***#>>>>>>>>>>>>oops concept in python<<<<<<<<<<<<<<<<<<***

class Student:

St\_name = "Yash"

st\_obj=Student()

print(st\_obj.St\_name)

class Student:

def info(self):

name = "Yash"

branch= "AI"

print(name)

print(branch)

print("this is info method")

st\_obj = Student()

st\_obj.info()

**#Inheritance in Python**

class A:

A\_name = "Yash"

class B:

B\_name = "Ankur"

class C(B,A):

C\_name = "Rohan"

obj = C()

print(obj.A\_name)

print(obj.B\_name)

print(obj.C\_name)

**#encapsulation**

class A:

name= "Yash"

\_name= "Ankur" #protected variable

\_\_name= "Rohan" #private variable

obj=A()

print(obj.name) #public variable

print(obj.\_name) #protected variable

# print(obj.\_\_name) #private variable (will raise an error)

obj.\_A\_\_name #accessing private variable using name mangling

**#method overiding**

class A:

def info(self):

print("This is class A")

class B(A):

def info(self):

print("This is class B")

obj = B()

obj.info() # This will call the info method of class B, overriding class A's method

**#method overloading**

class A:

def info(self,name):

print(f"Name is {name}")

def info(self, age):

print(f"Age is {age}")

obj = A()

obj.info("Yash") # This will call the second info method, as it overrides