Encapsulation : binding or wrapping data (variable) and code (methods) in a single unit is known as encapsulation .

Ex : class

JavaBean class

1. Class must be public
2. All variable must be private
3. For each variable we need to provide setter and getter method
4. Setter method to set the value
5. Getter method to get the value

```
public class Customer {
private int cid;
private String cname;

public void setCId(int cid) {
        this.cid = cid;
}
public int getCid() {
        return this.cid;
}

}
```

Java bean class is known as pure encapsulation class.

Inheritance : it is use to inherits or acquire properties and behaviour of old class to new class.

```
class OldClass  {                       super class or base class or parent class.
        properties
        behaviour
}
class NewClass extends OldClass{        sub class or child class or derived class.
        properties
        behaviour
}
```

Types of inheritance

1. Single inheritance : one super class and one sub class
   Class A {}
   Class B extends A {}
2. Multilevel inheritance : one super class and n number of sub class connected one by one
   Class A {}
   Class B extends A {}
   Class C extends B{}
   Class D extends C{}
3. Hierarchical inheritance : one super class and n number of sub class connected directly to super class
   Class A {}
   Class B extends A{}
   Class C extends A{}
   Class D extends A{}
4. Multiple inheritance : more than one super class and one sub class
   Class A {}
   Class B {}
   Class C extends A,B{} but Java doesn't support this type of inheritance. This type of inheritance we can achieve using interface.

   class Employee {

   id,name,salary Scanner

   readEmp(),

   disEmp() ;

   }

   class Manager extends Employee{

   numberOfEmp

   readMgr()

   disMgr()

   }

   class Developer extends Employee{

   projectName

   readDev

   disDev

   }

   class ProjectManager extends Manager{

clientName

readProgMgr()

disProMgr()

}

Polymorphism one name many forms or many implementation.

Compile time polymorphism or static binding or early binding we need to compile program using javac compiler

Example : Method overloading : the method have same name but different parameter list ie type of parameter list or number of parameter list must be different.

Run time polymorphism or late binding or dynamic

java interpreter

Example : Method overriding : The method have same name and same method signature. (number of parameter list, type of parameter list and return type must be same). To achieve method overriding we need inheritance.

Java provided annotation concept.

Annotation : meta-data, data about data.

Java provided lot of pre defined annotation. All annotation start with pre-fix @ followed by annotation name.

We can use annotation on class level, method level as well as property level.

Non Access specifiers keywords

1. abstract
2. final
3. static

abstract : abstract is a keyword we can use with method and class but not with variable.

1. Abstract method : method without body is known as abstract method or incomplete method.
   Syntax
   abstract void speed();
2. If class contains one or more abstract method that class we need to declare as abstract class.
   abstract class Bike {

   }
3. Any normal class extends abstract class that class must be provide body for all abstract method belong to that abstract class mandatory. They can ignore if that class itself ie sub class is abstract class.
4. Abstract class we can't create object.
5. Abstract class can contains normal as well as abstract method. ie zero or 1 or many.
6. Abstract class can contains constructor ie empty or parameter but we can't create object.
7. We can call normal method belong to abstract class through sub class object.

Final keyword

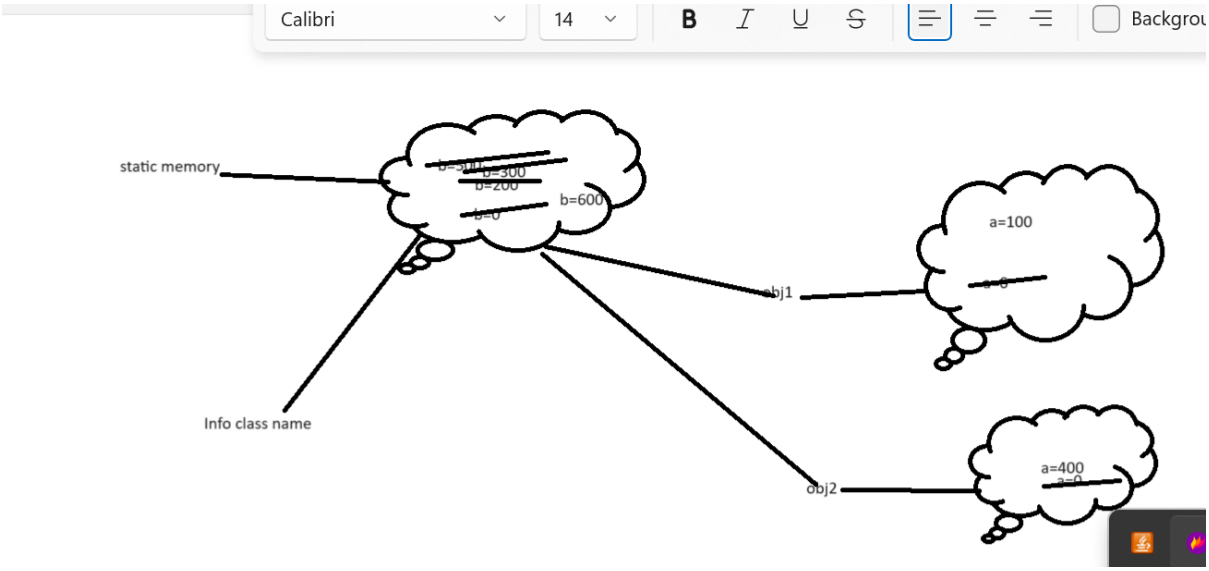
1. Final is a keyword we can use with variable, method and class.
2. Final variable : to declare constant variable in java we use final keyword.
   final int A=10;
   A=20; Error
3. final method : if method is final sub class can't override that method. but it an use it but can't override.
4. final class : if class is final we can't inherits or can't extends that class.

static keyword

1. static keyword we can use with variable and method but not with class.
2. static variable : if variable is static we can assign the value for that variable using class name object not required.
3. static method : if method is static we can call that method with help of class name object not required.
4. Even though we can assign the value for static variable with help object also as well as we can call static method with help of object also.
5. Inside non static method we can access both static as well as non static variable directly but variable must be part of same class.
6. Inside static method we can access only static variable directly but variable must be part of same class.
7.

Static Vs Non static (Heap memory).

Every class contains only one static memory.



Employee

Id,name,salary -→ instance variable

ClientId,TrainerId ,ProjectId static

Static is like a global to all objects.