'''

Q.Create a class Student with attributes pin, name and behaviour display\_details ,create 3

objects and call display\_details

class Student:

def \_\_init\_\_(self, name,pin,behaviour):

self.pin = pin

self.name = name

self.behaviour=behaviour

def display(self):

print("Pin: %d \nName: %s \nBehaviour:%s" % (self.pin, self.name,self.behaviour))

s1 = Student("Nagendra", 55,'good')

s2 = Student("Jaggu", 58,'average')

s3 = Student("sandeep", 56,'bad')

s1.display()

s2.display()

s3.display()

output:

Pin: 55

Name: Nagendra

Behaviour:good

Pin: 58

Name: Jaggu

Behaviour:average

Pin: 56

Name: sandeep

Behaviour:bad

'''

'''

Q.Create a class Player and derive 2 classes Batsman and Bowler with methods runs\_ scored

and wickets\_taken and an overridden method rating.

class Player:

x = 0

def \_init\_(self,name):

self.name=name

print(f"{self.name} joined")

def played\_count(self) :

self.x = self.x + 1

print(f'Played count of {self.name} is {self.x}')

def rating(self):

if self.wic>3:

print('Rating:5')

if self.runs>50:

print('Rating:5')

class Batsman(Player):

runs = 0

def score(self,s):

self.runs = self.runs + s

self.played\_count()

print(self.name,"score:",self.runs)

def rating(self):

if self.runs>90:

print('Rating:9')

else:

print('Rating:8')

class Bowler(Batsman):

wic=0

def wickets(self,w):

self.wic=self.wic+w

print(self.name,"wickets:",self.wic)

def rating(self):

if self.wic>8:

print('Rating:9')

else:

print('Rating:8')

s = Player("Dhoni")

s.played\_count()

j = Batsman("Virat")

j.score(50)

j.score(40)

j.rating()

w = Player("Bumrah")

s.played\_count()

i = Bowler("bhuvi")

i.wickets(5)

i.wickets(3)

i.rating()

output:

Dhoni joined

Played count of Dhoni is 1

Virat joined

Played count of Virat is 1

Virat score: 50

Played count of Virat is 2

Virat score: 90

Rating:8

Bumrah joined

Played count of Dhoni is 2

bhuvi joined

bhuvi wickets: 5

bhuvi wickets: 8

'''

'''

Q.Create a class with all types of variables public,private and protected

class Employee:

no\_of\_leaves = 8

var = 8

\_protec = 9

\_\_pr = 98

def \_\_init\_\_(self, aname, asalary, arole):

self.name = aname

self.salary = asalary

self.role = arole

def printdetails(self):

return f"The Name is {self.name}. Salary is {self.salary} and role is {self.role}"

@classmethod

def change\_leaves(cls, newleaves):

cls.no\_of\_leaves = newleaves

@classmethod

def from\_dash(cls, string):

return cls(\*string.split("-"))

@staticmethod

def printgood(string):

print("This is good " + string)

emp = Employee("harry", 343, "Programmer")

print(emp.\_Employee\_\_pr)

output:

98

'''