**Section 8: Object Oriented Programming**

**22.02.**

**68. Object Oriented Programming – Introduction**

class NameOfClass():

def \_\_init\_\_(self, param1, param2):

self.param1 = param1

self.param2 = param2

def some\_method(self):

print(self.param1)

**23.02.**

**69. Object Oriented Programming - Attributes and Class Keyword**

When instantiating a class, the \_\_init\_\_ method is automatically called

We can use instead of self any other name to link the method to the class

When we check the type of one class, there will be stated also to what file is linked

By convention the param sent in \_\_init\_\_ should have the same name as the attribute

Self is a reference for a particular instance of a class

**70. Object Oriented Programming - Class Object Attributes and Methods**

Class attributes can be defined in a class, but outside methods

For example:

class Dog:

breed = ‘Unknown’

We cannot access class attributes inside the class without self. Or ClassName.

Every method in a class must have the self parameter

Parameters of \_\_init\_\_ method can have default values

Class attributes can be access without using an instance

**24.02.**

**71. Object Oriented Programming - Inheritance and Polymorphism**

We can inherit without super or BaseClass.\_\_init\_\_, just by using class ChildClass(BaseClass)

This way is used when we don’t want to add other attributed in the ChildClass

The second option is to call BaseClass.\_\_init(self, …)

The third option is to call super().\_\_init\_\_(…)

In the ChildClass we can access methods and attributes from the BaseClass via super()

In Python there is multiple inheritance and method resolution order

In the ChildClass we can override the methods from the BaseClass

Polymorphism usually uses also an abtract BaseClass and ChildClasses that inherit the base one

These ChildClasses override methods from the BaseClass

Use case for polymorphism: file reader class for multiple formats