**OOP**

-------------

Everything is a object

Object is real time entity

Class is property and behavior of object

it is a blue print of a object

attributes: properties

methods: behavior and characteristics

EXAMPLE:

class: GOD

object: SHIVA

attributes: UNSEEN, has incredible powers, human form ,

behavior : does favour for humans (NOTE:except ENGINEERS)

class:Programming LANG

object: Python

attributes: no use of ; and brackets , human understandable

behavior: interpretes line by line

**4 Modules:**

>>encapsulation

packing related things as one

>>polymorphism (poly=many , morph=shapes)

same function but different object for different set of values

>> DATA abstraction

hiding unnecessary variables for user and other classes

>>inheritence

inherit or reuse code from base class or parent class

relation btw the class (parent and child )

**CLASS**

--------------------------

syntax:

class Classname:

stmt

mtds & attr

obj=Classname() <object> //instantiating a class

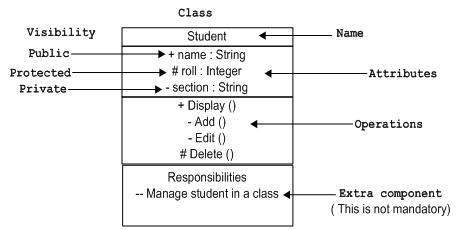
>>>creating a object / instance is called instantiation of that class

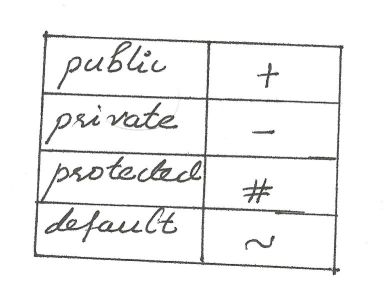
>>>memory will be allocated only when a object is created for a class .

>>>printing a object will give the address(hexa value ) of that instance of a class

>>> UML diagram is a representation of a class in a pictorial form

**UML diagram**





class Dog:

breed='lab'

age='adult'

def run(<self>,a):

print("runnning",a)

obj=Dog()

obj.run(a) <runs when self is mentioned in def>

Dog.run() <runs when there is no self is mentioned because there no obj/instance will created>

Example:

class Student:

def \_\_init\_\_(self,name,rno,dept,cgpa):

self.name=name

self.rno=rno

self.dept=dept

self.cgpa=cgpa

def print(self):

print(self.name,self.rno,self.dept,self.cgpa)

def hobby(self):

hby=input("enter hby")

print(hby)

def learn(self):

lrn=input("enter hrs")

print(lrn)

def sleep(self):

slp=input("enter hrs(slp)")

print(slp)

jeevan=Student('raghul','17cse64','cse','6.8')

jeevan.print()

jeevan.hobby()

jeevan.learn()

jeevan.sleep()