class employee:  
 number\_of\_employee=0  
 increment=2.5  
 def \_\_init\_\_(self, fname, lname, salary):  
 self.fname=fname  
 self.lname=lname  
 self.salary=int(salary)  
 employee.number\_of\_employee += 1  
 def increase(self):  
 self.salary=int(self.salary \* employee.increment)  
  
prem = employee('prem','prakash','120000')  
harry = employee('harry','jacson','90000')  
  
print(f"Total Number of Employee : {employee.number\_of\_employee}")  
print(f"Existing salary for Prem is:{prem.salary}")  
prem.increase()  
print(f"New salary, after increment of 2.5% for Prem is:{prem.salary}") *#o/p=New salary, after increment  
# of 2.5% for Prem is:300000*class programmer(employee): *#inheritance (New class, using new functions along with old one and real life use to  
 #increase salary of programmers seperatly.* def \_\_init\_\_(self,fname,lname,salary,language,exp):  
 super().\_\_init\_\_(fname,lname, salary) *# We use super(). so that this class can get old objects from super  
 #class* self.language=language  
 self.exp=exp  
 def increase(self):  
 self.salary=int(self.salary \*(employee.increment+2))  
 return self.salary *# We write this so that the code can understand that it has a return value, else it will not  
 #provice any output*prem=programmer('prem','prakash','120000','Python','3 years')  
sparsh=programmer('sparsh', 'singh','10','starting','fresher')  
prem.increase()  
print(prem.salary) *#O/P=540000*print(f"Total number of emplyoees are {employee.number\_of\_employee}") *#o/p=4*