1. Import is our way to use another file in our file. For example if I have a file with certain functions and I want to use one of the functions in another file, I can write import to the file and then all the functions in that file can be used in the current file.
2. If the file to be imported is in the same directory as the current file or in a subdirectory of it, you can use the file name directly without specifying a full path.

If not, you need to provide a full path.

1. \_\_name\_\_ is a built-in variable used to determine whether a Python script is being run as a main program or whether it is being imported as a module into another script. This is useful so that you can reuse modules in different scripts and still have the option of executing a particular block of code only when the script is being executed.
2. A class in python is a concept in OOP (object oriented programming), which is basically a way of creating objects, with their own functions and their own attributes.
3. We want to use classes over regular variables because first of all it is more orgsnized. Instead of having a lot of variables for each object all over the place, we do it in an organized way.

It also helps with the reusability, so when you define a class, you can add as many objects to it without creating more classes.

And also it is easier for testing and debugging, because it is clearer and more understandable and readable than having many variables.

1. Attribute of a class is basically a variable inside the object you create, for example if you have a class of person, you can create an attribute of their name, age, city, and more. And every new object you make, needs to have all the attributes you add at the beginning. An example is:

class Person():

def \_\_init\_\_(self, name, age):

self.name = name

self.age = age

1. Class objects are created based on the class blueprint and have their own unique data of the attributes and use the class’s functions with their own data.

An example is:

person1 = Person(“maya”, 19)

1. The constructor is defined with the \_\_init\_\_ method. This method is called automatically when an object is created by the class. It receives self as the first parameter, which points to the new object, and any other parameters needed for initialization.

The Destructor manages memory and cleans up objects when they are no longer referenced. You can implement it by calling the method \_\_del\_\_.

1. The keyword self is the name for the first parameter of \_\_init\_\_ in a class. It refers to the instance of the class itself.
2. Inheritance refers to the way you can create new classes that inherit from existing classes. It's a mechanism by which classes can define their behaviour and attributes by extending properties and methods from a parent class to a child

The parent is the class the child inherits from.

1. Polymorphism refers to when many classes and object uses the same functions and methods. So to do that we don’t use specific values but variables so whatever calls the function can use it for their own thing and it is more efficient.
2. The static method is in a class is when you have a function in a class that is not includes any attributes from the class, you will put above it the syntax: @staticmethod

In class, all the variable definded are static.

It is different from a normal function in a class, because a normal function uses the self method and is directly related to the attributes of the class, but static class is not.

To create a static function you use the syntax: @staticmethod.

An example is:

class Person:

@staticmethod

def sum\_nums(a, b):

return a + b

enums:

1. Enums are a way of giving clear names to unique values of int or string. You write like in a class and it is a way to be organized and to have your code be readable and more understandable.
2. We will want to use enums because it is ore readable, because you give meaningful names to values, and also it is safer because it ensures that valid values only are being used.