**Constructor chaining-**

A constructor is called from another constructor in the same class this process is known as **constructor chaining.**

It occurs through inheritance.

When we create an instance of a derived class, all the constructors of the inherited class (base class) are first invoked, after that the constructor of the calling class (derived class) is invoked.

Why:

One of the main use of constructor chaining is to avoid duplicate codes while having multiple constructor (by means of constructor overloading) and make code more readable.

Rules for constructor chaining

1. An expression that uses this keyword must be the first line of the constructor.
2. Order does not matter in constructor chaining.
3. There must exist at least one constructor that does not use this keyword.

**How to achieve constructor chaining in two ways**

1. Within the same class-

If the constructors belong to the same class, we use this keyword.

1. From the base class and derived class-

If the constructor belongs to different classes (parent and child classes), we use the super keyword to call the constructor from the base class.

**Program for constructor chaining within same class**

**package** com.test;

**public** **class** Chaining {

Chaining() {

**this**(5);

System.***out***.println("This is Default constructor");

}

Chaining(**int** x) { //this is parameterized constructor with int

**this**("Java");

System.***out***.println(x);

}

Chaining(String str) { //this is parameterized constructor with string

System.***out***.println(str);

}

**public** **static** **void** main(String args[]) {

Chaining chaining=**new** Chaining(); //calling default constructor here

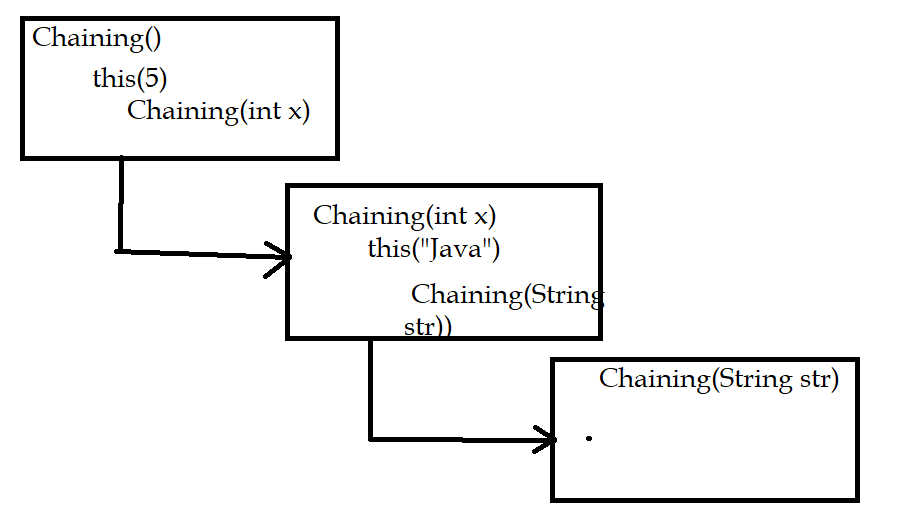
}

}

Output

Java

5



This is Default constructor

**Program for constructor chaining within super and sub class**

**package** com.test;

**public** **class** Base {

String name;

Base() {

**this**("");

System.***out***.println("No-argument constructor of base class....");

}

Base(String name) {

**this**.name = name;

System.***out***.println("Calling parameterized constructor of base class....");

}

}

**package** com.test;

**public** **class** Derived **extends** Base {

Derived() {

System.***out***.println("No-argument constructor of derived class");

}

Derived(String name) {

**super**(name); //calling base class constructor

System.***out***.println("Calling parameterized constructor of derived class");

}

**public** **static** **void** main(String args[]) {

Derived derived = **new** Derived("test"); //calling Derived class parameterized constructor

}

}

Output

Calling parameterized constructor of base class....

Calling parameterized constructor of derived class