Day-5 evening Assessment

# Python Oop practice questions

1.An instance method is defined inside a class and is called on an instance of that class. It automatically receives the instance as first object which is self.

Ex: class Dog:  
 def \_\_init\_\_(self,name):  
 self.name = name  
  
 def barks(self): #instance method  
 return f"{self.name} barks"  
  
my\_dog = Dog("Buddy") #creating instance of class dog  
print(my\_dog.barks()) #calling instance method

2. class Car:  
 def \_\_init\_\_(self,model):  
 self.model = model  
  
 def start\_engine(self):  
 return f"{self.model} engine started"  
  
my\_car=Car("Ford")  
print(my\_car.start\_engine())

3.Both ways are possible but using object is the most preferred way.

Ex: class Person:  
 def greet(self):  
 print("Hello from instance method")  
  
obj = Person()  
obj.greet() #using object  
Person.greet(obj) #using class

o/p: Hello from instance method

Hello from instance method

4. class Circle:  
 def \_\_init\_\_(self, radius):  
 self.radius = radius  
  
 def area(self):  
 return math.pi \* self.radius \*\* 2  
  
cir = Circle(5)  
print(cir.area())

5. It will give a type error

# Class Method Questions

6. A class method in Python is a method that is bound to the class and not the instance of the class(@classdecorator) is used. Where as instance method bounds to the instance method and the first parameter should be self.

Ex: class Car:  
 wheels = 4 # Class variable  
  
 def \_\_init\_\_(self, brand):  
 self.brand = brand # Instance variable  
  
 def show\_brand(self): # Instance method  
 return f"This car is a {self.brand}"  
  
 @classmethod  
 def show\_wheels(cls): # Class method accessing class variable  
 return f"Cars have {cls.wheels} wheels"  
   
  
car1 = Car("Toyota")  
  
# Instance method  
print(car1.show\_brand()) #Uses instance data  
  
# Class method  
print(Car.show\_wheels()) #Uses class data

7. class Employee:  
 company = " TechCorp"  
  
 @classmethod  
 def get\_company\_name(cls):  
 return cls.company  
  
print(Employee.get\_company\_name())

[8. @classmethod](mailto:8.@classmethod) decorator is used and class method is defined using this decorator.

9. class Counter:  
 count=0  
  
 @classmethod  
 def increment(cls):  
 cls.count+=1  
 return cls.count  
  
print(Counter.increment())  
print(Counter.increment())

o/p: 1

2

# Static Method Questions

10. Used when some functionality is related to the class but doesn't need access to class or instance data. Defined using @staticmethod decorator.

Ex: class Maths:  
 @staticmethod  
 def add(a,b):  
 return a+b  
 @staticmethod  
 def sub(a,b):  
 return a-b  
print(Maths.add(3,4))

o/p: 7

11. class MathsUtils:  
 @staticmethod  
 def is\_even(num):  
 if num % 2 == 0:  
 return True  
 else:  
 return False  
  
my\_maths = MathsUtils()  
print(my\_maths.is\_even(5))

12.-when you want to access and modify class variables,use a class method.

-when you don’t need class or instance data,use a static method.