1:Defining a class in Python with parameterized constructor

class Person:

def \_\_init\_\_(self, name, age):

self.name = name

self.age = age

p1 = Person("John", 36)

print(p1.name)

print(p1.age)

2:Defining a class in Python with default constructor

class Person:

def \_\_init\_\_(self):

self.name = input()

self.age = int(input())

p1 = Person()

print(p1.name)

print(p1.age)

3: classes with methods

class Person:

def \_\_init\_\_(self):

self.name = input()

self.age = int(input())

def myfunc(self):

print("Hello my name is " + self.name)

p1 = Person()

p1.myfunc()

4: **Modify Object properties**

class Person:

def \_\_init\_\_(self,name,age):

self.name = name

self.age = age

def myfunc(self):

print("Hello my name is " + self.name)

p1 = Person("John",22)

p1.myfunc()

p1.age=56

print(p1.age)

**5:Public access modifier :**

class Employee:

# constructor

def \_\_init\_\_(self, name, sal):

self.name = name;

self.sal = sal;

emp = Employee("Ironman", 999000);

print(emp.sal)

**6:Protected access modifie**r

class Employee:

def \_\_init\_\_(self, name, sal):

self.\_name = name;

self.\_sal = sal;

emp = Employee("Ironman", 999000);

print(emp.\_sal)

**7:Private access modifie**r

class Employee:

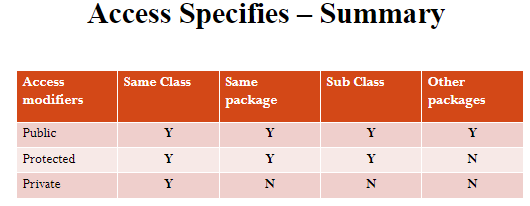
def \_\_init\_\_(self, name, s):

self.\_name = name;

self.\_\_sal = s;

emp = Employee("Ironman", 999000);

print(emp.\_\_sal)



8:Class variables VS instance variables

Case1:

class Employee:

org\_name="Infosys"

def \_\_init\_\_(self, name, s):

self.name = name;

self.sal = s;

def display(self):

print(self.name,self.sal,self.org\_name)

emp1 = Employee("Ironman", 999000);

emp1.display()

emp2 = Employee("XYZ", 500000);

emp2.display()

emp3 = Employee("ABC", 340000);

emp3.display()

Output:

Ironman 999000 Infosys

XYZ 500000 Infosys

ABC 340000 Infosys

Case2:

class Employee:

org\_name="Infosys"

def \_\_init\_\_(self, name, s):

self.name = name;

self.sal = s;

def display(self):

print(self.name,self.sal,self.org\_name)

emp1 = Employee("Ironman", 999000);

emp1.display()

Employee.org\_name="CTS"

emp1 = Employee("Ironman", 999000);

emp1.display()

emp2 = Employee("XYZ", 500000);

emp2.display()

emp3 = Employee("ABC", 340000);

emp3.display()

output:

Ironman 999000 Infosys

Ironman 999000 CTS

XYZ 500000 CTS

ABC 340000 CTS

Case3:

class Employee:

org\_name="Infosys"

def \_\_init\_\_(self, name, s):

self.name = name;

self.sal = s;

emp1 = Employee("Ironman", 999000);

emp2 = Employee("xyz", 23000);

emp3 = Employee("pqr", 45000);

emp1.org\_name="CTS"

print(emp1.org\_name)

print(emp2.org\_name)

print(emp3.org\_name)

Single Inheritance:

class University:

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

self.univ\_name = name;

class student(University):

def \_\_init\_\_(abc, uname,name, r):

University.\_\_init\_\_(abc,uname)

abc.sname = name;

abc.roll=r

def display(abc):

print(abc.univ\_name,abc.sname,abc.roll)

s=student("Amrita","Rahul",32)

s.display()