1. Inheritence means one class is deriving data from the single parent class but in case of multiple inheritance, a subclass is derived from more than one parent classes. Suppose A and B are the classes and class C is deriving data from A and B. We can say this is multiple inheritance.

class parent\_1:

pass

class parent\_2:

pass

class child(parent\_1, parent\_2):

pass

1. Lets say a is a object of parent class A and b is a object of child class B. Then b can use all the variables and methods of parent class A to perform the operations. This is known as delegation. Following is the example of delegation.

class A:

def \_\_init\_\_(self,name,age):

self.name=name

self.age=age

def f\_a(self):

return 'this is class A'

class B(A):

def \_\_init\_\_(self,number,\*args):

super(B,self).\_\_init\_\_(\*args)

self.number=number

a=A('shivansh',30)

b=B(10,'TESHU',25)

a.f\_a()

b.f\_a()

1. Suppose there are multiple class say, class A, class B and class C and their respective objects are a,b and c. So when we are going to call the final class C and we wanted to utilise the items of the other previous classes A and B then we use their objects a and b when we create the final class object c by passing a and b into that.

class A:

def \_\_init\_\_(self):

pass

a=A()

class B:

def \_\_init\_\_(self):

pass

b=B()

class C:

def \_\_init\_\_(selfo,bject\_a,object\_b):

pass

c=C(a,b)

1. Function whcih is defined insid ethe class and take the instance of a class as its first argument is known as bound function. This is because it bounded by the class. It can only perfrom the operation by using the class object.

# example of bound method

class A:

def \_\_init\_\_(self):

pass

def func(self):

return 'hello world'

obj = A()

obj.func

<bound method A.func of <\_\_main\_\_.A object at 0x0000023858ED79C8>>

1. When we create a class and put some attributes inside it, those can be used by us within the class. But it may happen that someone other than us can also be use the same attributes with similar name from outside or inside the class. So in that case there will be change in our program codes. This will change the entire meaning of that code. So in order to make it safe from other users we use double underscore ( ) as a notation for private attribues or methods So that others cannot use it and if they able to use it then they cannot be able to access it.

It can be access with some special way

class person:

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

self.name = name

self.\_\_age = 18

def \_\_detail(self):

return self.name,self.\_\_age

ob = person("priyanka")

ob.name

ob.\_person\_\_age

ob.\_person\_\_detail()