LET'S PY by shakti Jaiswal

# OOPs Part - 2

## Using members of one class inside another class:

**We can use members of one class inside another class by using the following ways**

**1. By Composition (Has-A Relationship)**
**2. By Inheritance (IS-A Relationship)**

## 1. By Composition (Has-A Relationship):

**By using Class Name or by creating object we can access members of one class inside another class is nothing but composition (Has-A Relationship).**

**The main advantage of Has-A Relationship is Code Reusability.**

## 2. By Inheritance(IS-A Relationship):

**What ever variables, methods and constructors available in the parent class by default available to the child classes and we are not required to rewrite. Hence the main advantage of inheritance is Code Reusability and we can extend existing functionality with some more extra functionality.**

**Syntax :**
**class childclass(parentclass):**

**What ever methods present in Parent class are automatically available to the child class and hence on the child class reference we can call both parent class methods and child class methods.**

## IS-A vs HAS-A Relationship:

**If we want to extend existing functionality with some more extra functionality then we should go for IS-A Relationship**

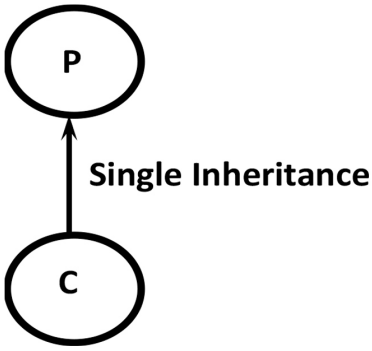
**If we dont want to extend and just we have to use existing functionality then we should go for HAS-A Relationship**

**Eg: Employee class extends Person class Functionality**
**But Employee class just uses Car functionality but not extending**

# Types of Inheritance:

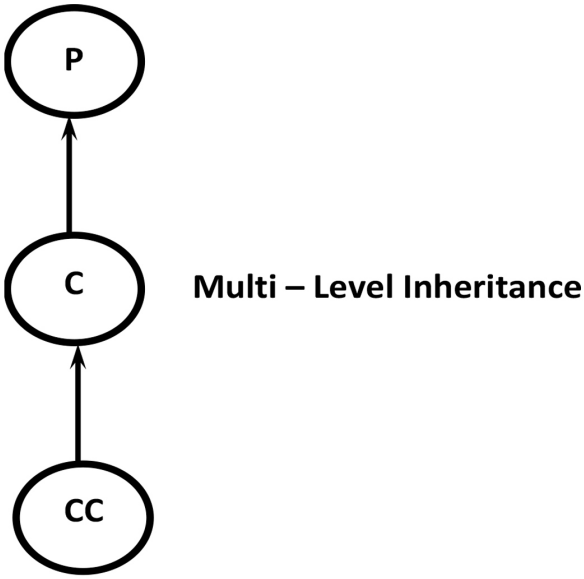
## 1. Single Inheritance:

**The concept of inheriting the properties from one class to another class is known as single inheritance.**



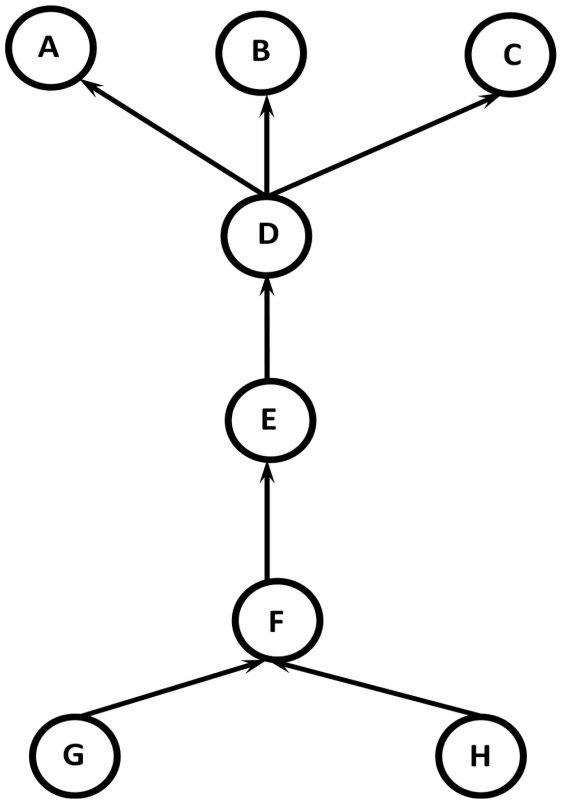

## 2. Multi Level Inheritance:

**The concept of inheriting the properties from multiple classes to single class with the concept of one after another is known as multilevel inheritance**



## 5. Hybrid Inheritance:

**Combination of Single, Multi level, multiple and Hierarchical inheritance is known as Hybrid Inheritance.**



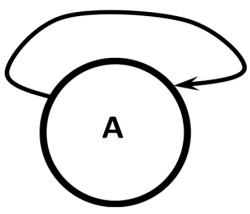

## 6. Cyclic Inheritance:

**The concept of inheriting properties from one class to another class in cyclic way, is called Cyclic inheritance.Python won't support for Cyclic Inheritance of course it is really not required.**

**Eg - 1:**

**class A(A):pass**

**NameError: name 'A' is not defined**



# Method Resolution Order (MRO):

In Hybrid Inheritance the method resolution order is decided based on MRO algorithm.
This algorithm is also known as C3 algorithm.
Samuele Pedroni proposed this algorithm.
It follows DLR (Depth First Left to Right)
i.e Child will get more priority than Parent.
Left Parent will get more priority than Right Parent

MRO(X)=X+Merge(MRO(P1),MRO(P2),...,ParentList)

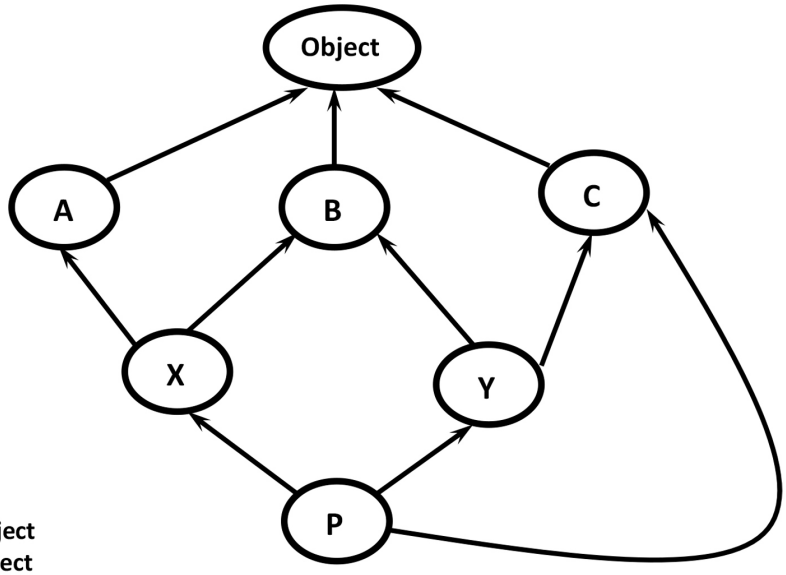
## Head Element vs Tail Terminology:

Assume C1,C2,C3,...are classes.
In the list : C1C2C3C4C5....
C1 is considered as Head Element and remaining is considered as Tail.

## How to find Merge:

Take the head of first list
If the head is not in the tail part of any other list,then add this head to the result and remove it from the lists in the merge.
If the head is present in the tail part of any other list,then consider the head element of the next list and continue the same process.

Note: We can find MRO of any class by using mro() function.
print(ClassName.mro())




mro(A)=A,object
mro(B)=B,object
mro(C)=C,object
mro(X)=X,A,B,object
mro(Y)=Y,B,C,object
mro(P)=P,X,A,Y,B,C,object

## Finding mro(P) by using C3 algorithm:

**Formula:** MRO(X)=X+Merge(MRO(P1),MRO(P2),...,ParentList)

mro(p)= P+Merge(mro(X),mro(Y),mro(C),XYC)
= P+Merge(XABO,YBCO,CO,XYC)
= P+X+Merge(ABO,YBCO,CO,YC)
= P+X+A+Merge(BO,YBCO,CO,YC)
= P+X+A+Y+Merge(BO,BCO,CO,C)
= P+X+A+Y+B+Merge(O,CO,CO,C)
= P+X+A+Y+B+C+Merge(O,O,O)
= P+X+A+Y+B+C+O

# super() Method:

super() is a built-in method which is useful to call the super class constructors,variables and methods from the child class.

## How to call method of a particular Super class:

We can use the following approaches

**1. super(D,self).m1()**
It will call m1() method of super class of D.

**2. A.m1(self)**
It will call A class m1() method

## Various Important Points about super():

**Case-1:** From child class we are not allowed to access parent class instance variables by using super(),Compulsory we should use self only.
But we can access parent class static variables by using super().

**Case-2:** From child class constructor and instance method, we can access parent class instance method,static method and class method by using super()