

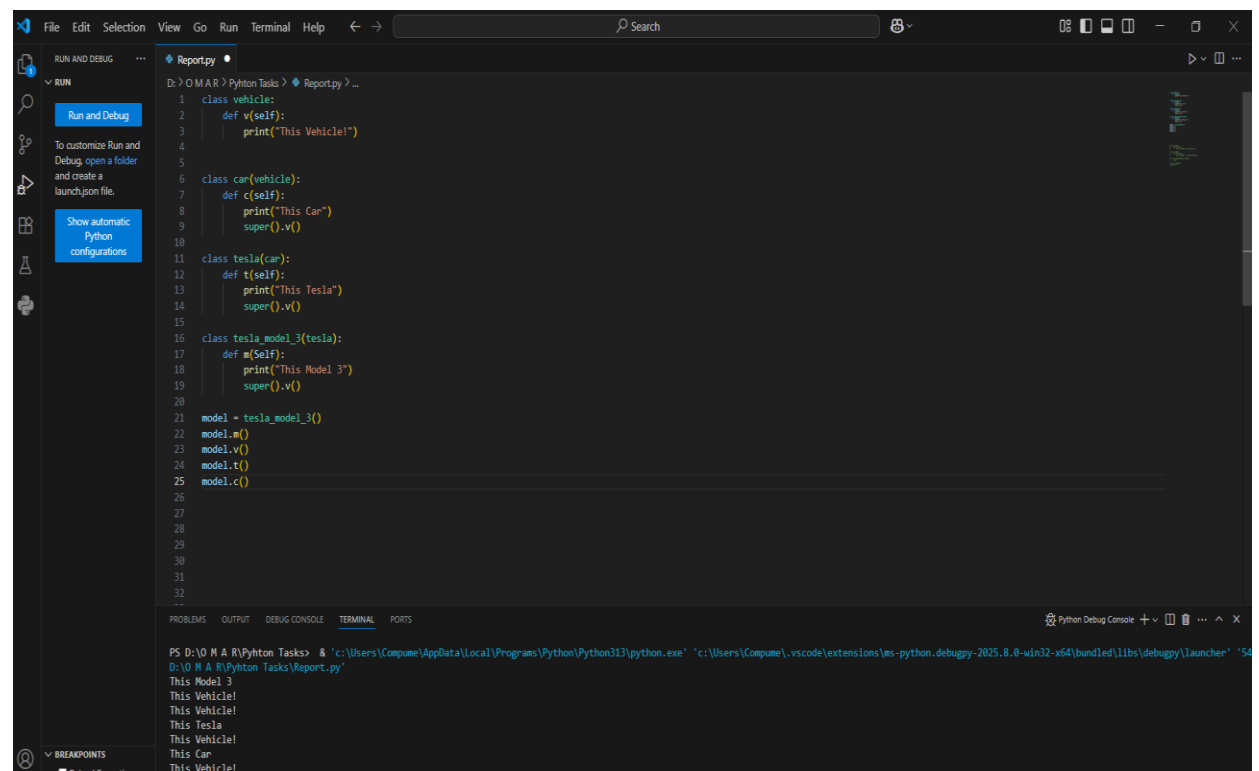
1- How super Function handle Multiple Inheritance ?

When a class inherits from multiple classes, the super() function does not just refer to the immediate parent.

Instead, it follows the Method Resolution Order (MRO) – a predefined order that Python uses to search for methods.

So when super () is called inside a method, Python checks the next class in the MRO chain.

Example:



```
File Edit Selection View Go Run Terminal Help
D:\> O M A R > Python Tasks > Report.py > ...

1 class vehicle:
2     def v(self):
3         print("This Vehicle!")
4
5
6 class car(vehicle):
7     def c(self):
8         print("This Car")
9         super().v()
10
11 class tesla(car):
12     def t(self):
13         print("This Tesla")
14         super().v()
15
16 class tesla_model_3(tesla):
17     def m(self):
18         print("This Model 3")
19         super().v()
20
21 model = tesla_model_3()
22 model.m()
23 model.v()
24 model.t()
25 model.c()
26
27
28
29
30
31
32
```

Python Debug Console

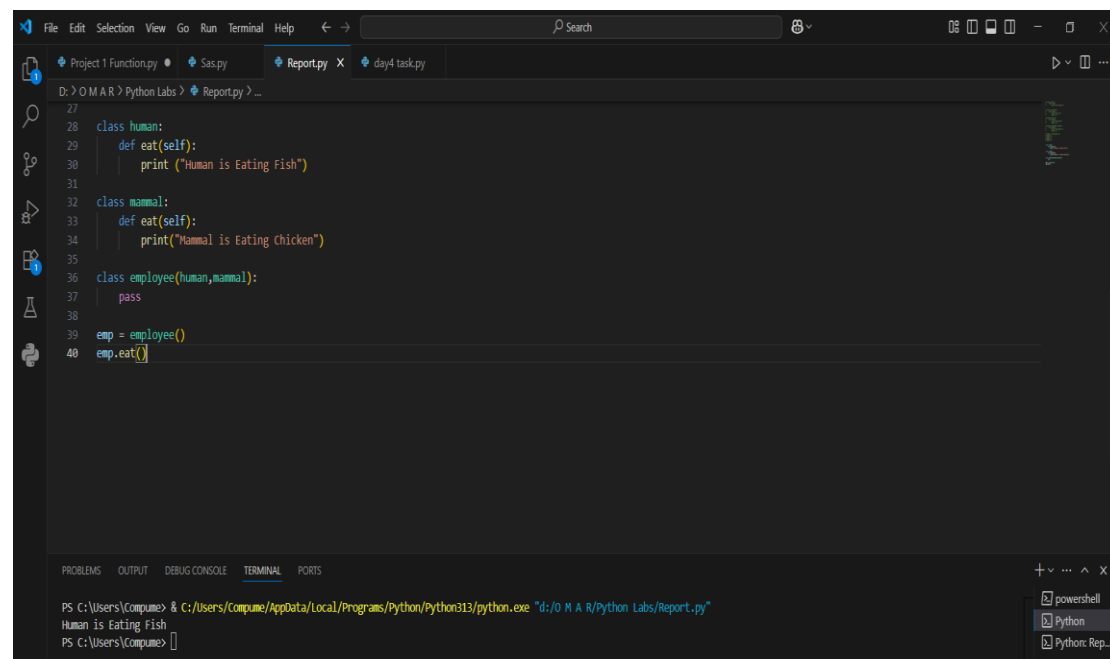
```
PS D:\O M A R\Python Tasks> & 'c:\Users\Compuer\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\Compuer\vscode\extensions\ms-python.debugpy-2025.8.0-win32-x64\bundle\libs\debugpy\launcher' '54'
D:\O M A R\Python Tasks> Report.py
This Model 3
This Vehicle!
This Vehicle!
This Tesla
This Vehicle!
This Car
This Vehicle!
```

2- . If Human and Mammal Have the same method like eat but with different Implementation. When Child[Employee] calls eat method how python handle this case ?

When two parent classes (Human , Mammal) define a method with the same name (eat)), and a child class (Employee) inherits from both, Python will call the version of eat() based on the Method Resolution Order (MRO).

In the example above, Employee (Human , Mammal) means human comes first in the MRO, so its eat() method is the one that gets executed

Example:



The screenshot shows a Python IDE with a file named `Report.py`. The code defines three classes: `human`, `mammal`, and `employee`. The `human` class has an `eat` method that prints "Human is Eating Fish". The `mammal` class has an `eat` method that prints "Mammal is Eating Chicken". The `employee` class inherits from both `human` and `mammal` and does not have its own `eat` method. An instance of `employee` is created and the `eat` method is called. The output in the terminal shows "Human is Eating Fish", indicating that Python's Method Resolution Order (MRO) prioritizes the `human` class's `eat` method over the `mammal` class's `eat` method.

```
27
28 class human:
29     def eat(self):
30         print ("Human is Eating Fish")
31
32 class mammal:
33     def eat(self):
34         print("Mammal is Eating Chicken")
35
36 class employee(human,mammal):
37     pass
38
39 emp = employee()
40 emp.eat()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\Compu> & C:\Users\Compu\AppData\Local\Programs\Python\Python313\python.exe "d:/O M A R/Python Labs/Report.py"

Human is Eating Fish

PS C:\Users\Compu>