OOPs concepts

Class

Object

Abstraction

Encapsulation

Inheritance

Polymorphism

Class It is the design or blueprint of any entity Class: Member Variable and Member method

Inheritance Extracting features from the existing class Types

1. Single inheritance
2. Multilevel inheritance
3. Hierarchical inheritance
4. Multiple inheritance

Abstraction giving only essential information to the user eg: while booking a ticket

Data Encapsulation binding the member method and member variable.wrapping up variables and methods into a single entity.user without access cannot access the file

Polymorphism Ability to take more than one form eg.game(football,volleyball,badminton,free fire,pubg) Types

1. Method overloading def area(r): pass #circle def area(l,b): pass #rectangle
2. Method overriding class A: def area(r):

```
        pass #circle
```

   class B(A): def area(a):

```
        pass #square
```

In [3]:

```
class parentClass:
    #member variable - parent
    #member method(Function) - parent
    pass
class childClass(parentClass):
    ''' member variable - parent
        member method(Function) - parent
        member variable - child
        member variable - child '''
    pass
```

In [4]:

```
#single
class parentClass:
    pass
class childClass(parentClass):
    pass
```

In [5]:

```
#multilevel
class grandparentClass:
    pass
class parentClass(grandparentClass):
    pass
class childClass(parentClass):
    pass
```

In [7]:

```
#hierarchical inheritance
class parentClass:
    pass
class sibClass1(parentClass):
    pass
class sibClass2(parentClass):
    pass
#hierarchical inheritance
class EntranceExam:
    pass
class eng(EntranceExam):
    pass
class mbbs(EntranceExam):
    pass
```

In [8]:

```
#multiple inheritance
class father():
    pass
class mother():
    pass
class child(father,mother):
    pass
```

In [11]:

```
#Abstraction
class abstract():
    #abstarcter method
    pass
class test(abstract):
    pass
class display:
    pass
```

In [12]:

```
#Data Encapsulation
class sample:
    a = 10
    def show(self):
        pass
    pass
```

#polymorphism(Method Overloading) #same function name diffferent number of parameter

#polymorphism(Method Overriding) #same function name same number of parameter

Object - instance of the class member variable and member method is accessed thru class.

In [1]:

```
class sample:
    def simple(self):
        print('Welcome')
    pass
#object

#object_name = class_name()
obj = sample()
obj.simple()
```

```
Welcome
```

Class is a desingn Object is a instance

In [ ]:

```
 In Class
    Member variable is known as attributes
    Member methods is known as functions
```

In [2]:

```
#Sample program for object and class
class Computer:  #defining a class
    def config(self):  #any function inside the class should have the parameter self
        print("AMD Ryzen,nvidia,3060 ti,16GB,512GB Storage")

comp1 = Computer()  #comp1 was a object inside the class computer
Computer.config(comp1) #class name+ . + function name + parameter
```

```
AMD Ryzen,nvidia,3060 ti,16GB,512GB Storage
```