# Java static keyword

* The static keyword in Java is used for memory management mainly. We can apply static keyword with variables, methods, blocks and nested classes. The static keyword belongs to the class than an instance of the class.
* The static can be:
  + Variable (also known as a class variable)
  + Method (also known as a class method)
  + Block
  + Nested class

## 1) Java static variable

* If you declare any variable as static, it is known as a static variable.
* The static variable can be used to refer to the common property of all objects (which is not unique for each object), for example, the company name of employees, college name of students, etc.
* The static variable gets memory only once in the class area at the time of class loading.

### Advantages of static variable

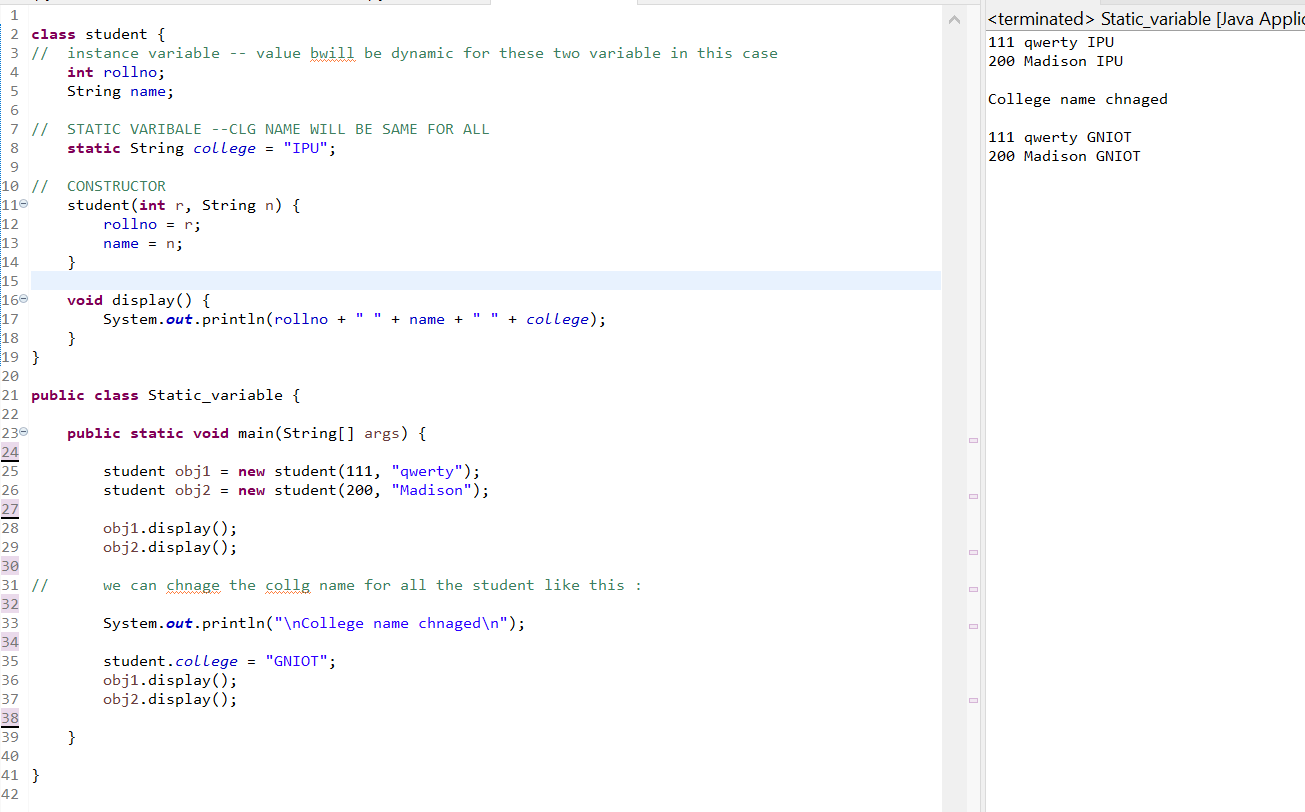
* It makes your program **memory efficient** (i.e., it saves memory).

#### Understanding the problem without static variable

1. **class** Student{
2. **int** rollno;
3. String name;
4. String college="ITS";
5. }

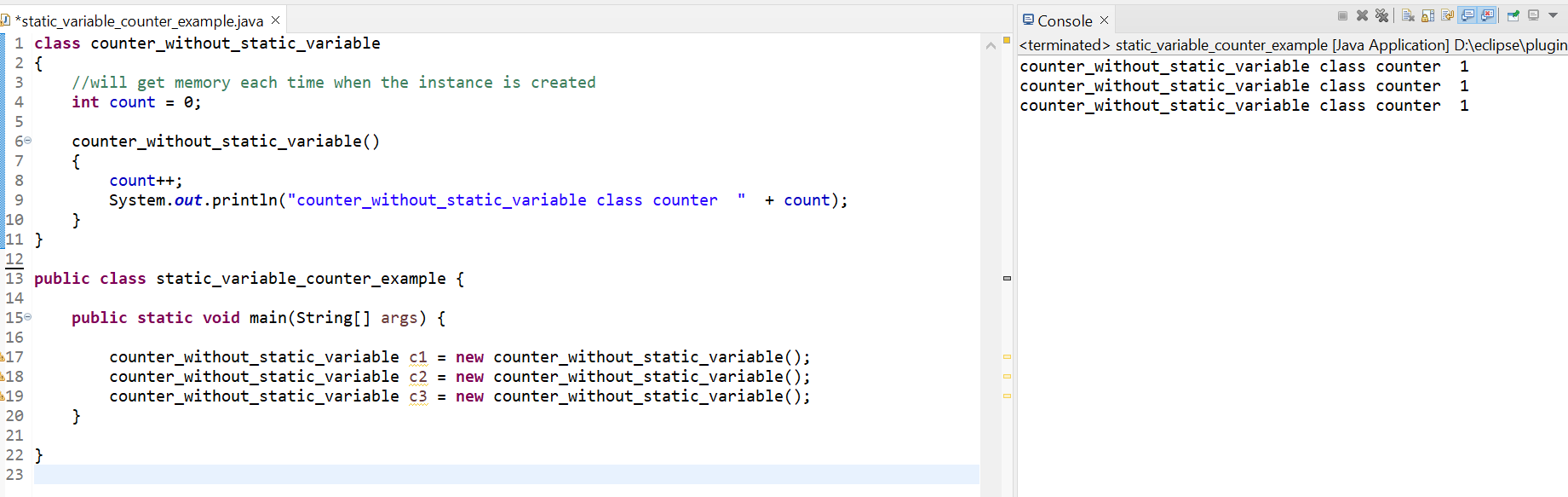
* Suppose there are 500 students in my college, now all instance data members will get memory each time when the object is created. All students have its unique rollno and name, so instance data member is good in such case. Here, "college" refers to the common property of all [objects](https://www.javatpoint.com/object-and-class-in-java). If we make it static, this field will get the memory only once.
* Java static property is shared to all objects.

EXample :



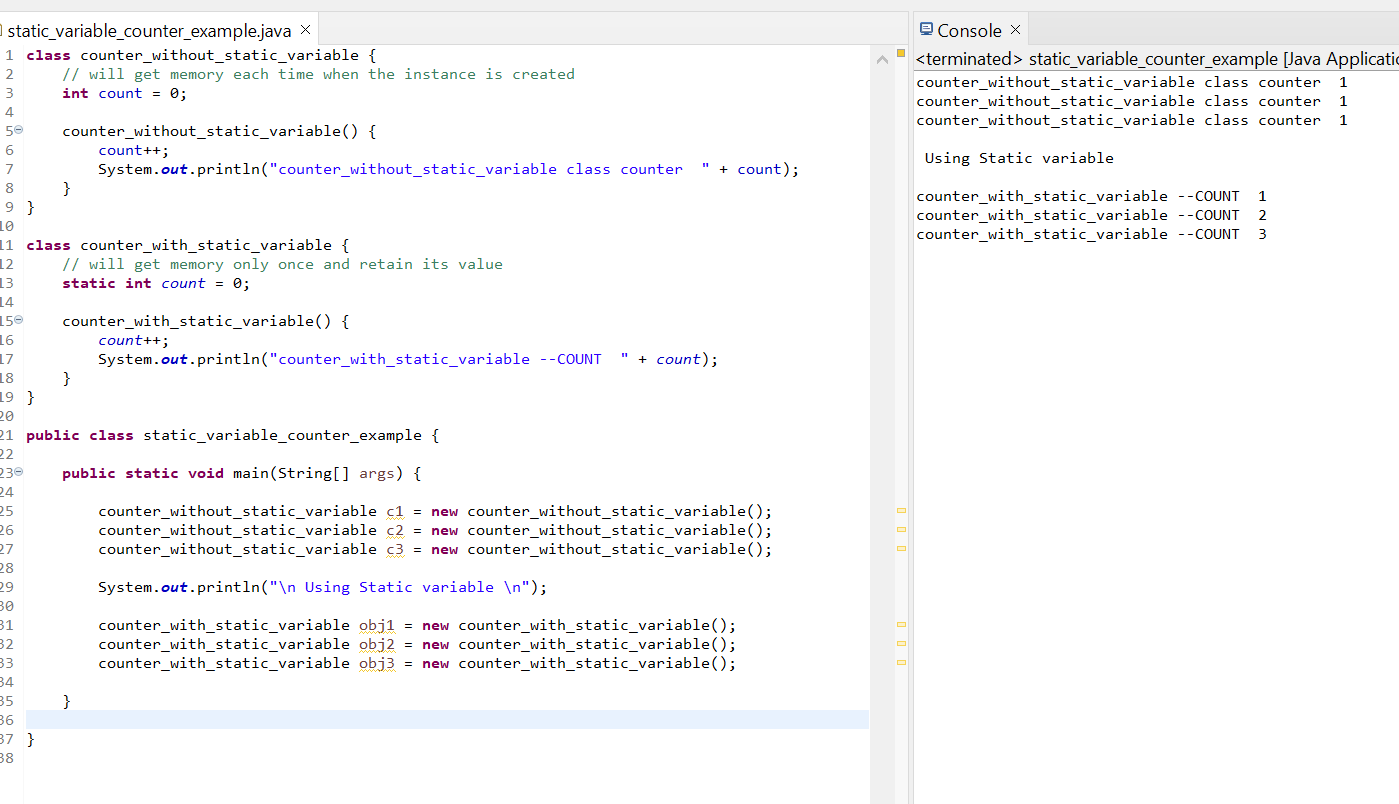
### Program of the counter without static variable

* In this example, we have created an instance variable named count which is incremented in the constructor.
* Since instance variable gets the memory at the time of object creation, each object will have the copy of the instance variable.
* If it is incremented, it won't reflect other objects. So each object will have the value 1 in the count variable.



### Program of counter by static variable

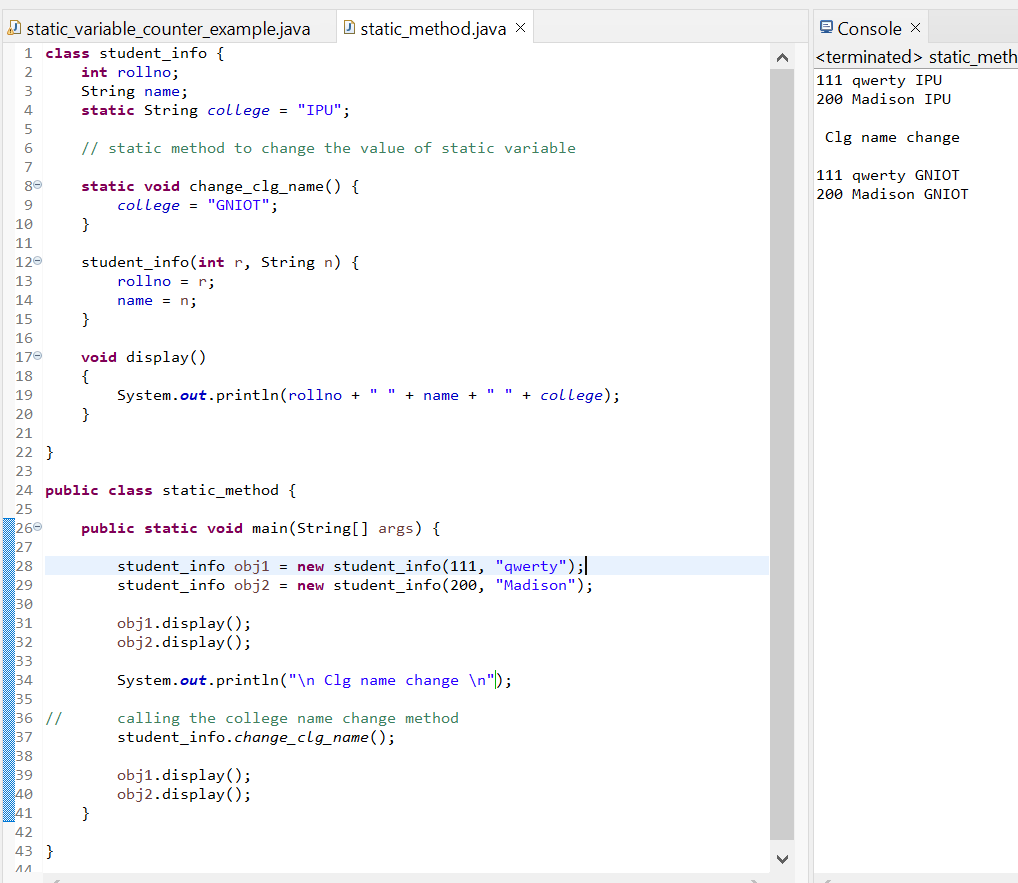
* As we have mentioned above, static variable will get the memory only once, if any object changes the value of the static variable, it will retain its value.



## 2) Java static method

If you apply static keyword with any method, it is known as static method.

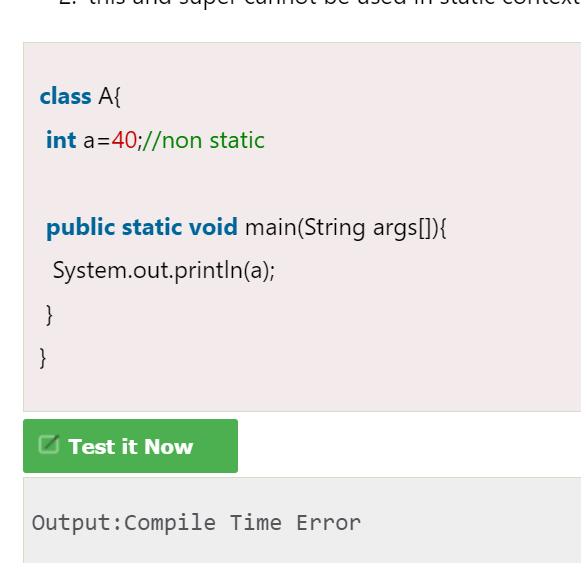
1. A static method belongs to the class rather than the object of a class.
2. A static method can be invoked without the need for creating an instance of a class.
3. A static method can access static data member and can change the value of it.



### Restrictions for the static method

There are two main restrictions for the static method. They are:

1. The static method can not use non static data member or call non-static method directly.
2. this and super cannot be used in static context.

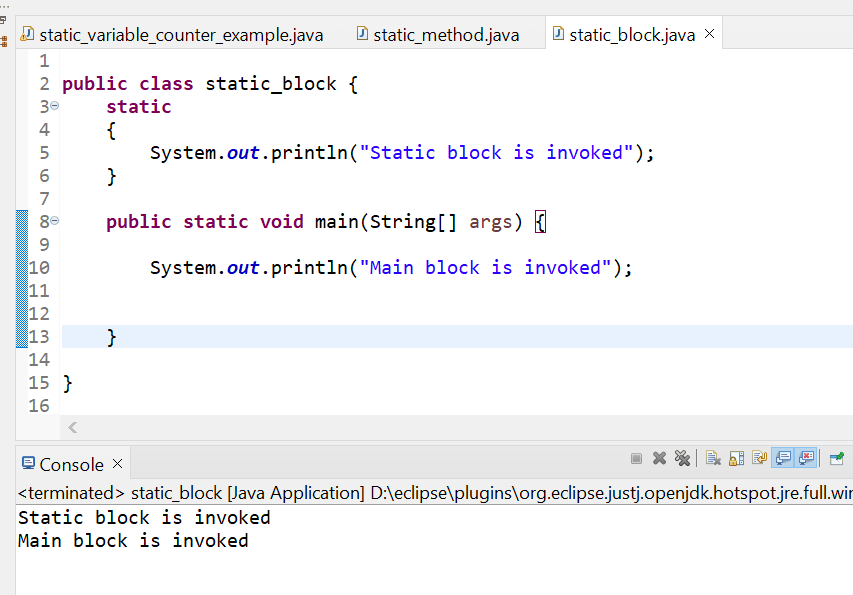


### Q) Why is the Java main method static?

Ans) It is because the object is not required to call a static method. If it were a non-static method, [JVM](https://www.javatpoint.com/jvm-java-virtual-machine) creates an object first then call main() method that will lead the problem of extra memory allocation.

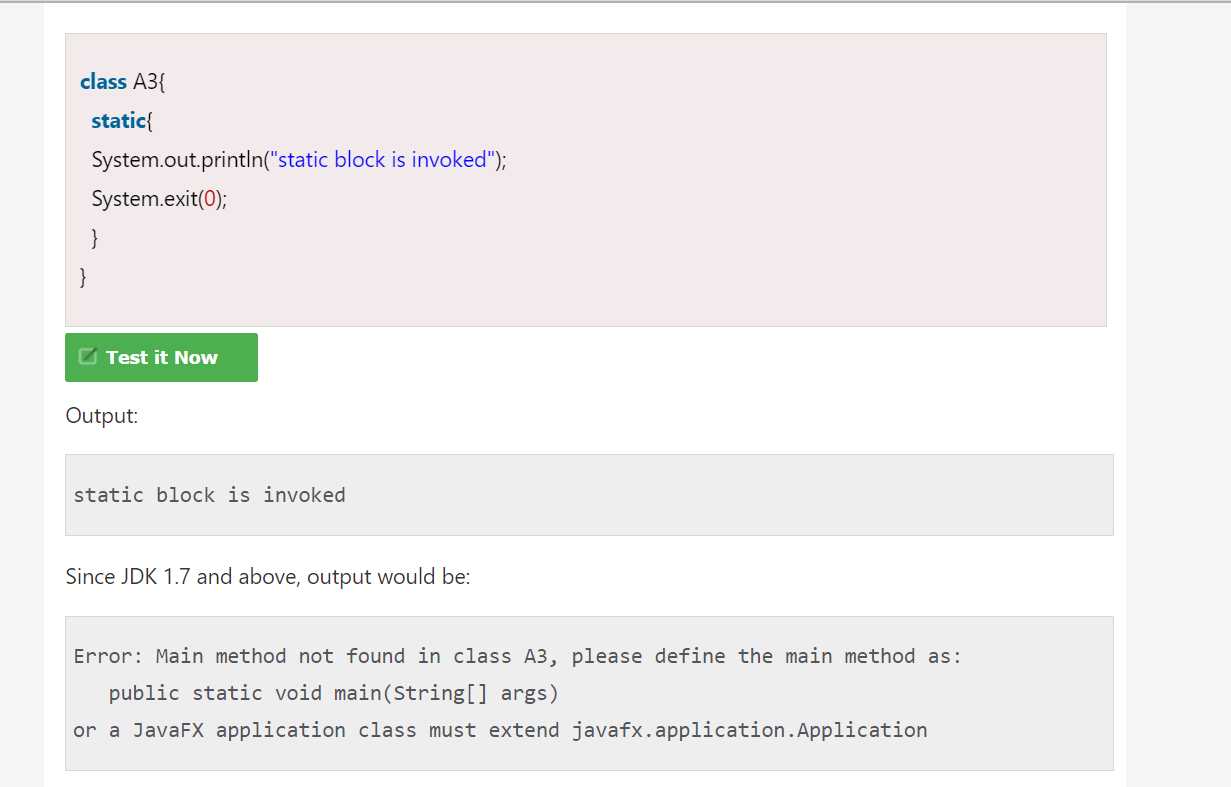
## 3) Java static block

* Is used to initialize the static data member.
* It is executed before the main method at the time of classloading.



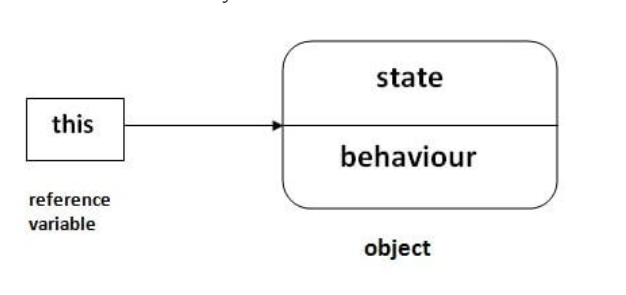
### Q) Can we execute a program without main() method?

Ans) No, one of the ways was the static block, but it was possible till JDK 1.6. Since JDK 1.7, it is not possible to execute a Java class without the [main method](https://www.javatpoint.com/java-main-method).



# this keyword in Java

* There can be a lot of usage of **Java this keyword**. In Java, this is a **reference variable** that refers to the current object.



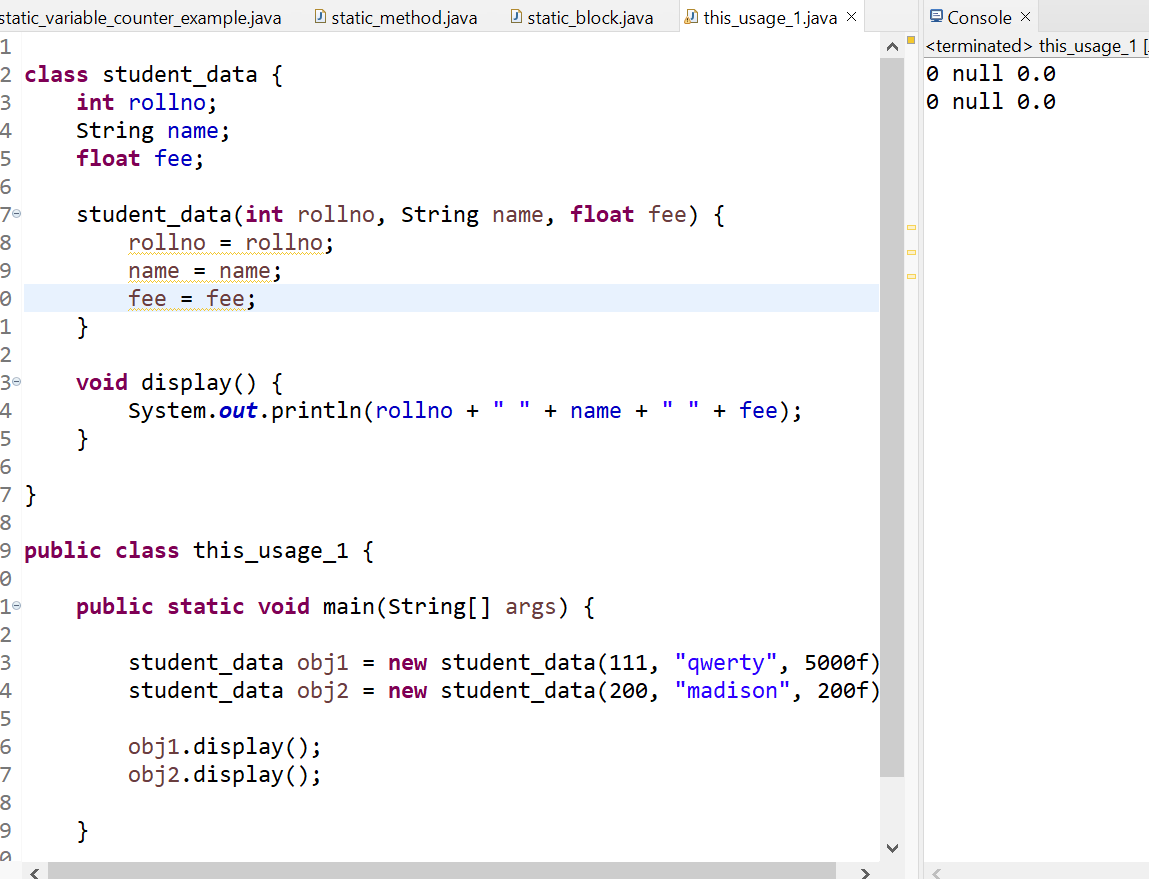
## Usage of Java this keyword

Here is given the 6 usage of java this keyword.

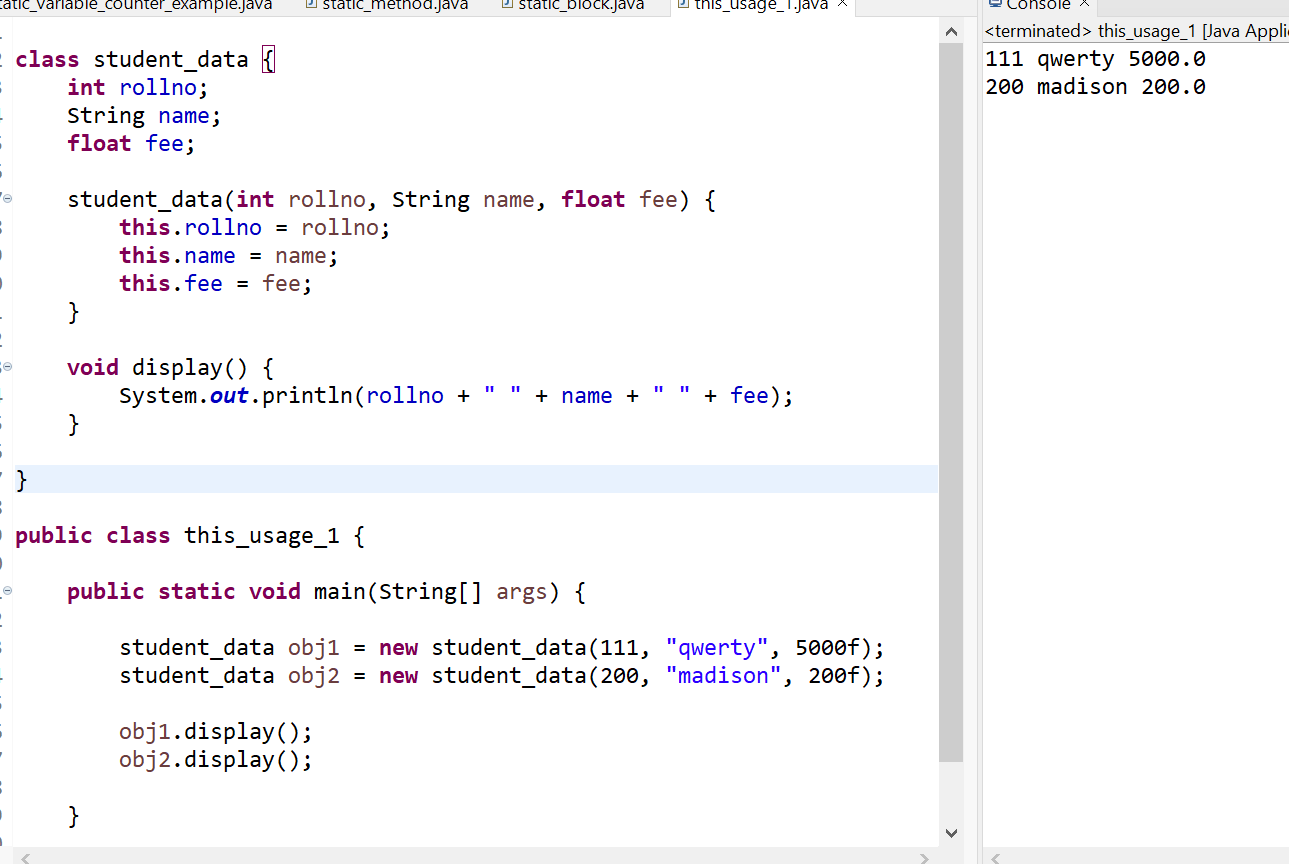
1. this can be used to refer current class instance variable.
2. this can be used to invoke current class method (implicitly)
3. this() can be used to invoke current class constructor.
4. this can be passed as an argument in the method call.
5. this can be passed as argument in the constructor call.
6. this can be used to return the current class instance from the method.

### 1) this: to refer current class instance variable

* The this keyword can be used to refer current class instance variable. If there is ambiguity between the instance variables and parameters, this keyword resolves the problem of ambiguity.

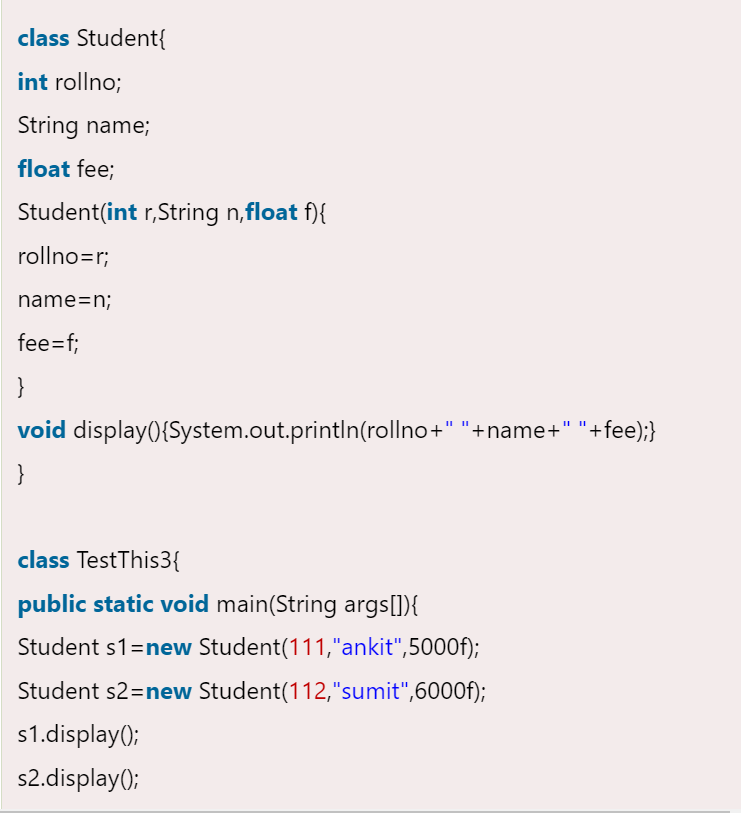


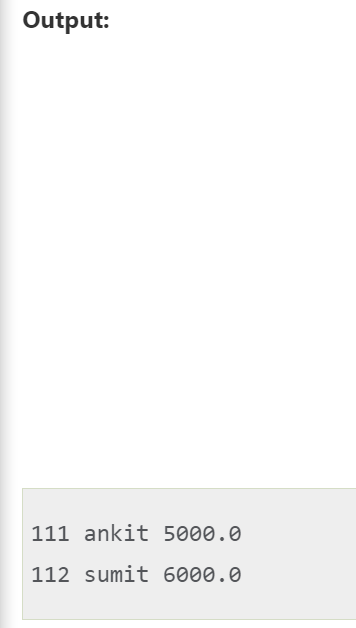
In the above example, parameters (formal arguments) and instance variables are same. So, we are using this keyword to distinguish local variable and instance variable.



It is better approach to use meaningful names for variables. So we use same name for instance variables and parameters in real time, and always use this keyword.

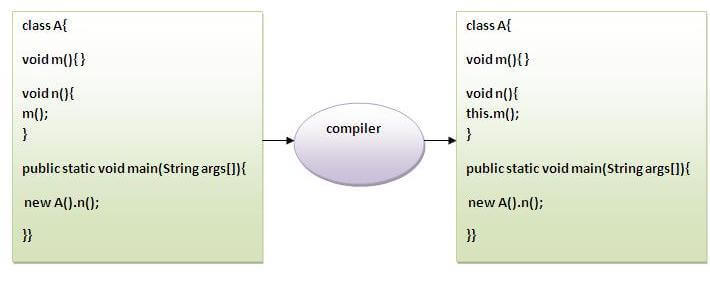
Example :

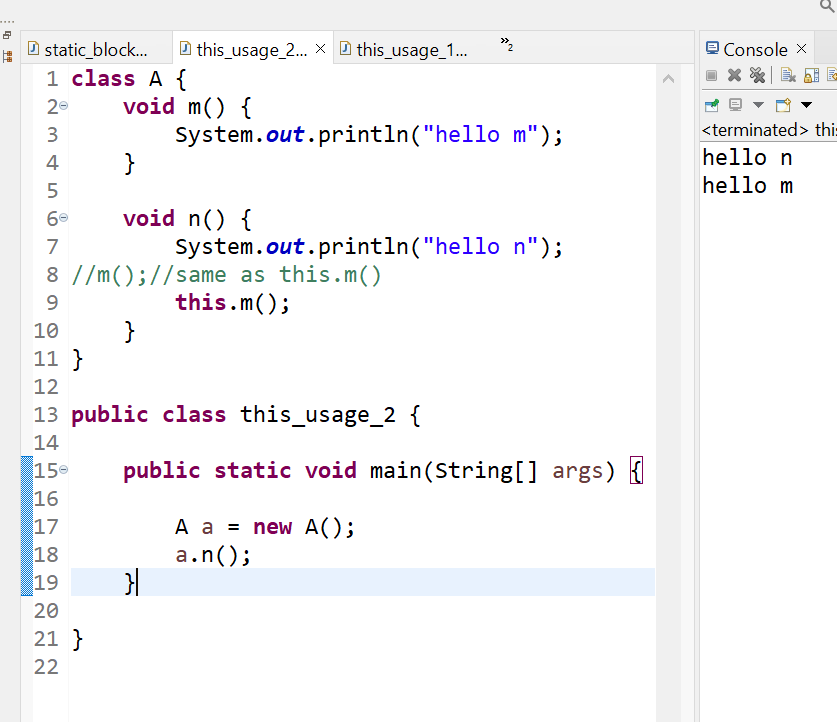




### 2) this: to invoke current class method

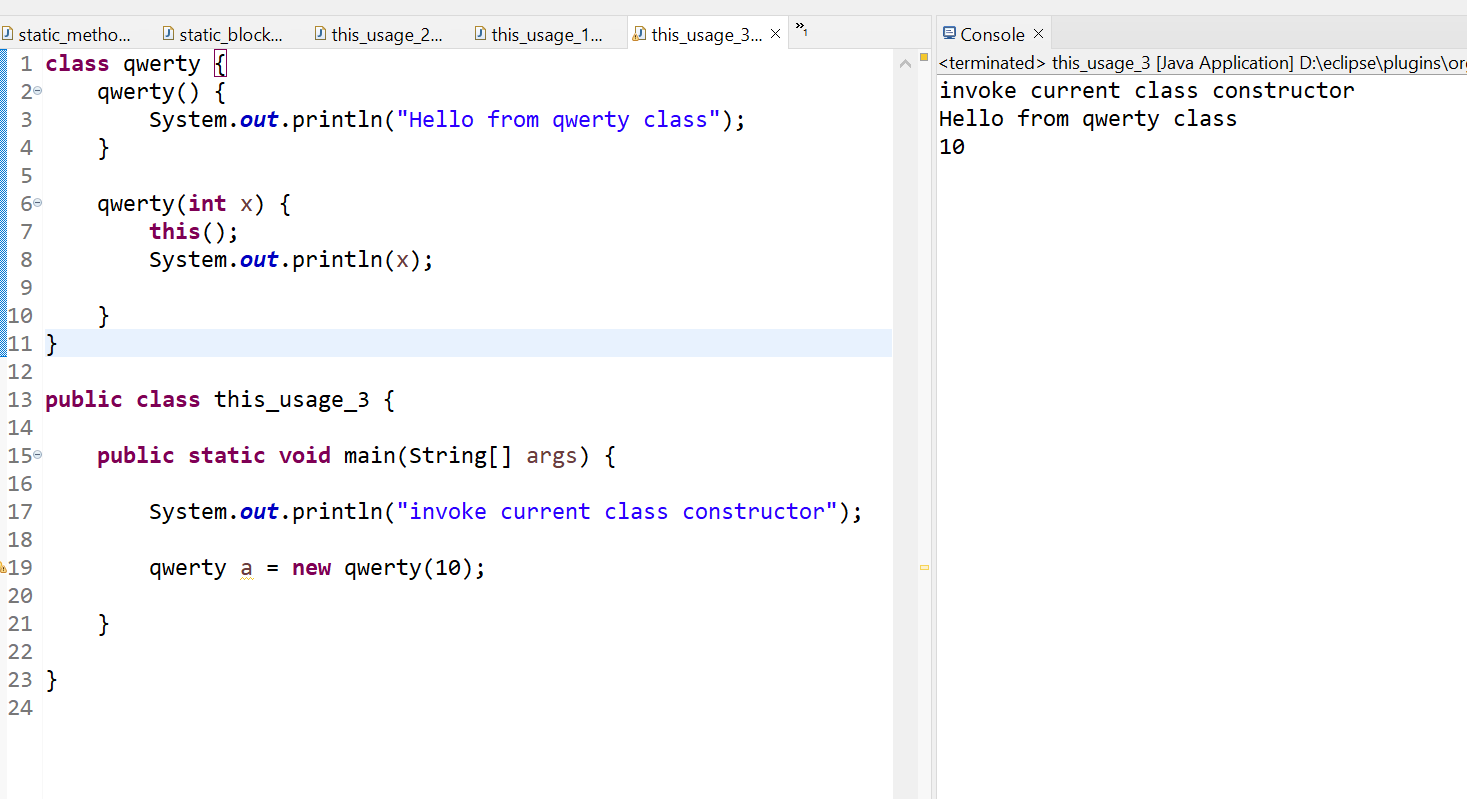
* You may invoke the method of the current class by using the this keyword.
* If you don't use the this keyword, compiler automatically adds this keyword while invoking the method.

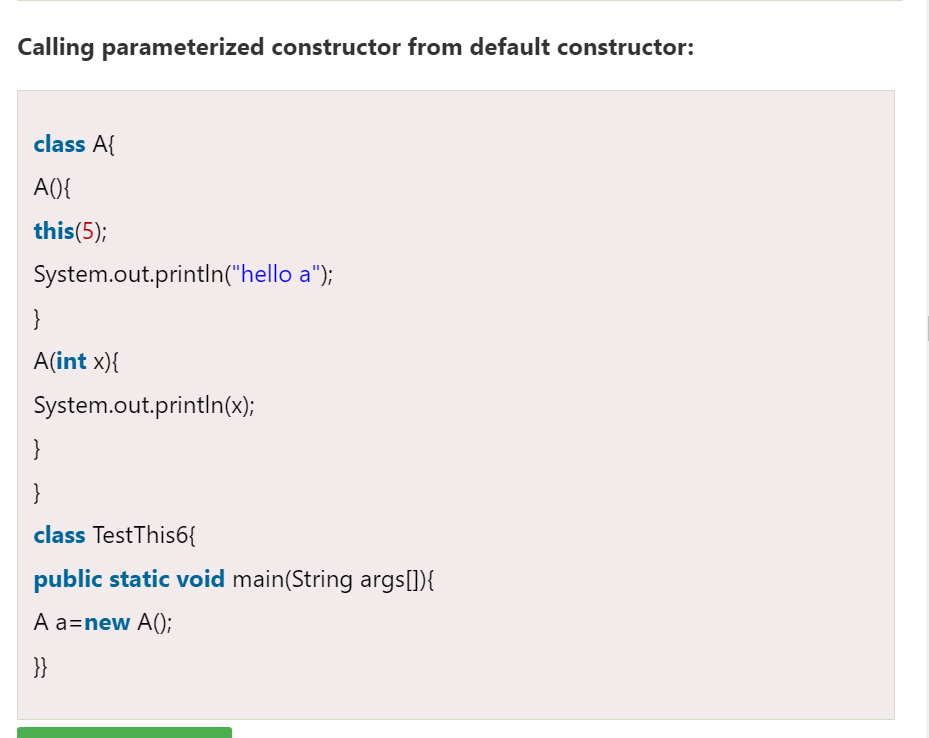




### 3) this() : to invoke current class constructor

The this() constructor call can be used to invoke the current class constructor. It is used to reuse the constructor. In other words, it is used for constructor chaining.





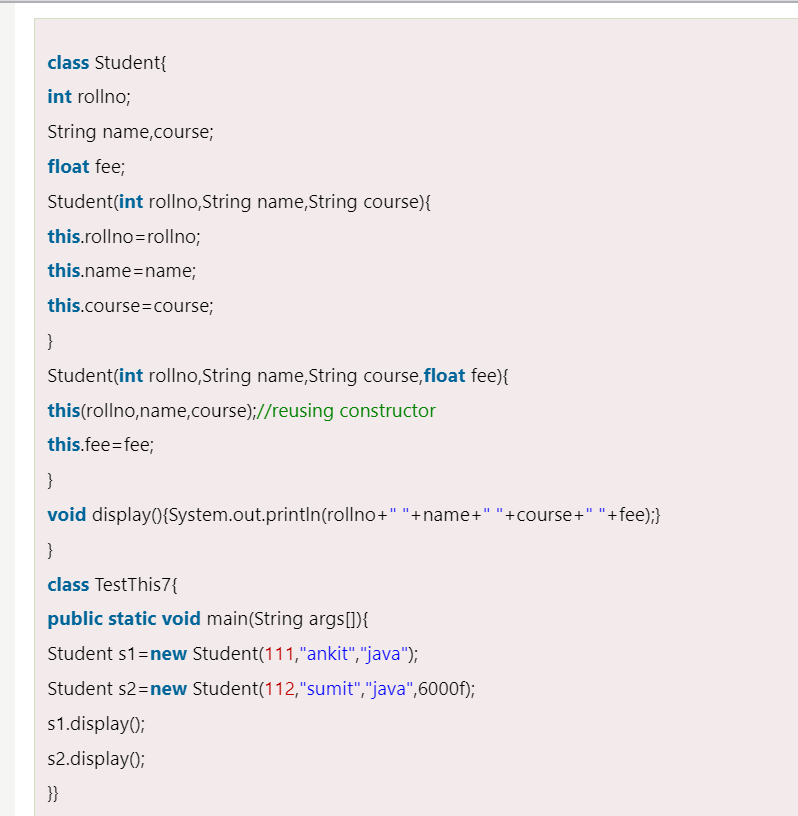
Output :

5

hello a

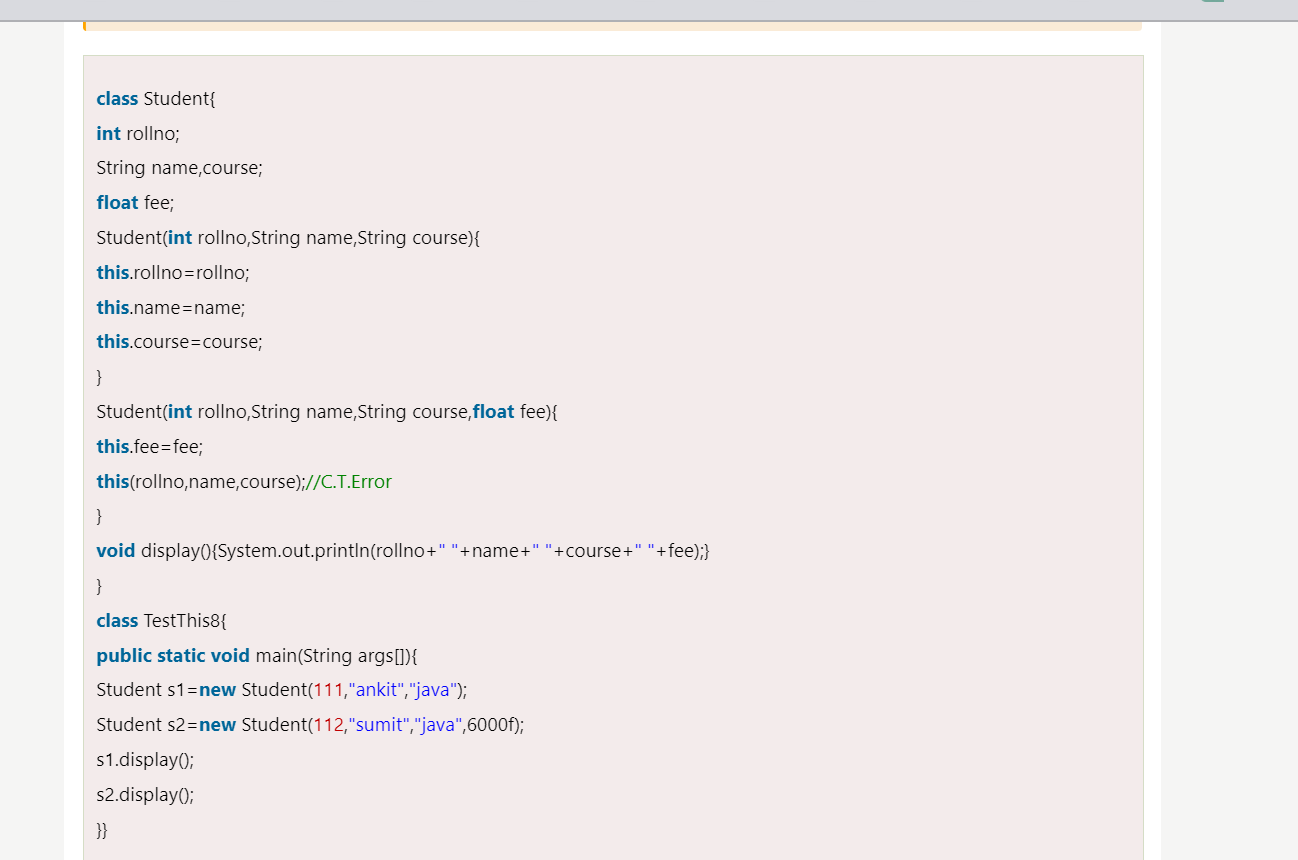
### Real usage of this() constructor call

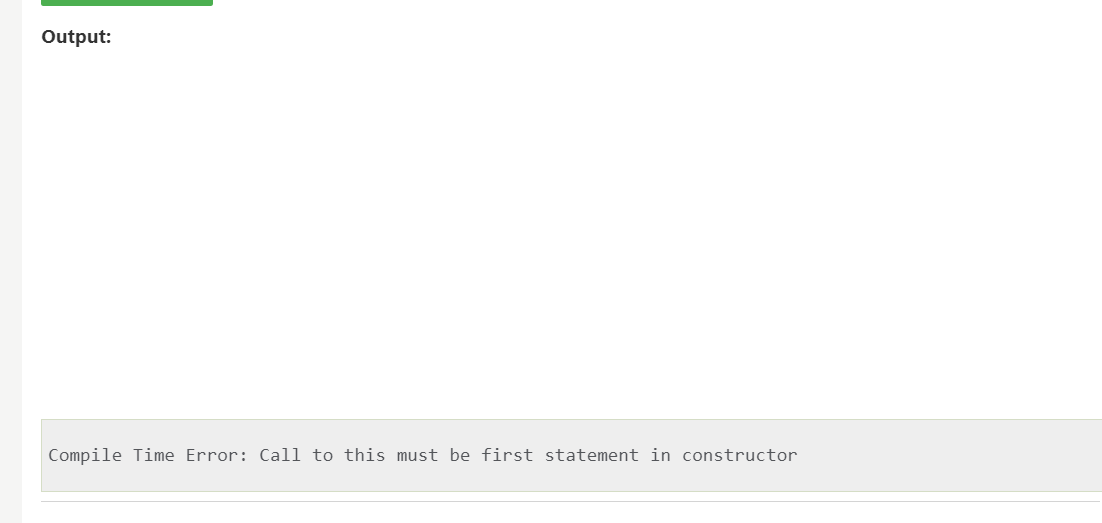
* The this() constructor call should be used to reuse the constructor from the constructor. It maintains the chain between the constructors i.e. it is used for constructor chaining. Let's see the example given below that displays the actual use of this keyword.



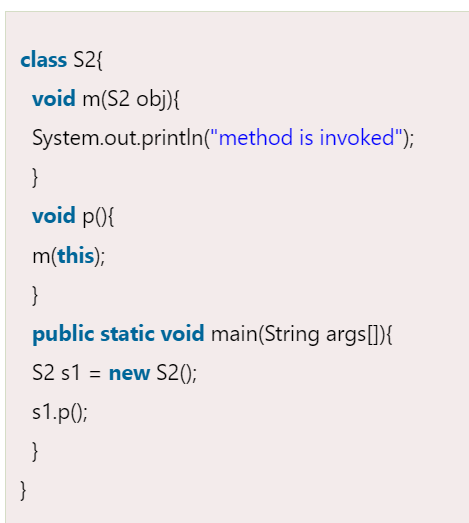


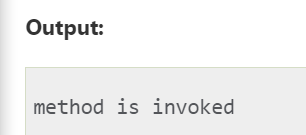
**Rule: Call to this() must be the first statement in constructor.**





### 4) this: to pass as an argument in the method

* The this keyword can also be passed as an argument in the method. It is mainly used in the event handling. Let's see the example:
* 

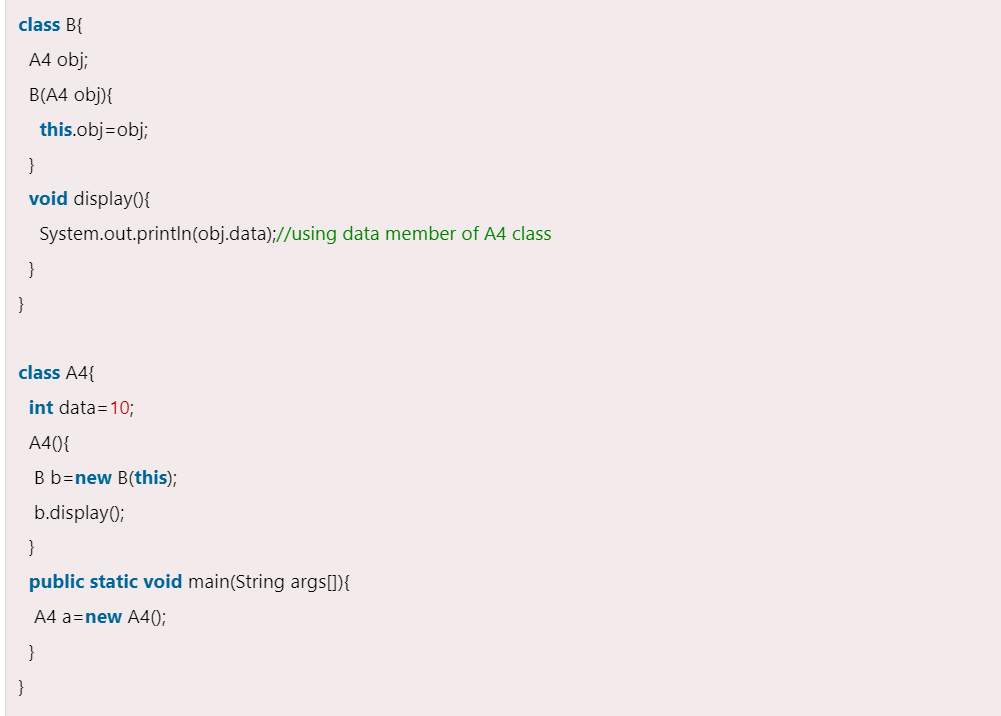


### Application of this that can be passed as an argument:

In event handling (or) in a situation where we have to provide reference of a class to another one. It is used to reuse one object in many methods.

### 5) this: to pass as argument in the constructor call

We can pass the this keyword in the constructor also. It is useful if we have to use one object in multiple classes



Output:10

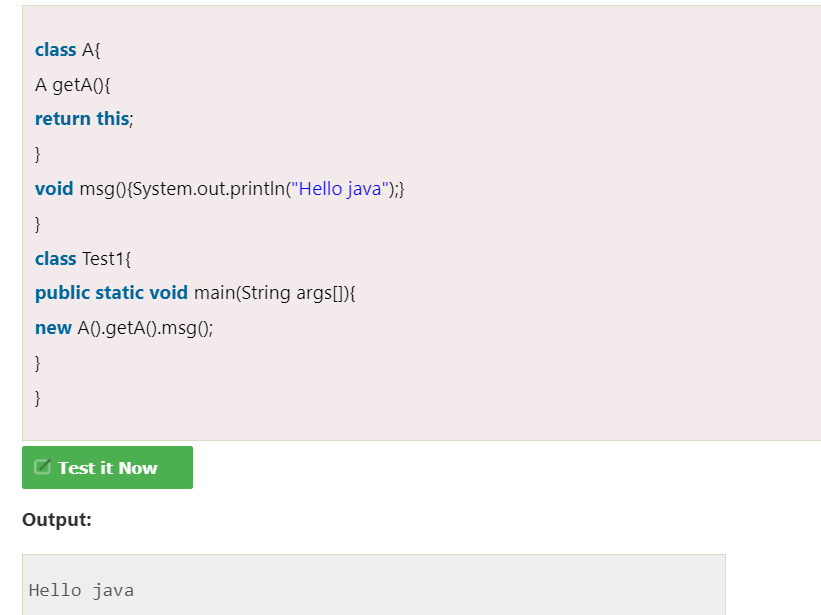
### 6) this keyword can be used to return current class instance

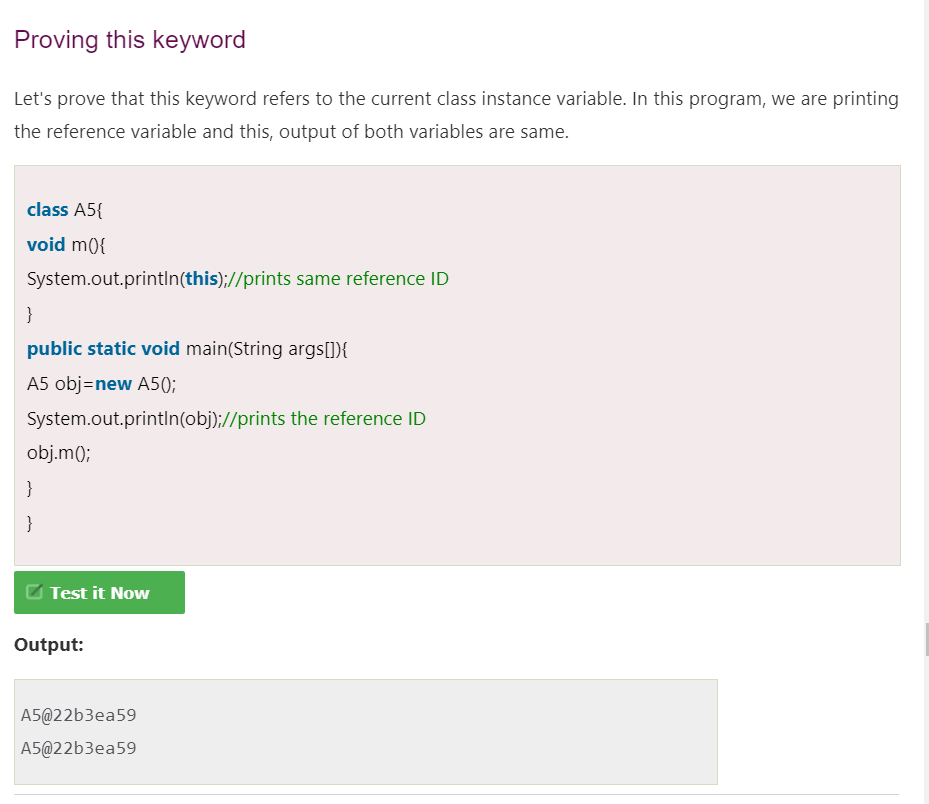
We can return this keyword as an statement from the method. In such case, return type of the method must be the class type (non-primitive). Let's see the example:

### Syntax of this that can be returned as a statement

1. return\_type method\_name(){
2. **return** **this**;
3. }

### Example of this keyword that you return as a statement from the method





## **The finalize( ) Method**

* It is possible to define a method that will be called just before an object's final destruction by the garbage collector. This method is called finalize( ), and it can be used to ensure that an object terminates cleanly.
* For example, you might use finalize( ) to make sure that an open file owned by that object is closed.
* To add a finalizer to a class, you simply define the finalize( ) method. The Java runtime calls that method whenever it is about to recycle an object of that class.
* Inside the finalize( ) method, you will specify those actions that must be performed before an object is destroyed.



* Here, the keyword protected is a specifier that prevents access to finalize( ) by code defined outside its class.
* This means that you cannot know when or even if finalize( ) will be executed. For example, if your program ends before garbage collection occurs, finalize( ) will not execute.