OOP's

→ Object Oriented Programming System / Structure

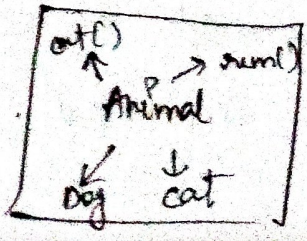
→ OOP is a programming paradigm / methodology

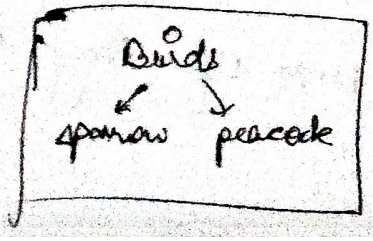
- OOP's paradigm
- Procedural paradigm
- functional paradigm
- logical paradigm
- structured paradigm

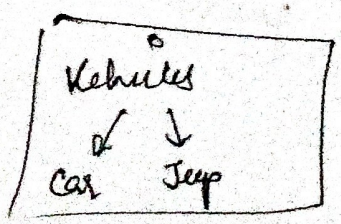
→ 6 main pillars of OOP's are -

- → class
- → objects & method
- → inheritance
- → polymorphism
- → Abstraction
- → Encapsulation

Paradigm can also be termed as method to solve some problems or do some task. Programming paradigm is an approach to solve problems using some programming language or also we can say that a method







OOP's:

class

1. class is the collection of object
2. class is not a real world entity. It is just a template or blueprint or prototype.
3. class does not occupy memory.

⇒ syntax: access modifier className

body { - methods
- constructors
- fields
- blocks
- nested class }

Method:

1. A set of codes which perform a particular task.

Advantages of methods:

1) code reuseability
2) code optimization

syntax:

accessmodifier returntype methodName (list of parameters)

{

}

object:

1. object is an instance of class
2. object is real world entity
3. object occupies memory.

object consist of

1) identity name
2) state / Attribute
3) Behaviours.

How to create an object:

- new keyword
- new instance() method
- clone() method
- deserialization
- factory methods

OOP's

1) Declaration : Animal bugo;
2) Instantiation : bugo = new Animal();
3) Initialization

Syntax: `Animal bugo = new Animal();`

Note : . dot operator is used to call method

→ `bugo.eat();`

```
class Animal
{
    public void eat()
    {
        System.out.println("I am eating");
    }
    public static void main(String[] args)
    {
        System.out.println("1");
    }
}
```

Animal bugo = new Animal();
bugo.eat();

Note : If a task can be done by single object then new object should not be created as it will occupy extra memory.

OOP's: 1. by reference variable

```
Class Animal
{
    String color;
    int age;
    public static void main(String[] args)
    {
        Animal bugo = new Animal();
        bugo.color = "black";
        bugo.age = 10;
```

→ Sop(bugo.color + "\n" + bugo.age);

2) By using method.

```
class Animal
  {
   String color;
   int age;
   void initObj(String c, int a)
      {
        color = c;
        age = a;
       }
       void display()
         {
          System.out.println(color + " " + age);
         }
       psvm()
         {
          Animal bugo = new Animal();
             bugo.initObj("black", 10);
             bugo.display();

         }
       }
```