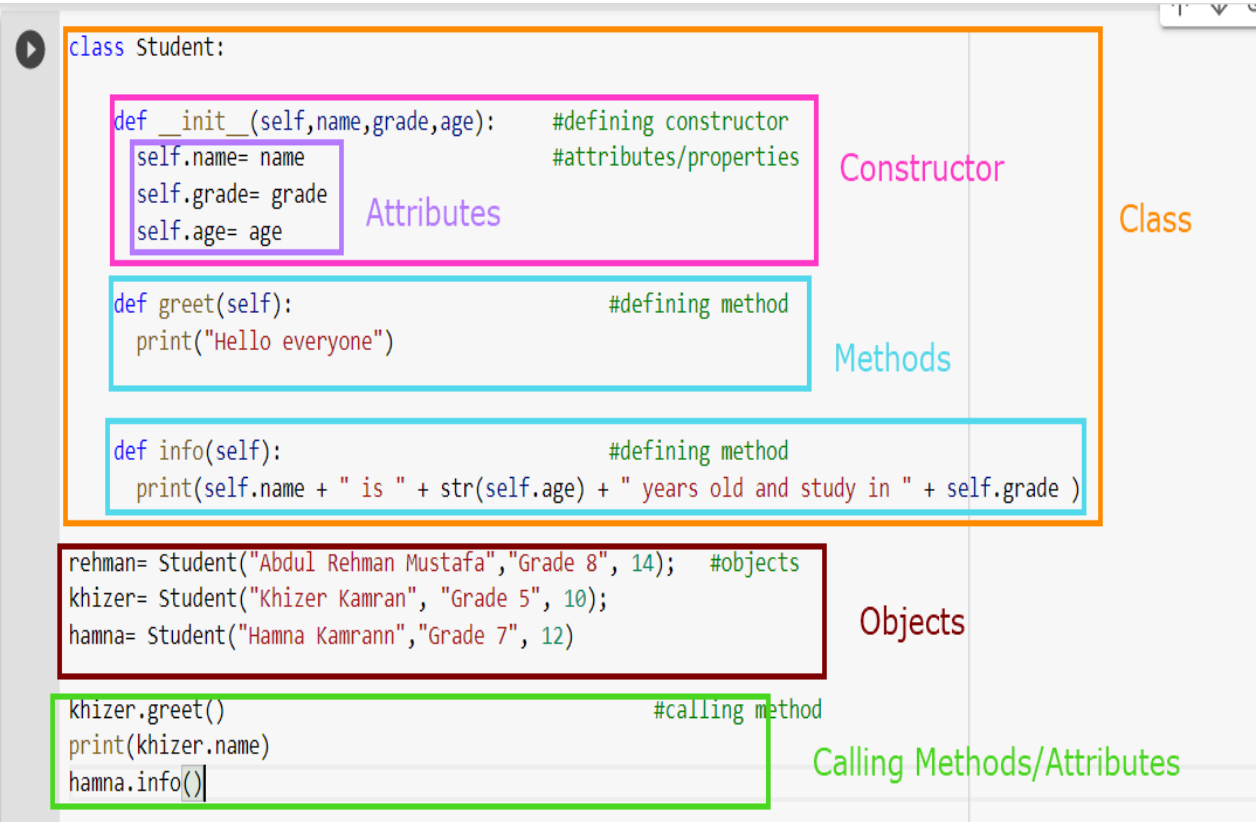
# ADVANCE PYTHON FOR KIDS-SESSION 11

## OBJECT ORIENTED PROGRAMMING

- Python is an object oriented programming language. Unlike procedure oriented programming, where the main emphasis is on functions, object oriented programming stresses on objects.
- An object is simply a collection of data (variables) and methods (functions) that act on those data. Similarly, a class is a blueprint for that object.
- We can think of class as a sketch (prototype) of a house. It contains all the details about the floors, doors, windows etc. Based on these descriptions we build the house. House is the object.
- As many houses can be made from a house's blueprint, we can create many objects from a class. An object is also called an instance of a class and the process of creating this object is called **instantiation**.

## DESCRIPTION:

- Class
- Objects
- Methods/Functions
- Constructor (objects initializer method)
- Attributes



## ADDING ATTRIBUTES TO A CLASS

- Inside the class, an `__init__` function has to be defined with `def`.
- `__init__` must always be present! It takes one argument: `self`, which refers to the object itself.
- Inside the function, the `pass` keyword is used as of now, because Python expects you to type something there. Remember to use correct indentation!
- Inside the function, when you type some attributes/properties, the `pass` keyword is not used. Remember to use correct indentation!
- The attributes you make inside the function is also added with the arguments of function.
- To access an object's attributes in Python, you can use the dot notation. This is done by typing the name of the object, followed by a dot and the attribute's name.

### SYNTAX

```
class ClassName:

  def __init__(self, attribute1, attribute2):          #defining __init__ function
    self.attribute1 = attribute1                        #initilaize attributes/properties
    self.attribute2 = attribute2

#creating objects with arguments
Object1 = ClassName(value for attribute1, value for attribute2)
Object2 = ClassName(value for attribute1, value for attribute2)
```

### CLASS

- Like function definitions begin with the def keyword in Python, class definitions begin with a class keyword.
- The first string inside the class is called docstring and has a brief description about the class. Although not mandatory, this is highly recommended.

### OBJECT

- Object could be used to access different attributes.
- It can also be used to create new object instances (instantiation) of that class. The procedure to create an object is similar to a function call.

### CODE

```
class Student:

  def __init__(self,name,grade,age):
    self.name= name              #attributes/properties
```

```
    self.grade= grade
    self.age= age

rehman= Student("Abdul Rehman Mustafa","Grade 8", 14);   #objects
khizer= Student("Khizer Kamran", "Grade 5", 10);
hamna= Student("Hamna Kamrann","Grade 7", 12)

print(rehman.name)
print(khizer.age)
print(hamna.grade)
print(khizer.name + "is " + str(khizer.age) + " years old and study in " +khizer.grade )
```

OUTPUT

```
Abdul Rehman Mustafa
10
Grade 7
Khizer Kamranis 10 years old and study in Grade 5
```

## METHOD:

- Functions inside the class are called Methods.
- These methods can be defined using “def” keyword with “self” as the necessary argument.
- The mandatory method should be named with __init__ and is called “Constructor”. It is used to initialize the objects.
- You can create as many methods inside a class as you want.

SYNTAX:

```
class ClassName:

   #defining constructor
   def __init__(self,attribute1,attribute2,attribute3):
     self.attribute1= attribute1
     self.attribute2= attribute2
     self.attribute3= attribute3

   def MethodName(self):       #defining method
     ---body of method----

#creating objects
Object1= ClassName(value for attribute1, value for attribute2, value for attribute3);


Object1.MethodName()          #calling method
```

CODE

```
class Student:

   def __init__(self,name,grade,age):    #defining constructor
     self.name= name                  #attributes/properties
     self.grade= grade
     self.age= age

   def greet(self):                     #defining method
     print("Hello everyone")


   def info(self):                      #defining method
     print(self.name + " is " + str(self.age) + " years old and study in " + self.grade )

rehman= Student("Abdul Rehman Mustafa","Grade 8", 14);   #objects
khizer= Student("Khizer Kamran", "Grade 5", 10);
hamna= Student("Hamna Kamrann","Grade 7", 12)

khizer.greet()
                              #calling method
khizer.info()
hamna.info()
```

OUTPUT

```
Hello everyone

Khizer Kamran is 10 years old and study in Grade 5
Hamna Kamrann is 12 years old and study in Grade 7
```

CODE

```
class Student:

   def __init__(self,name,grade,age):      #defining constructor
     self.name= name                   #attributes/properties
     self.grade= grade
     self.age= age


   def birthday(self):                  #defining method
     self.age+=1      #x=x+1
```

```
rehman= Student("Abdul Rehman Mustafa","Grade 8", 14);   #objects
khizer= Student("Khizer Kamran", "Grade 5", 10);
hamna= Student("Hamna Kamrann","Grade 7", 12)

print(khizer.age)
khizer.birthday()                    #calling method
print(khizer.age)
```

OUTPUT

```
10
11
```

## PASSING ARGUMENTS TO METHODS

CODE

```
class Student:

   def __init__(self,name,grade,age):        #defining constructor
     self.name= name                    #attributes/properties
     self.grade= grade
     self.age= age

   def info(self):                      #defining method
     print(self.name + " is " + str(self.age) + " years old and study in " + self.grade )

   def TotalMarks(self,x,y,z):              #defining method
     print(x+y+z)

   def setfrnd(self,frnd):                  #defining method
     self.frnd=frnd
     frnd.frnd=self

rehman= Student("Abdul Rehman Mustafa","Grade 8", 14);   #objects
khizer= Student("Khizer Kamran", "Grade 5", 10);
hamna= Student("Hamna Kamrann","Grade 7", 12)

khizer.TotalMarks(5,8,2)
hamna.TotalMarks(10,7,3)                    #calling method

khizer.setfrnd(rehman)
print(khizer.frnd.grade)
khizer.frnd.info()
```

OUTPUT

```
15
20
Grade 8
Abdul Rehman Mustafa is 14 years old and study in Grade 8
```