**Object oriented programming:**

* it phenomenon of working with real time entities will lead us to work with the concept name called object oriented program language as we are working real time entities the built in data types are not suffienct are enough to store the details of real world entities because the real world entities will have various types, properties, structures, functionalities are present  this all are not suitable to store are manipulate the information in built in data types so that’s why we are going to use the user defined data type called class and object
* through class and object, we are creating our own structure and functionalities

**why oops?**

* to store the data in organized and classify the data to perform the functionalities to data in clear structure
* easy maintains and manipulate the data

By using the oops we can reduce the code difficulty and code reusability

oops following the four principles

* inheritance (acquiring the properties from one class to another class reduce the existing task and code repetition)
* encapsulation (to provide the security to the data)
* polymorphism (code reusability, to perform the many tasks by using one interface)
* abstraction (hiding the implementation of logic)

**Class:**

* it is a stru
* class is a blue print of an instance(object)
* class is a structure or plan to create a object
* class is a user defined data types that specify the states and behaviour of a real time entity
* the states are nothing but property, attributes, data are the real-world entities
* and behaviour is the functionality or operation about real time entity
* class can be created by help “class” keyword by using class we are designing or preparing the template to create an object

**object:**

* an object is a variable of an user defined class type
* object is an instance of a class
* object is a reference of an class memory
* class will specify that what to store and what to do
* object will store what we specify by the class
* object is an real time entity to store the real time data by through class

|  |
| --- |
| syntax: objectname= class classname(arguments)  if we pass the arguments that store inside the object memory |

* when we create the object the controller will create memory inside the value space that object memory have unique address that assign the object name(variable)
* The object name with address is stored the variable space
* What are the class members are available in class memory those members derived object memory
* All the object member are created by through function are constructor or directly we can store the object address

**State or Properties:**

if we consider a employees as a class then the company\_name,regno,MBL(main branch location) all are common for all the employees and name of employee email phone these all the data under the category of specific states of specific employee

States are segregated into 2 categories:

* Generic states(static states/class member/generic attributes/class attributes)
* Specific states(non-static states/object members/specific attributes/object attributes)

1. Generic states: These are the properties that are real world entities and common for all the objects which belong to the class these will be stored at class dict and can be accessed by using the class name and object name
   1. This states are common for class and objects Ex. If we consider a bank as an example bank name location phone isfc loan rate or FD rate of interest will be common for all the customers of the bank so the states are the generic states
      1. These data that stored in the variable which is called as static states
      2. Static states: these variable that contains the generic data that defined for all the real world entities which stores inside class memory and can be accessed by all the members
      3. Modification of generic states written to class name will have an impact on class dict object dictionaries

**Creating the generic attributes of the class**

|  |
| --- |
| **Syntax:**  class Cname:  var1=val1  var2=val2  var3=val3 |

intialization of static states

class apple:

    a=100

    b=200

    c=300

class sample:

    pass

sample.a=100

sample.b=200

sample.c=300

class demo:

    pass

def create\_static(cls):

    cls.a=100

    cls.b=200

    cls.c=300

create\_static(demo)

class pen:

    def create(cls):

        cls.a=100

        cls.b=200

        cls.c=300

pen.create(pen)

class paper:

    @classmethod

    def create(cls):

        cls.a=100

        cls.b=200

        cls.c=300

paper.create()

**Initialize the non-static states:**

Class sample:

Pass

Obj=sample()

Obj.variable=value

Obj.variable1=value1

Class sample:

Pass

Obj=sample()

Def create(self,arg)

Self.variablename=value

Create(obj,arg)

Class sample:

Def create(self,args):

Self.variablename=value

Obj=sample()

Sample.create(obj,arg)

Obj.create()

Class sample:

Def \_\_init\_\_(self,arg):

Self.variablename=arg

Obj=sample(arguments)

So this method must have at least one argument to store the address of an object memory according to industrial standrand we call that with the name called self

Class name:

Def \_\_init\_\_(self)

Self.varname=value

Obj=name()args

Why we need the behaviour?

To specify the data