

OOPs- PYTHON

Dr Sivabalan,
Technical Training Advisor
Sivabalan.n@nttdata.com

Agenda

1. Basics of OOP
2. Types of variables & methods
3. Inheritance
4. Polymorphism
5. Encapsulation
6. Abstraction
7. Interface

Requirements

1. Python basic knowledge
2. Functional programming

What is class?

- ✓ Class is a template/blueprint/prototype for creating objects.
- ✓ Every object belong to some class
- ✓ Email class:- email1 + email2 + email3 +email4

What is class?

attributes:-

heading
participants
attachments

methods:-

send()
save_as_draft()



Email1:-

heading:- taking leave
participant:- xyz
attachments:- form.pdf



Email2:-

heading:- require help
participant:- abc
attachments:- pic.jpg

What is class?

- ✓ Class is a collection of attributes and methods.
- ✓ Class is a collection of objects.
- ✓ Technically, class is a user-defined datatype.

What is constructor?

- ✓ Special method used for initializing objects with attributes
- ✓ It is `__init__()` method
- ✓ First arguments is 'self'.

Types of constructor?

- ✓ Parameterized constructor
- ✓ Non-Parameterized constructor
- ✓ Default Constructor

Accessing Class Members

How to access class members?

- ✓ Class members :- Attributes(variables) + Actions(Methods)
- ✓ We can access these variables using object outside the class.
- ✓ Syntax:-
 - Accessing attribute:- `object_name.variable_name`
 - Accessing method:- `object_name.method_name()`

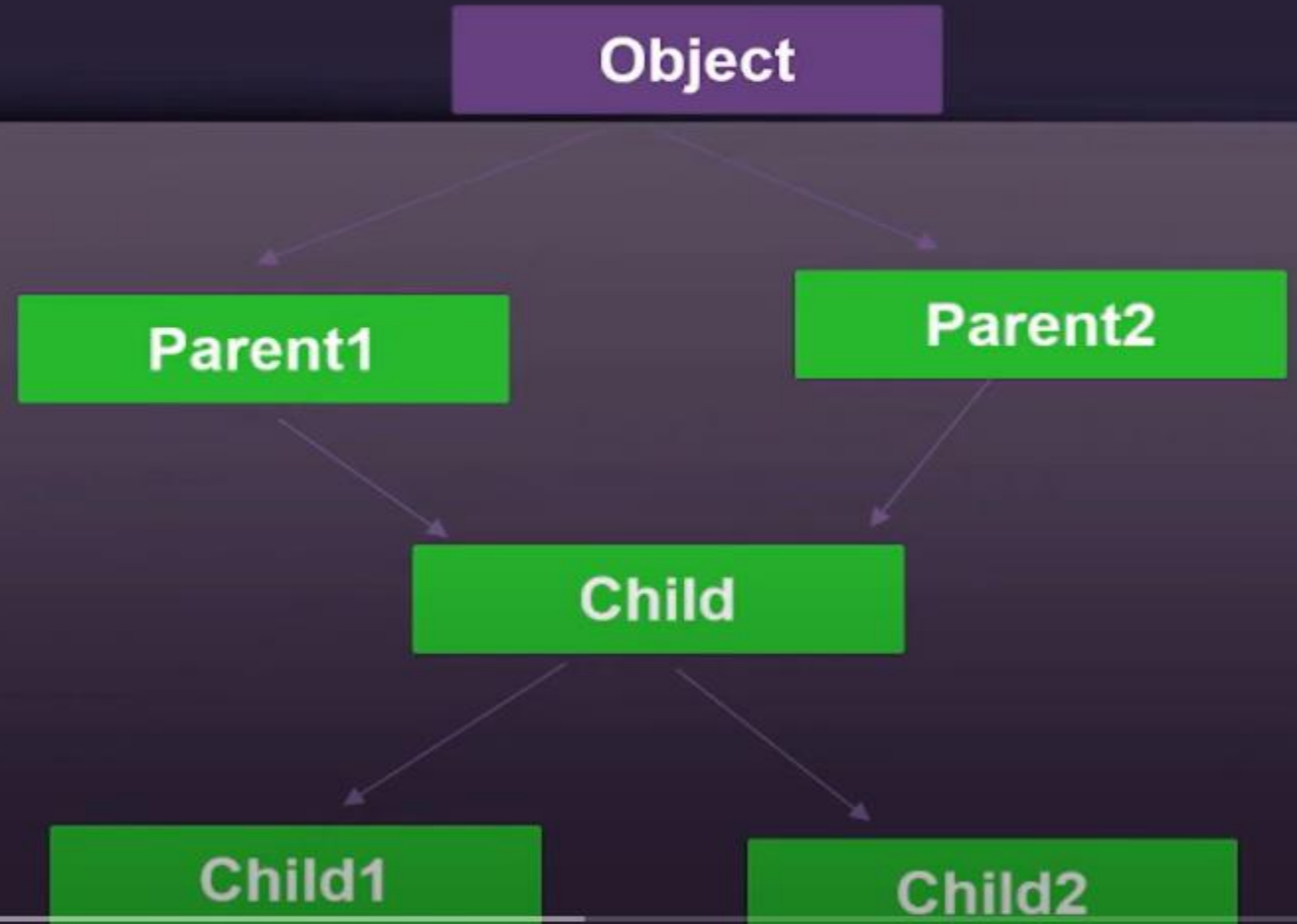
Built-in Class Functions

Following are built-in class functions:-

- ✓ `getattr(object_name, attribute_name)`
- ✓ `setattr(object_name, attribute_name, new_value)`
- ✓ `delattr(object_name, attribute_name)`
- ✓ `hasattr(object_name, attribute_name)`

Hybrid Inheritance:-

- ✓ It contains multiple type of inheritance.



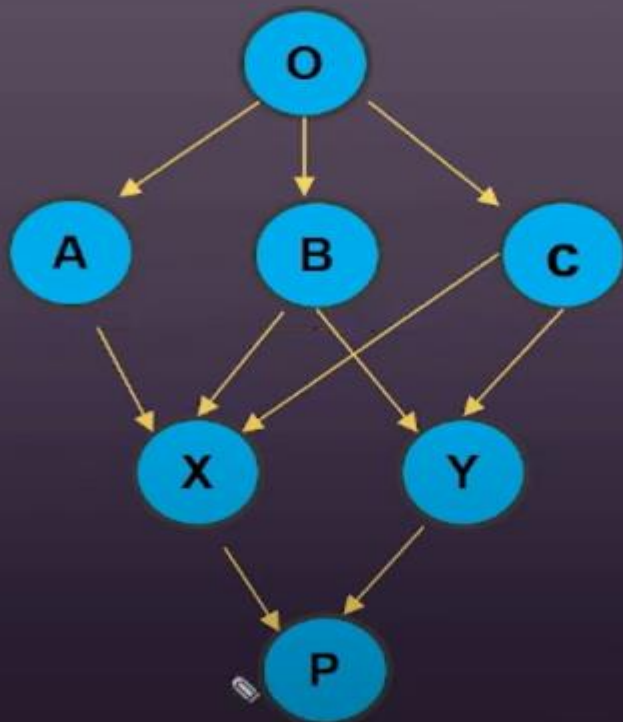
What is MRO?

- ✓ MRO represents how properties (attributes+methods) are searched in inheritance.

Rule -01

- ✓ Python First search in child class and then goes to parent class.
- ✓ Priority is to child class

Rule -02 MRO Follows 'Depth First Left to Right approach'



- ✓ `mro(o):- Object`
- ✓ `mro(A):- A,O`
- ✓ `mro(B):- B,O`
- ✓ `mro(C):- C,O`
- ✓ `mro(X):- X,A,B,C,O`
- ✓ `mro(Y):- Y,B,C,O`

Encapsulation

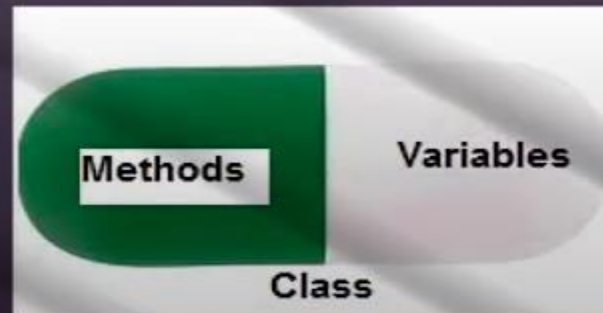
Topics:-

- ✓ What is **Encapsulation** in python?
- ✓ Need of **Encapsulation** in Python
- ✓ **Access Modifiers** in python
- ✓ **Name mangling** concept
- ✓ Making private method

What is Encapsulation in python?



- ✓ Wrapping up **data and methods working on data together** in a single unit (i.e class) is called as encapsulation.



Access Modifiers in Python :-

- ✓ Generally, we restrict data access outside the class in encapsulation.
- ✓ Encapsulation can be achieved by declaring the data members and methods of a class as private.
- ✓ Three access specifiers:- public, private, **protected**

Access Modifiers in Python :-

- ✓ Public member:- Accessible anywhere by using object reference.
- ✓ Private member:- Accessible within the class. Accessible via methods only.
- ✓ **Protected member:-** Accessible within class and it's subclasses

Polymorphism

Topics:-

- ✓ What is Polymorphism in python?
- ✓ Examples of polymorphism
- ✓ Polymorphism in built-in functions

Real life analogy

You

In front of parent



study, career, exams etc

In front of friends



movies, Netflix, series ,gf-bf etc

What is Polymorphism in python?

- ✓ Polymorphism in python is an ability of python object to take many forms.
- ✓ If a variable, object, method performs different behaviour according to situation is called as polymorphism.

Polymorphism with inheritance

Polymorphism in functions and objects

Destructor in python

In OOP, we have two terms

➤ Constructor

➤ Destructor



Reverse of each other

What is Destructor?

- A special method which destroys objects and releases resources tied to objects.
- Destructor is called automatically when object is destroyed.

What is purpose of Destructor ?

- Releasing objects tied to destroyed objects

X = 100

Y = 200

database connection

cache created

file handling done



X

Y

DB object
Cache vars
Handler
etc

Below are two conditions when destructor is called :-

- Reference counting reaches to 0.
- When variable goes out of scope

Note:- In Python, The special method `__del__()` is used to define a destructor.

Object Creation & Deletion in python:-

```
emp = Employee('Jay',50000)
```



Arguments passed here for
constructor to initialize variables

Object is created using
__new__() method

Object is initialized using
__init__() method

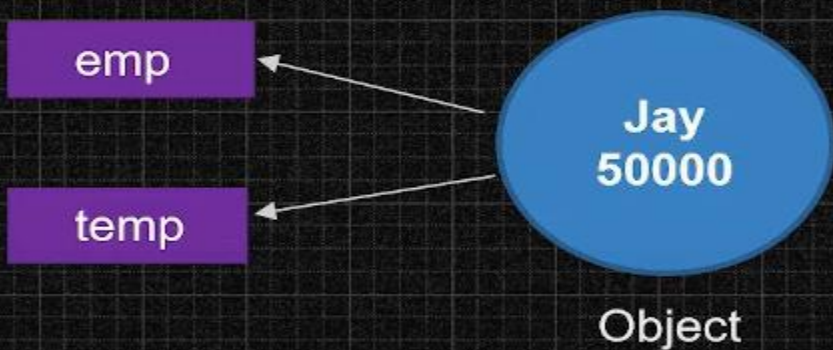
```
del emp
```

Destructor invoked using
__del__() method

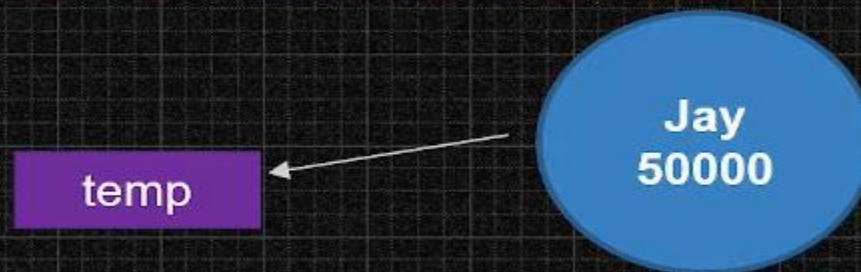
Working of Destructor in python:-

1. Create Object

```
emp = Employee('Jay',50000)  
temp=emp
```



2. delete emp :- del emp



`__del__()` method will not be invoked as we still have one reference

3. delete temp :- del temp

Object is ready for garbage collection, call the destructor