**OOP Object Programming Language**

It is a programming approach based on classes and object. In simple Language OOp is a method to represent the real world entity

**Class**

Class is a blueprint for an object like Real world entity has some properties which is represented by class variable and method in programming

Class classname

#variables

#methods

**Object**

As we know class is a logical entity while an object is a physical or real entity that works on class data

1. Each object has a distinct role or responsibility
2. Object create space on memory as per class members

Obj.name = class.name()

**Constructor**

Constructor is a special function that get automatically called when object of class created

def \_\_init\_\_(self);

#code

Type of constructor

1.Default constructor

2.Parametetarized constructor

**Default Constructor**

Default constructor are also called empty constructor because it doesn’t have any parameters

Class classname

def \_\_init\_\_(self);

#code

**Parameterized Constructor**

Parameterized Constructor accept arguments along with self it is known as parameterized Constructor

Class classname

def \_\_init\_\_(self,name);

#code

**Access Modifier**

Access Modifier are used to set the limit of member accessibility

var Public(same class, same package, Derived class, Other class)

\_var Protected(same class, same package, Derived class)

\_\_var Private(same class)

**Inheritance**

When we define a class that inherits all the properties of other class called inheritance

Syntax

Class Father:

Properties

Class Son(Father):

Properties

Types

1.Simple inheritance

2.Multiple inheritance

3.Multi level inheritance

4.Hierarchial inheritance

5.Hybrid inheritance

**1.Simple inheritance**

Simple inheritance is nothing but which contain only one parent and only one child class

class Father:

Properties

class Son(Father):

Properties

**3.Multi level inheritance**

In this inheritance we have one parent class and multiple child classes

Father->son->gson

**2.Multiple inheritance**

A class contain more child class and only one child class is called multiple inheritance

Father+mom = son

**4.Hierarchial inheritance**

Inheritance which contain one parent class and multiple child classes but each class but each child class can access parent class property

Mother-> son1|son2

**Encapsulation**

Python provides access to all the variable and method globally by using encapsulation we can restrict the variable and method access globally by making it private or protected

1.single underscore[protected]

2.double underscore[private]

**Abstraction**

Abstraction is a process of hiding the implementation details from the users only the hightlighted set of services provided to the user

**Abstract class**

Abstract class is a class that contain one or more abstact method is called abstract class

1.an object of an abstract class cannot be created

2.python provides abc module to work with abstraction

3.we use @abstractmethod decorates to define abstract method

from abc import ABC:

def fun(self):

#body

@abstractmethod

def fun2(self):

**Interface**

Interface is nothing but abstract class contain only abstact method but not any normal method

**Polymorphism**

Polymorphism means ability to take various form

Print(5+5)add

Print(“5”+”5”)concade

It can be performed by two types

1.Overloading

2.Overriding

**Method Overloading**

Whenever class contain more than one methods with same name and different types parameter called overloading

**Method Overriding**

Whenever we writing method name with same signature in parent & child called method overriding

For calling second method you have to add super().class name

**Multithreading**

Multithreading is a technique which allows a CPU to execute multiple threads of one process at the same time.

#why Multithreading

The purpose of Multithreading is to run multiple tasks and function at the same time

Thread class method

1.run() 2.start() 3.join() 4.isAlive() 5.Setname() 6.getname()

from threading import Thread