Constructor calling order in case of multilevel inheritance:-

Suppose, there’s an inheritance relationship exists between class A and class B and there’s another inheritance relationship exists between class B and class C.

Suppose, this is the representation of the multilevel inheritance we are talking about.

Now, if we write a program like the following one:-

class A

{

…………………………….

……………………………. //declaration of private members of the class A

public A()

{

System.out.println(“Constructor of class A”);

…………………………….

……………………………. //rest of the implementation of constructor A

}

…………………………….

……………………………. //definition of public members of the class A

}

class B extends A

{

…………………………….

……………………………. //declaration of private members of the class B

public B()

{

System.out.println(“Constructor of class B”);

…………………………….

……………………………. //rest of the implementation of constructor B

}

…………………………….

……………………………. //definition of public members of the class B

}

class C extends B

{

…………………………….

……………………………. //declaration of private members of the class C

public c()

{

System.out.println(“Constructor of class C”);

…………………………….

……………………………. //rest of the implementation of constructor c

}

…………………………….

……………………………. //definition of public members of the class C

}

Then if we want to create an instance of the class c, by its default constructor,

i.e. by using the following statement:-

C c1=new C();

the following statements will be displayed:-

Constructor of class A

Constructor of class B

Constructor of class C

This is the constructor invoking order if we want to create an instance of the sub-class or the derived class in the case of multilevel inheritance.