**Java**

***Inheritance***

Inheritance

* Same inheritance concept of C++ in Java with some modifications
* In Java,

– One class inherits the other using ***extends*** keyword

– The classes involved in inheritance are known as ***superclass*** and ***subclass***

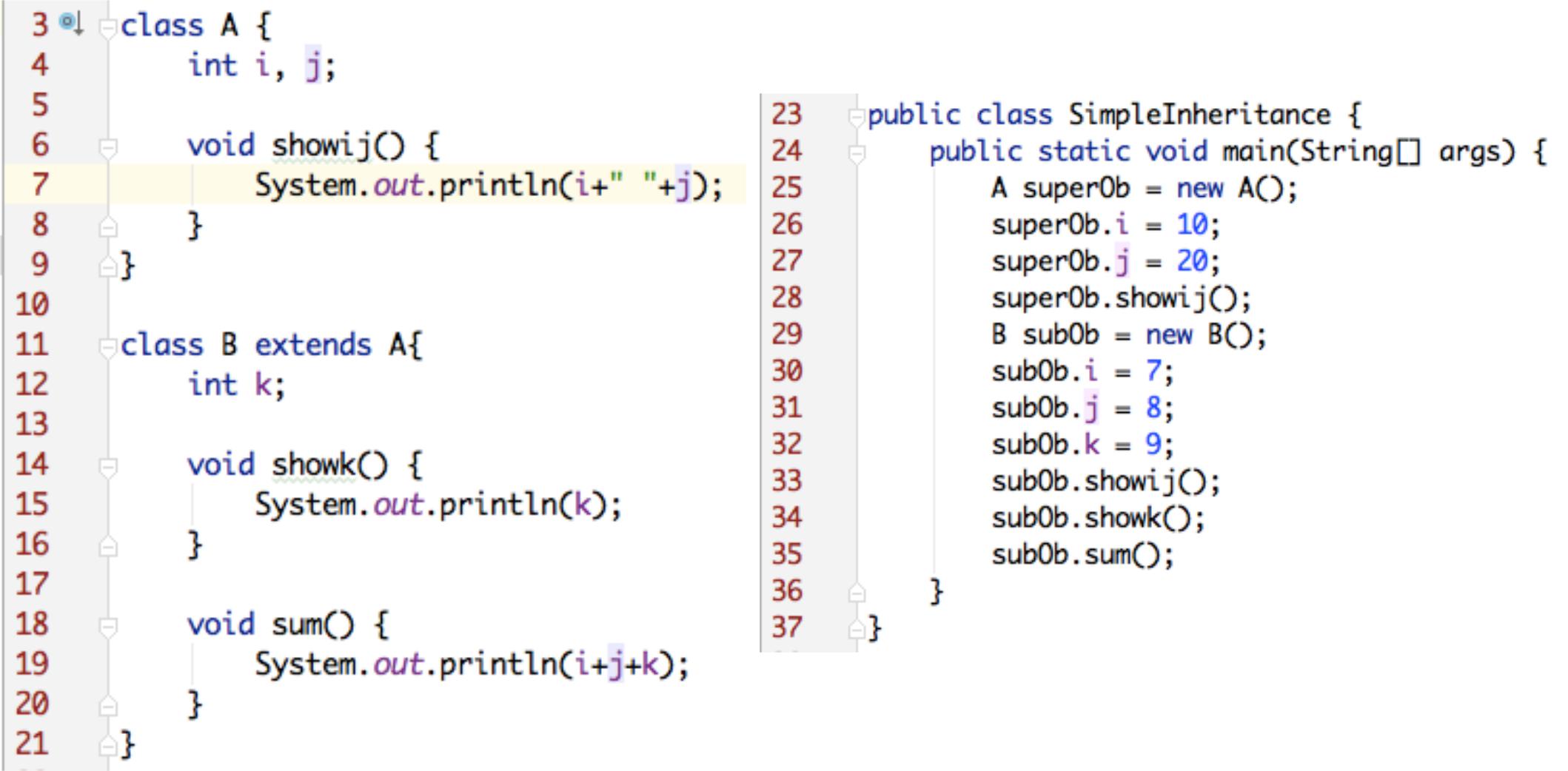
– ***Multilevel*** inheritance but no ***multiple*** inheritance

– There is a special way to call the superclass’s ***constructor***

– There is automatic ***dynamic method dispatch***

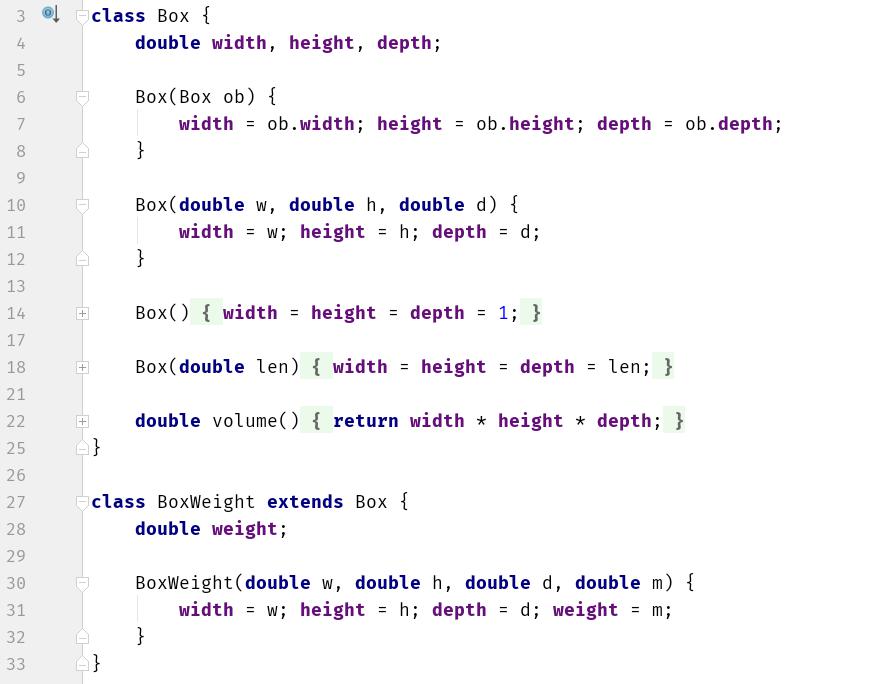
Prepared By - Rifat Shahriyar 2

Simple Inheritance



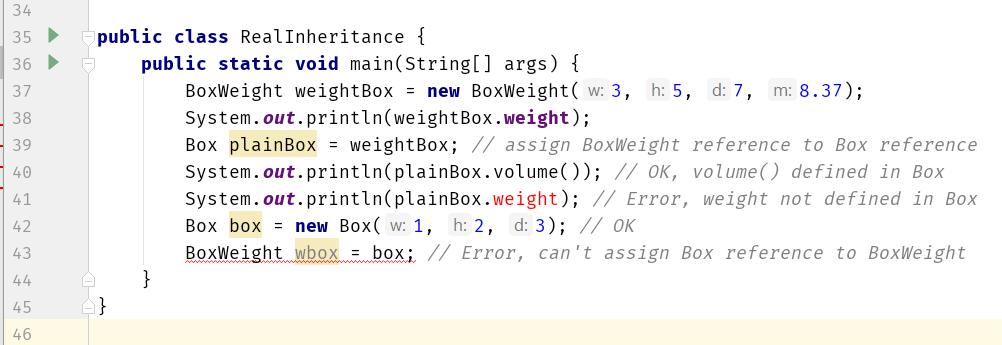
Prepared By - Rifat Shahriyar 3

Practical Example



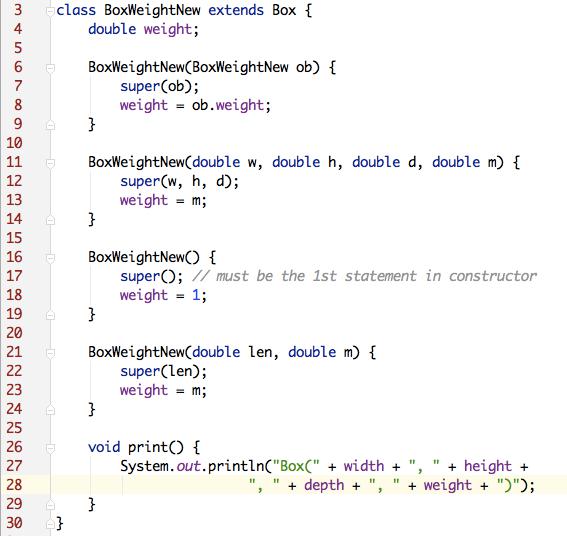
Prepared By - Rifat Shahriyar 4

Superclass variable reference to Subclass object



Prepared By - Rifat Shahriyar 5

Using super



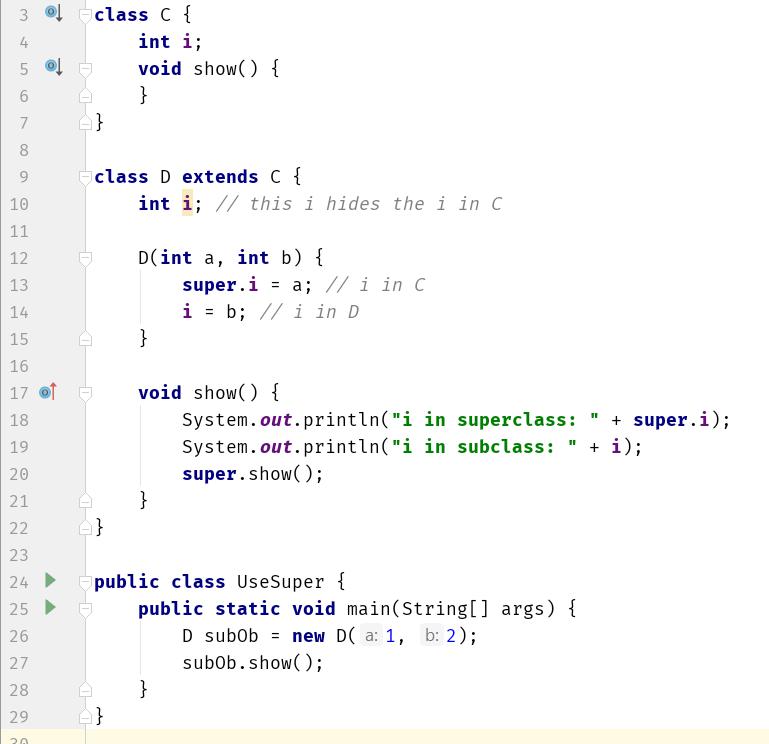
Prepared By - Rifat Shahriyar 6

Using super



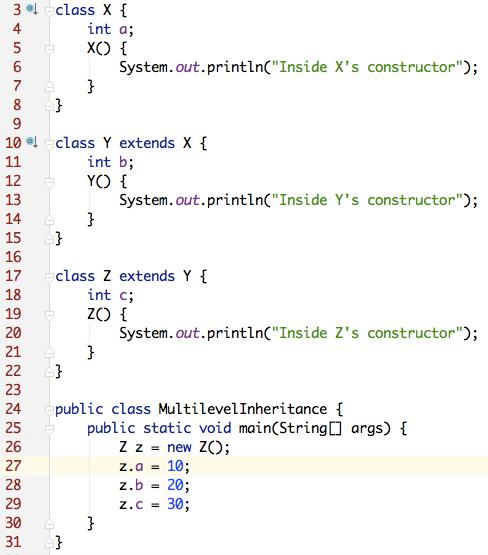
Prepared By - Rifat Shahriyar 7

Using super



Prepared By - Rifat Shahriyar 8

Multilevel Inheritance



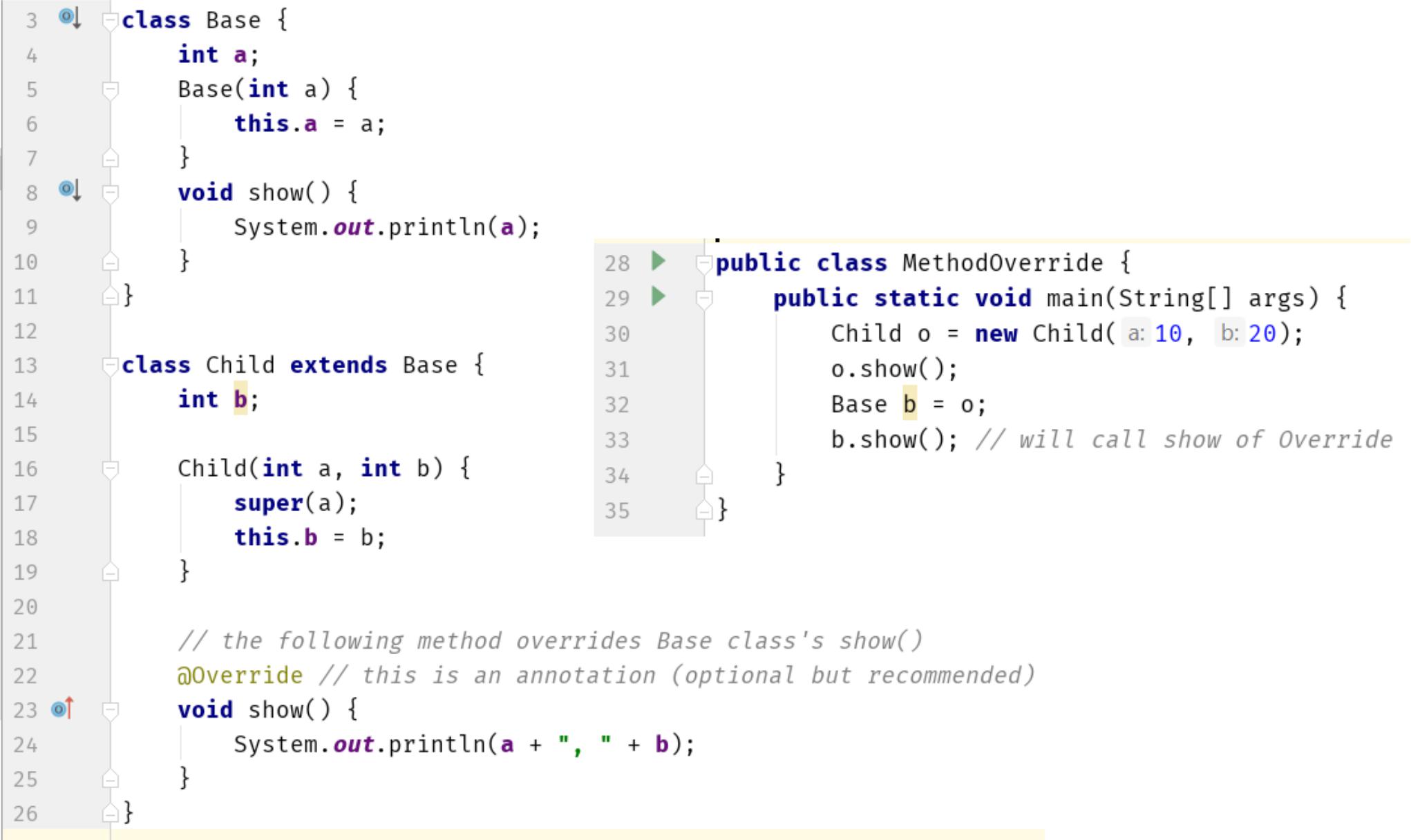
**Inside X's constructor**

**Inside Y's constructor**

**Inside Z's constructor**

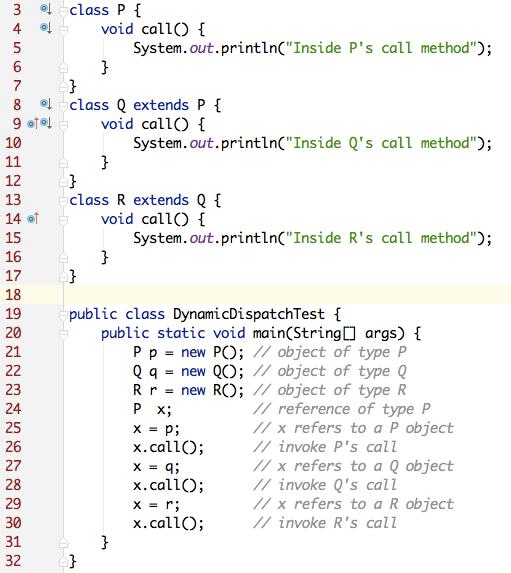
Prepared By - Rifat Shahriyar 9

Method Overriding



Prepared By - Rifat Shahriyar 10

Dynamic Method Dispatch



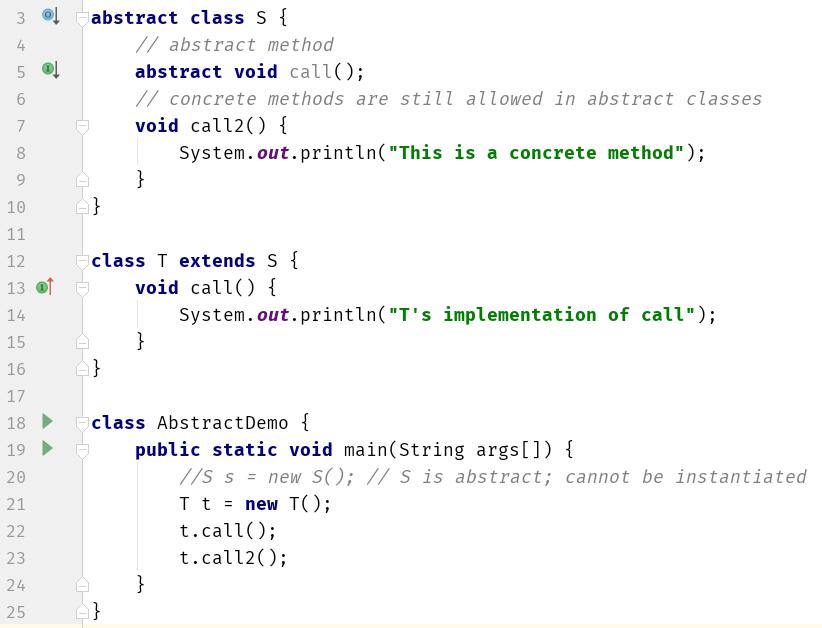
Prepared By - Rifat Shahriyar 11

Abstract Class

* ***abstract class A***
* contains abstract method ***abstract method f()***
* No instance can be created of an abstract class
* The subclass must implement the abstract method
* Otherwise the subclass will be a abstract class too

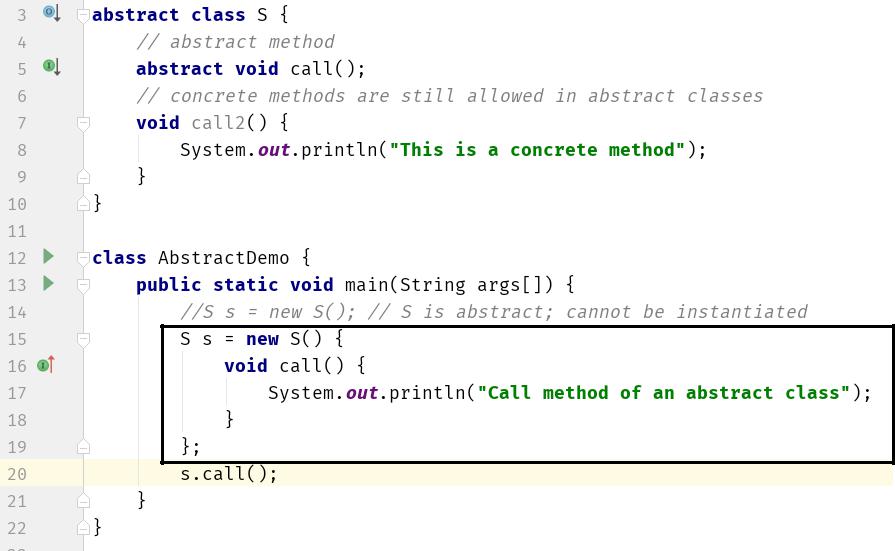
Prepared By - Rifat Shahriyar 12

Abstract Class



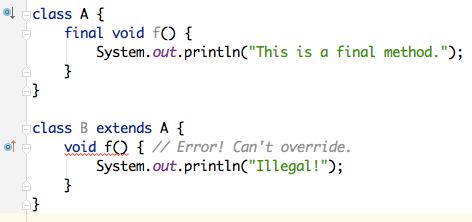
Prepared By - Rifat Shahriyar 13

Anonymous Subclass

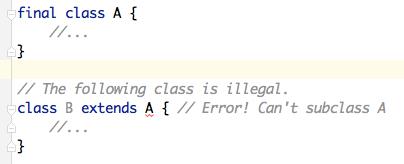


Prepared By - Rifat Shahriyar 14

Using final with Inheritance



***To prevent overriding***



***To prevent inheritance***

Prepared By - Rifat Shahriyar 15

Object Class

* There is one special class, ***Object***, defined by Java
* All other classes are subclasses of Object
* That is, Object is a superclass of all other classes
* This means that a reference variable of type Object can refer to an object of any other class
* Also, since arrays are implemented as classes, a variable of type Object can also refer to any array

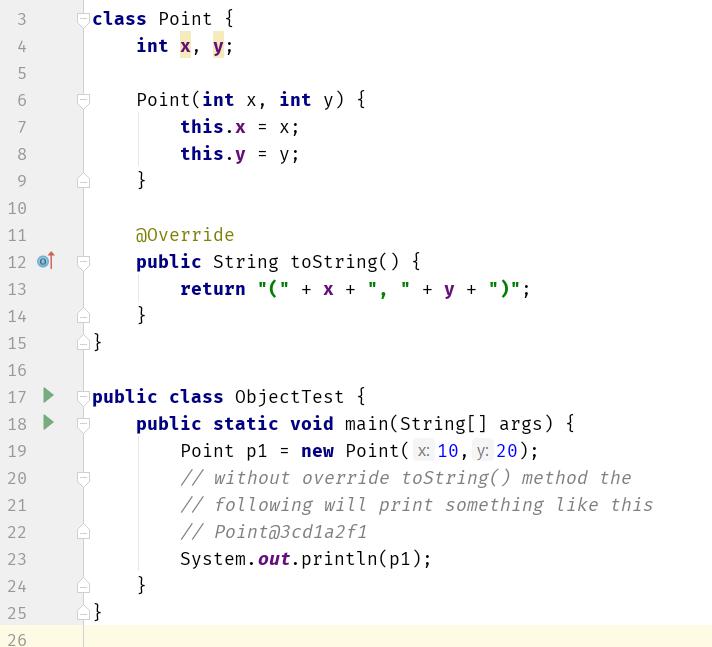
Prepared By - Rifat Shahriyar 16

Object’s toString()

* The **toString**( ) method returns a string that contains a description of the object on which it is called
* Also, this method is automatically called when an object is output using println()
* Many classes override this method
* Doing so allows them to provide a description specifically for the types of objects that they create

Prepared By - Rifat Shahriyar 17

Object’s toString()



Prepared By - Rifat Shahriyar 18

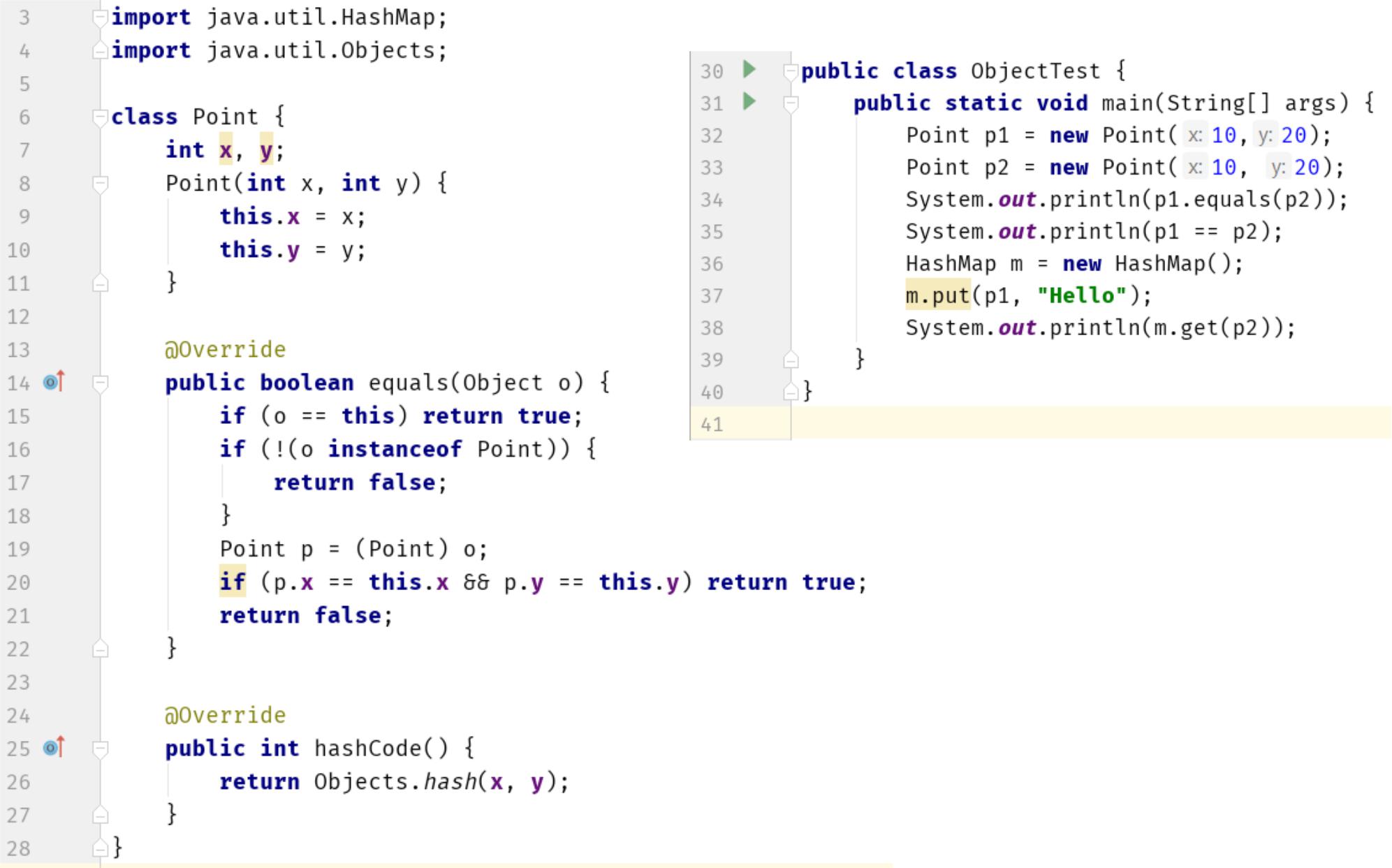
Object’s equals() and hashCode()

* **==** is a reference comparison, whether bothvariables refer to the same object
* Object’s **equals**() method does the same thing
* String class override **equals**() to check contents
* If you want two different objects of a same class to be equal then you need to override **equals**() and **hashCode**() methods

– **hashCode**() needs to return same value to work properly as keys in Hash data structures

Prepared By - Rifat Shahriyar 19

Object’s equals() and hashCode()



Prepared By - Rifat Shahriyar 20