**Python : Decorator and Generator**

A guide on decorator and generator using oops

**Decorator :**

Decorator is allows to programmers to change behavior of the function or the class. Decorator allows us to wrap another function to extend behavior of the wrapper function.

Creating decorator :

Text

Description automatically generated

**How wrapper class works :**

wrapper takes argument from the inner. inner class calls the func object passes to the function. and it returns inner class using return function. @wrapper function is used to decorate or modify the function. In this code we wrap inside\_decorator class in it.

**Decorator using oops :**

Text

Description automatically generated

By using \*args and \*\*kwargs in the decorator function, the function is able to be used with any number of positional arguments, keyword arguments, or a combination of both.

**Output:**

Text

Description automatically generated

**Generator :**

Python Generators are the functions that return the traversal object and used to create iterators. It traverses the entire items at once. It is a lengthy process to create iterators. It is like regular function but generator uses yield in place of the return.

This made the function easier to write and much more clear than an approach using instance variables like self.index. and self.data.

There are two types of generator.

1. Generator-function

2. Generator-object

We use two class Prime and Employee by using inheritance. Employee class is inherited by the prime class. prime class print prime number between 0 to 21.

Text

Description automatically generated

references : <https://docs.python.org/3.11/tutorial/classes.html#generators>

<https://www.geeksforgeeks.org/generators-in-python/>

* Made by **Kirtan mangukiya(20CE051)** and **Jay Panchal(20CE068)**
* **Charusat University(2022).**
* -Guided by **MRUGENRASINH RAHEVAR**