DAY-4

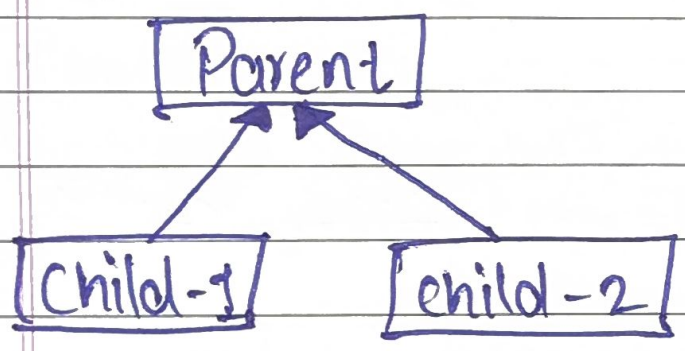
Types of Inheritance

- Simple
- Heirarichal
- Multilevel
- Multiple
- Hybrid

1> Simple :

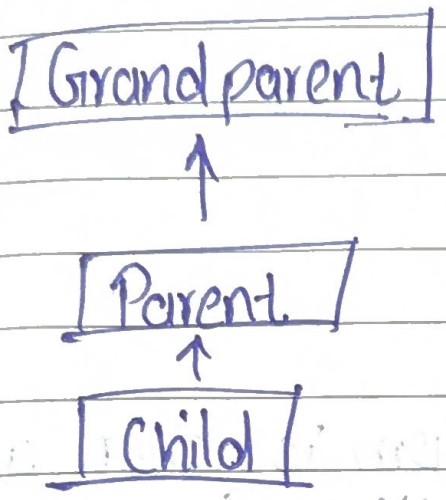
- What we have studied till now.

2> Heirarichal :



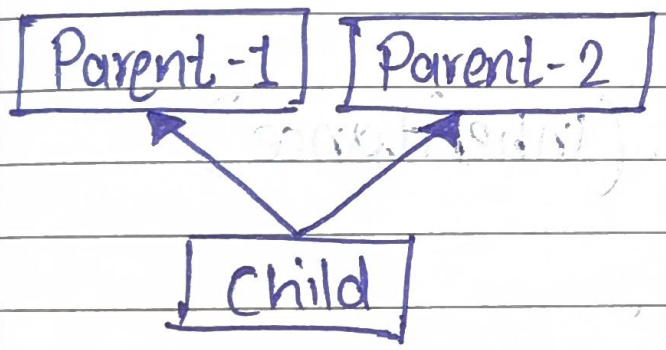
- Both children get properties of parent.
- Children are not related to eachother.

3) Multilevel :



- lower levels get properties from upove levels.
- Super() of each class can reach out to just the parent.

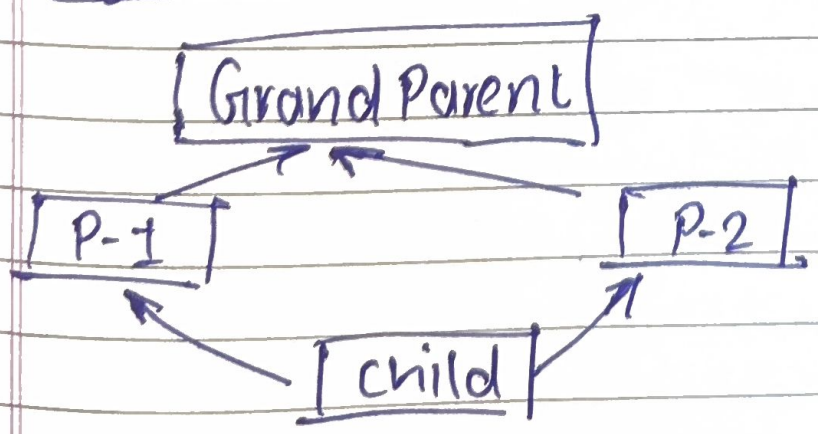
4) Multiple :



- not allowed in Java
- Ambiguity solved in the order of inheritance, get properties of both parent.
- MRO (Method Resolution Order) followed in resolving ambiguity.

- class c (P_1, P_2)
- Priority: $P_1 > P_2$

5) Hybrid :



- Mix of inheritance type.

Diamond Problem

Polymorphism :

- Poly → multiple / many
- Morph → shape / form
- In simple words, polymorphism is when a single entity can take multiple forms.
- In python, we achieve polymorphism by below:

1. Method overriding. (Inheritance)
2. Method Overloading.
3. Operator Overloading.

- ~~Major benefit is that code becomes~~

1> Method Overloading:

- When in a same class we have two different methods having same name but different behaviour depending on the input.

For e.g.

```
class Shap:
    def area(self, r):
        return 3.14*r*r
    def area(self, l, b):
        return l*b
```

This format is not supported in python.

- Code becomes more clean & readable.

To implement Method Overloading:

```
class Shape:
    def area(self, a, b=0):
        if b==0:
            return 3.14*a*a
        else:
            return a*b
```

Operator Overloading:

The behaviour of the same operant is different according to input.

For eg:

1) 'Hello' + 'World'
   ↳ 'HelloWorld'

2) 4 + 5
   ↳ 9

3) [1,2,3] + [4,5,6]
   ↳ [1,2,3,4,5,6]

Abstraction:

- Abstraction → hidden
- Nothing to write more, refer notebook.
- decorator: @abstractmethod.
- Abstract class:
  - Having atleast one abstract method.
- We cannot make object of the abstract class.