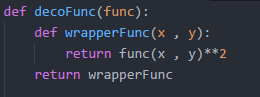
**DECORATORS:**

Decorators in programming is a special software design pattern which uses the functions and modifies them without actually changing the body of the function. That is the decorators are used to wrap a new functionality to our function without changing the source code of the function. Decorators in python are no different. We will now observe the concept of decorators in python:



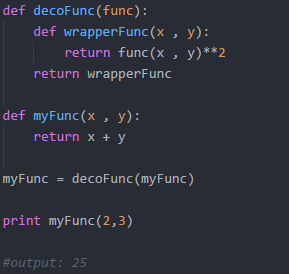
We have the above function which returns the summation of the two arguments passed as parameters to the function. Now what if we want this function to not only return the summation but also return the square of the summation, something like (x + y)2 without changing the original body of the function. So let us design the decorator function that is going to make this magic happen:



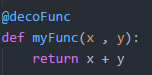
Now what the above function (the decorator) does is, takes a function as a parameter. This decorator function has got another function within itself (which is fine in python) which is the function that is going to add the extra modified feature to our function myFunc(x , y). This function has the same number of parameters as our original function (only for this example) because at the end, this the function that is going to be returned to our existing function. So now what this wrapperFunc() is doing is calling the function it received as a parameter ( technically its parent function received) with its arguments and squaring the result and returns the final result. Now this wrapperFunc is returned by the decoFunc function which is our decorator to the function that needs decoration! So the only thing left to do to complete decorating myFunc is:



Calling the decoFunc with myFunc (itself) as parameter and getting the returned function that is the wrapperFunc. So the whole thing becomes:



So now that we are calling the myFunc function with the arguments 2 & 3 instead of returning only the summation of the two arguments we are getting the square of the summation. In short the function myFunc gets decorated.

Now python has got a syntactic sugar for decorators (python 2.4 & above). Instead of  this we can do this.

So finally we can decorate any function in this way:

