## Ques. What is Class?

- The class is a collection of objects.
- It is a logical entity that has some specific attributes and methods.
- To define a class in Python, you can use the class keyword, followed by the class name and a colon. Inside the class, an **init** method has to be defined with def. This is the initializer that you can later use to instantiate objects. It's similar to a constructor in Java. **init** must always be present! It takes one argument: self, which refers to the object itself. Inside the method, the pass keyword is used as of now, because Python expects you to type something there.

```
class ClassName:
<statement-1>
            .
            .
<statement-N>
```

- **Instantiating object**

```
mohit = ClassName()
print(mohit)
```

**Example 1**

```
class Person:
  def __init__(self, name, age):
    self.f_name = name
    self.age = age

p1 = Person("John", 36)

print(p1.f_name)
print(p1.age)
```

**Example**

```
class Dog:

    def __init__(self, name, age):
        self.name = name
        self.age = age

    def bark(self):
        print("bark bark!")
```

```
    def doginfo(self):
        print(self.name + " is " + str(self.age) + " year(s) old.")

ozzy = Dog("Ozzy", 2)
skippy = Dog("Skippy", 12)
filou = Dog("Filou", 8)

ozzy.bark()
skippy.doginfo()
filou.doginfo()

Output:-
bark bark!
Skippy is 12 year(s) old.
Filou is 8 year(s) old.
```

- we have shown two ways of accessing the values of those attributes. One, by directly using the class name and the other by using an object(class instance). Assigning a class to a variable is known as object instantiation.

```
class Scaler:
    Course1 = 'Python'
    Course2 = 'C++'
    Course3 = 'Java'
# Accessing the values of the attributes
print(Scaler.Course1)
print(Scaler.Course3)
# Accessing through object instantiation.
obj= Scaler()
print(obj.Course2)

Output:-
Python
Java
C++
```

- If we change the value of the attribute using the class name, then it would change across all the instances of that class. While if we change the value of an attribute using class instance(object instantiation), it would only change the value of the attribute in that instance only.

```
class Scaler:
    Course = 'Python'
# Changing value using Class Name
Scaler.Course = 'Machine Learning'
obj= Scaler()
print(obj.Course)
# Changing value using Class Instance
obj.Course = 'AI'
print(obj.Course)   # Value will change in this instance only
```

```
print('Using class instance would not reflect the changes to other instances')
print(Scaler.Course) # Value haven't changed
obj2= Scaler()
print(obj2.Course)   # Value haven't changed

Output:-
Machine Learning
AI
Using class instance would not reflect the changes to other instances
Machine Learning
Machine Learning
```