URBAN EDGE

Chapter 11 - Inheritance & more on OOPs

Inheritance is a way of creating a new class from an existing class

Syntax:

```
class Employee:                    → Base Class
    # Code
    ...

class Programmer (Employee):       → Derived or child class
    # Code
```

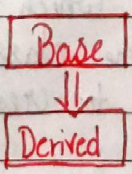
We can use the methods and attributes of Employee in Programmer object.
Also, we can overwrite or add new attributes and methods in Programmer class.

Types of Inheritance

1. Single inheritance
2. Multiple inheritance
3. Multilevel inheritance

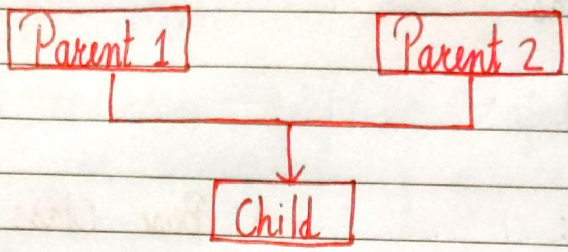
Single Inheritance

Single inheritance occurs when child class inherits only a single parent class



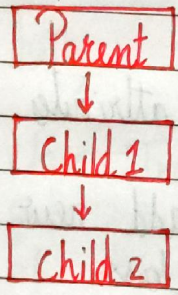
Multiple Inheritance

Multiple inheritance occurs when the child class inherits from more than one parent class.



Multilevel Inheritance

When a child class becomes a parent for another child class



Super() method

Super method is used to access the methods of a super class in the derived class

```
super().__init__()
```

↳ calls constructor of the base class

Class methods

A class method is a method which is bound to the class and not the object of the class.

@classmethod decorator is used to create a class method

Syntax to create a class method:

```
@classmethod
def (cls, p1, p2):
    ...
```

@property decorators
Consider the following class

```
class Employee:
    @property
    def name(self):
        return self.ename
```

if e = Employee() is an object of class employee, we can print(e.name) to print the ename/call name() function.

@.getters and @.setters
The method name with @property decorator is called getter method
We can define a function + @name.setter decorator like below:

```
@name.setter
def name(self, value):
    self.ename = value
```

Operator overloading in Python
Operators in python can be overloaded using dunder methods.
These methods are called when a given operator is used on the objects.

Operators in python can be overloaded using the following methods:

$p_1 + p_2 \rightarrow p_1.\_\_add\_\_(p_2)$

$p_1 - p_2 \rightarrow p_1.\_\_sub\_\_(p_2)$

$p_1 * p_2 \rightarrow p_1.\_\_mul\_\_(p_2)$

$p_1 / p_2 \rightarrow p_1.\_\_truediv\_\_(p_2)$

$p_1 // p_2 \rightarrow p_1.\_\_floordiv\_\_(p_2)$

Other dunder/magic methods in python

`__str__()` → used to set what gets displayed upon calling `str(obj)`

`__len__()` → used to set what gets displayed upon calling `__len__()` or `len(obj)`