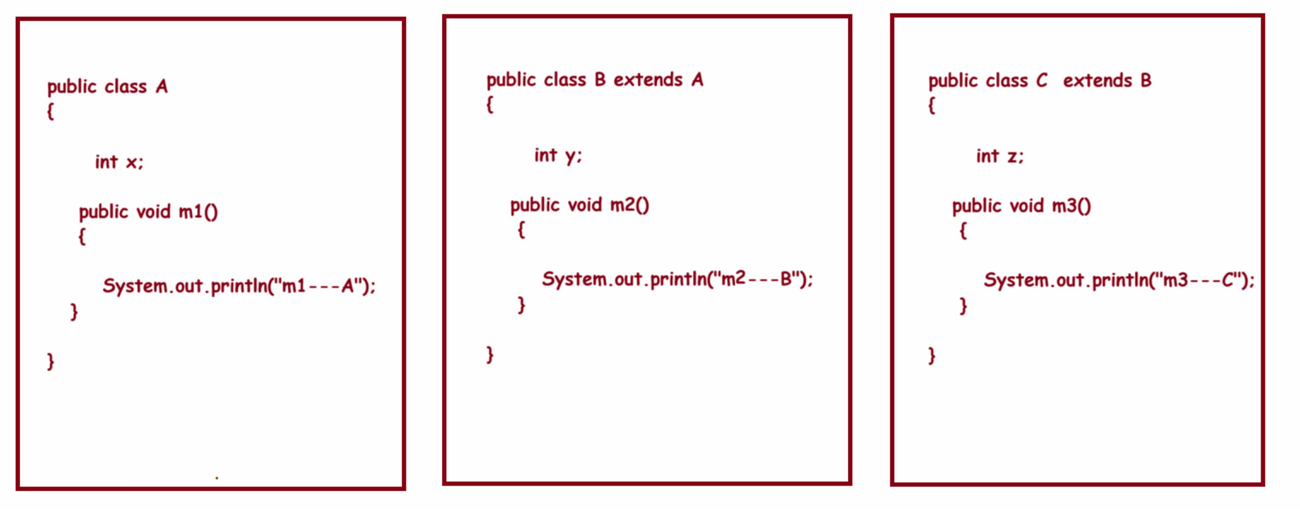
Monday, October 7, 2024   11:39 AM

Multilevel Inheritance

```
package com.cjc.multilevel;

public class A {

        int x = 10;

        public void m1() {
                System.out.println("m1--A");
        }

}
---------------------------------------------------------------
package com.cjc.multilevel;

public class B extends A{

        int y = 20;

        public void m2() {
                System.out.println("m2---B");
        }
}
-------------------------------------------------------------
package com.cjc.multilevel;

public class C extends A{

        int z = 30;

        public void m3() {
                System.out.println("m3---C");
        }
}


------------------------------------------------------------
package com.cjc.multilevel;

public class Test {

public static void main(String[] args) {

                System.out.println("main----start");
```

```
                    A a = new A();
                    a.m1();
                    System.out.println(a.x);
System.out.println("--------------------------------");

                    B b = new B();
                    b.m2();
              b.m1();
                    System.out.println(b.y);
                    System.out.println(b.x);
System.out.println("---------------------------------");
                    C c  = new C();
                    c.m3();
                    c.m1();
                    System.out.println(c.z);
                    System.out.println(c.x);


          }
}
```



```
public class A
{

      int x;

    public void m1()
    {

        System.out.println("m1---A");
    }

}
```

```
public class B extends A
{

      int y;

    public void m2()
     {

        System.out.println("m2---B");
     }

}
```

```
public class C  extends B
{

      int z;

    public void m3()
     {

        System.out.println("m3---C");
     }

}
```

Method and Variable Compilation and Running Rule

A a = new B();

1) Method and Variable Compilation Rule :

- Method and variable always compiles from reference class (A.java) if it is not present in reference class then it will check in there parent class (Object.java)

2) Variable Running Rule :

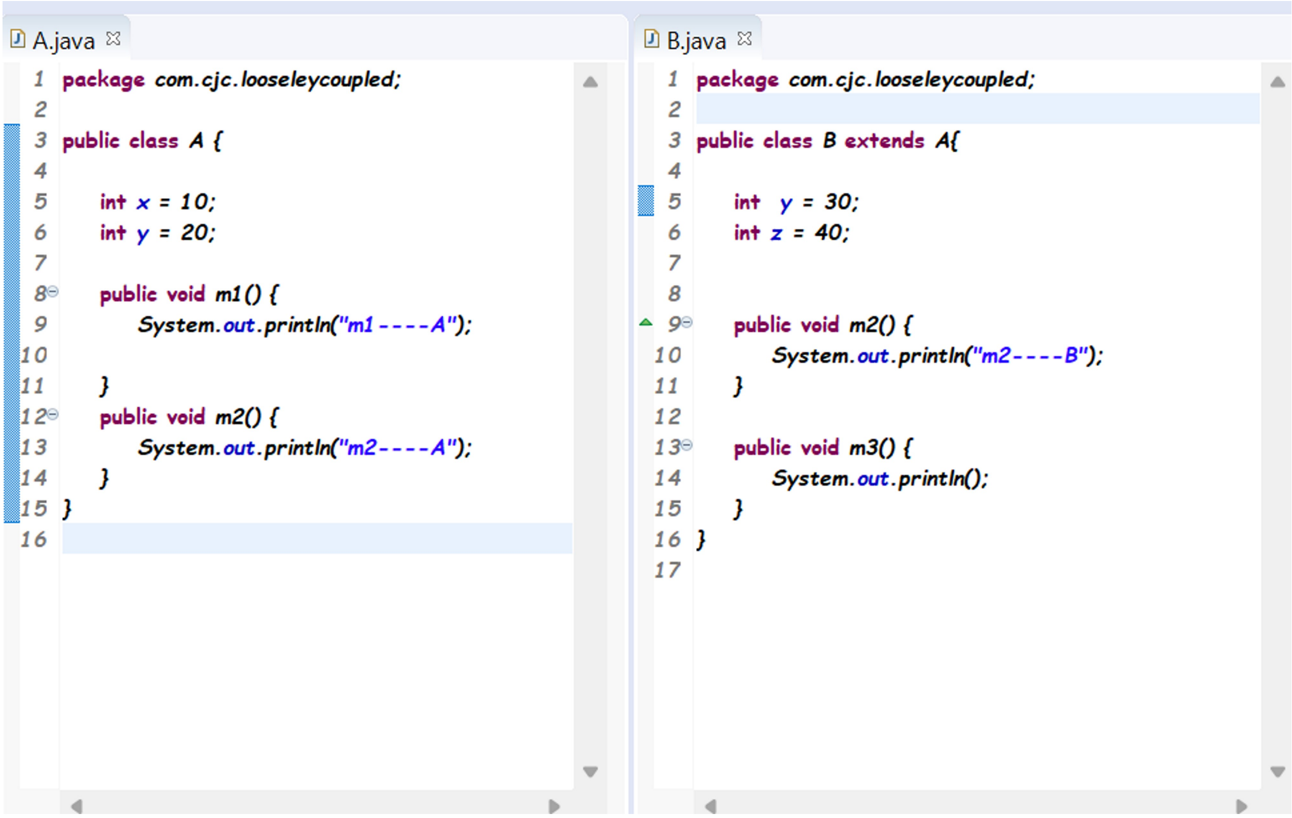
- Variable always runs from reference class (A.java) and if it is not present in reference class Then it run from parent class (Object.java)

2) Variable Running Rule :

- Variable always runs from reference class (A.java) and if it is not present in reference class Then it run from parent class (Object.java)

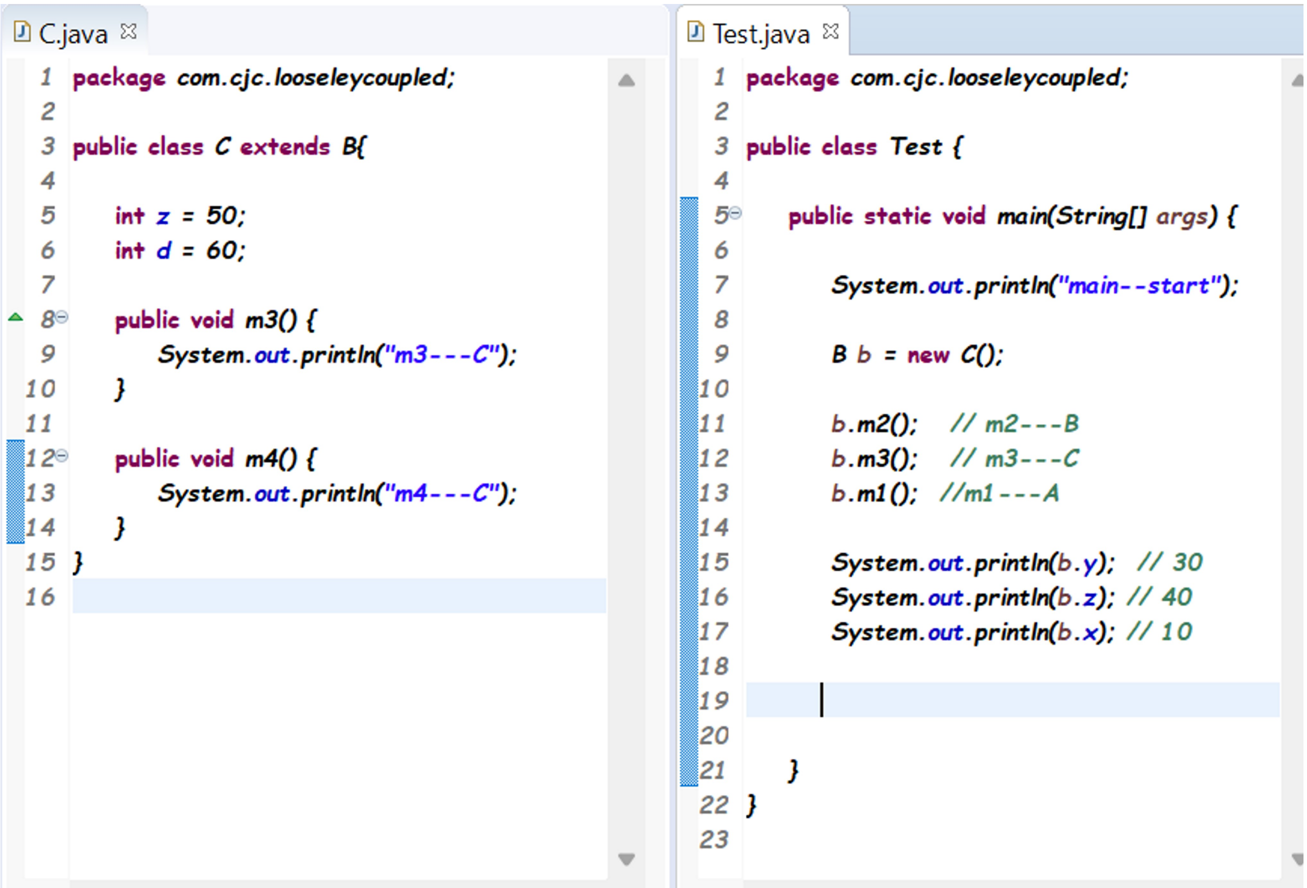
3) Method Running Rule :

- Method running always starts from Constructor class (B.java) and if it is not present in in constructor class then it will run from parent class (A.java)



```java
package com.cjc.looseleycoupled;

public class A {

    int x = 10;
    int y = 20;

    public void m1() {
        System.out.println("m1----A");

    }
    public void m2() {
        System.out.println("m2----A");
    }
}

```

```java
package com.cjc.looseleycoupled;

public class B extends A{

    int  y = 30;
    int z = 40;


    public void m2() {
        System.out.println("m2----B");
    }

    public void m3() {
        System.out.println();
    }
}

```



```java
package com.cjc.looseleycoupled;

public class C extends B{

    int z = 50;
    int d = 60;

    public void m3() {
        System.out.println("m3---C");
    }

    public void m4() {
        System.out.println("m4---C");
    }
}

```

```java
package com.cjc.looseleycoupled;

public class Test {

    public static void main(String[] args) {

        System.out.println("main--start");

        B b = new C();

        b.m2();   // m2---B
        b.m3();   // m3---C
        b.m1();  //m1---A

        System.out.println(b.y);  // 30
        System.out.println(b.z); // 40
        System.out.println(b.x); // 10



    }
}

```

A a1 = new A();
A a2 = new B();
A a3 = new C();

B b1 = new B();
B b2 = new C();

C c1 = new C();