PYTHON OOPs

Inheritance : The process of inheriting the properties of one class by another class is called as inheritance.

· It is the best features in the concept of object oriented programming.

· It refers to defining a new class with little or no modifications to an existing class. The new class is called Derived or Child class and the existing class is called Base class or Parent class.

· Best feature of inheritance is code re-usability.

Syntax:

```
class Baseclass:
        Body of base class
class Derivedclass(Baseclass):
        Body of derived class
```

Example:

```
class Student:
        def __init__(self,name):
                self.name=name
        def getName(self):
                return self.name
        def isStudent(self):
        return "{} is from  Srichaitanya".format(self.name)
class School(Student):
        def details(self):
        return "{} is famous in Andhra
       Pradesh".format(self.name)
stu = School("Naveen")
print (stu.getName(),stu.isStudent())
```

```
stu = School("Narayana")

print (stu.getName(),stu.details())
```

Output :

```
Naveen Naveen is from Srichaitanya

Narayana Narayana is famous in Andhra Pradesh
```

Multiple Inheritance : The process of inheriting all features of base classes to the derived classes is called multiple inheritance.

· This is same as inheritance but difference here we can take multiple base classes to a subclass.

Syntax :

```
class One:
        pass
class Two:
        pass
class Three(One,Two):
        pass
```

Example :

```
class One:
        def __init__(self):
                self.state1 = "Tamil Nadu"
                print ("One class")
class Two:
        def __init__(self):
                self.state2 = "Telangana"
                print ("Two class")
class Three(One,Two):
        def __init__(self):
```

```
                One.__init__(self)
                Two.__init__(self)
                print ("Three")
        def output(self):
                print (self.state1, self.state2)
I = Three()
I.output()
```

Output :

```
One class
Two class
Three
Tamil Nadu Telangana
```

Multilevel inheritance : Inheriting properties from a child class and its associated parent class by a another class is called multilevel inheritance.

- Both parent and child properties will be inherited to the new child class.

Syntax :

```
class Parent:
        pass
class Child1(Parent):
        pass
class Child2(Child1):
        pass
```

Example :

```
class Cricket:
        def game(self):
                print ("Cricket is famous game")
class Batsmen(Cricket):
```

```
        def batsmen(self):
            print ("Modern Batsmen are ruling the cricket")
    class Leading(Batsmen):
        def leading(self):
            print ("Sachin is the leading runscorer in the world")
    a = Leading()
    a.game()
    a.batsmen()
    a.leading()
```

Output :

```
Cricket is famous game
Modern Batsmen are ruling the cricket
Sachin is the leading runscorer in the world
```

Access Parent class in subclass : Accessing a parent class from its subclass can be done in two ways:

Parent class name : In this method we can use parent class name to access the value in the parent class.

Example:

```
class Parent:
    def __init__(self,name):
        self.name = name
class Child(Parent):
    def __init__(self,name,country):
        Parent.name = name
        self.country = country
    def Output(self):
    print (Parent.name, self.country)
```

```
d = Child("Rajesh", "America")
d.Output()
```

Output :

```
Rajesh America
```

• Parent.name can be replaced with self.name both perform same actions.

Super () : In this method to access parent we will use super().

Example :

```
class Parent:
    def __init__(self,name):
        self.name = name
class Child(Parent):
        def __init__(self,name,sport):
            super(Child,self).__init__(name)
            self.sport = sport
        def print(self):
            print (self.name,":", self.sport)
d = Child("Dhoni", "Cricket")
d.print()
b = Child("Messie","Football")
b.print()
```

Output :

```
Dhoni : Cricket
Messie : Football
```

• issubclass() and isinstance() are used to check the relations between classes and instances.

Issubclass() : It is a python Boolean function which is used to check the condition whether the give class is subclass of another class according to the condition and returns True if condition is satisfied or False.

Syntax :

```
Issubclass(child,parent)
```

Example :

```
class First(object):
    pass
class Second(First):
    pass
class Third(Second):
    pass
print (issubclass(Second,First))
print (issubclass(First,Second))
print (issubclass(Third,First))
print (issubclass(Third,Second))
```

Output :

```
True
False
True
True
```

• In example **First** not the subclass of **Third** but it returned **True** because **Second** is the subclass of **Third** and **First** is subclass of **Second**.

Isinstance() : It is a function used to check whether the object is instance of the class or instance of any subclass of that class and return True.

Syntax :

isinstance(object,class)

Example :

```
class First:
        pass
class Second(First):
        pass
class Third(Second):
        pass
a = Second()
b = First()
c = Third()
print (isinstance(b, Second))
print (isinstance(a, First))
print (isinstance(c, First))
```

Output :

```
False
True
True
```