Search Task

* **Dunder methods**

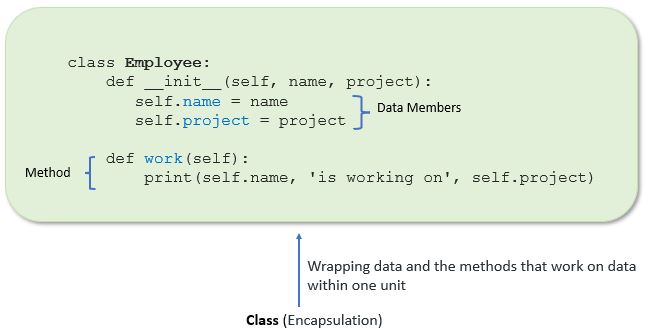
Dunder methods in python are the special methods and end with double underscores. they are also called magic methods. Dunder methods meant the invocation happens internally from the class on a certain action.

Examples for dunder methods :

* \_\_init\_\_,  it gets called whenever an object is created automatically
* \_eq\_\_, the == operation
* \_\_ne\_\_, the != operation
* \_\_lt\_\_, the < operation
* \_\_gt\_\_, the > operation
* \_\_le\_\_, the <= operation
* \_\_ge\_\_, the >= operation
* \_\_abs\_\_, the result of the built-in abs() function (absolute value)
* \_\_add\_\_, \_\_iadd\_\_, \_\_radd\_\_, the + operator
* \_\_sub\_\_, the - operator
* \_\_mul\_\_, the \* operator
* \_\_div\_\_, the / operator
* \_\_mod\_\_, the % operator (modulo)
* \_\_pow\_\_, the \*\* operator
* \_\_and\_\_, the & operator (binary AND)
* \_\_or\_\_, the | operator (binary OR)
* \_\_xor\_\_, the ^ operator (binary XOR)
* **\* args and \*\*kwargs**
* \*args and \*\*kwargs  are special keyword which allows function to take variable length argument.
* \*args is used to send a **non-keyworded** variable length argument list to the function.
* \*\*kwargs allows you to pass **keyworded** variable length of arguments to a function. You should use \*\*kwargs if you want to handle **named arguments** in a function.

# **Encapsulation**

[Encapsulation](https://en.wikipedia.org/wiki/Encapsulation_(computer_programming)" \t "https://pynative.com/python-encapsulation/_blank) in Python describes the concept of **bundling data and**[methods](https://pynative.com/python-instance-methods/)**within a single unit**. So, for example, when you create a [class](https://pynative.com/python-classes-and-objects/), it means you are implementing encapsulation. A class is an example of encapsulation as it binds all the data members ([instance variables](https://pynative.com/python-instance-variables/)) and methods into a single unit



* Abstraction

Abstraction is used to hide the internal functionality of the function from the users. The users only interact with the basic implementation of the function, but inner working is hidden. User is familiar with that **"what function does"** but they don't know **"how it does."**

**Examples :** Another example is, when you use TV remote, you do not know how pressing a key in the remote changes the channel internally on the TV. You just know that pressing + volume key will increase the volume.