09-02-2013 | SATURDAY

POLYMORPHISM

P87

```
1) class A
   {
     void test1()
     {
       Sop("running test1() of class A");
     }
   }
   class B extends A
   {
     void test1()
     {
       Sop("running test1() of class B");
     }
   }

   class C extends B
   {
     void test1()
     {
       Sop ("running test1() of class C");
     }
   }
   class X
   {
     void sample1 (A a1)
     {
       a1.test1();
     }

   class Run9
   {
     psvm (String[] args)
     {
       Sop ("program starts...");
       X x1 = new X();
       x1.sample1 (new A());
       x1.sample1 (new B());
       x1.sample1 (new C());
```

1*) A method declared with reference type argument can hold any type of object which is a sub class to the reference type.

<u>Example</u>: If a method() takes an argument of A-type, while invoking that method, we can pass any object which is derived from A class type.

While passing a subclass object, the compiler does auto upcast to the argument type.

2*) Inside the method if we have to refer the sub class members using upcasted object, then that object should be downcasted to the sub class type.

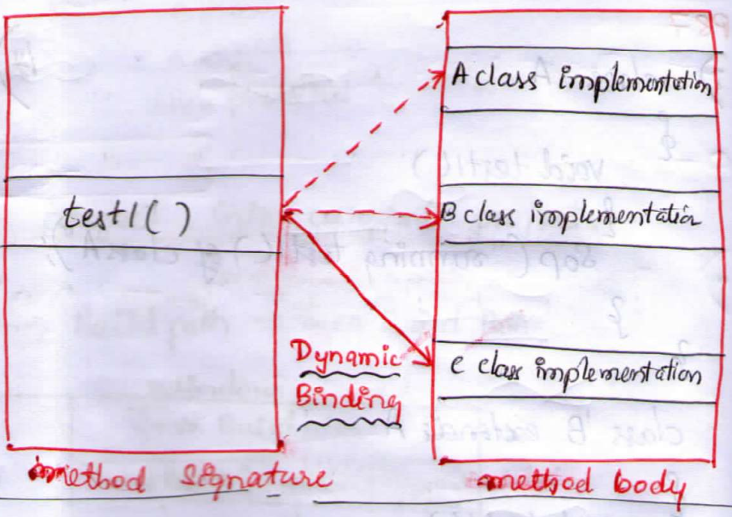
```
Sop ("program ends...");

}
```

O/p :

```
Program starts...
running test1() of class A
running test1() of class B
running test1() of class C
Program ends...
```

Run Time Polymorphism



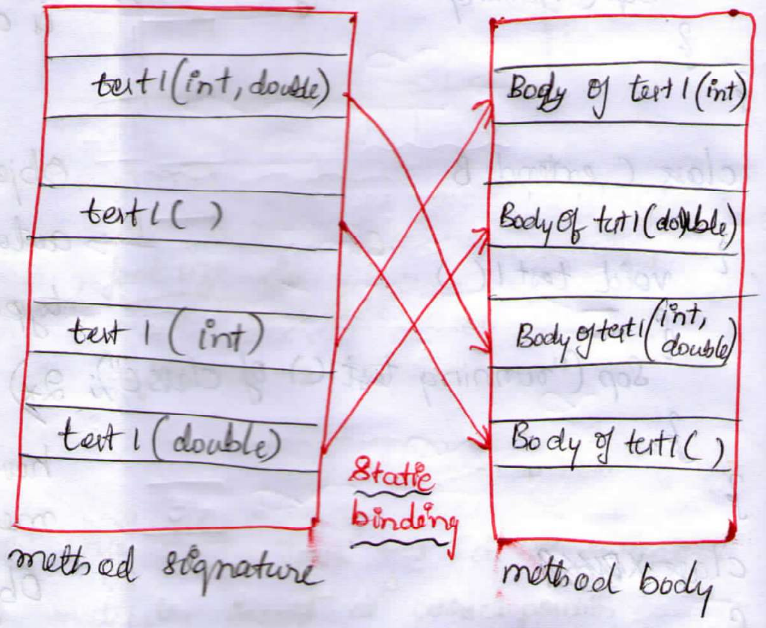
Compile time polymorphism :

```
class A
{
    test1( )
    {
    ______
    }

    test1(int a)
    {
    ______
    }

    test1(double d)
    {
    ______
    }

    test1(int a, double d)
    {
    ______
    }
```



1*) An object behaving differently in a diff condition is known as polymorphism.

2*) Two types of polymorphism is supported in java

1* Compile time polymorphism

2* Run time polymorphism

3*) In Compile time polymorphism method behavior is binded to the method signature at the time of compilation by compiler. This binding is static binding because once binded it cannot change throughout the program. This is also known as Static Polymorphism

4*) Examples of compile time polymorphism is method Overloading & constructor overloading.

5*) In Run-time polymorphism the method behavior is binded to method signature at the execution time done by jvm. The binding happens based on object created. The binding can be changed in the program execution, hence it is known as dynamic binding. This is also known as Dynamic polymorphism

6*) Example : method overriding.
To achieve run time polymorphism following condition should be satisfied.

1* Inheritance (or) Is-A relationship

2* Method overriding

3* Upcast