PYTHON OOPs CONCEPTS

CLASS AND OBJECTS:
In Python, a class is a blueprint or template for creating objects.

An object is like a thing that has certain characteristics and can do certain things.

For example, you can think of a class called "Car" that represents cars in general. The class "Car" would define what a car is and what it can do.

In a class, you have two main things: attributes and methods.

```
89
90  class Car:
91      def __init__(self, color, model, doors):
92          self.color = color
93          self.model = model
94          self.doors = doors
95
96      def start_engine(self):
97          print("Engine started!")
98
99      def accelerate(self):
100         print("Car is accelerating.")
101
102 myCar1 = Car("red", "Ford", 5)
103 myCar2 = Car("green", "Hundai", 4)
104
105 print(myCar1.color)
106 print(myCar2.model)
107
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  COMMENTS

PS C:\Users\Ram prasath> & "C:/Users/Ram prasath/AppData/Local/Programs/Python/Pytho
red
Hundai
PS C:\Users\Ram prasath>
```

Attributes are like the properties or characteristics of an object.

In this above example, the attributes are '**color**', '**model**', and **'doors'**. And the methods are

'start_engine' and **'accelerate'**.

WHAT IS THE 'SELF' KEYWORD IN CLASS?

- **`self`** is a special parameter in Python that refers to the instance of a class.
- It is commonly used as the **first parameter** in instance methods within a class definition.
- When a method is called on an instance of a class, **`self`** is automatically passed as the first argument to that method.
- **`self`** allows you to access and modify the instance's attributes and methods within the class.
- By convention, the name **`self`** is used, but it is not a strict requirement. You can use any name as long as it is the first parameter in the method definition.
- You don't need to pass an argument explicitly for **`self`** when calling instance methods. Python takes care of it behind the scenes.
- The use of **`self`** enables object-oriented programming concepts such as **encapsulation** and code reusability.

Example of class without '*SELF*' keyword:

As you can see, without self you will face the below error.

```
134
135   class SampleClass:
136
137       def sampleMethod(name):
138           print("my name is ", name)
139
140
141   # declare an object
142   obj = SampleClass()
143
144   obj.sampleMethod("David")
145
146
147
148
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   COMMENTS

PS C:\Users\Ram prasath> & "C:/Users/Ram prasath/AppData/Local/Programs/Python/Python311/python.exe
Traceback (most recent call last):
  File "c:\Users\Ram prasath\main.py", line 144, in <module>
    obj.sampleMethod("David")
TypeError: SampleClass.sampleMethod() takes 1 positional argument but 2 were given
PS C:\Users\Ram prasath>
```

Example of class With '*SELF*' keyword:

```
134
135   class SampleClass:
136
137       def sampleMethod(self, name):
138           print("my name is ", name)
139
140
141   # declare an object
142   obj = SampleClass()
143
144   obj.sampleMethod("David")
145
146
147
148
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  COMMENTS
PS C:\Users\Ram prasath> & "C:/Users/Ram prasath/AppData/Local/Programs/Python/
my name is  David
PS C:\Users\Ram prasath>
```

More Example of Class with *'SELF' keyword:*

```
134
135     class Person:
136 +
137         def personalDetails(self, name, age, gender):
138             print("my name: ", name)
139             print("age: ", age)
140             print("gender: ", gender)
141
142         def addressDetails(self):
143             print("City: ", self.city)
144             print("State: ", self.state)
145             print("Country: ", self.country)
146
147     #  declare an object
148     obj = Person()
149
150     obj.personalDetails("Vignesh", 24, "Male")
151     obj.city = "Chennai"
152     obj.state = "Tamilnadu"
153     obj.country = "India"
154     obj.addressDetails()
155
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  COMMENTS
PS C:\Users\Ram prasath> & "C:/Users/Ram prasath/AppData/Local/Programs/Python/P
my name:  Vignesh
age:  24
gender:  Male
City:  Chennai
State:  Tamilnadu
Country:  India
PS C:\Users\Ram prasath>
```

CONSTRUCTOR

A constructor in Python is a special method within a class that is automatically called when an object is created from that class.

The constructor method is defined using the __init__ function name.

The constructor method takes the self parameter as the first argument, which refers to the instance of the class being created.

EXAMPLES OF CONSTRUCTOR

```python
class ConstructorExample:
    def __init__(self):
        self.name = "David"
        self.age = 12
        self.gender = "male"


obj = ConstructorExample()
print("my name is :", obj.name)
print("my age is :", obj.age)
print("my gender is :", obj.gender)
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  COMMENTS
PS C:\Users\Ram prasath> & "C:/Users/Ram prasath/AppData/Local/Programs/Pyt
my name is : David
my age is : 12
my gender is : male
PS C:\Users\Ram prasath>
```

```python
class ConstructorExample:
    def __init__(self, name, age, gender):
        self.name = name
        self.age = age
        self.gender = gender
    def details(self):
        print("my name is :", self.name)
        print("my age is :", self.age)
        print("my gender is :", self.gender)


obj = ConstructorExample("david", 24, "male")
obj.details()
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  COMMENTS

PS C:\Users\Ram prasath> & "C:/Users/Ram prasath/AppData/Local/Programs/Python/P
my name is : david
my age is : 24
my gender is : male
PS C:\Users\Ram prasath>
```

```python
class ConstructorExample:
    def __init__(self, name, age, gender):
        self.name = name
        self.age = age
        self.gender = gender
        print("**********************************************************************")
        print("my name is '%s' and my age is '%s' and my gender is '%s'" % (self.name, self.age, self.gender))
        print("**********************************************************************")

    def details(self, city, state, country):
        print("city name : ", city)
        print("------------------")
        print("state name : ", state)
        print("------------------")
        print("country name : ", country)
        print("------------------")


obj = ConstructorExample("david", 24, "male")
obj.details("Chennai", "Tamilnadu", "India")
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  COMMENTS

PS C:\Users\Ram prasath> & "C:/Users/Ram prasath/AppData/Local/Programs/Python/Python311/python.exe" "c:/Users/Ram prasath/main.
**********************************************************************
my name is 'david' and my age is '24' and my gender is 'male'
**********************************************************************
city name :  Chennai
------------------
state name :  Tamilnadu
------------------
country name :  India
------------------
PS C:\Users\Ram prasath>
```