Oops

Polymorphism : One name many forms or many implementation.

In java we can achieve two type of polymorphism

1. Compile time polymorphism or early binding or static binding
   Example : Method overloading
   The method java same name but different parameter list ie type of parameter list or number of parameter list must be different.
2. Run time polymorphism or late binding or dynamic binding
   Example : Method overriding
   The method have same name as well as same method signature(number of parameter list, type of parameter list and return type must be same).
   To achieve method override we need inheritance concept.

To check sub class method overriding super class method.

Annotation : it is a meta-data , meta data mean data about data. Annotation is like a decorator in angular.

1. We can use annotation. Java provided lot of pre defined annotation all annotation start with pre-fix @ followed by annotation name. we can use annotation in java on class, variable as well as method.
   @Override This annotation we need to use in sub class method. If sub class method overriding super class method then it doesn't show any exception. If sub class didn't override super class method it will display error.

Non access specifiers keywords.

1. abstract
   abstract is keyword we can use with method and class but with variable.
   a. abstract method : method without body or incomplete method or without curly braces.
      abstract returntype methodName(parameterList);
      abstract void speed();
   b. if class contains one or more than one method as abstract we need declare that class as abstract class.
      abstract class className {

      }
   c. which ever class extends abstract class that class must be provided the body for all those abstract method belong to that class mandatory.
   d. abstract class can contains zero or one or all abstract methods.
   e. abstract class we can't create the object.

2. final :
   final keyword we can use with variable, method and class.
   a. to declare a constant value in java we use final variable.
      final int A=10;
      A=20; Error
   b. final method : if method is final sub class can't override that method. But they can call it or use it.
      final void speed() {

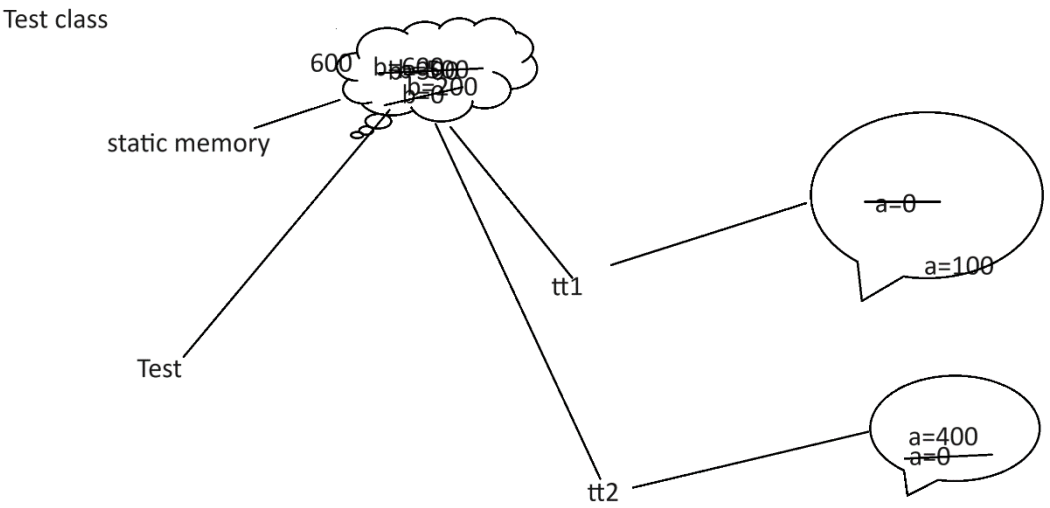
      }
   c. final class : if class is final we can't extends or inherits that class.
      we can't use abstract and final together.

3. static

   static keyword :

   static keyword we can use with variable and method but not with class.

   a. static variable : if variable is a static we can assign the value for that variable using class name.
   b. static method : if method is static we can call that method with help of class name.
   c. we can access static method as well as static variable using object also.
   d. Inside a non static method we can access static as well as non static variable directly.
   e. Inside static method we can access only static variable directly.



Static property global for all object.

So in employee class id,name,salary,desg are instance and manager id or client id must be static.

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

syntax to declare the interface

```
interface interfaceName {
        variables;
        methods;
}
```

By default all variable part of interface are public static and final.

By default all method in interface are public and abstract.

```
interface Abc {
        public static final int A=10;
        static final int B=20;
        final int C=30;
        int D=40;
        int E;              Error
        public abstract void dis1();
        abstract void dis2();
        void dis3();
        void dis4() {}                  // Error
}
```

Interface contains constant variable and abstract methods.

```
interface Abc {
        int A=10;
        void dis1();
}
interface Xyz {
        int B=20;
        void dis2();
}
interface Mno extends Abc,Xyz{
        int C=30;
        void dis3();
}
class Demo implements Abc,Xyz{
        void dis1() {          }
        void dis2() {          }
}
```

Like a class one interface can extends another interface as well as interface can extends more than one interface but class can't. Abc and Xyz is super interface and Mno is sub interface ie multiple inheritance.

Class always implements interface. Class can implements more than one interface. Whichever class implements any interface that class must be provide the body for all those method which belongs those interface.

Access specifiers while method overriding

public

private

protected

default (nothing)

Super class / interface	sub class
default	default public protected
protected	protected public
public	public
private	we can’t override that method

Abstract Class Vs Interface

1. Abstract class can contain normal as well as final variable but interface contains only final variable.
2. Abstract class can contain normal as well as abstract method but interface contains only abstract method.
3. Any normal class can extends only one abstract class but it can implements more than one interface.
4. Using abstract class we can achieve partial abstraction but using interface we can achieve 100% abstraction.

Interface provide specification and class provide implementation.

package : package is a collection of classes and interface which have same name but different functionality. Package is just like a directory or folder.