class **Vehicle**():  
 '''  
 *The instantiation operation (“calling” a class object) creates an empty object. Many classes like to create objects with  
 instances customized to a specific initial state. Therefore a class may define a special method named \_\_init\_\_(),  
 like this:*  
 '''  
 color ="red" # class attribute should not be changed but we can change outside the class  
 pi=3.14  
 V\_cnt=0 # class attribute  
  
 def \_\_init\_\_(self,body,make): #Constructor Method,by default it will call when we create a object  
 self.vehicle\_body=body  
 self.vehicle\_make=make  
 Vehicle.V\_cnt +=1  
 ''''  
 When a class defines an \_\_init\_\_() method, class instantiation automatically invokes \_\_init\_\_() for the newly-created class   
 instance. So in this example, a new, initialized instance can be obtained by:  
 '''  
 def get\_vehicle\_count(self)-> int:  
 return Vehicle.V\_cnt  
  
 def isMatch(self,str\_word):  
 if self.vehicle\_make==str\_word:  
 print(f" Matches {self.vehicle\_make} and {str\_word}")  
 else:  
 print(f"no Match {self.vehicle\_make} and {str\_word}")  
  
 def drive(self):  
 print('Vehicle Driving..')  
  
class **Truck**(Vehicle):  
 def drive(self): #override the method ,inheritance  
 print('Truck Driving..')  
  
class **motercycle**(Vehicle):  
 def drive(self): #override the method ,inheritance  
 print('MoterCycle Driving is very fast ..')  
  
def main():  
 car1 = Vehicle('Jeep','Toyota')  
 car2 = Vehicle('SUV', 'Honda')  
 car3 = Vehicle('Truck', 'RAM')  
 car4 = Vehicle('Hatch Back', 'Maruti')  
 #Vehicle.color = 'White' # class attribute  
 print(car1.vehicle\_make +' '+ car1.vehicle\_body +' '+car1.color)  
 print(car2.vehicle\_make +' '+ car2.vehicle\_body +' '+car2.color)  
 print(car3.vehicle\_make +' '+ car3.vehicle\_body +' '+car3.color)  
 print(car4.vehicle\_make +' '+ car4.vehicle\_body +' '+car4.color)  
 car1.color = 'Black' # instance Variable  
 #print(type(car1)) #<class '\_\_main\_\_.Vehicle'>  
 print("===================================")  
 print(car1.vehicle\_make + ' ' + car1.vehicle\_body + ' ' +car1.color)  
 print(car2.vehicle\_make + ' ' + car2.vehicle\_body + ' ' + car2.color)  
 print(car3.vehicle\_make + ' ' + car3.vehicle\_body + ' ' + car3.color)  
 print(car4.vehicle\_make + ' ' + car4.vehicle\_body + ' ' + car4.color)  
 car2.isMatch('Toyota')  
 print(f"No# of cars {car1.get\_vehicle\_count()}")  
 truck1 = Truck('Tata Vajra','TATA')  
 truck2 = Truck('Toyota Aurmer','Toyota')  
 truck3 = Truck('Tesla Truck','TESLA')  
 print(truck1.vehicle\_make + ' ' + truck1.vehicle\_body + ' ' + truck1.color)  
 print(truck2.vehicle\_make + ' ' + truck2.vehicle\_body + ' ' + truck2.color)  
 print(truck3.vehicle\_make + ' ' + truck3.vehicle\_body + ' ' + truck3.color)  
 print(f"No# of Trucks {truck1.get\_vehicle\_count()}")  
 print(truck1.drive())  
 motercycle1 = motercycle('Yamaha ', 'BAJAJ')  
 print(motercycle1.drive())  
 for v in [truck1,motercycle1,car1]: # three diff objects  
 v.drive() # same method defined for three diff objects /classes   
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 print('main Starts ......')  
 main()  
 print("\*\*\*\*DONE MAIN\*\*\*\*")  
  
  
C:\Pyspark\_Installed\Python3.9.6\python.exe "C:/Users/shail/Downloads/mystyff/mystyff/yp/py/py class/Class\_Test.py"

main Starts ......

Toyota Jeep red

Honda SUV red

RAM Truck red

Maruti Hatch Back red

===================================

Toyota Jeep Black

Honda SUV red

RAM Truck red

Maruti Hatch Back red

no Match Honda and Toyota

No# of cars 4

TATA Tata Vajra red

Toyota Toyota Aurmer red

TESLA Tesla Truck red

No# of Trucks 7

Truck Driving..

None

MoterCycle Driving is very fast ..

None

Truck Driving..

MoterCycle Driving is very fast ..

Vehicle Driving..

\*\*\*\*DONE MAIN\*\*\*\*

class Emp(object):  
 def \_\_init\_\_(self,name,salary):  
 self.name=name  
 self.salary=salary  
 return None  
 def display(self):  
 print(f"The name is: {self.name}\nThe salary is: {self.salary}")  
 return None

outside class  
emp1=Emp('Ramu',56000)  
emp2=Emp("Naren",90000)  
  
emp1.display()  
emp2.display()

The name is: Ramu

The salary is: 56000

The name is: Naren

The salary is: 90000

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Collection\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

x = 'hi Shailendra Joshi'  
y = 'not working today relaxing..'  
z="python is very easy"  
str="python./i pot porking"  
  
'''  
print(dir(x))  
print(x.title())  
print(y.capitalize()) # only first char will be in caps  
print(x.swapcase())  
print(x.upper())  
print(x.casefold())  
print(str)  
print(str.strip('not'))  
print(str.count('not'))  
print(str.index('pot'))  
print(str.index('p',11))  
print(str.find('z',11))  
print(str.index('z',11))  
'''  
  
#sal = eval(input("Please enter your Salary: "))  
#print(sal\*20/100)  
  
my\_list=[10,20,30,40,"python",50,60,40,80,90]  
print(my\_list[0])  
print(my\_list[0:10])  
my\_list[0]=100  
print(my\_list[:])  
#print(dir(my\_list))  
print(my\_list.index(40))  
print(my\_list.index(40,4)) #from 4 position  
my\_list.append(200)  
#my\_list.clear()  
print(my\_list)  
  
my\_list.insert(0,10)  
print(my\_list)  
my\_new\_list =['Hi','Shailendra12']  
my\_list.append(my\_new\_list)  
print( my\_list)  
my\_list.extend(my\_new\_list)  
print(my\_list)  
#my\_list.pop() #remove last position data  
print(my\_list.pop()) #remove last position data also shows what was deleted  
print(my\_list)  
  
#print(my\_list.pop(0)) #remove position data also shows what was deleted aslo  
#my\_list.remove(len(my\_list))  
#my\_list.sort()  
  
'''  
my\_list.remove('Hi')  
my\_list.remove('Shailendra')  
my\_list.remove('python')  
'''  
  
print(my\_list)  
#my\_list.sort()  
print(my\_list)  
my\_list.reverse()  
print(my\_list)  
#my\_list.sort(reverse=True)  
#print(my\_list)  
  
print('tuple')  
my\_tuple=(4756191565,475675454,10000022)  
print(bool(my\_tuple))  
my\_new\_tuple=(4756191565,475675454,['Hi','Shailendra'],10000022)  
print(my\_new\_tuple[2]) #['Hi', 'Shailendra']  
print(my\_new\_tuple[2][1]) #Shailendra  
print(my\_new\_tuple[2][0]) #Hi  
#my\_tuple[0]=100000000 # immutable Error  
print(my\_tuple[0])  
x=1,2,3  
y=19,  
print(type(x))  
print(type(y))  
my\_dict={'fruit':'apple','animal':'fox',1:'one','two':2}  
print(my\_dict[1])  
print(my\_dict['animal'])  
#print(my\_dict['three']) #error  
print(my\_dict.get('three') ) #None  
  
  
my\_dict['three'] =3  
print(my\_dict)  
my\_dict['three'] =56  
print(my\_dict)  
print(my\_dict.keys())  
print(my\_dict.values()) #dict\_values(['apple', 'fox', 'one', 2, 56])  
print(my\_dict.items()) #dict\_items([('fruit', 'apple'), ('animal', 'fox'), (1, 'one'), ('two', 2), ('three', 56)])  
  
  
#str= input("Enter a String(word) : ")  
  
'''  
str=' Hi Shailendra how are you doing'  
print(str.split())  
my\_lst=str.split()  
print(type(my\_lst))  
#print(my\_lst)  
#my\_lst.reverse()  
#print("First " ,my\_lst)  
# OR  
print(my\_lst[::-1])  
output=" ".join(my\_lst[::-1])  
print(output)  
#print(my\_lst[::-1])  
  
my\_int\_list =[1,2,3,4,5,6,7]  
my\_int\_list.reverse()  
print(my\_int\_list)

C:\Pyspark\_Installed\Python3.9.6\python.exe "C:/Users/shail/Downloads/mystyff/mystyff/yp/py/py class/Python\_Test.py"

10

[10, 20, 30, 40, 'python', 50, 60, 40, 80, 90]

[100, 20, 30, 40, 'python', 50, 60, 40, 80, 90]

3

7

[100, 20, 30, 40, 'python', 50, 60, 40, 80, 90, 200]

[10, 100, 20, 30, 40, 'python', 50, 60, 40, 80, 90, 200]

[10, 100, 20, 30, 40, 'python', 50, 60, 40, 80, 90, 200, ['Hi', 'Shailendra12']]

[10, 100, 20, 30, 40, 'python', 50, 60, 40, 80, 90, 200, ['Hi', 'Shailendra12'], 'Hi', 'Shailendra12']

Shailendra12

[10, 100, 20, 30, 40, 'python', 50, 60, 40, 80, 90, 200, ['Hi', 'Shailendra12'], 'Hi']

[10, 100, 20, 30, 40, 'python', 50, 60, 40, 80, 90, 200, ['Hi', 'Shailendra12'], 'Hi']

[10, 100, 20, 30, 40, 'python', 50, 60, 40, 80, 90, 200, ['Hi', 'Shailendra12'], 'Hi']

['Hi', ['Hi', 'Shailendra12'], 200, 90, 80, 40, 60, 50, 'python', 40, 30, 20, 100, 10]

tuple

True

['Hi', 'Shailendra']

Shailendra

Hi

4756191565

<class 'tuple'>

<class 'tuple'>

one

fox

None

{'fruit': 'apple', 'animal': 'fox', 1: 'one', 'two': 2, 'three': 3}

{'fruit': 'apple', 'animal': 'fox', 1: 'one', 'two': 2, 'three': 56}

dict\_keys(['fruit', 'animal', 1, 'two', 'three'])

dict\_values(['apple', 'fox', 'one', 2, 56])

dict\_items([('fruit', 'apple'), ('animal', 'fox'), (1, 'one'), ('two', 2), ('three', 56)])