3/04/2020
FRIDAY

OOPs

OBJECT ORIENTED PROGRAM.

→ Like other programming languages python support OOPL.

→ Pyth OOP helps to develop applications using OO Approach.

→ In python we can easily create & use classes & objects.

Major principle of ob-or-program

$$\begin{tikzcd} \text{object} \arrow[r] & \text{attributes} \\ & \text{behavior} \end{tikzcd}$$

→ class

→ Method

→ Inheritance

→ Polymorphism

→ Data abstraction

→ Encapsulation

→ The concept of python focuses on creating reusab Code. This concept is also known as DRY (Don't repeat Yourself)

Object

The object is an entity that has state & behaviour
It may be any real-world object like mouse,
Keyboard, chair, table, pen etc.

→ Everything in python is object & almost everything
has attributes & methods.
→ All the functions have a built-in attribute
__doc__, which returns the doc string defined
in the function source code.
→ object has two characters
1) Attributes
2) Behaviour

Eg:- Parrot is an object
1) name age colour are attributes
2) singing dancing are behavior.

Class
↳ Class is a collection of object, its a logical
entity that has some specific attributes & methods.
Eg:- If you have an employee class then it should
Contain an attribute & Method i.e email id, name,
age etc.

Syntax

```
class <class name>
    <statement 1>
    .
    .
    .
    <statement N>
```

> A class is a blue print for the object

↳ We can think of Class as a sketch of a parrot with labels. It Contains all the details aboult the name, colour size etc.

↳ So, Based on these descriptions we can study about the parrot, So, here Parorot is an object

Eg:

```
class Parrot :
    pass
    <statements>
```

↳ Here we use Class keyword to define an empty Class. Parrot:

↳ From class we Create (Construct Instances).

A n instance is a specific object created from a particular class.

↳ An object is also called as Instance
So, An instance is an instantiation of a class
↳ when a class is defined only the description for the object is defined. Therefore, no memory or storage is allocated.

Eg:-
```
obj = Parrot ( )
```
here obj is object of class Parrot

Suppose we have details of parrot we can build the class & objects of parrot

```
class Parrot:
        speciecies = "bird"

        def __init__(self, name, age):
            self.name = name
            self.age = age
- - - - - - - - - - - - - - -
    # Instantiate the Parrot class
                                                       → Name
                                                         Parameter
    blu = Parrot ("Blu", 10)          → Age parameter
    woo = Parrot ("Woo", 15)
     ↳ object Name  → Class Name
```

~~print~~("

In Parrot there are ~~at~~ class attributes & Instance attri

So, we can Access the independently.

print("blue is a {}". formate(blue . __class__ . species))

print ("woo is a {}". formate(woo . __class__ . species))

---

print {"{} is {} years old". formate(blue.name, blue.age)

pt {"{} is {} years old". formate (woo.name, woo.age)

↳ Here parrot is class & we created instances of Parrot. Here blue & woo are references (value) to our new objects.

↳ Then, we access the class attribute using __class__ species. class attributes.

__class__ . species. Class attribute

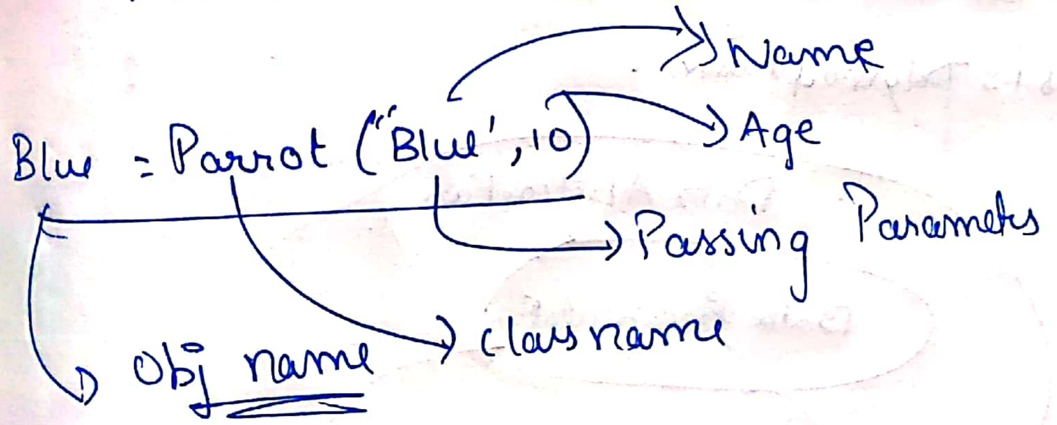
// In the similar way we can also create Methods to the class

Methods

→ Methods are functions defined inside the body of class. They are used to define the behaviour of an object.

Methods are functions defined inside the body of a class. They are used to define the behaviour of an object.

```
class Parrot:

def __init__(self, name, age):

    self.name = name

    self .age = age

def sing (self, song):
    Print (" Sing a song "){

def dance (self):
    Return " { } is now dancing ". format(self.name)
```



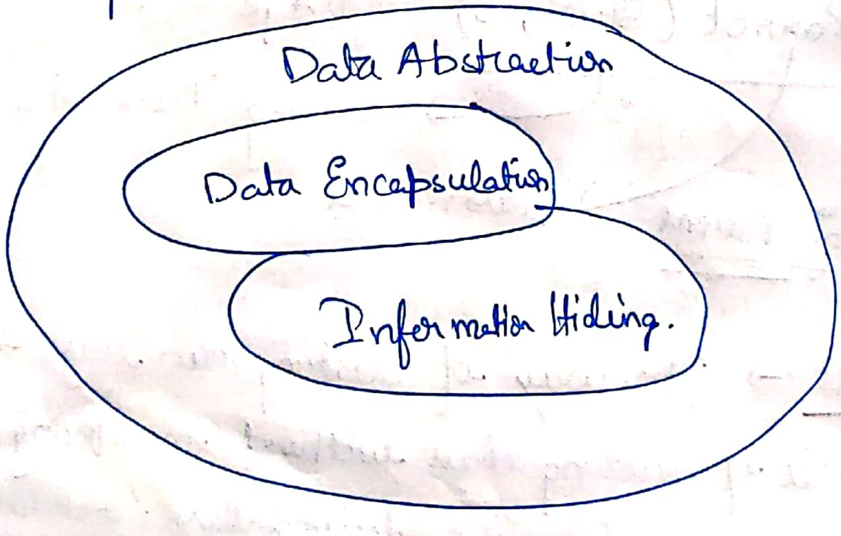
Inheritance → The way of creating new class for using details of existing class without modifying it. The newly formed class is a derived class (or child class). Similarly, the existing class is a base class (a parent class)

ENCAPSULATION

Using OOP in Python we Can restrict access to Methods & Variables. This prevent data from direct modification which is Called encapsulation.
In python we denote private attribute using, underscores prefix i.e single "_" or double "__".

POLYMORPHISM

→ Ability to use Common interface for Multiple form (data types).
→ Suppose, we need to colour a shape, there are Multiple shape option (rectangle, square, circle) we Can use same method to colour any shape. This is Polymorphism.



Scanned with CamScanner

ABSTRACTION

> The Process of hiding the real implementation of an Application from the user and emphasizing Only on usage of it