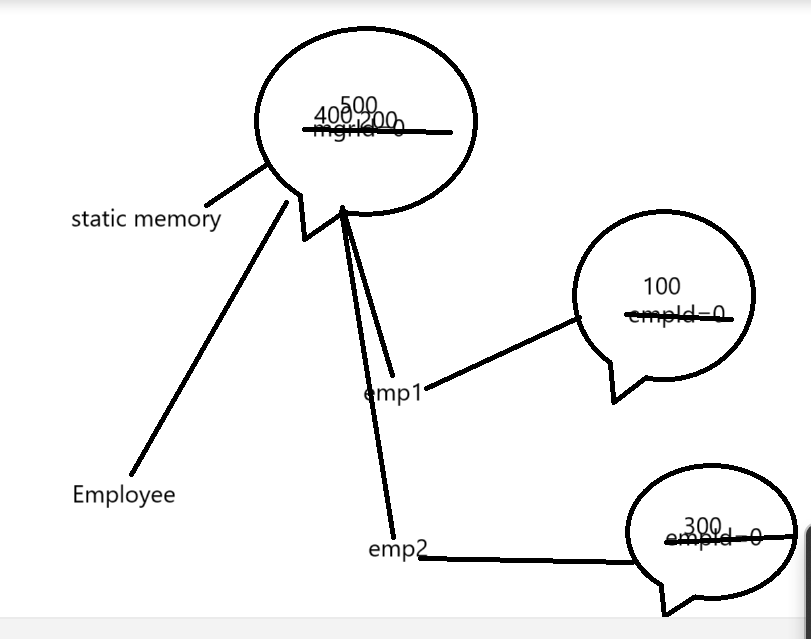
**super keyword**

super() constructor chaining. By default every sub class constructor super() parameter present. It always call super class empty constructor.

**Static keyword :** static is non access specifiers we can use with variable and method but not with class. (if class is inner class or nested class we can use static keyword but can’t use for outer class).

1. Static variable : if variable is static we can assign the value for those variable using class name.
2. Static method : if method is static we can call that method with help of class name.
3. Static variable and static method we can call through object also.
4. Inside a static method we can’t access non static variable directly.

Every class contains only one static memory.



Final keyword we can use with variable, method and class.

1. final variable : to declare constant value in java we use final keyword.

final int A=100;

float int PI=3.142f;

1. final method : if method is final we can’t override that method.
2. final class : if class is final we can’t extends or inherits that class.

**interface :** interface is a type of reference data type which also known as 100% pure abstract class.

syntax

interface interfaceName {

fields;

methods;

}

By default all methods inside a interface are public static and **final**. Interface contains only final variable.

By default all method part of interface are public and abstract.

interface Abc {

public static final int A=10;

**int A1=20;**

public abstract void dis1();

**void dis2();**

**void dis();**

**int M=100;**

}

interface Xyz {

static **int M=200;**

int B=20;

void dis3();

**void dis();**

}

interface Mno **extends Abc**,**Xyz**{

Int C=30;

void dis4();

}

class Test implements Abc,Xyz{

}

In java one interface can extends more than one interface.

**Interface vs Abstract class**

1. interface contains only final variable but abstract class may or may not.
2. interface contains only abstract method but abstract class can contain normal as well as abstract methods.
3. Interface doesn’t contain constructor but abstract class can contain default constructor.
4. Whichever class extends or implements interface that class must be provide the body for all those method part of interface or abstract class.

Interface provide the specification and class provide the implementation.

Using abstract class we can achieve partial abstraction using interface we can achieve 100% abstraction.

**Access specifiers : using access specifiers we can expose the visibility of variable, methods, class and constructor. s**

Java provided 4 types of access specifiers

1. private
   1. private we can use with instance variable, static variable, method (non static as wells static), constructor but we can’t use with class and local variable.
   2. Scope : within a same class.
2. default (nothing):
   1. we can use with all.
   2. Scope : within a same package.
3. protected
   1. private we can use with instance variable, static variable, method (non static as wells static), constructor but we can’t use with class and local variable.
   2. Scope : within a same package as well as other package if sub class.
4. public
   1. private we can use with instance variable, static variable, method (non static as wells static), constructor, class but not with local variable.
   2. Scope : same package as well as other package.

Package : package is a collection of classes and interface. If we want to write more than one class or interface which have same name but different purpose using package we can write. Package is like a directory or folder