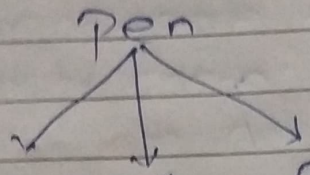
OBJECTS ORIENTED PROGRAMMING

- Class — Group of these entities
- Object — Entities in the real world

Blue Print of an object

ex -



Pen → Object class

blue, Yellow, Red → Attributes / Property

does occupy memory

changeColor → function

Data member
Method
Constructor
Nested class
Interface

* Access Modifiers / specifier
  help to restrict the scope of class constructor, variable, method or data member
  It provide security, accessibility.

	Within class	Within package	outside package	" by subclass
1) private	y	N	N	N
2) Default	y	y	N	N
3 protected	y	y	N	y
4) public	y	y	y	y

Get - to return the value
Set - to modify the value
this - Refer to the current oblect

```
class pen
{
    int tip;
    void setTip (int tip)
    {
        this.tip = tip;
    }
}
```

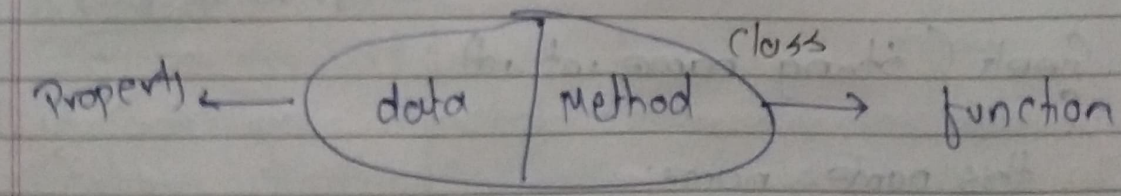
Refer class variable

temp. variable

1) Encapsulation
2) Inheritance
3) Abstraction
4) Polymorphism

Encapsulation-

Wrapping up of data & methods under a single unit. It also implements data hiding.



It is the way of hiding the implementation detail of a class from outside access & only exposing a public interface that can be used to interact within a class

Page No.
Date

Constructor-

is a special method which is invoked automatically at the time of oblect creation

1) Constructor have same name as class
2) Constructor doesn't have a Return type
3) Constructor are only called once, at oblect creation
4) Memory allocation happen when constructor called
5) Used to initalize oblect.-type of Method

Constructor are different from Java Method.

- Constructor must have the same name as class,
- it doesn't return any return type, while method have return type, or void if doey n't return any value
- Constructor are called only once at the time of oblect creation, Method called no of time.

Type - 1) Default (No Parameter)
2) Parameterized
3) Copy (passing another oblect arg, value etc).

```
Greek ( String name, int id)
{
  this.name= name;
  this.id  = id;
}
PSVM.
Greek gk = new Greek("Ash",18);
```

```
Greek gk2 = new Greek
(gk);
```

Destructors -
The Garbage collector automatic delete the unsend oblect to free up the memory.
It destroy oblect, for free heap memory by destroying unreachable oblect.

Type - 1) Minor or incremental Garbage "
2) Malor or full Garbage "

Ways - 1) By nulling the refrence
Employee e = new Employee();
e = null.

2) By assigning a refrence to another
Employee e1 = new Employee();
Employee e2 = new Employee();
e1 = e2;
Now, the first oblect referred by 1, is available for garbage collection.

3) By anonymous oblect - (ગુમનામ)
new employee()

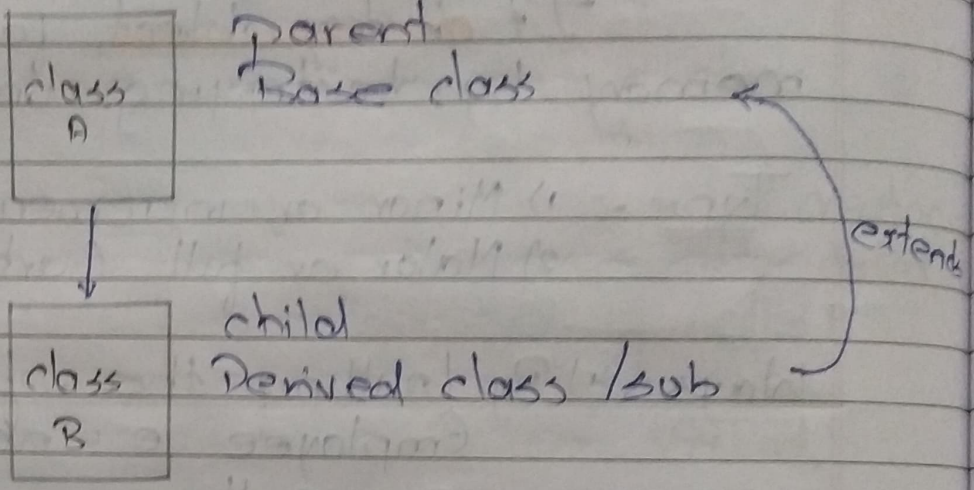
① finalize() -
invoked each time before the oblect is garbage collected, this method is used to perform cleanup process.
Protected void finalize() { }

② G( ) - found in System & Runtime class
used for cleanup
public static void g() { }

Page No.
Date

Inheritance

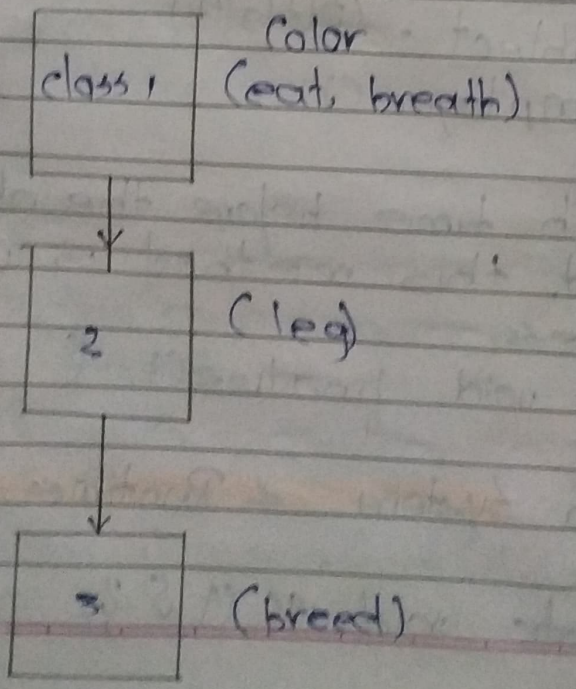
is when properties and method of base class are passed on to a derived class.



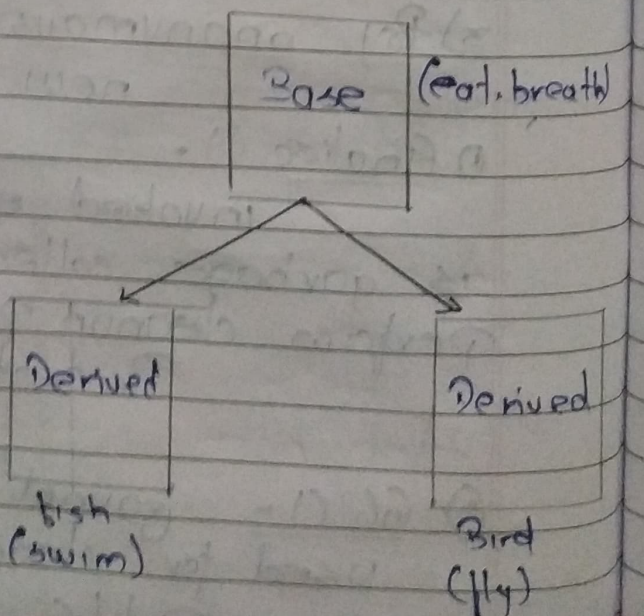
Types-

1) Single level
2) Multi level
3) Hierarchial
4) hybrd
5) Multiple

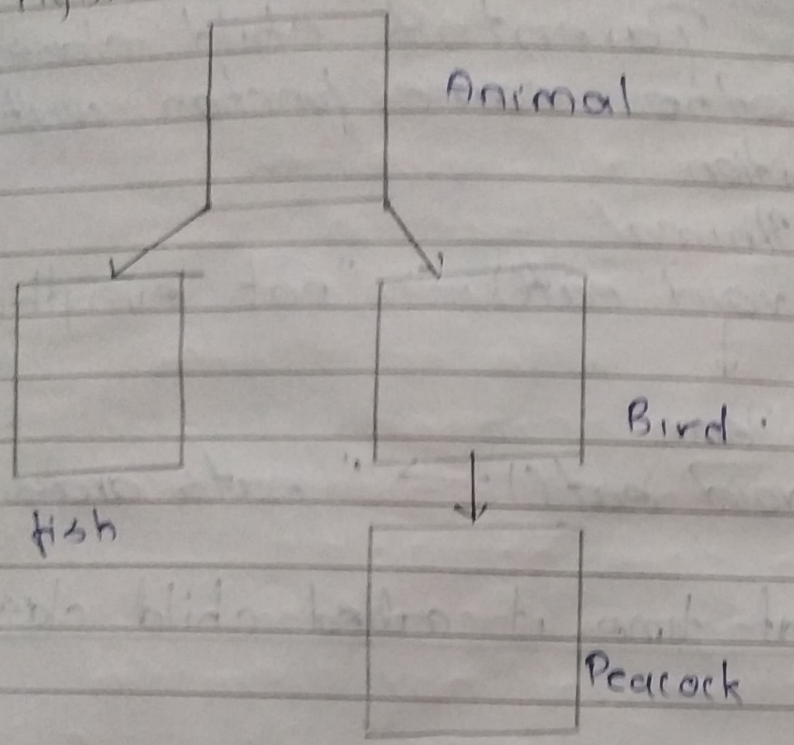
2) Multi -



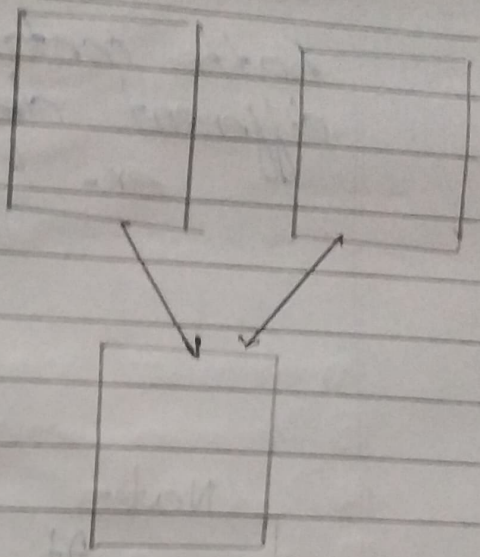
3) Hierarchial



4) hybrid.



5) Multiple



In Java it can form using interface

Polymorphism

poly → Many
morph → form

Type - 1) Compile (Static) Method Overloading
2) Run (dynamic) Method Overriding

Method overloading.

Multiple function with the same name but different parameter (Type, Count)

ex- class Calculator {

Count ← Type ← sum (int a, int b) | 1+2
sum (int a, int b, int c) | 1+2+3
sum (float a, float b) | 1.5+2.5

}

Method Overriding-

Parent & child classes both contain the same function with a different definition

ex- Normal
void eat() → "eat everything"

↓

Dear
void eat() → "eats grass"

Note-
At that time it called child class

Package in Java.

is a group of similar type of classes, interface and sub-package

Type

User-Define package
ex- MyPackage

In-build package
ex- java.lang
java.io
java.util

* Every class is a part of some package
* If package name is specified, the file must be in a subdirectory called name
* It consider as data encapsulation (data-hiding)
* Providing control access

Abstraction-

Hiding all the unnecessary detail & showing only the important part to the user.

Abstract → idea (ଅବଧାରଣା)

It achieve by 1) Interface 2) Abstract class

Abstract class-

1) Declare with abstract keyword.
2) Can't create an instance (object) of abstract class
3) Can have abstraction / non abstract method.
4) Can have constructore (parameterized, default)
5) Implementation not possible i.e { }

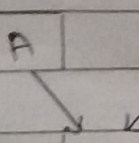
Parent Constor call

i.e

class —— X —— object
Method —— X —— parameter { }

Interface

① blueprint of a class
② Used for Multiple inheritance (it is not done by class, so it done by interface)
③ Total abstraction.
④ Declare with Interface keyword.
⑤ Class → extends ; Interface → implement
⑥ All method are public, abstract & without implementation.
⑦ Used to achieve total abstraction
⑧ Variable in the interface are final, public & static



Page No.
Date

Static keyword-

is used to share the same variable or method of a given class.
it can be 1) Properties
2) function
3) Blocks
4) Nested classes

① Share memory allocation
② Access without object instance
③ Associate with class, not object
④ Can't access non static method
⑤ Can be overloaded, but not overridden
Advantage- 1) Memory Efficiency 2) Improve Perform.
3) Global accessibility 4) Encapsulation

Super keyword.

Refer immediate parent class obj
- to access parent property
- " " " function
- " " " constructor

- Eliminate the confusion bet^n superclass & subclass that have method with the same name.

1) Enable reuse of code
2) Support polymorphism
3) Provide access to parent class behavior
4) Allows for customization of behavior