Important tricks for Final by Riyadh 😊

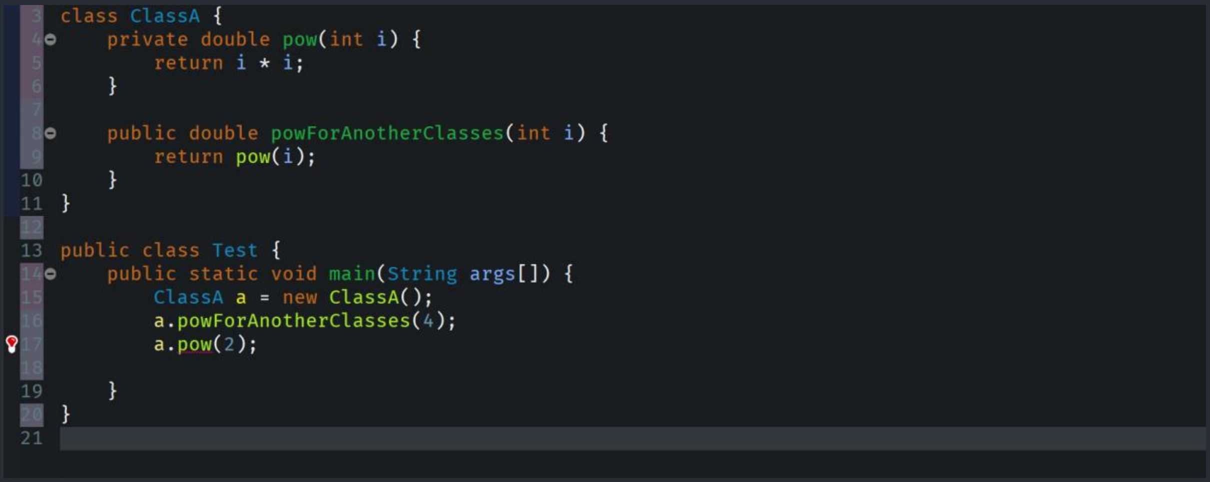
هذا الفايل يتتبع فايل اخر خاص بالميد
رابط الفايل :
https://drive.google.com/file/d/15rSCvG-xWayARa-BRpGLqcJRqkwXswIx/view?usp=sharing
شرح لفايل الميد :
https://www.youtube.com/watch?v=327IsporLEA&list=PLT10mtvjaTpbFl05HdQgDs0BZ2TJew1CB&index=2&t=9s
شرح لهذا الفايل :
https://youtu.be/wshSMzI0Xng

تجميعات سابقة محلولة مع الشرح:
حل اختبار نهائي 2018 -
https://www.youtube.com/watch?v=0-21Xb_9WIE&list=PLT10mtvjaTpbFl05HdQgDs0BZ2TJew1CB&index=4
حل اختبار نهائي 2019 -
https://www.youtube.com/watch?v=vJ99Nyov0KM&list=PLT10mtvjaTpbFl05HdQgDs0BZ2TJew1CB&index=5

Twitter : @ftcksu , @RLYADLN

Private : private methods or attributes can called only in the same class, impossible to call it outside the class.

Public : public methods or attributes it is the same as default, can called outside or inside the class with an “object” and it is impossible to call it without declaring an object.

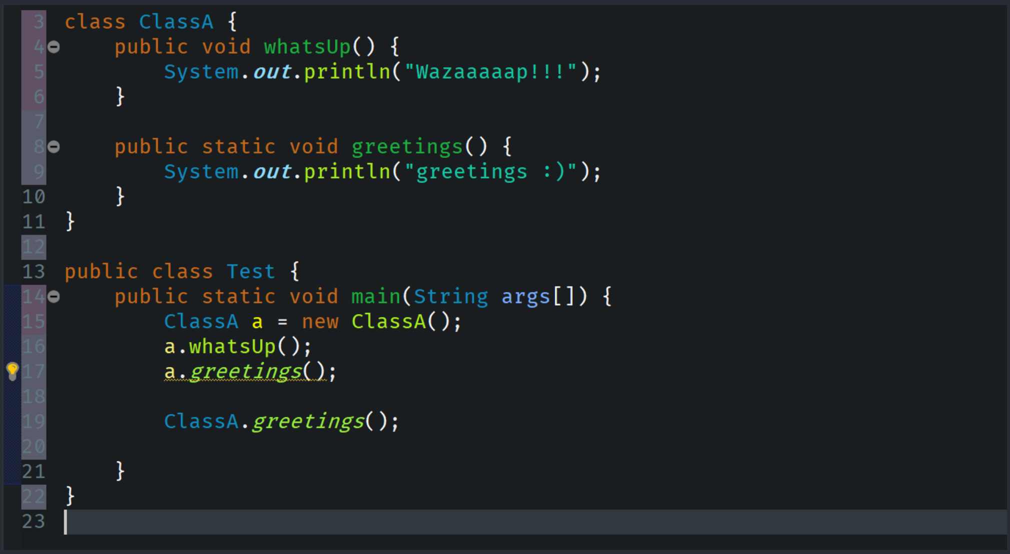


```
class ClassA {
    private double pow(int i) {
        return i * i;
    }

    public double powForAnotherClasses(int i) {
        return pow(i);
    }
}

public class Test {
    public static void main(String args[]) {
        ClassA a = new ClassA();
        a.powForAnotherClasses(4);
        a.pow(2);

    }
}
```

If we see here, the error because (pow) method is a private method for that we cannot call it outside the class, also there is a public method (powForAnotherClasses) that returning the private method so we can use it in another class.

Static : Statics methods or attributes have no affect to the object, can call outside inside the class without or with the object, also it cannot call any atrributes or methods that are not static

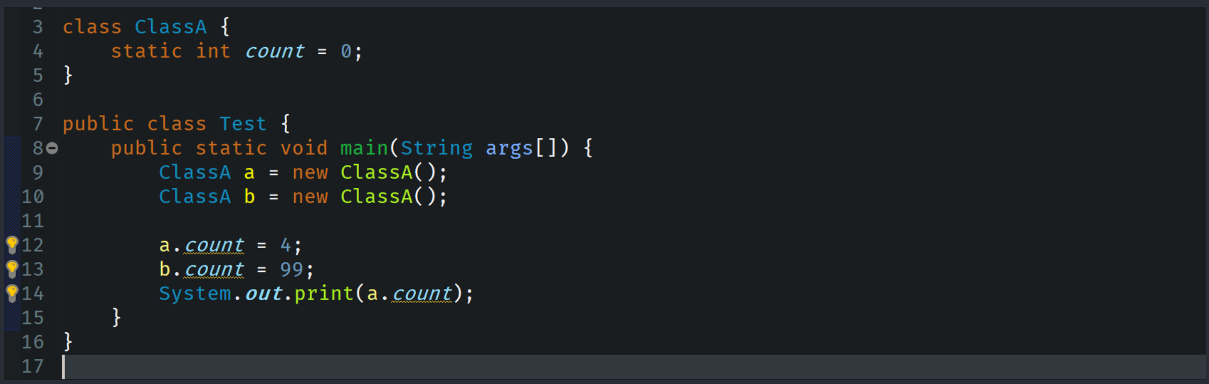
Twitter : @ftcksu , @RLYADLN



```java
class ClassA {
    public void whatsUp() {
        System.out.println("Wazaaaaap!!!");
    }

    public static void greetings() {
        System.out.println("greetings :)");
    }
}

public class Test {
    public static void main(String args[]) {
        ClassA a = new ClassA();
        a.whatsUp();
        a.greetings();

        ClassA.greetings();

    }
}
```

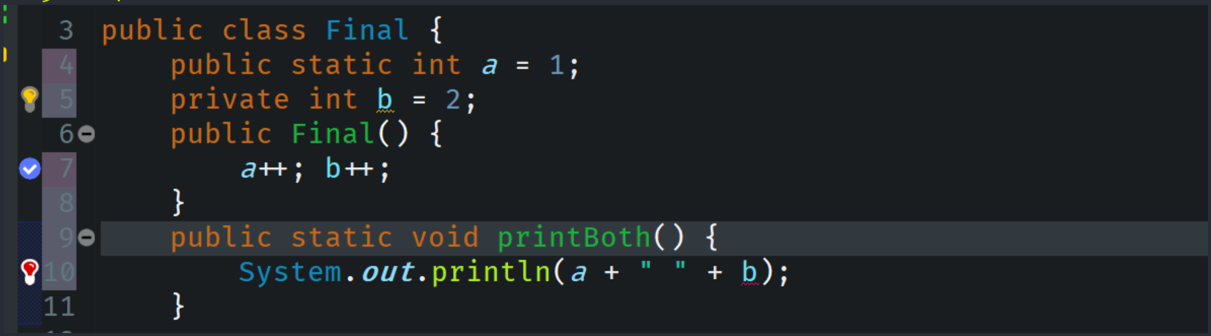
If we look closely we see two methods (whatsUp) and (greetings)
And these methods are public and static respectively.

In main we declared an object a from ClassA and we call the two methods, then we call the static method with the Class ~~not the object,~~ this is valid , but if we do it in the public method like (ClassA.whatsUp) it will become a compilation error



```java
class ClassA {
    static int count = 0;
}

public class Test {
    public static void main(String args[]) {
        ClassA a = new ClassA();
        ClassA b = new ClassA();

        a.count = 4;
        b.count = 99;
        System.out.print(a.count);
    }
}
```

Another helpful example to understand the static , the output will be 99 because the static variable or method have not affect in the object, but have affect in the class.



```java
 3 public class Final {
 4     public static int a = 1;
 5     private int b = 2;
 6     public Final() {
 7         a++; b++;
 8     }
 9     public static void printBoth() {
10         System.out.println(a + " " + b);
11     }
```

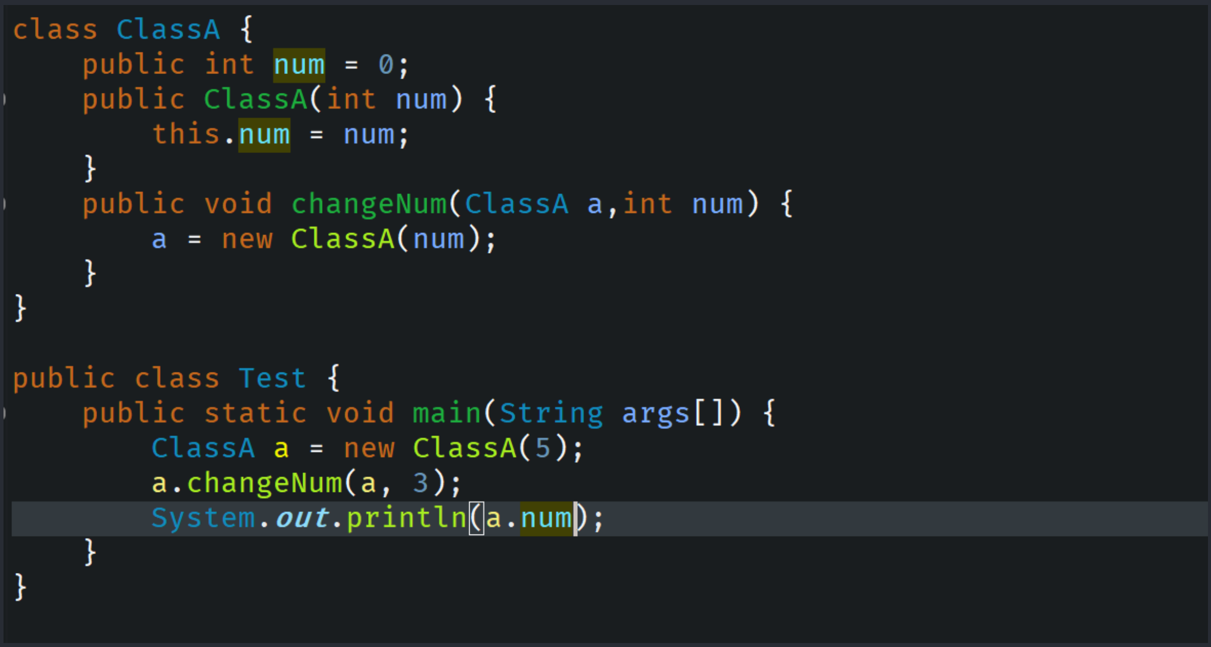
Error in line 10 , calling non-static variable in an static method

Twitter : @ftcksu , @RLYADLN

Method parameter :

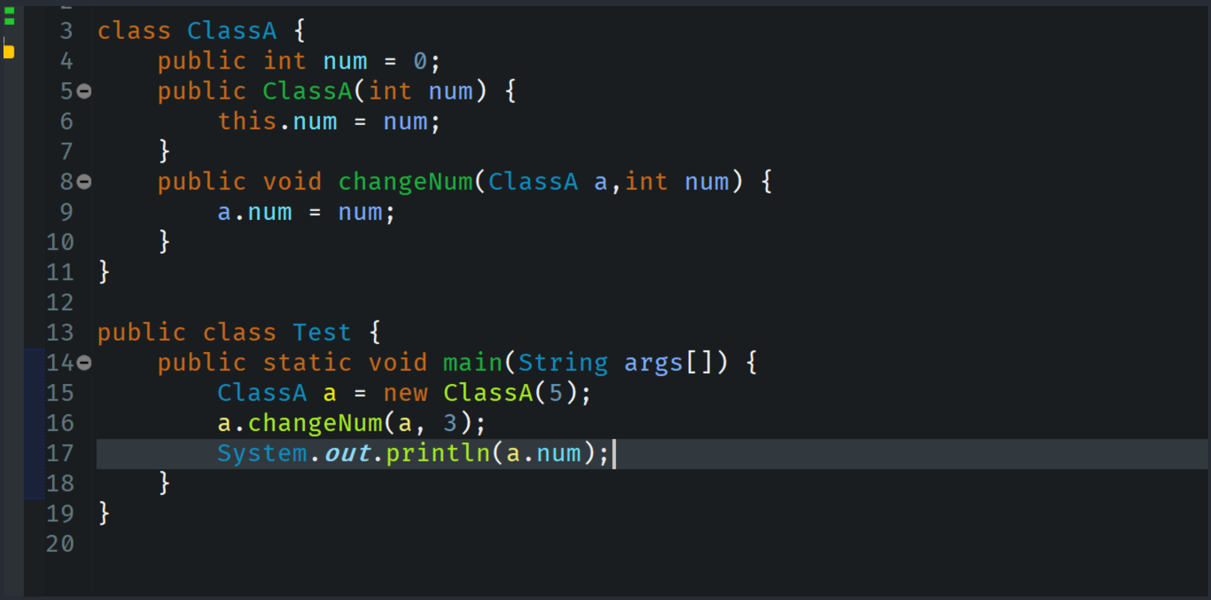
If we change the whole parameter in the method we will not get the change of it.

E.G:



```java
class ClassA {
    public int num = 0;
    public ClassA(int num) {
        this.num = num;
    }
    public void changeNum(ClassA a,int num) {
        a = new ClassA(num);
    }
}

public class Test {
    public static void main(String args[]) {
        ClassA a = new ClassA(5);
        a.changeNum(a, 3);
        System.out.println(a.num);
    }
}
```

OUTPUT = 5

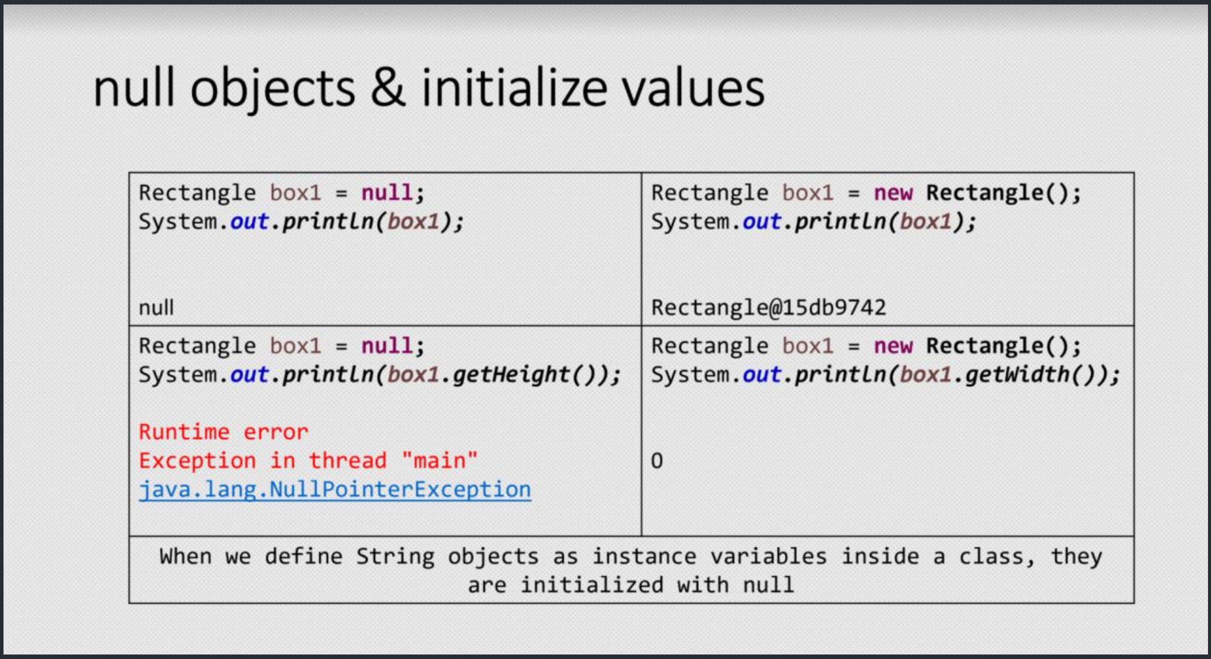


```java
class ClassA {
    public int num = 0;
    public ClassA(int num) {
        this.num = num;
    }
    public void changeNum(ClassA a,int num) {
        a.num = num;
    }
}

public class Test {
    public static void main(String args[]) {
        ClassA a = new ClassA(5);
        a.changeNum(a, 3);
        System.out.println(a.num);
    }
}
```

OUTPUT = 3

The diffrenece between the two codes, is in the first code we change the whole parameter, in the second code we change an attribute in the parameter.

Twitter : @ftcksu , @RLYADLN



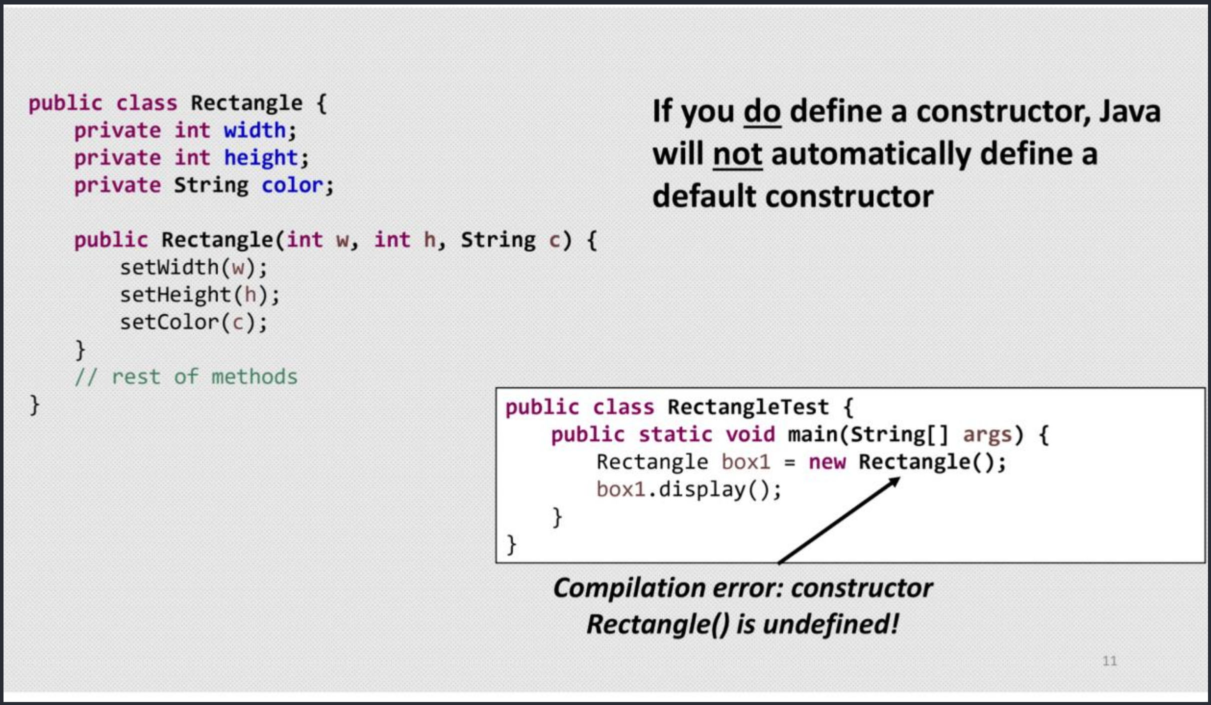
null objects & initialize values

`Rectangle box1 = null;`
`System.out.println(box1);`
 	
null
	`Rectangle box1 = new Rectangle();`
	`System.out.println(box1);`
	
	Rectangle@15db9742

`Rectangle box1 = null;`
`System.out.println(box1.getHeight());`
 	
Runtime error
Exception in thread "main"
java.lang.NullPointerException
	`Rectangle box1 = new Rectangle();`
	`System.out.println(box1.getWidth());`
	
	0

When we define String objects as instance variables inside a class, they are initialized with null	

Same with Strings



```
public class Rectangle {
    private int width;
    private int height;
    private String color;

    public Rectangle(int w, int h, String c) {
        setWidth(w);
        setHeight(h);
        setColor(c);
    }
    // rest of methods
}
```

If you do define a constructor, Java will not automatically define a default constructor

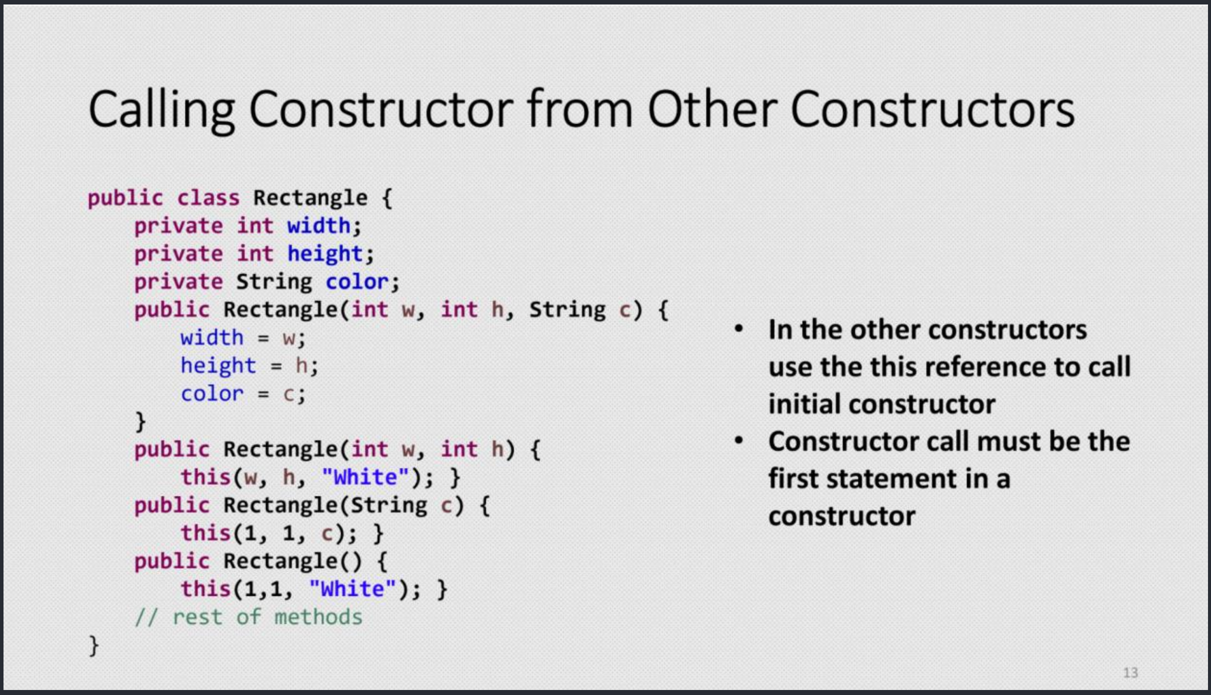
```
public class RectangleTest {
    public static void main(String[] args) {
        Rectangle box1 = new Rectangle();
        box1.display();
    }
}
```

Compilation error: constructor Rectangle() is undefined!

11

Twitter : @ftcksu , @RLYADLN

CALLING ANOTHER CONSTRUCTER FROM ANOTHER CONSTRUCTER PROBLEMS:



Calling Constructor from Other Constructors

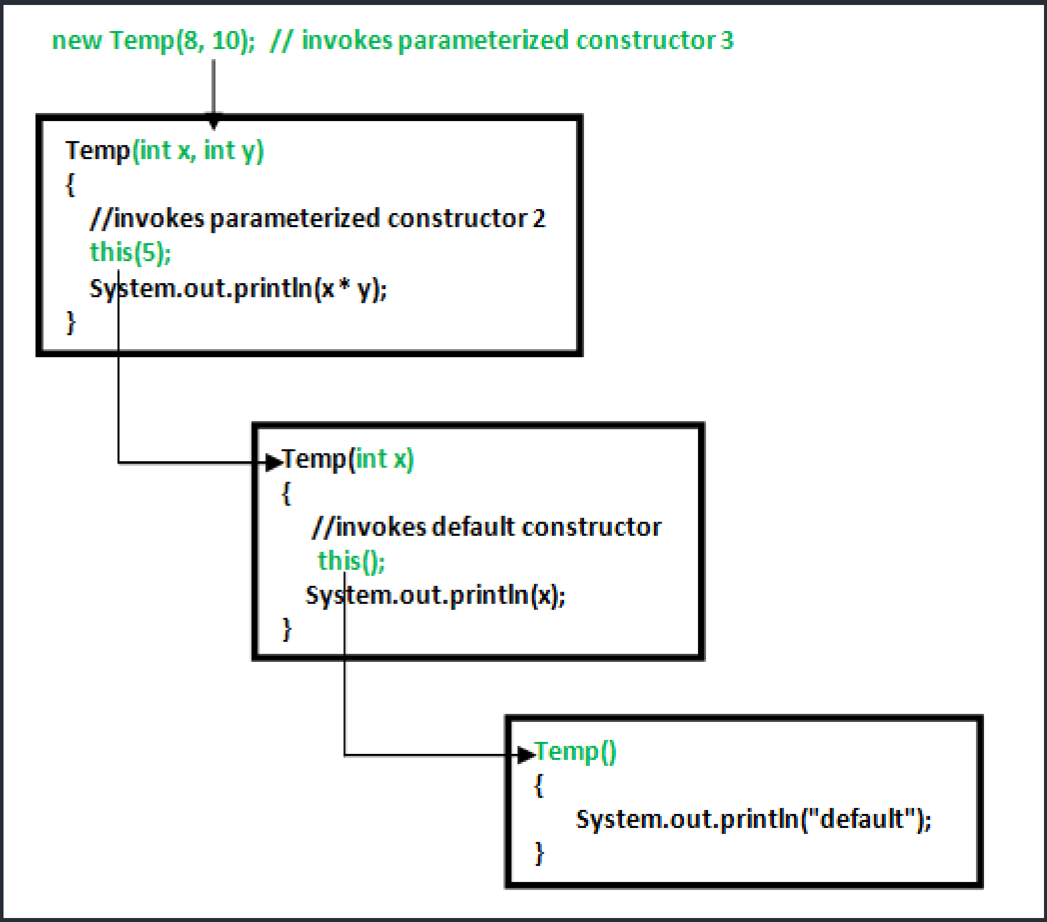
```
public class Rectangle {
    private int width;
    private int height;
    private String color;
    public Rectangle(int w, int h, String c) {
        width = w;
        height = h;
        color = c;
    }
    public Rectangle(int w, int h) {
        this(w, h, "White"); }
    public Rectangle(String c) {
        this(1, 1, c); }
    public Rectangle() {
        this(1,1, "White"); }
    // rest of methods
}
```

- In the other constructors use the this reference to call initial constructor
- Constructor call must be the first statement in a constructor

13

Here if you want to call a constructer you must call it the first statement

Twitter : @ftcksu , @RLYADLN

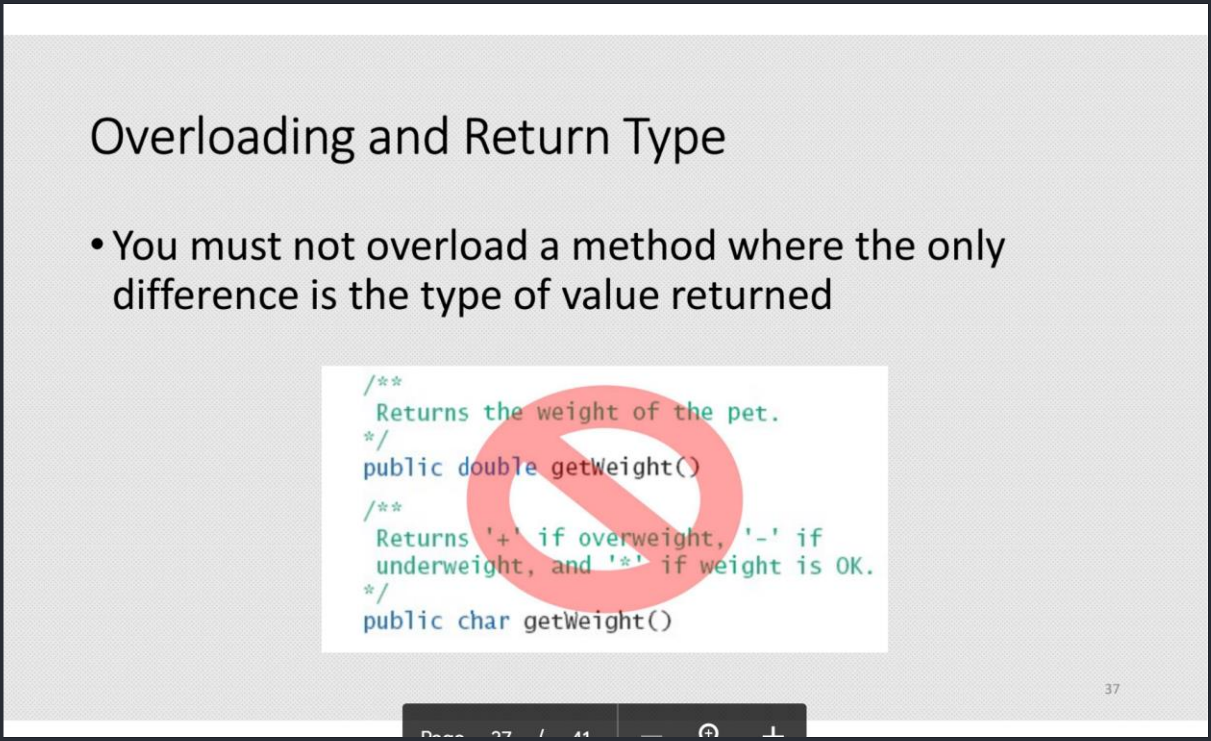


```
new Temp(8, 10);  // invokes parameterized constructor 3

Temp(int x, int y)
{
   //invokes parameterized constructor 2
   this(5);
   System.out.println(x * y);
}

Temp(int x)
{
   //invokes default constructor
   this();
   System.out.println(x);
}

Temp()
{
     System.out.println("default");
}
```

```java
// Java program to illustrate Constructor Chaining
// within same class Using this() keyword
class Temp
{
    // default constructor 1
    // default constructor will call another constructor
    // using this keyword from same class
    Temp()
    {
        // calls constructor 2
        this(5);
        System.out.println("The Default constructor");
    }

    // parameterized constructor 2
    Temp(int x)
    {
        // calls constructor 3
        this(5, 15);
        System.out.println(x);
    }

    // parameterized constructor 3
    Temp(int x, int y)
    {
        System.out.println(x * y);
    }

    public static void main(String args[])
    {
        // invokes default constructor first
        new Temp();
    }
}
```

```
OUTPUT = 75

5

The Default constructor
```

Method overload tricks :



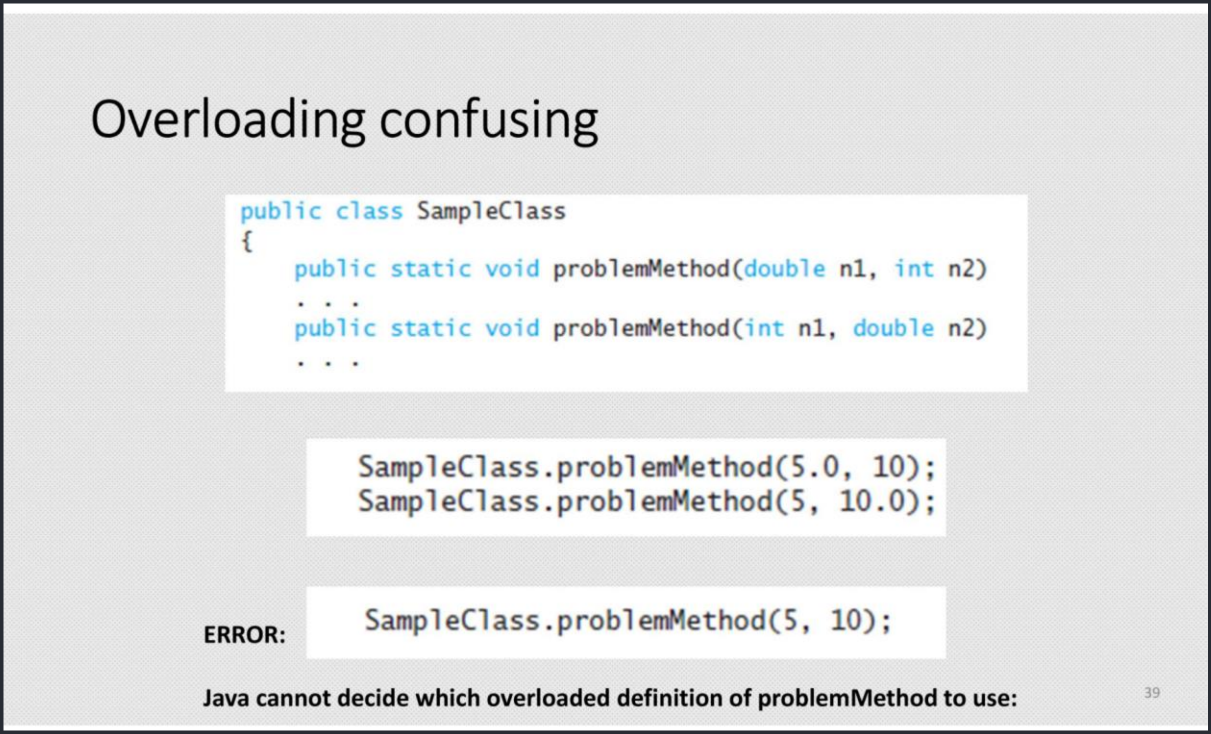
Overloading and Return Type

- You must not overload a method where the only difference is the type of value returned

```
/**
 Returns the weight of the pet.
*/
public double getWeight()

/**
 Returns '+' if overweight, '-' if
 underweight, and '*' if weight is OK.
*/
public char getWeight()
```

37

Compilation error



Overloading confusing

```
public class SampleClass
{
    public static void problemMethod(double n1, int n2)
    . . .
    public static void problemMethod(int n1, double n2)
    . . .
```

```
SampleClass.problemMethod(5.0, 10);
SampleClass.problemMethod(5, 10.0);
```

ERROR:

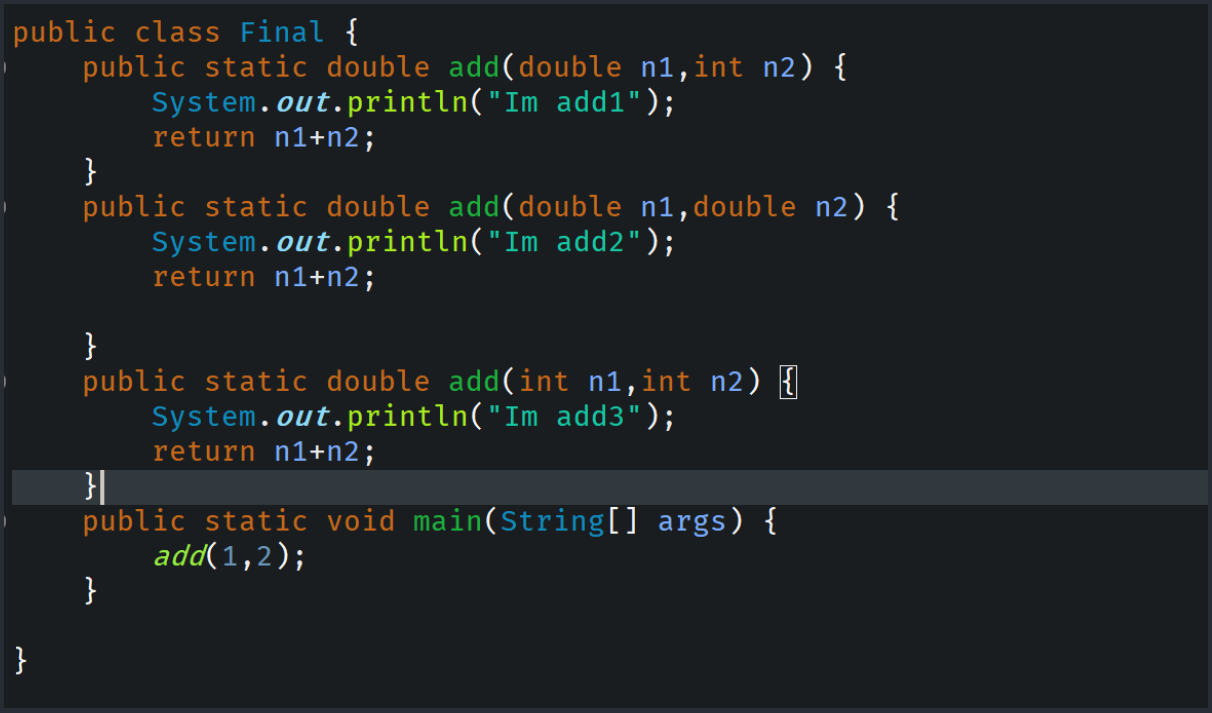
```
SampleClass.problemMethod(5, 10);
```

Java cannot decide which overloaded definition of problemMethod to use:

39

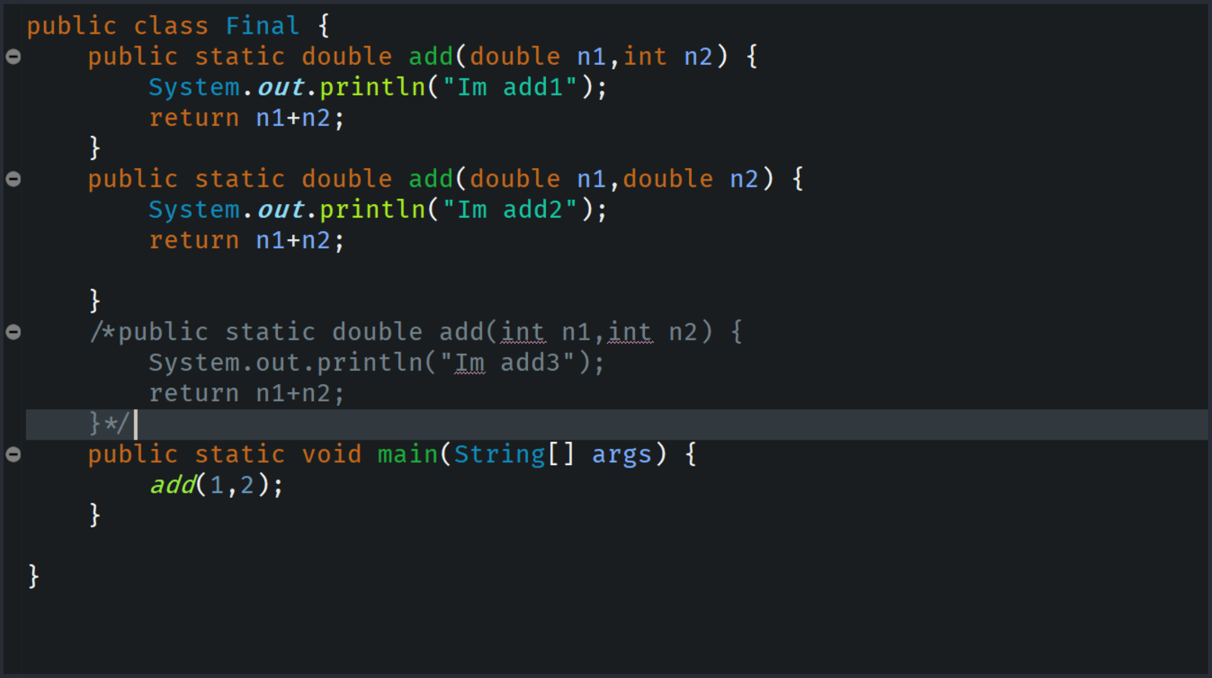
Twitter : @ftcksu , @RLYADLN

Method preferring parameters :



```java
public class Final {
    public static double add(double n1,int n2) {
        System.out.println("Im add1");
        return n1+n2;
    }
    public static double add(double n1,double n2) {
        System.out.println("Im add2");
        return n1+n2;

    }
    public static double add(int n1,int n2) {
        System.out.println("Im add3");
        return n1+n2;
    }
    public static void main(String[] args) {
        add(1,2);
    }

}
```

```
OUTPUT = Im add3
```

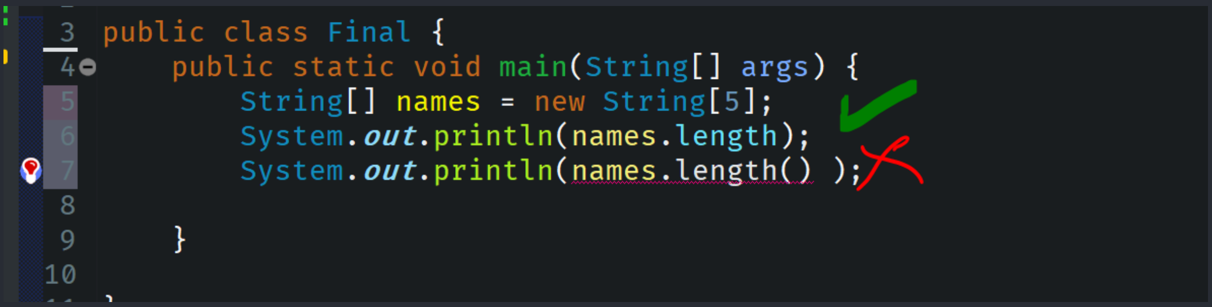


```java
public class Final {
    public static double add(double n1,int n2) {
        System.out.println("Im add1");
        return n1+n2;
    }
    public static double add(double n1,double n2) {
        System.out.println("Im add2");
        return n1+n2;

    }
    /*public static double add(int n1,int n2) {
        System.out.println("Im add3");
        return n1+n2;
    }*/
    public static void main(String[] args) {
        add(1,2);
    }

}
```

```
OUTPUT = Im add1
```

Twitter : @ftcksu , @RLYADLN

Length() vs length trick:



```java
public class Final {
    public static void main(String[] args) {
        String[] names = new String[5];
        System.out.println(names.length);
        System.out.println(names.length() );

    }

}
```

length is for arrays, length() is for Strings

Twitter : @ftcksu , @RLYADLN