OOPS in Python

```
class student:
    name="siddhesh"


s1=student()
print(s1.name)
```

```
siddhesh
```

```
class student:
    name="Badakh"

s1=student()
print(s1.name)

s2=student()
print(s2.name)
```

```
Badakh
Badakh
```

init

```
class student:

    def __init__(self,name,marks):
      self.name=name
      self.marks=marks
      print("adding new student in database")

s1=student("siddhesh",90)
print(s1.name,s1.marks)

s2=student("rahul",80)
print(s2.name,s2.marks)
```

```
adding new student in database
siddhesh 90
adding new student in database
rahul 80
```

Attribute

```
class student:
    clg_name="Mit college"
    name="rahul"

    def __init__(self,name,marks):
```

```
        self.name=name
        self.marks=marks
        print("adding new student in database")
s1=student("Siddhesh",90)
print(s1.name,s1.marks)
```

```
adding new student in database
Siddhesh 90
```

METHODS

```
class student:
    clg_name="MIT college"

    def __init__(self,name,marks):
        self.name=name
        self.marks=marks

    def Welcome(self):
        print("WELCOME STUDENT")

s1=student("siddhesh",99)
print(s1.name,s1.marks)
s1.Welcome()
```

```
siddhesh 99
WELCOME STUDENT
```

Abstration

```
class car:
    def __init__(self):
        self.acc=False
        self.brk=False
        self.clutch=False
    def start(self):
        self.clutch=True
        self.acc=True
        print("car started")
car1=car()
car1.start()
```

```
car started
```

Encapsulation

```
class Account:
    def __init__(self, bal, acc):
        self.balance = bal
```

```
        self.account_no = acc

    def debit(self, amount):
        self.balance -= amount
        print("Rs.", amount, "was debited")
        print("Total balance =", self.get_balance())

    def credit(self, amount):
        self.balance += amount
        print("Rs.", amount, "was credited")
        print("Total balance =", self.get_balance())

    def get_balance(self):
        return self.balance

acc1=Account(10000,12345)
print(acc1.balance)
print(acc1.account_no)

acc1.debit(10000)
acc1.credit(10000)
```

```
10000
12345
Rs. 10000 was debited
Total balance = 0
Rs. 10000 was credited
Total balance = 10000
```

Inheritance

```
class Animal:
    def speak(self):
        print("Animal Speaking")
#child class Dog inherits the base class Animal
class Dog(Animal):
    def bark(self):
        print("dog barking")
d = Dog()
d.bark()
d.speak()
```

```
dog barking
Animal Speaking
```

```
class Animal:
    def speak(self):
        print("Animal Speaking")
#The child class Dog inherits the base class Animal
class Dog(Animal):
```

```
    def bark(self):
        print("dog barking")
#The child class Dogchild inherits another child class Dog
class DogChild(Dog):
    def eat(self):
        print("Eating bread...")
d = DogChild()
d.bark()
d.speak()
d.eat()
```

```
dog barking
Animal Speaking
Eating bread...
```

```
class Calculation1:
    def Summation(self,a,b):
        return a+b;
class Calculation2:
    def Multiplication(self,a,b):
        return a*b;
class Derived(Calculation1,Calculation2):
    def Divide(self,a,b):
        return a/b;
d = Derived()
print(d.Summation(10,20))
print(d.Multiplication(10,20))
print(d.Divide(10,20))
```

```
30
200
0.5
```

Method Overloading

```
class Animal:
    def speak(self):
        print("speaking")
class Dog(Animal):
    def speak(self):
        print("Barking")
d = Dog()
d.speak()
```

```
Barking
```

Method Overriding

```
class Bank:
    def getroi(self):
```

```
        return 10;
class SBI(Bank):
    def getroi(self):
        return 7;

class ICICI(Bank):
    def getroi(self):
        return 8;
b1 = Bank()
b2 = SBI()
b3 = ICICI()
print("Bank Rate of interest:",b1.getroi());
print("SBI Rate of interest:",b2.getroi());
print("ICICI Rate of interest:",b3.getroi());
```

```
Bank Rate of interest: 10
SBI Rate of interest: 7
ICICI Rate of interest: 8
```