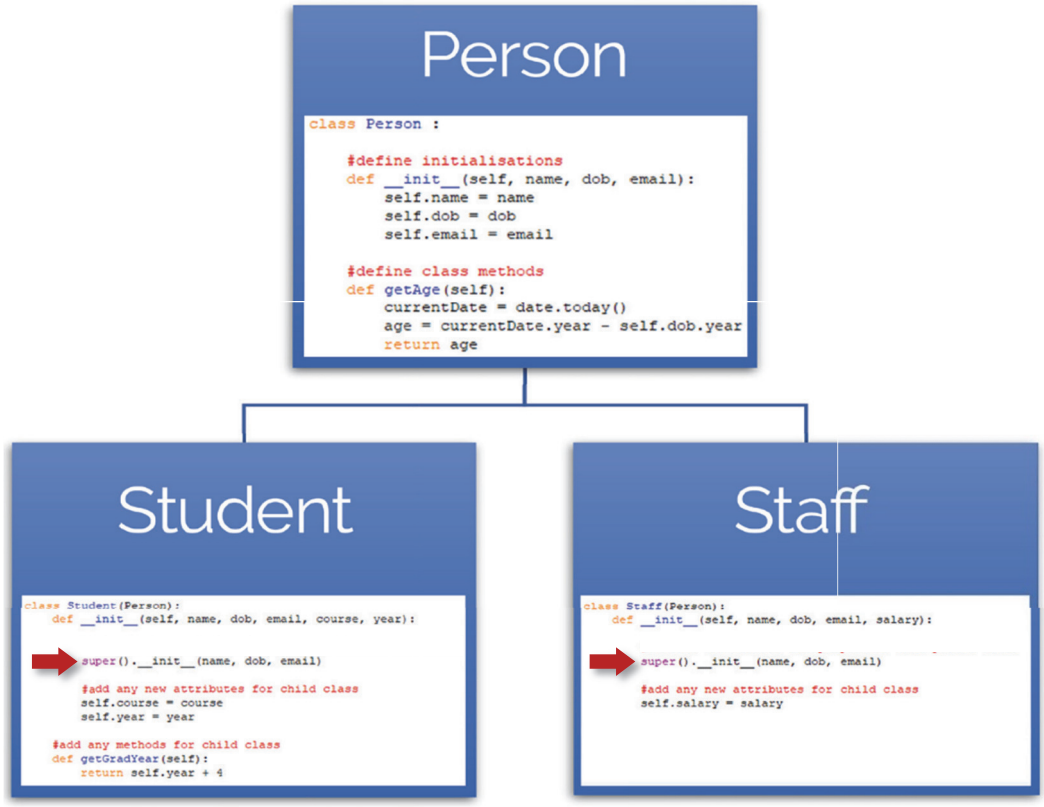
`Merc.`**`speed`** `= 66`

Inheritance

A class can inherit all the methods and attributes from another class.



If a class called ‘person’ had name, age, and dob. We could define two other child classes called ‘student’ and ‘staff’. Both can inherit the methods and attributes from the ‘person’ class.



When defining your child classes, you might need to access methods defined in the parent. To do this, use the `super()` function. The `super()` function allows us to refer to the parent class explicitly.

For example, if you need to access the attributes in the parent, you'll need to call the parent's `__init__()` constructor using `super()` in the child class.

```
super().__init__(name, dob, email)
```