**Name: Abdelwahab abdelsalam wahba**

**Email:** [**abdelwaba.abdelsalam00@gmail.com**](mailto:abdelwaba.abdelsalam00@gmail.com)

**Phone number: 01006581926**

**Github link:** [**https://github.com/abdelwahab-abdelsalam/AMIT-Training**](https://github.com/abdelwahab-abdelsalam/AMIT-Training)

**Name of instructor: Ahemd Khaled**

**Search about Dunder methods, \*args, \*\*kwargs, encapsulation and abstraction**

**Dunder methods**

Dunder methods are methods that have 2 underscores before and after it for example \_\_init\_\_, \_\_abs\_\_, \_\_and\_\_, \_\_len\_\_, \_\_sizeof\_\_ .

The name dunder stands for double under (D Under ), It is also called magic methods.

First of all, \_\_init\_\_ is the method that is called when creating any object, it lets the class initialize the object’s attributes.

We can use the function dir() to see all the properties and methods of the desired object without its values, For example dir(str) used in the attached notebook file used to get all the properties and methods in the object str (string).

From the methods that were obtained from dir(str) we defied a string called it x and put a string in it then we used \_\_add\_\_ to add to the string as shown in the examples notebook file, the, the \_\_add\_\_ methods is what is called when you call + on 2 objects.

And so, from the above examples \_\_abs\_\_ is a method in the int object we assigned y a negative value then we used \_\_abs\_\_ to get the absolute of the number.

We used \_\_sizeof\_\_ in the attached notebook file and the expected result would be that it is going to return the size of the string but the result was not the exact size of the string but it returns the internal memory size for the object.

Unlike \_\_sizeof\_\_ \_\_len\_\_ returns the size of the object itself not how much space it took for storage.

And finally, we used \_\_and\_\_ which is what is called when you for example a and b which is a logical operation between a and b the same is applied when or is used, we use the \_\_or\_\_ method.

**\*arg and \*\*kwargs**

In functions we pass simply pass multiple arguments for the operation or an argument list that has all the values as shown in the attached notebook file.

This works fine until you need to pass arguments the number of which is determined at runtime then you have to use \*arg which allow you to pass a varying number of values like shown in example in the notebook file this way we can send the to the function whatever number of values every time we call it when you send the arguments to the function it is no longer a list being sent to the function but a varying number of arguments that get put in a single iterable object named arg or whatever it is named the object itself is a tuple.

Similar to \*arg \*\*kwarg works just like it but it accepts keyword arguments in simpler terms it accepts named values and puts them with their keys in a dictionary when they are sent to the function.

**Encapsulation and** **Abstraction**

**Encapsulation** is an OOP concept that wraps data and functions into one component it acts as protective layer so that any access to the wrapped data is denied by any code outside of the class itself in doing so it provides better data security.

Encapsulation can be done with two different methods they are declaring the defined data in the class as private or declaring them as protected.

To define the data or methods as private we put the prefix ‘\_\_’ to it.

Private methods and data can be only accessed withing the class itself and cannot be accessed from the outside.

An example is showing the attached notebook file shows how it written and how it cannot be accessed from the outside.

On the other hand, protected data and methods can be accessed within the same class and any class that inherits the main class.

Abstraction is also an OOP concept that instead of showing everything it focused on the object’s relevant data or methods and hides all the other irrelevant data or methods this is done to simplify and improve efficiency and generalize the objects.

**References**

* <https://www.tutorialsteacher.com/python/magic-methods-in-python#:~:text=Python%20%2D%20Magic%20or%20Dunder%20Methods,class%20on%20a%20certain%20action>
* <https://www.geeksforgeeks.org/args-kwargs-python/>
* <https://realpython.com/python-kwargs-and-args/>