CHP 07 OOPs

1- Python classes and objects

(i) - Class :- A class in Python is a blueprint for creating objects. It define the behavior of the object that will be created from it.

How to create a class in Python

```
Class classname
    class body
```

(ii) - Object :- Object is an instance of a class. You cannot access class properties without an object

How to create an object

```
my_class = classname()
```

This code will create an object named my_class.

2- Constructor in Python

i)- Constructor is a special member function which calls whenever object created of that class.

ii)- Constructor is identified by (__) underscore.

iii)- **__init__ function**:- This function is called constructor.

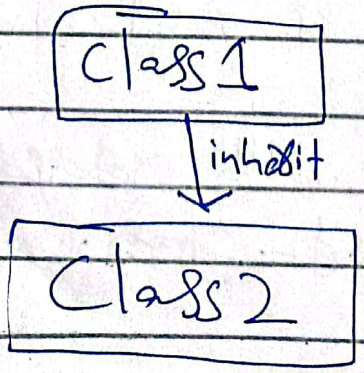
How to define constructor

```
def __init__(self):
    Constructor body
```

3- Inheritance in Python

i)- Python supports inheritance, it even support multiple inheritance

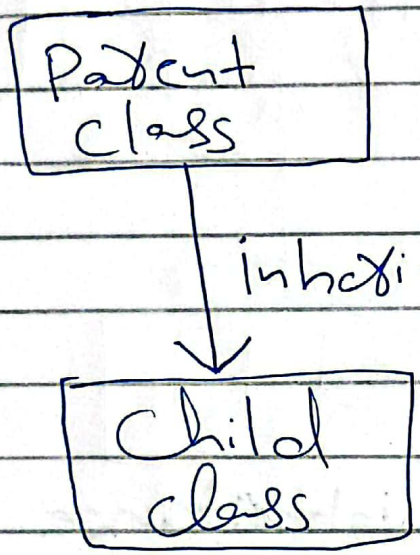
ii)- Class can inherit from one class to another. For example:



Types of inheritance

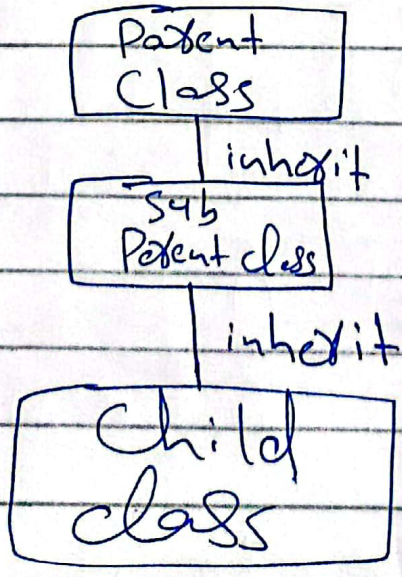
i)- Single inheritance

i)- In single inheritance there is only one parent class and one child class. For example:



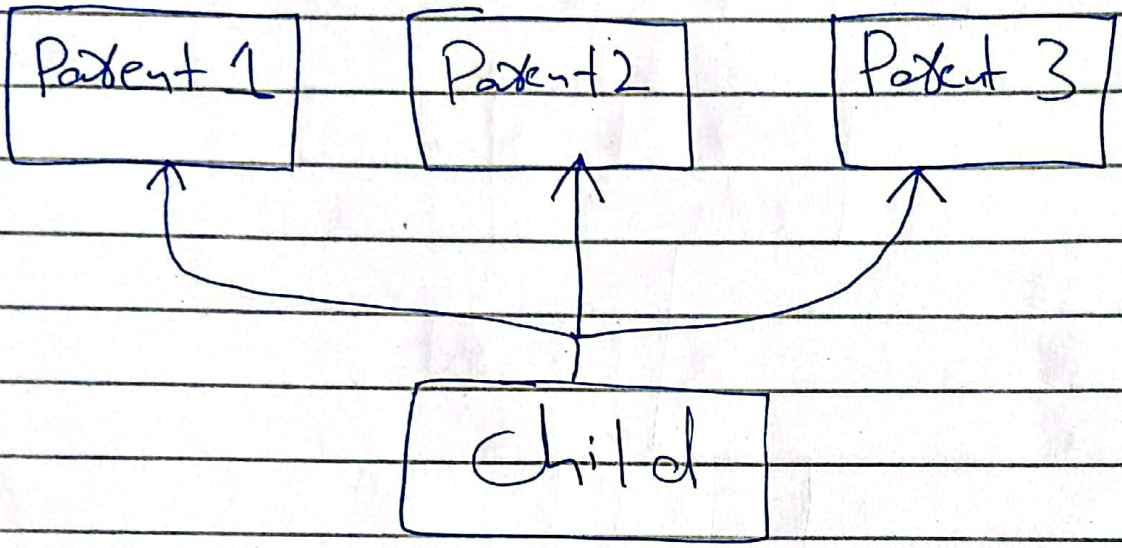
ii)- Multilevel inheritance

i)- In multi-level there are level of class



iii)- Multiple inheritance

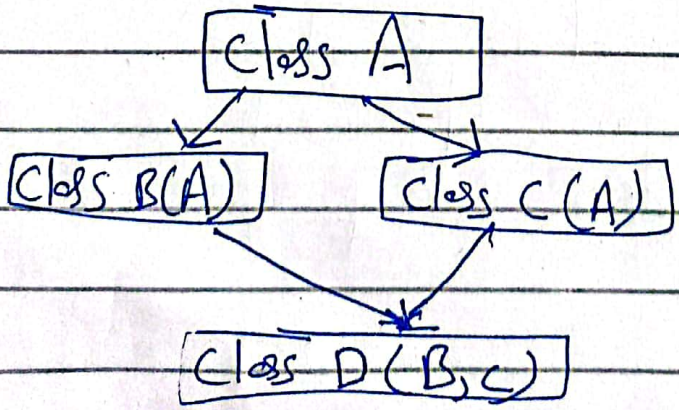
Inheritance which contains more parent classes and only only child class is called multiple inheritance.



iv)- Hybrid inheritance

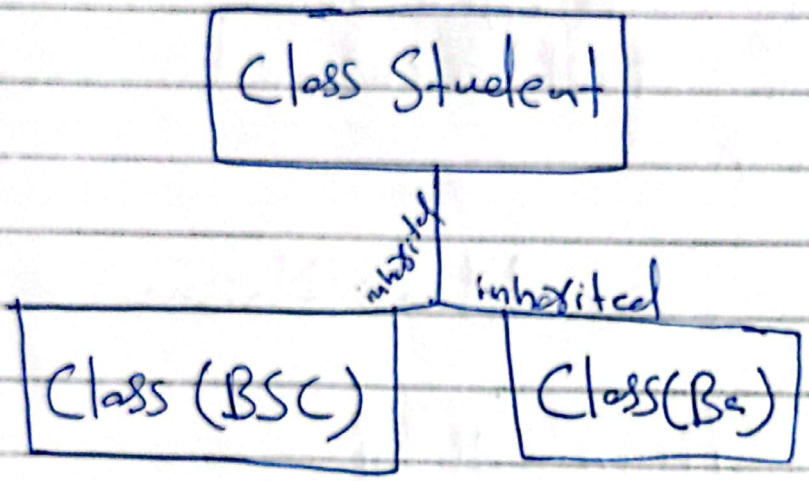
i) Hybrid inheritance is nothing but a collection of more then one inheritance

ii)- This can be useful for creating classes that have a complex set of features

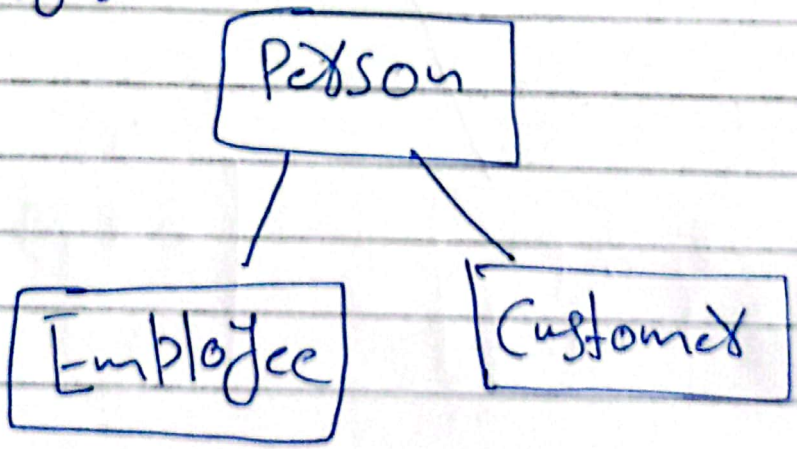


v)- Hierarchical inheritance

Hierarchical inheritance is a type of inheritance where multiple subclasses inherit from a single parent class.



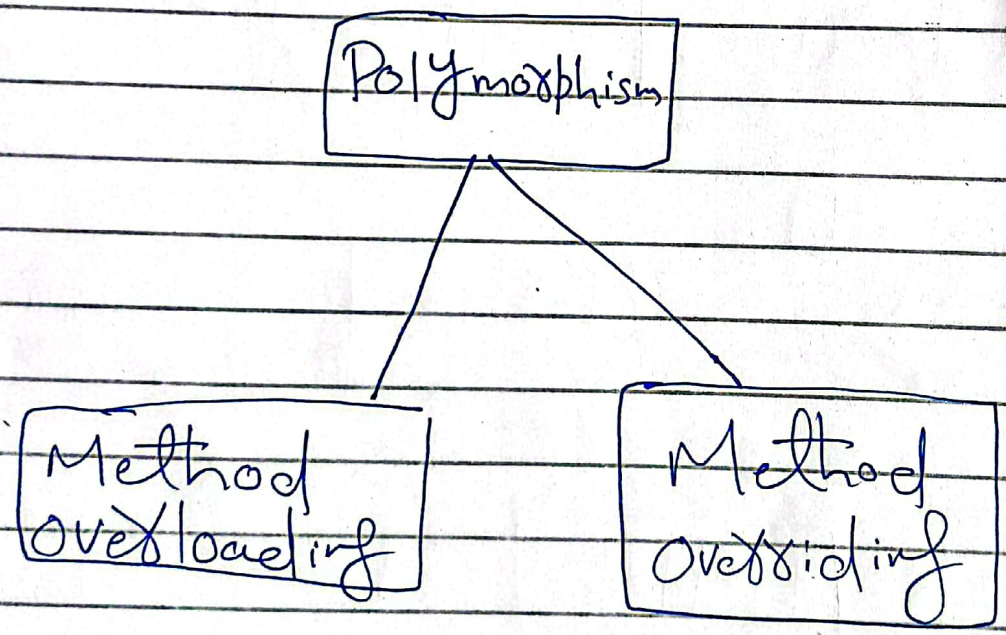
Similarity:



4- Polymorphism in Python

Polymorphism is the ability of a method to behave differently depending on the type of object it is called on.
In Python, Polymorphism is achieved through inheritance and method overriding.

Types of Polymorphism



i)- Method overloading

Python doesnot support method overloading because it is an interpreted language with dynamic typing

ii)- Method overriding

i)- Method overriding is the example of run-time polymorphism.
ii)- Method overriding allow a subclass to provide a new implementation for a method that is already defined in its parent class.

Rules of method overriding

* The method name, return type and parameters must be same
* Inheritance is mandatory

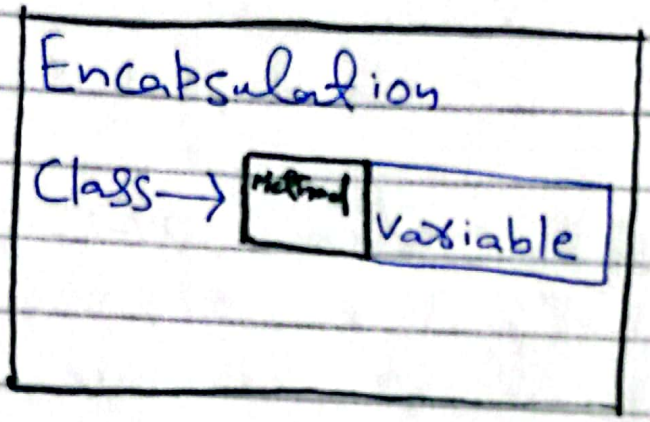
5- Abstraction in Python

i)- Abstraction is a process of hiding the implementation details of an object or a system from the user.
ii)- In Python, Abstraction can be achieved using abstract classes and methods. For example:

```
Class AbstractClass:
    def abstractmethod(self):
        Pass
```

16- Encapsulation in Python

The process of wrapping up variables and methods into a single entity is known as encapsulation. For exp



How to achieve Encapsulation

i)- _ (Single underscore) == Protected

ii)- __ (double underscore) == Private

7- Super keyword

Super keyword refer to the object of super class. Whenever the super class and sub class variable and method name both are

Same then we used super keyword