ASSESSMENT-1

**Power Calculation**

[bookmark\_border](https://doselect.com/)

* subject Coding

**DESCRIPTION**

**Environment Specifications & Instructions**

**Allowed Languages**

* Python2
* Python3

**Build Expectation**

**Problem Statement**

John has just studied OOPSconcepts in python.

Now he wants to implement the concept of method overloadingin the problem of power calculation. The function ***Power*** takes at most three arguments *a, b, c*.

* If none of *a,b,c* is given, it returns 0
* If only *a* is given, it returns a
* If both *a* and *b* are given, it returns *a^b* (i.e. power(*a,b*))
* If all *a,b,c* are given, it returns *a^(b^c)***(i.e. *power(a,power(b,c))*)**

**Task**

Your task is to solve this problem by implementing following class and functions:

**Specifications**

class Power:

function init (variable a, variable b, variable c):

//init is the \_\_init\_\_(self,variables) function **defined** in python.

member variable a= a

member variable b= b

member variable c= c

function find\_power():

**if** none of a,b,c is **given**:

**return** 0

**else** **if** only a is **given**:

**return** a

**else** **if** a **and** b both are **given**:

**return** a^b (i.e. power(a,b))

**else** **if** all a,b,c are **given**:

**return** a^(b^c) (i.e. power(a,power(b,c)))

//**no** need to handle remaining cases

You are done. No need to implement ***main()*** function. Once you implement the above class and function, your code will be tested by calling the class and function.

ASSESSMENT-2:

**Python Decorators**

**DESCRIPTION**

**Environment Specifications & Instructions**

**Build Expectation**

**Task**

Implement a Python decorator that should take whatever the decorated function returns, and writes it to a file in a new line.

For the sake of this problem, let us assume that the decorated functions always return a string.

**Constraints**

The decorator should be named ***log\_message***and should write to the file ***/tmp/decorator\_logs.txt***.

**Example usage:**

@log\_message

**def** **a\_function\_that\_returns\_a\_string**():

**return** "A string"

@log\_message

**def** **a\_function\_that\_returns\_a\_string\_with\_newline**(s):

**return** "{}\n".format(s)

@log\_message

**def** **a\_function\_that\_returns\_another\_string**(string=""):

**return** "Another string"

On running in python shell,

>>> a\_function\_that\_returns\_a\_string()

>>> a\_function\_that\_returns\_a\_string\_with\_newline("Newline String")

>>> a\_function\_that\_returns\_another\_string(string="Another String")

Should write the string to the file.

$ cat /tmp/decorator\_logs.txt

A string

Newline string

Another string