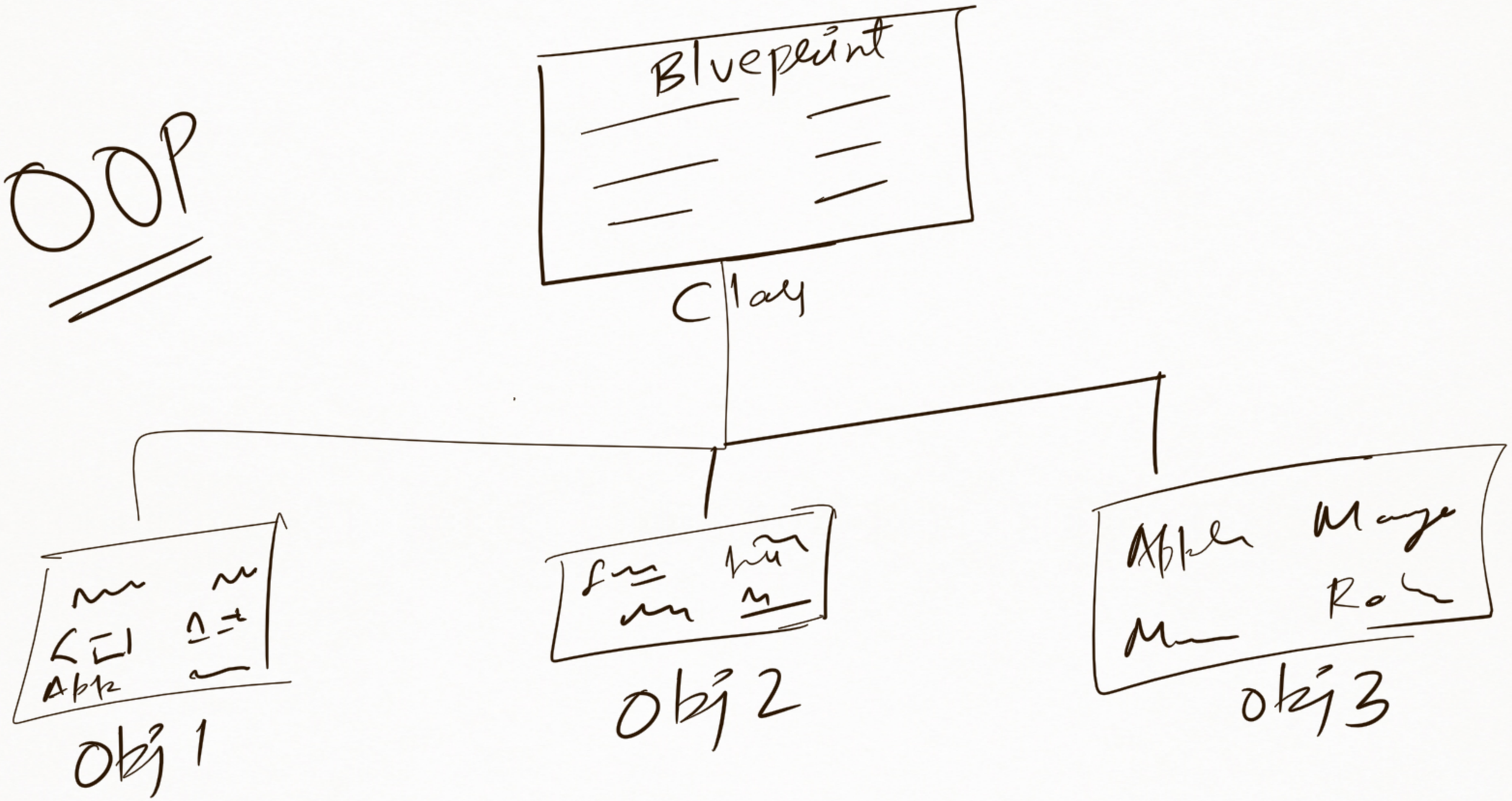
OOP



Class

- Methods
  - Part of Obj meth
    - classmethod — @classmethod (cls, ---
    - Objectmethod (self, ---
    - staticmethod — @staticmethod — X
      - → does not manipulate obj or class var
      - → used for class functionalities
  - ✓ magic methods
    - __ len __ (self)
    - __ str __ (self)

class Vehicle

__init__(self, A, B)

↑ object method

* → It runs when we create an object of this class.

initialiser

constructor

obj_ref = Vehicle(Mohit, Singh)

class Vehicle

→ variables

→ obj var

→ class var

class Vehicle
def test(self, name):
y = 34
y2 = y + name

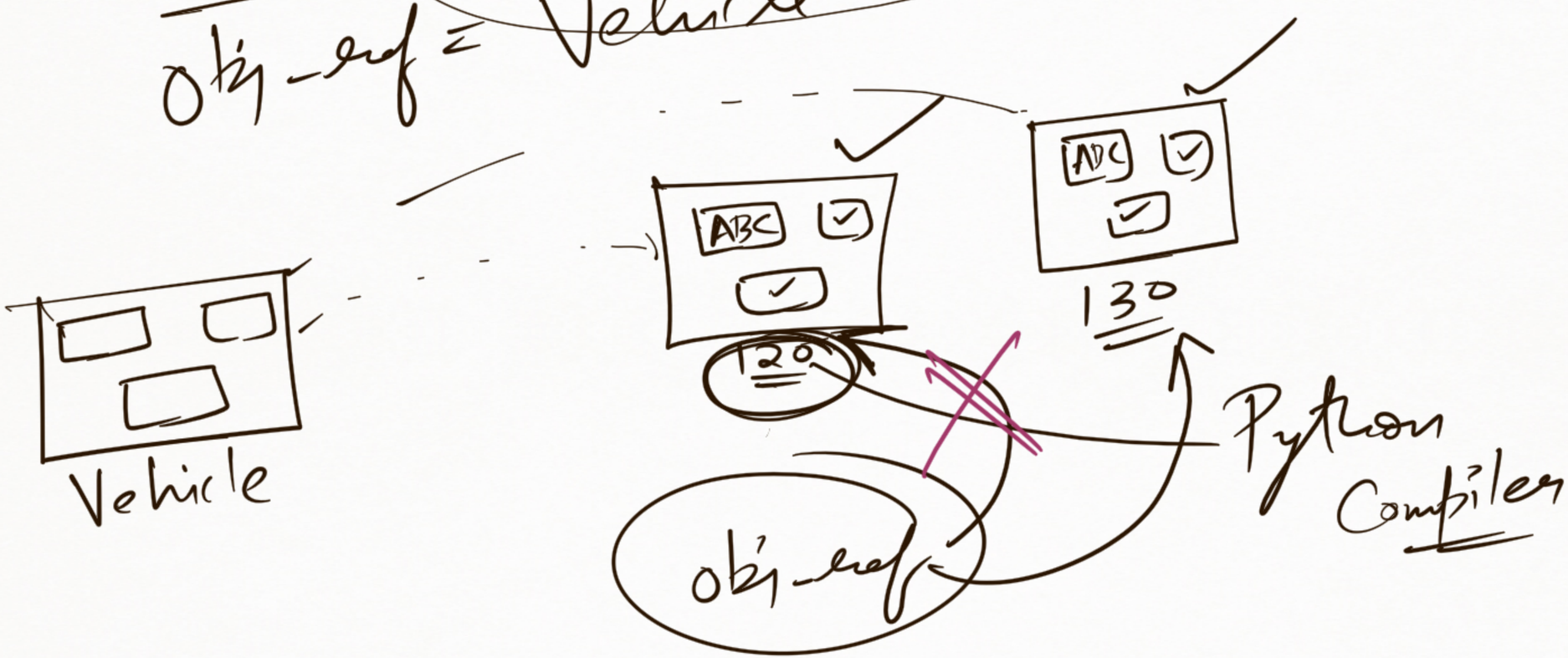
③ Define in obj methods.

① obj var ⇒ ① self is used to refer obj var.
② separate copy per each obj.

① class var ⇒ ① cls ✓ Vehicle
② same for each obj
③ Beginning before any method.

obj_ref = Vehicle ("ABC")

obj_ref = Vehicle ("ABC")

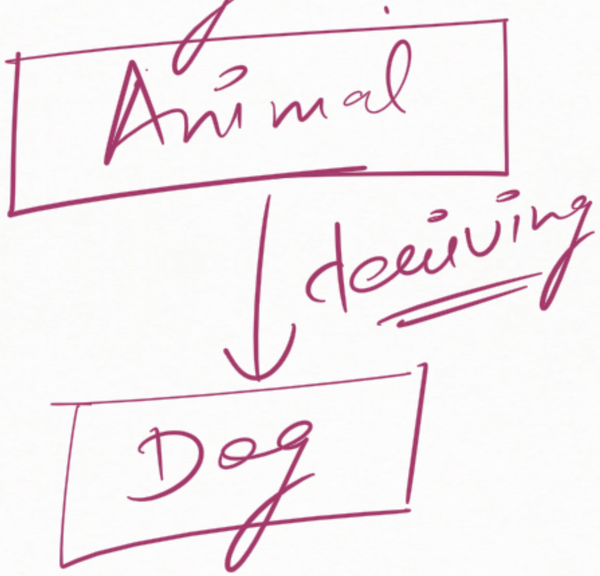


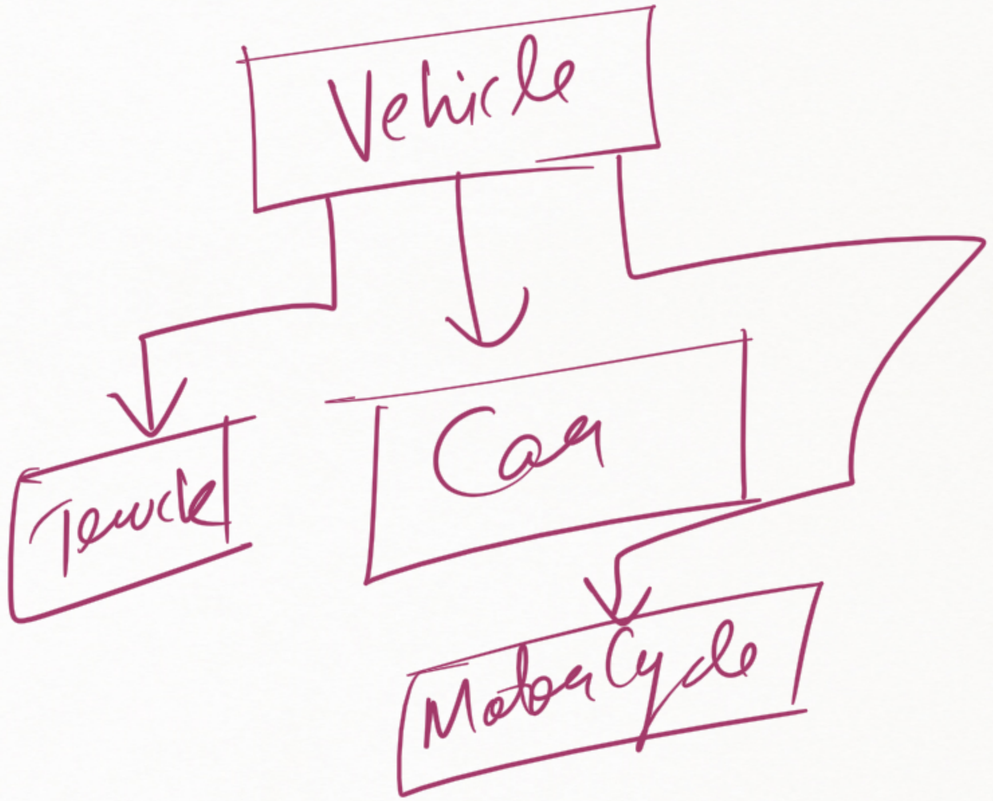
Inheritance

Deriving properties and methods from existing Class.

Var

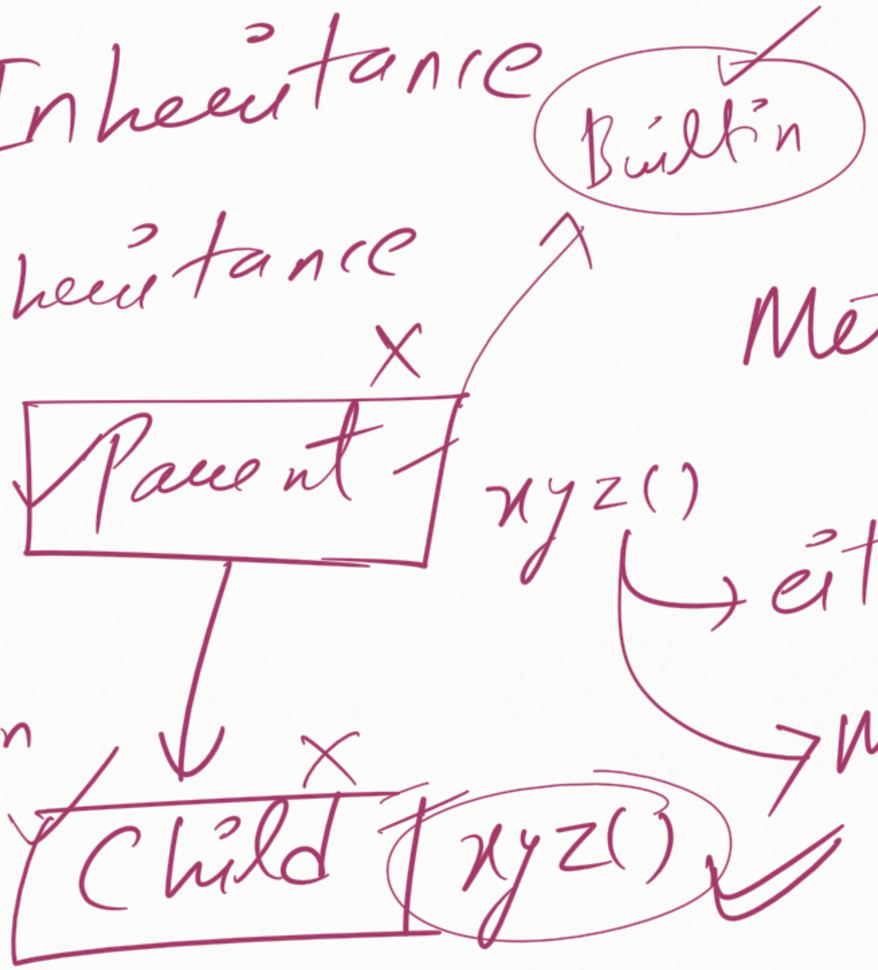
* Reusability
* Reduce redundancy





Types of Inheritance

1. Single Inheritance



Method Overriding?

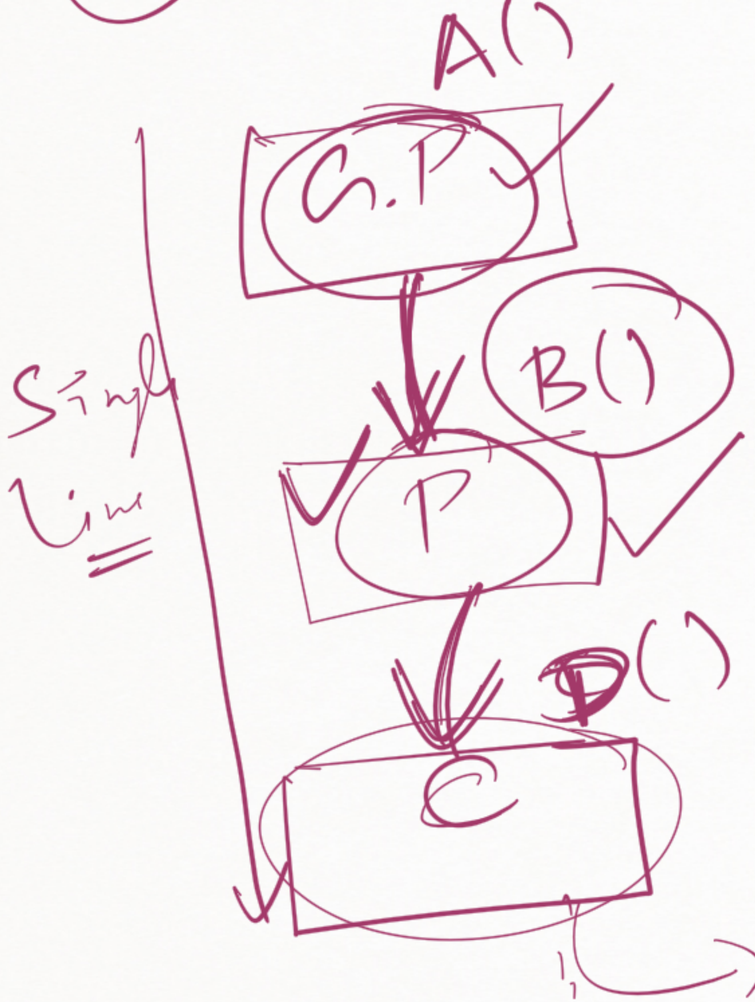
xyz()
→ either use it
OR
Modify it

MRO
Method Resolution order
child, Parent, builtin

xyz
help(obj)

obj = Child()
obj.xyz => child one

② Multilevel Inheritance



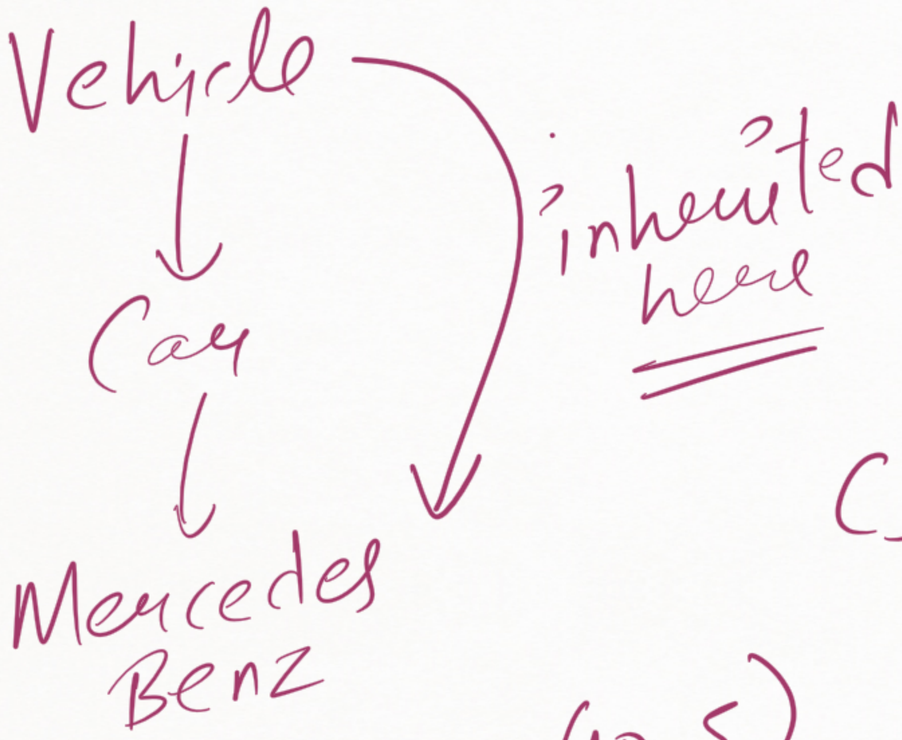
MRO

GC, C, P, G.P ✓

Method Overriding

G.C.B() ?

what methods we can override in G.C



real

$C_1 = 10 + 5j$

$C_2 = 20 + 3j$

imaginary

$C_3 = C_1 + C_2$

$= 30 + 8j$

$C_1 = Complex(10, 5)$

$C_2 = Complex(20, 3)$

$C_4 = C_1 - C_2$

$= -10 + 2j$

Magic Methods

```
[ ]: ## MAGIC METHODS
```

```
[ ]: MAKE A CLASS ON COMPLEX NUMBERS ANS USE VARIOUS MAGIC METHODS.
```

```
[8]: class Complex:
         def __init__(self,real,imag):
             self.real = real
             self.imag = imag

         def __add__(self,other):
             return Complex(self.real+other.real,self.imag+other.imag)
```

called whenever I use the "+" operator between 2 Complex objects

```
[9]: c1 = Complex(10,5)
     c2 = Complex(5,3)
```

```
10]: c3 = c1+c2
     print(c3.real,c3.imag)
```

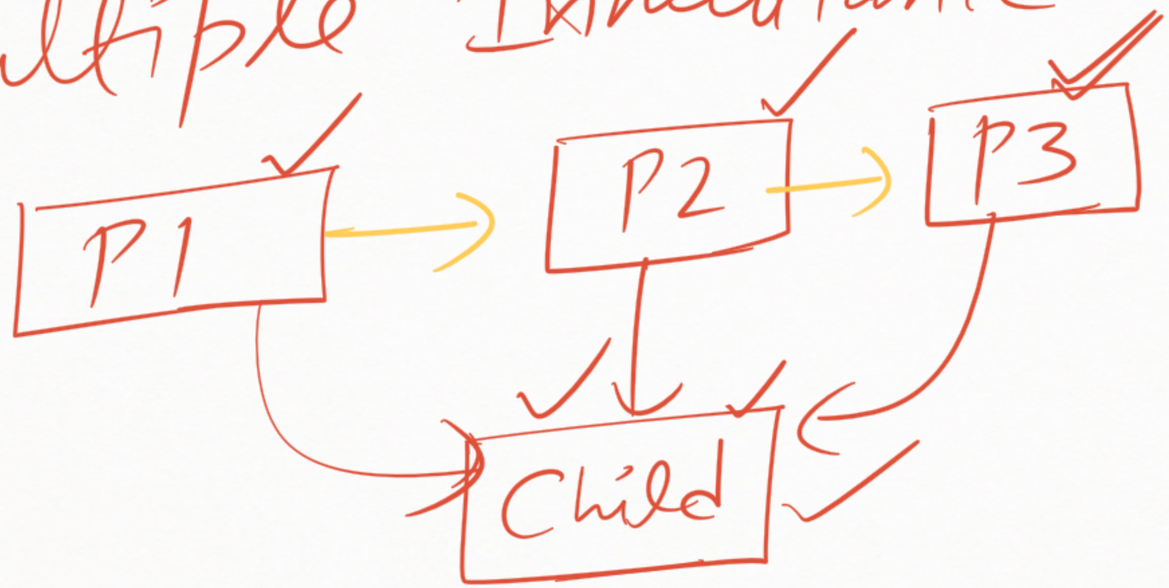
10+5j
5+3j

c1 ---> self
c2 ---> other

c3 = 15+8j

Note
Only __add__ works
It is predefined in Python.

③ Multiple Inheritance

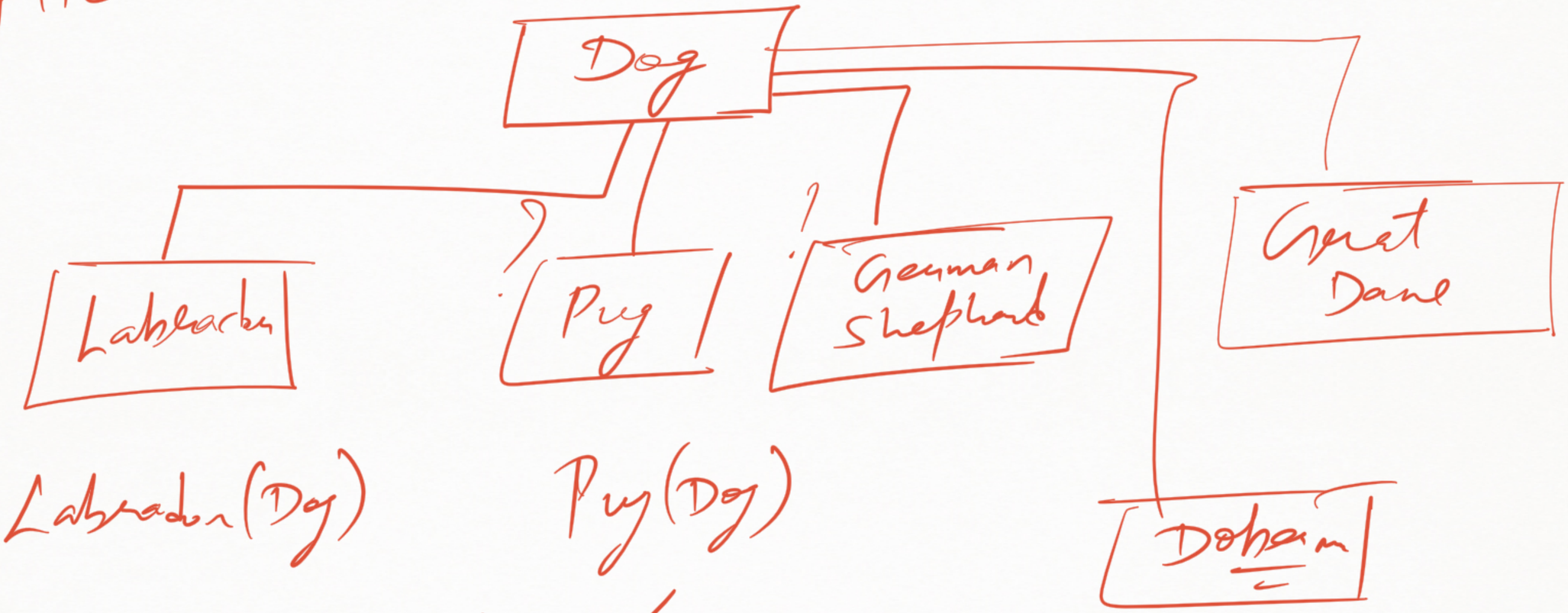


class Child(P1, P2, P3)

MRO => Child, P1, P2, P3, builtin

Method Overriding

④ Hierarchical Inheritance

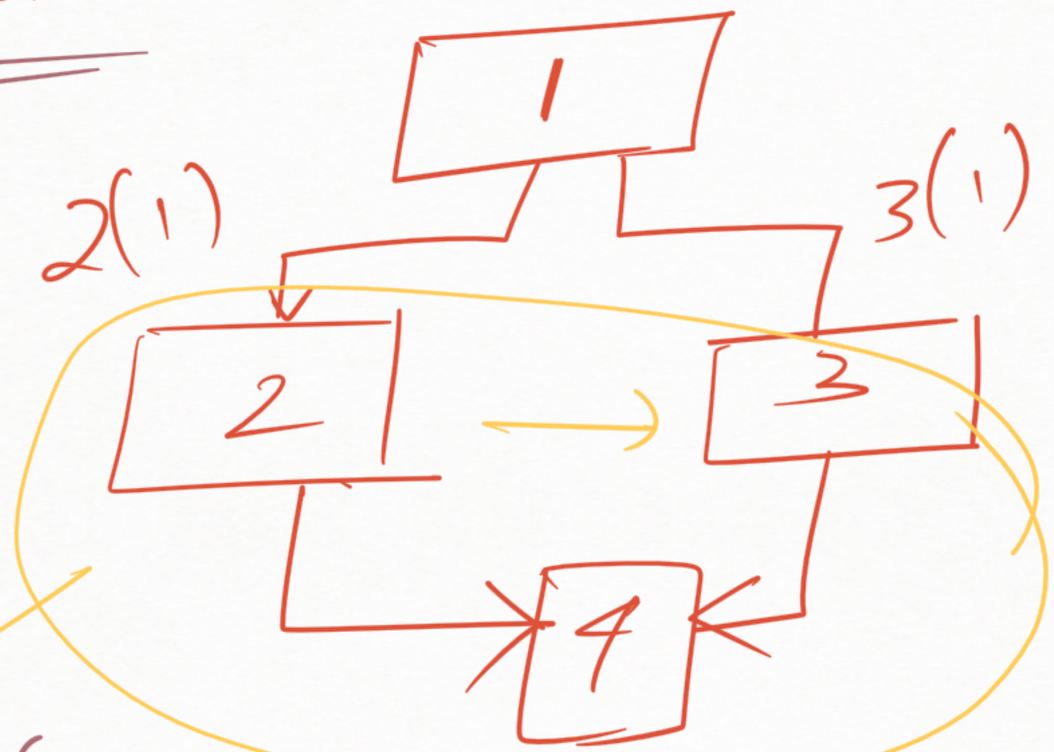


Labrador(Dog)

Pug(Dog)

Several MRO's ✓

⑤ Hybrid Inheritance



MRO

4, 2, 3, 1

Multiple

* there is more to MRO