1716

polymorphism :-

We use for method Overloading & Overriding

Method Overloading :-

It consists of same name but different parameters.

Ex:
```
class Demo:
    def m1(self):
        print ("0 parameters")
    def m1(self,a):
        print ("1 parameter")
    def m1(self,a,b):
        print ("2 parameters")

o = Demo()
o.m1()
```

o/p: " Error, it missing 2 parameters "

If we remove all parameters def m1(self,a) as def m1(self): & def m1(self,a,b): as def m1(self): then the output will be

o/p: 2 parameters

In method overloading, it try to Executes only last method.

∴ We can conclude that method OverLoading is not going to support by python.

As there in java & in other languages we use data types, but in python we no need.

Eg:
```
class Demo:
    def m1(self, a):
        print("0 parameters", a)

o = Demo()
o.m1("dharma")
o.m1(10)
o.m1(2.5)
```

O/p:- 0 parameters dharma

0 parameters 10

0 parameters 2.5

In python, By default a support is there for method Overloading.

So we can conclude in python there is no need for method Overloading.

```
Class A:
    def m1(self):
        print("I'm from m1")

Class B(A):
    def m1(self):
        print("I don't need implementation")

o=B()
o.m1()
```

o/p:- I don't need implementation

Here same method name is taken & it takes second method.

If we also want to implement parent method.

```
Class A:
    def m1(self):
        print("I'm from m1")

Class B:
    def m1(self):
        print("I don't need implementation")
        super().m1()

o=B()
o.m1()
```

o/p:- I don't need implementation

I'm from m1.

Method Overriding means with same method name but implementation is different.

let take one more Example

```
class Parent:
    def car(self):
        print("Benz Car")
class Child(Parent):
    def car(self):
        print("BMW")
o=Child()
o.car()
```

o/p:- BMW

Super() is a keyword that helps to print from parents class

```
class Parent:
    def car(self):
        print("Benz Car")
class Child(Parent):
    def car(self):
        print("BMW")
        super().car()
o=child()
o.car()
```

o/p → BMW
Benz