Fundamentals of Object Oriented Programming

CSN- 103

Dr. R. Balasubramanian
Associate Professor
Department of Computer Science and Engineering
Indian Institute of Technology Roorkee
Roorkee 247 667
balarfcs@iitr.ac.in
https://sites.google.com/site/balaiitr/

Constructors

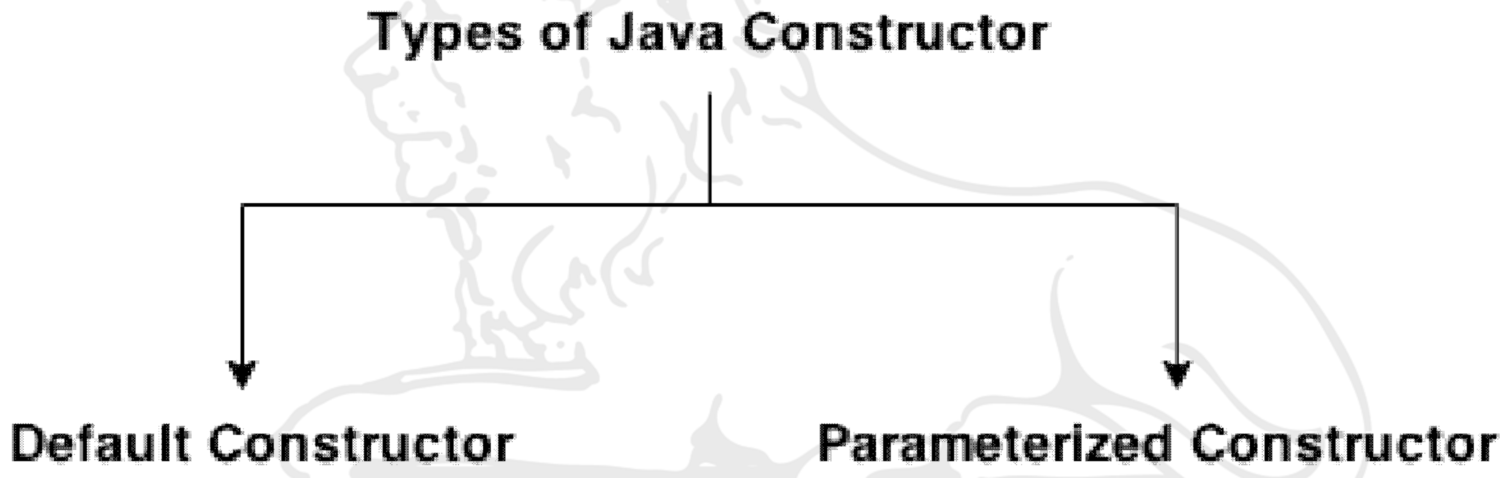
- **Constructor in java** is a *special type of method* that is used to initialize the object.
- Java constructor is *invoked at the time of object creation*. It constructs the values i.e. provides data for the object that is why it is known as constructor.

Rules for creating java constructor

- There are basically two rules defined for the constructor.
  – Constructor name must be same as its class name
  – Constructor must have no explicit return type

Types of java constructors

- There are two types of constructors:
  - Default constructor (no-arg constructor)
  - Parameterized constructor

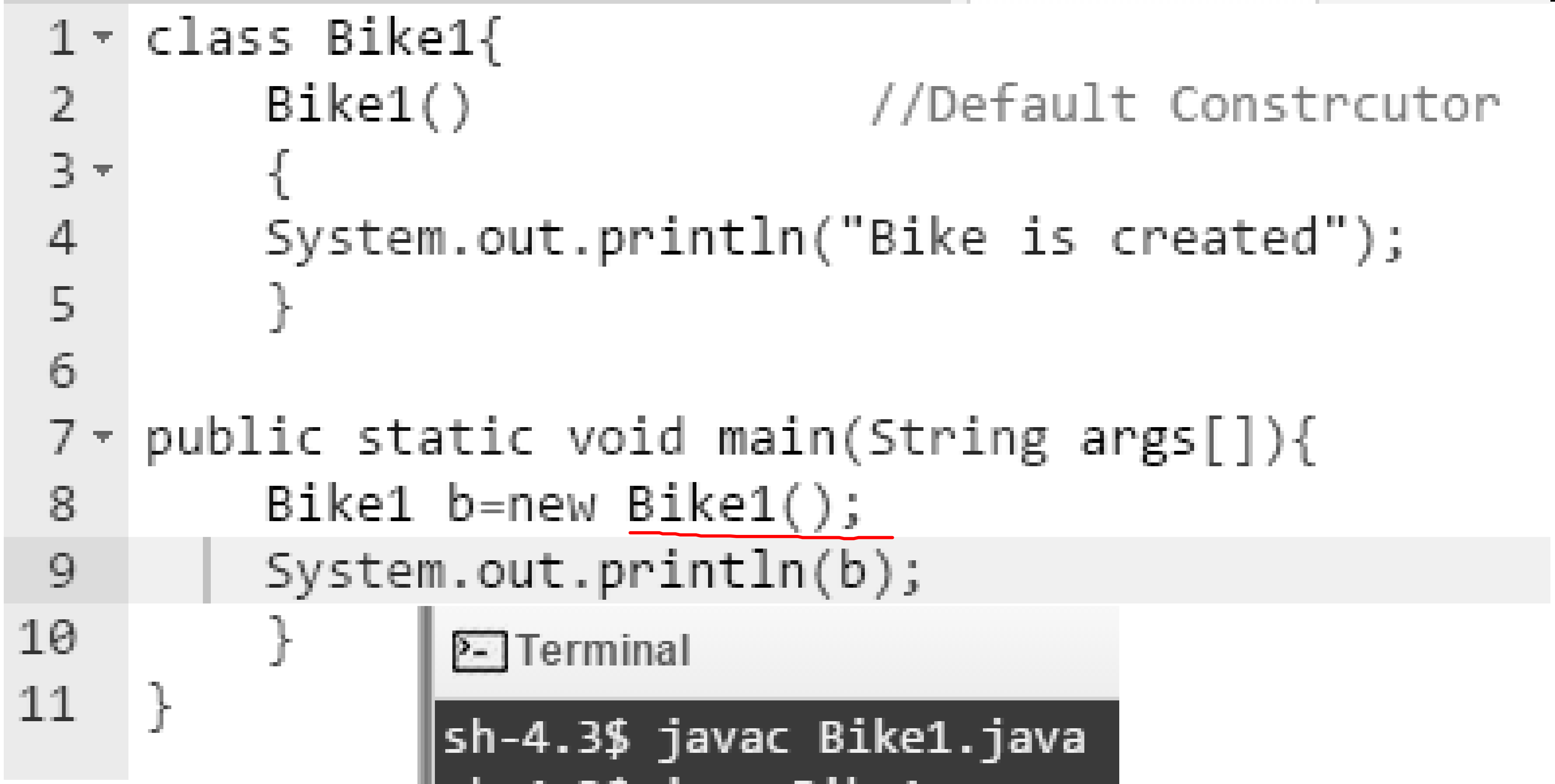


Java Default Constructor

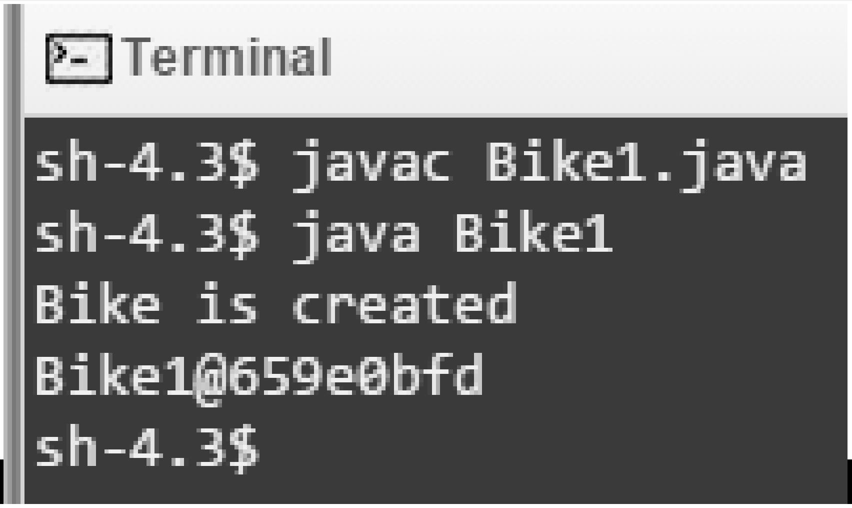
- Java Default Constructor
- A constructor that have no parameter is known as default constructor.
- **Syntax of default constructor:**

```
<class_name>(){}
```

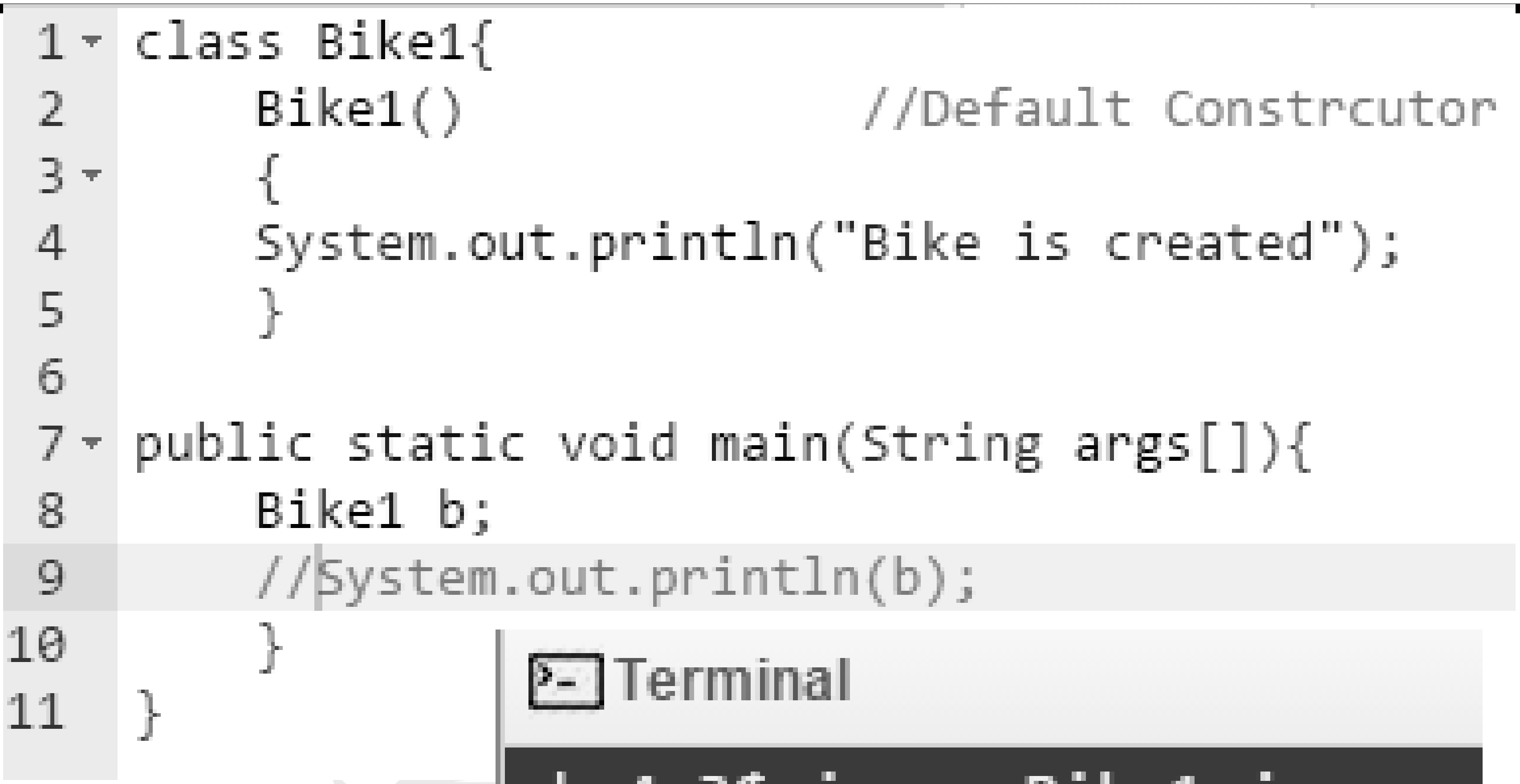
Example



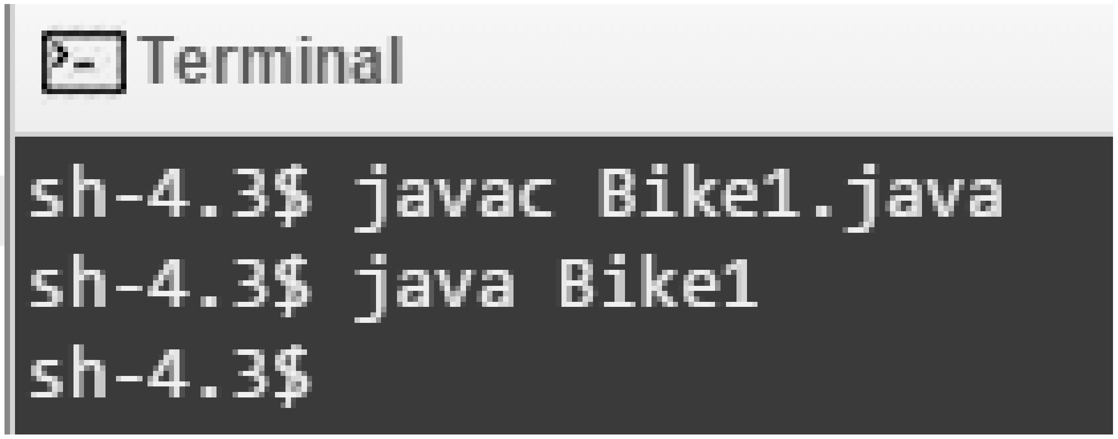
```java
class Bike1{
     Bike1()                     //Default Constrcutor
     {
     System.out.println("Bike is created");
     }

public static void main(String args[]){
     Bike1 b=new Bike1();
     System.out.println(b);
     }
}
```



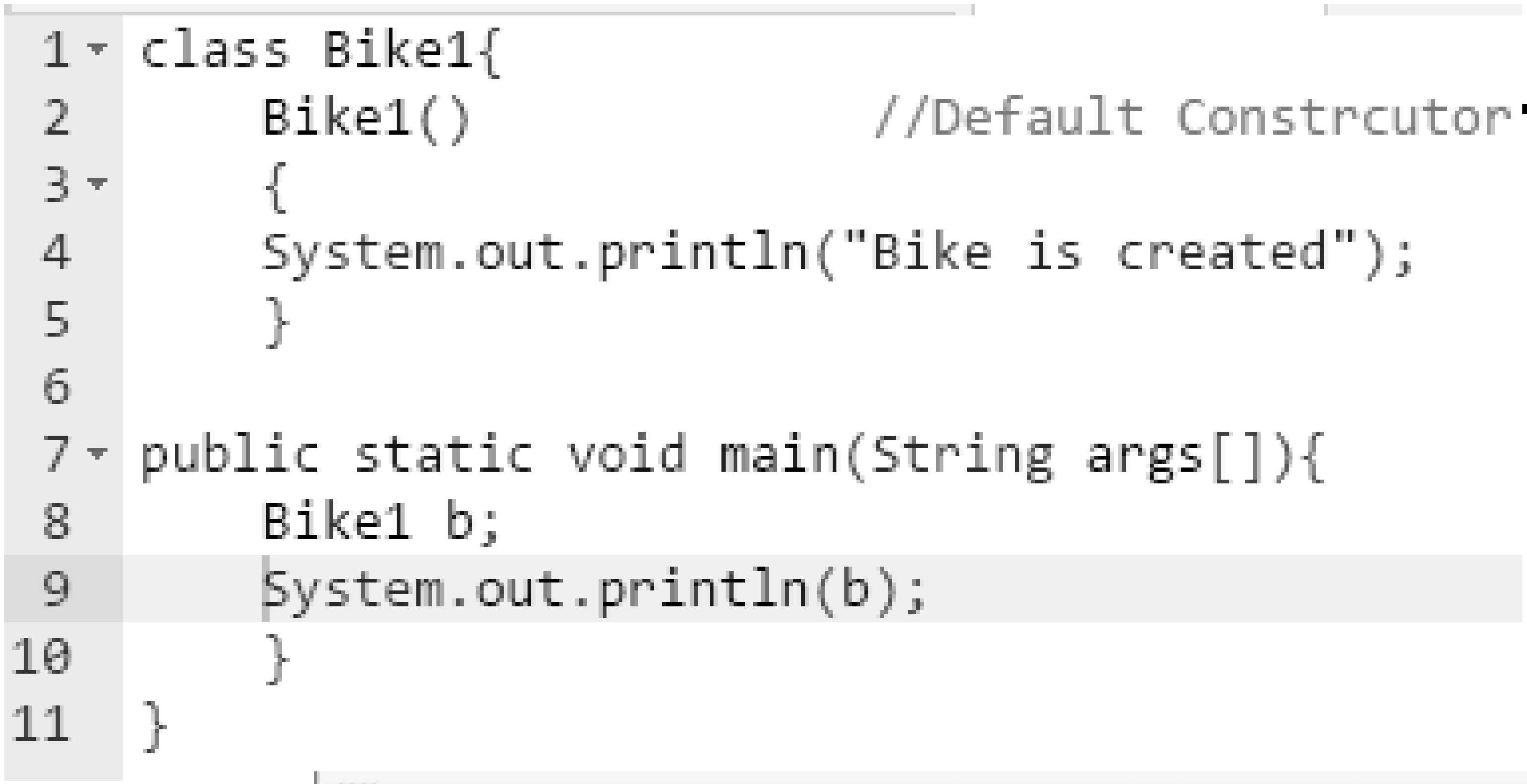
```
sh-4.3$ javac Bike1.java
sh-4.3$ java Bike1
Bike is created
Bike1@659e0bfd
sh-4.3$
```



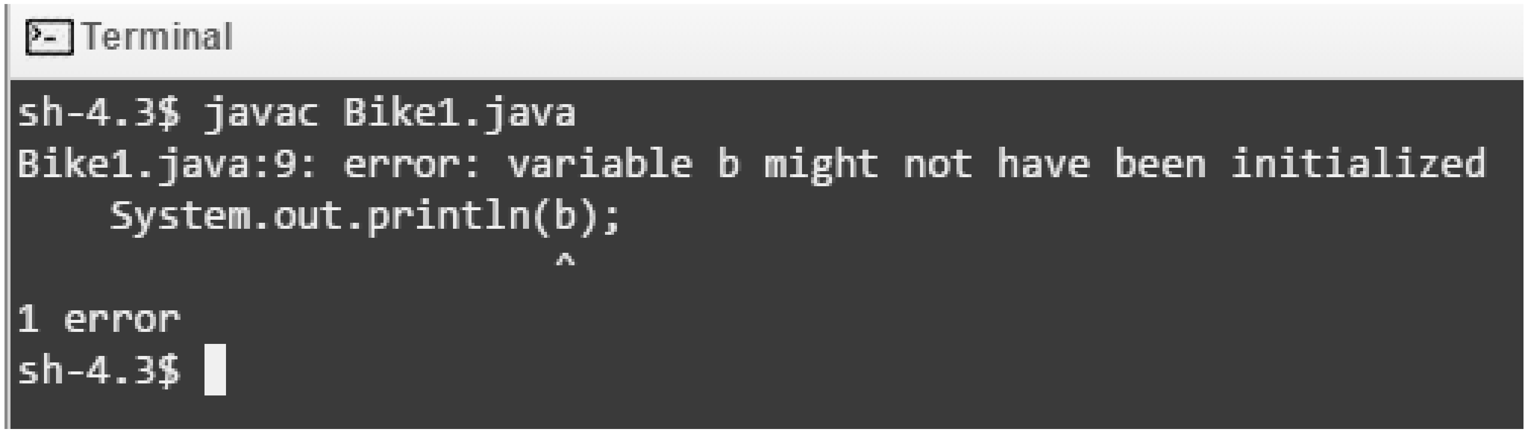
```java
class Bike1{
    Bike1()                     //Default Constrcutor
    {
    System.out.println("Bike is created");
    }

public static void main(String args[]){
    Bike1 b;
    //System.out.println(b);
    }
}
```



```
sh-4.3$ javac Bike1.java
sh-4.3$ java Bike1
sh-4.3$
```

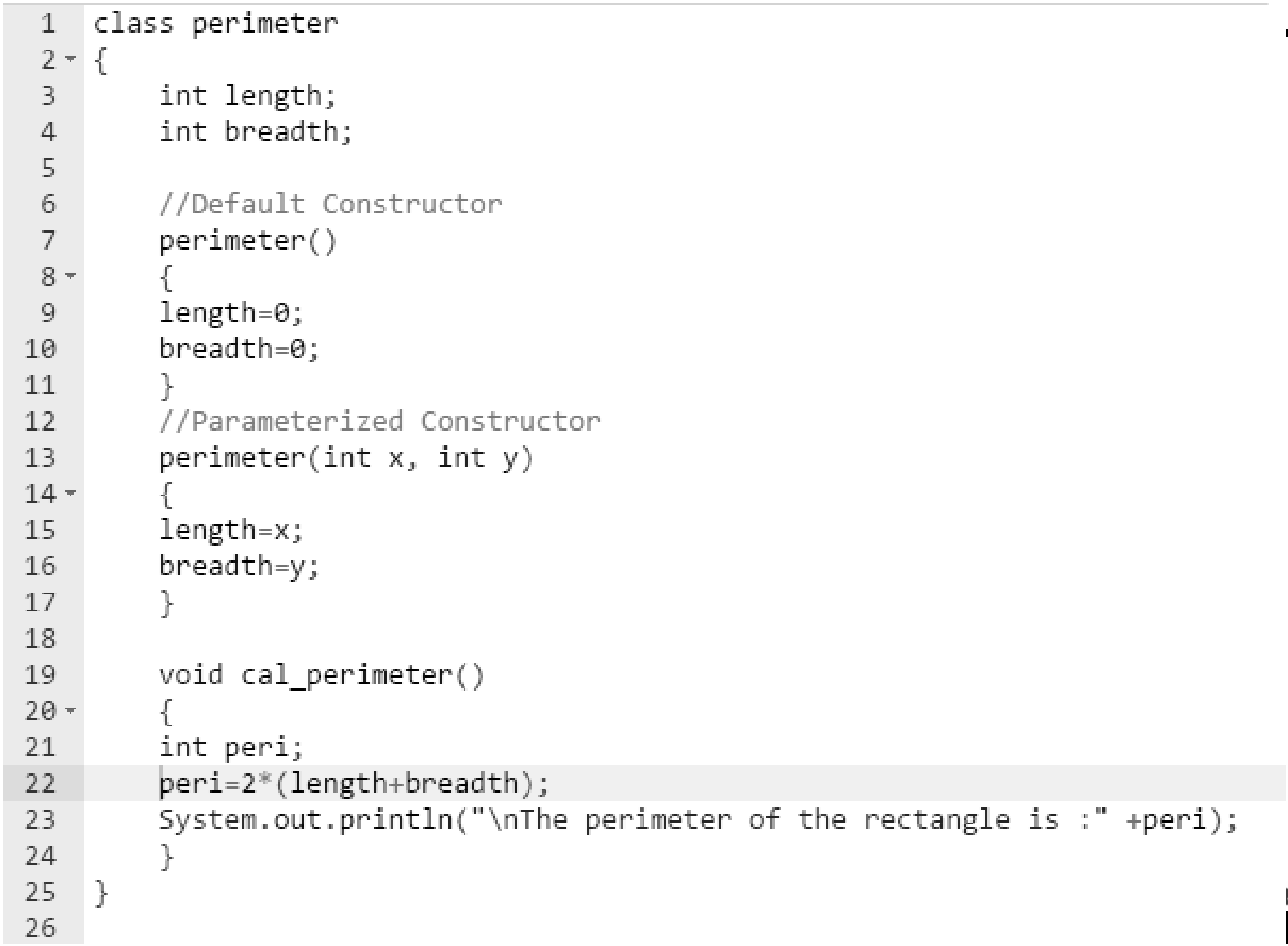


```java
class Bike1{
    Bike1()                 //Default Constrcutor
    {
    System.out.println("Bike is created");
    }

public static void main(String args[]){
    Bike1 b;
    System.out.println(b);
    }
}
```

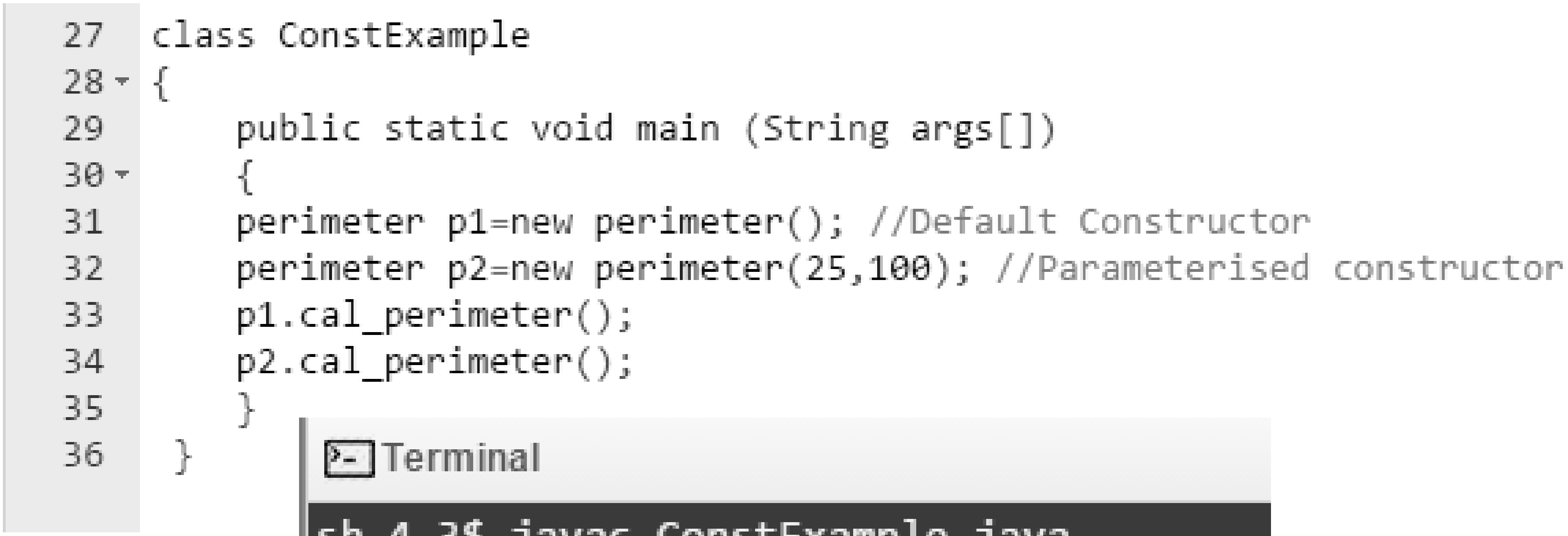


```
sh-4.3$ javac Bike1.java
Bike1.java:9: error: variable b might not have been initialized
    System.out.println(b);
                       ^
1 error
sh-4.3$
```

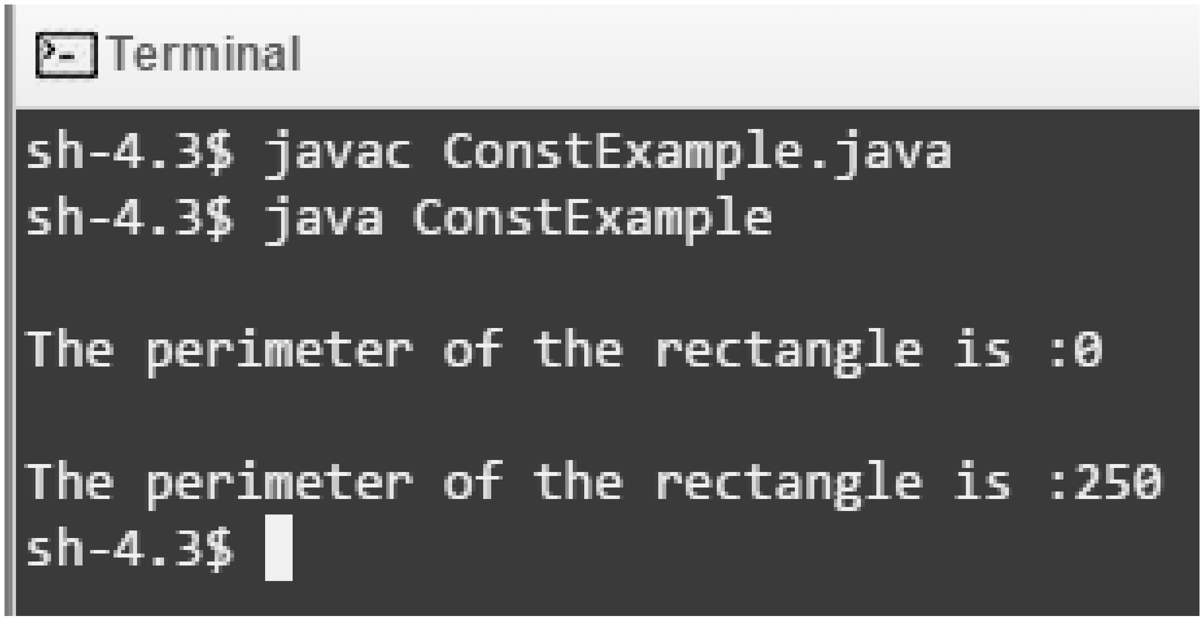
Default and Parameterized Constructor



```java
class perimeter
{
    int length;
    int breadth;

    //Default Constructor
    perimeter()
    {
    length=0;
    breadth=0;
    }
    //Parameterized Constructor
    perimeter(int x, int y)
    {
    length=x;
    breadth=y;
    }

    void cal_perimeter()
    {
    int peri;
    peri=2*(length+breadth);
    System.out.println("\nThe perimeter of the rectangle is :" +peri);
    }
}

```



```java
class ConstExample
{
    public static void main (String args[])
    {
    perimeter p1=new perimeter(); //Default Constructor
    perimeter p2=new perimeter(25,100); //Parameterised constructor
    p1.cal_perimeter();
    p2.cal_perimeter();
    }
 }
```



Terminal

```
sh-4.3$ javac ConstExample.java
sh-4.3$ java ConstExample

The perimeter of the rectangle is :0

The perimeter of the rectangle is :250
sh-4.3$
```

JAVA static word

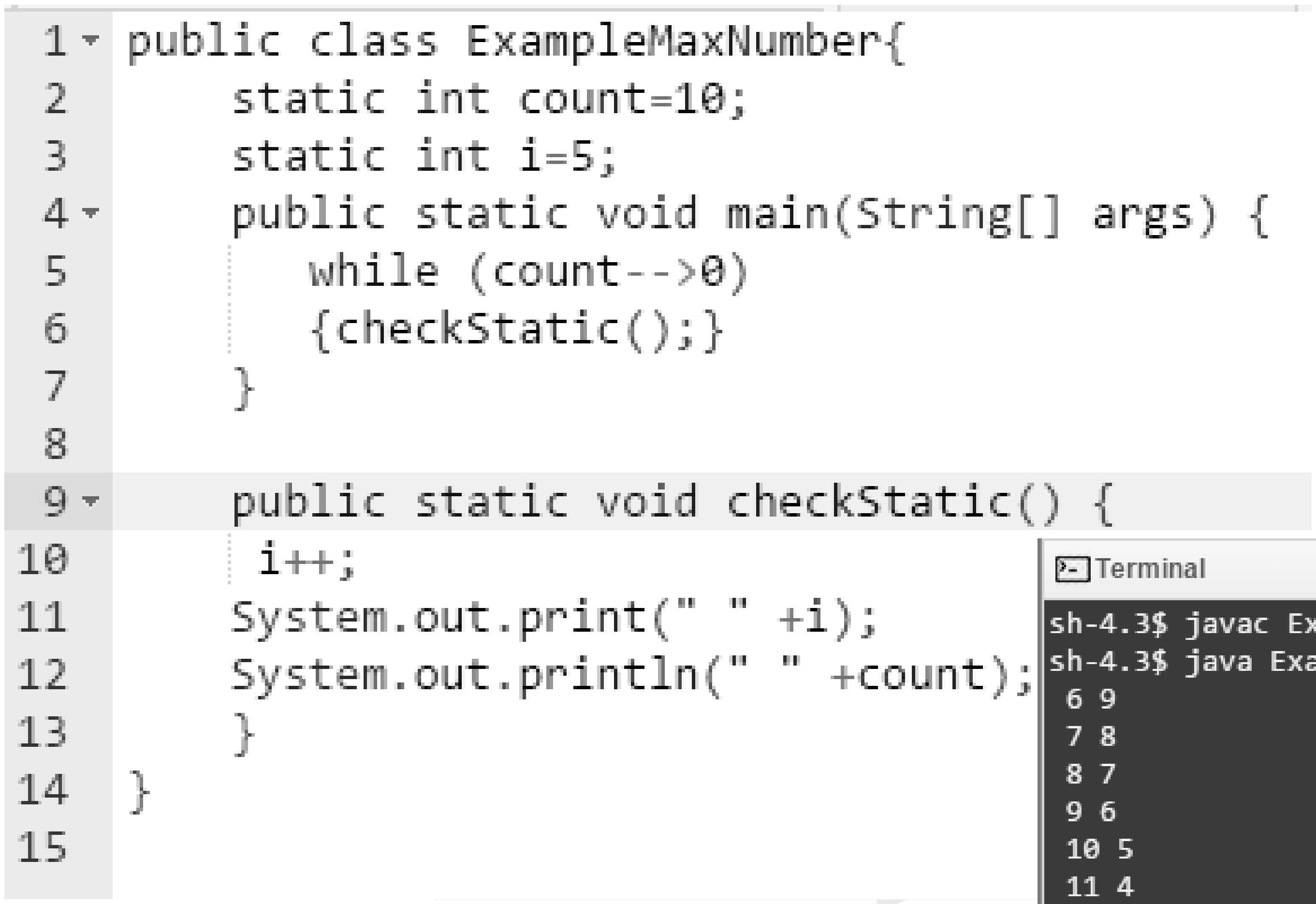
- The **static keyword** in java is used for memory management mainly.
- We can apply java static keyword with
    - Variables
    - methods
    - blocks
    - nested class.

Java static variable

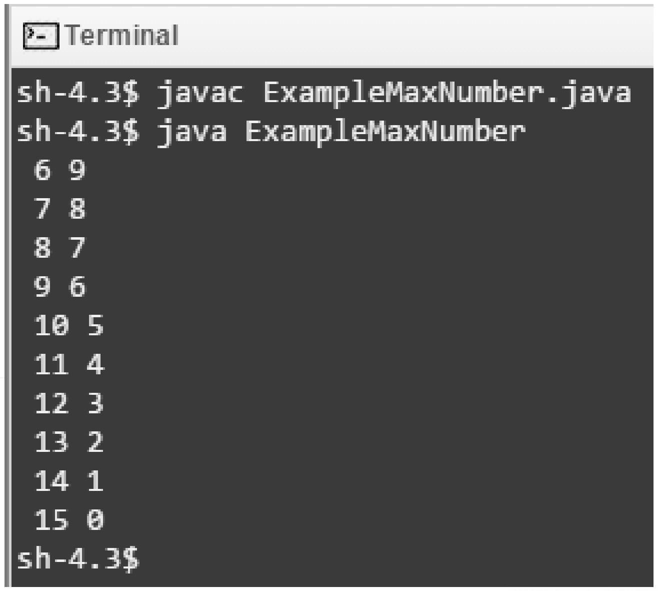
- If you declare any variable as static, it is known static variable.
- The static variable can be used to refer the common property of all objects (that is not unique for each object) e.g. company name of employees, college name of students etc.
- The static variable gets memory only once in class area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

- **Advantage of static variable**
  - It makes your program **memory efficient** (i.e it saves memory).

Static variable

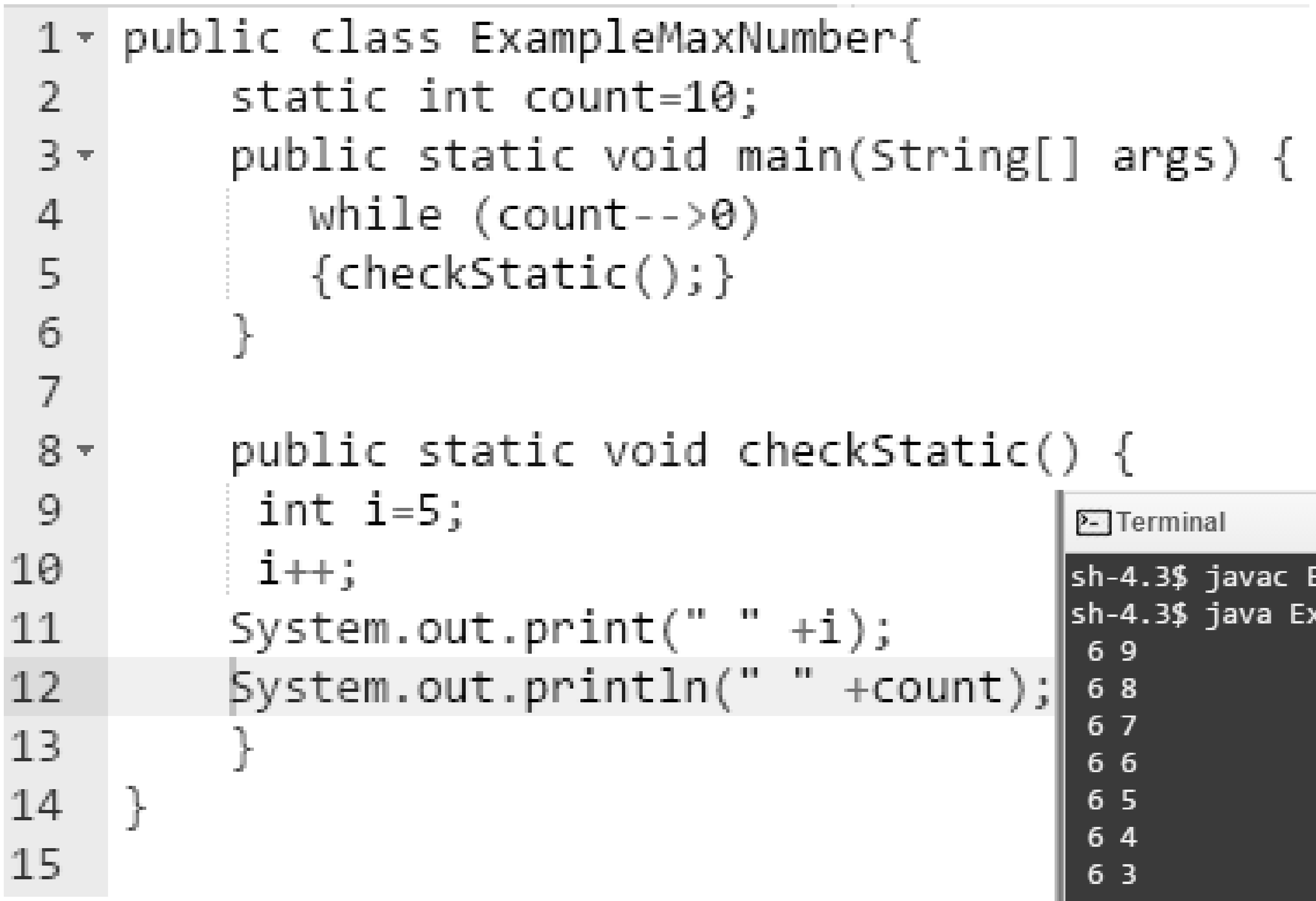


```java
public class ExampleMaxNumber{
    static int count=10;
    static int i=5;
    public static void main(String[] args) {
       while (count-->0)
       {checkStatic();}
    }

    public static void checkStatic() {
     i++;
    System.out.print(" " +i);
    System.out.println(" " +count);
    }
}

```

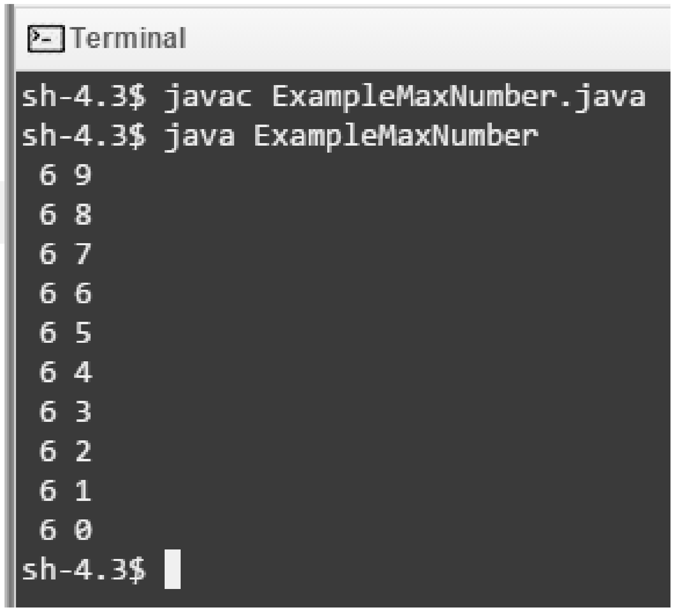


Terminal

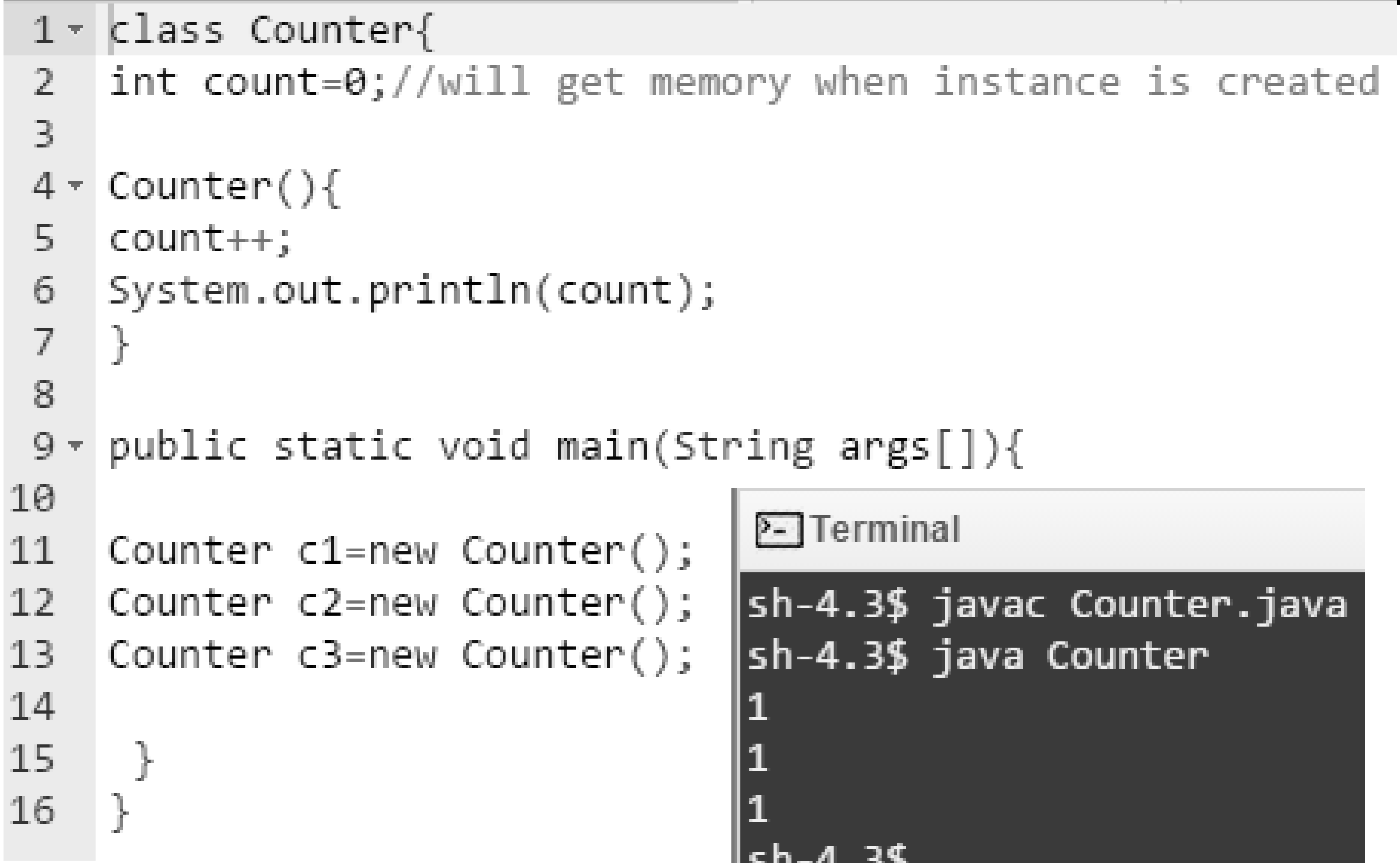
```
sh-4.3$ javac ExampleMaxNumber.java
sh-4.3$ java ExampleMaxNumber
 6 9
 7 8
 8 7
 9 6
 10 5
 11 4
 12 3
 13 2
 14 1
 15 0
sh-4.3$
```



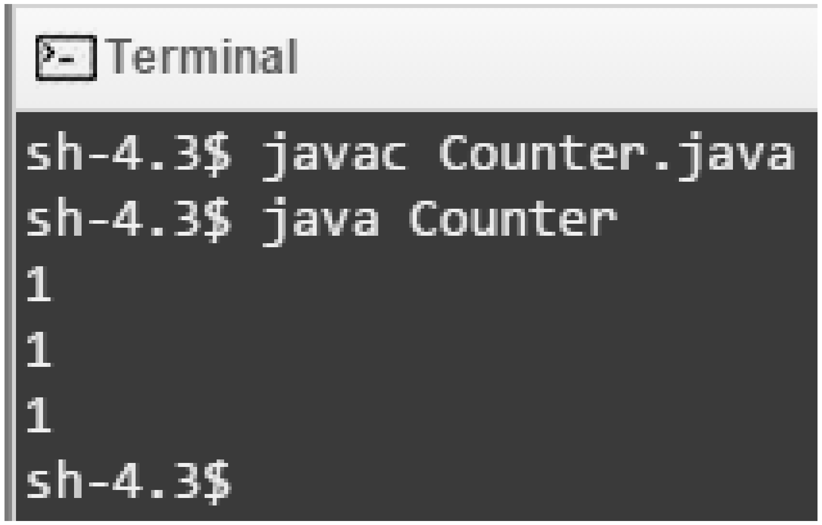
```java
public class ExampleMaxNumber{
    static int count=10;
    public static void main(String[] args) {
       while (count-->0)
       {checkStatic();}
    }

    public static void checkStatic() {
     int i=5;
     i++;
    System.out.print(" " +i);
    System.out.println(" " +count);
    }
}

```



```
sh-4.3$ javac ExampleMaxNumber.java
sh-4.3$ java ExampleMaxNumber
 6 9
 6 8
 6 7
 6 6
 6 5
 6 4
 6 3
 6 2
 6 1
 6 0
sh-4.3$
```

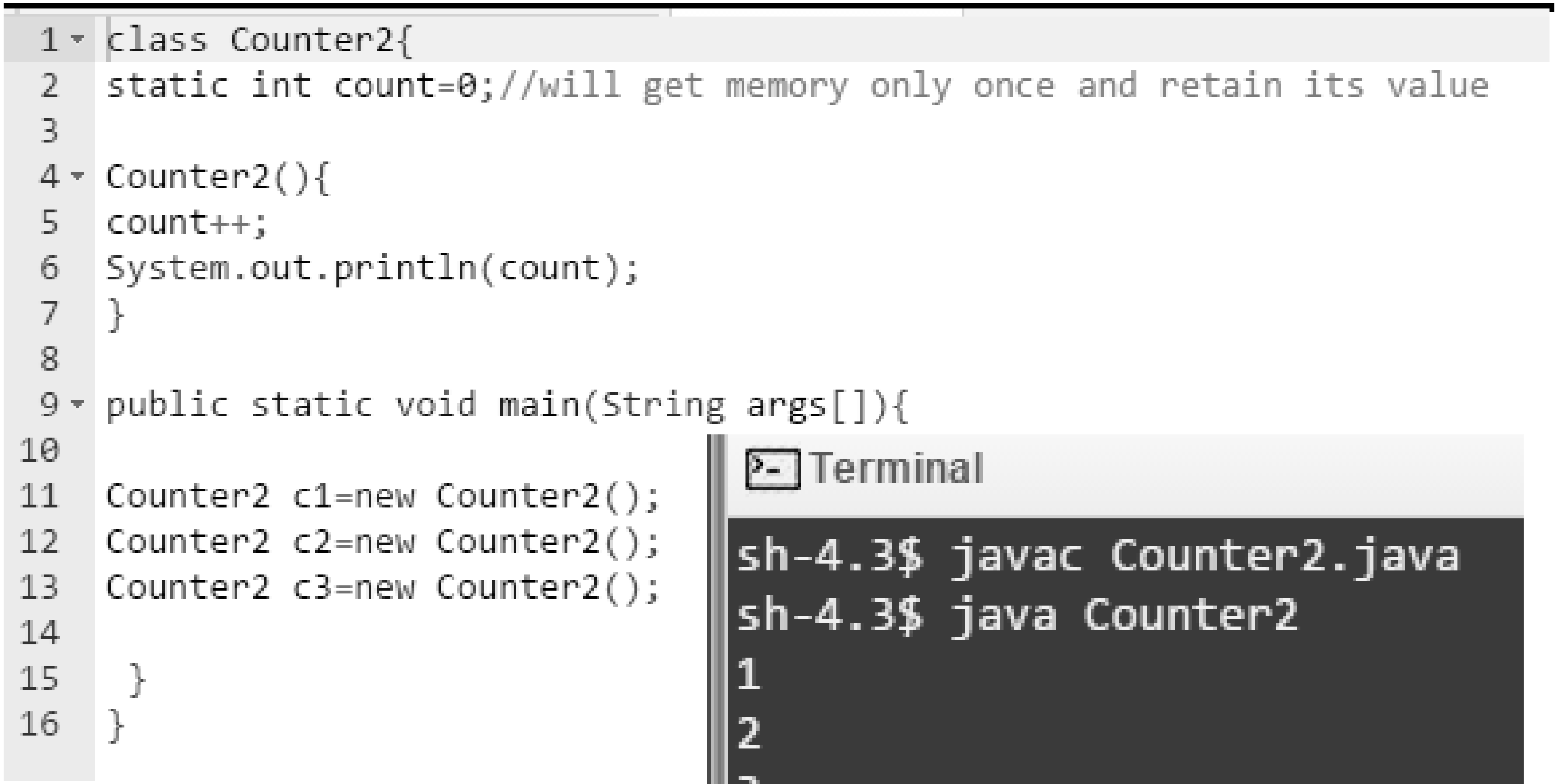


```java
class Counter{
int count=0;//will get memory when instance is created

Counter(){
count++;
System.out.println(count);
}

public static void main(String args[]){

Counter c1=new Counter();
Counter c2=new Counter();
Counter c3=new Counter();

 }
}
```

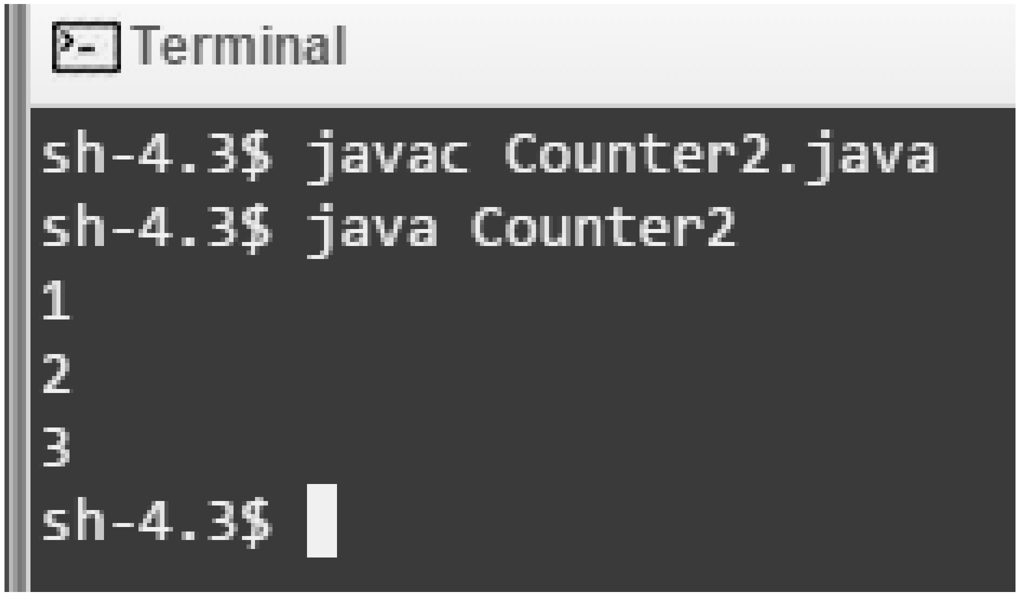


Terminal

```
sh-4.3$ javac Counter.java
sh-4.3$ java Counter
1
1
1
sh-4.3$
```



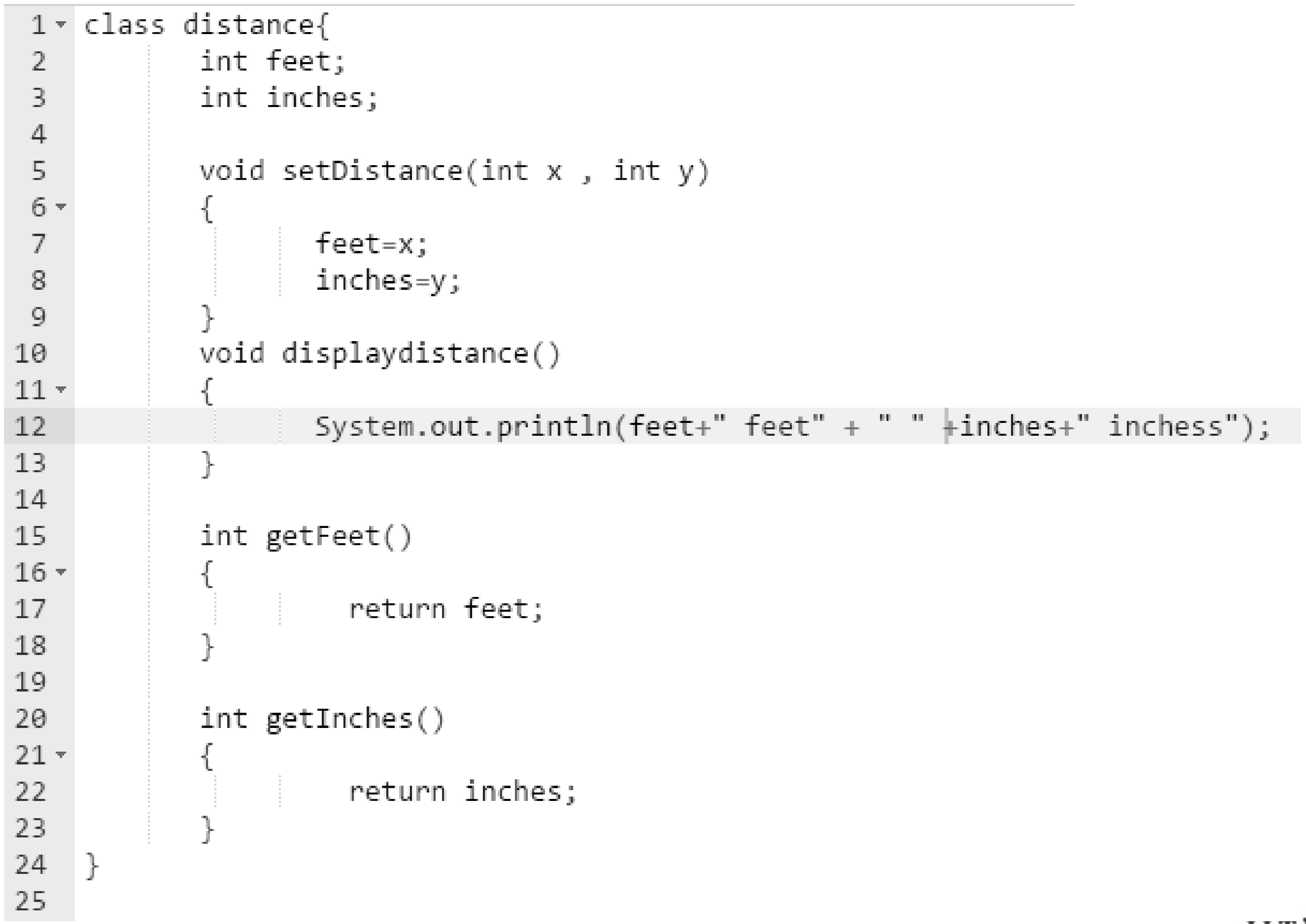
```java
class Counter2{
static int count=0;//will get memory only once and retain its value

Counter2(){
count++;
System.out.println(count);
}

public static void main(String args[]){

Counter2 c1=new Counter2();
Counter2 c2=new Counter2();
Counter2 c3=new Counter2();

 }
}
```



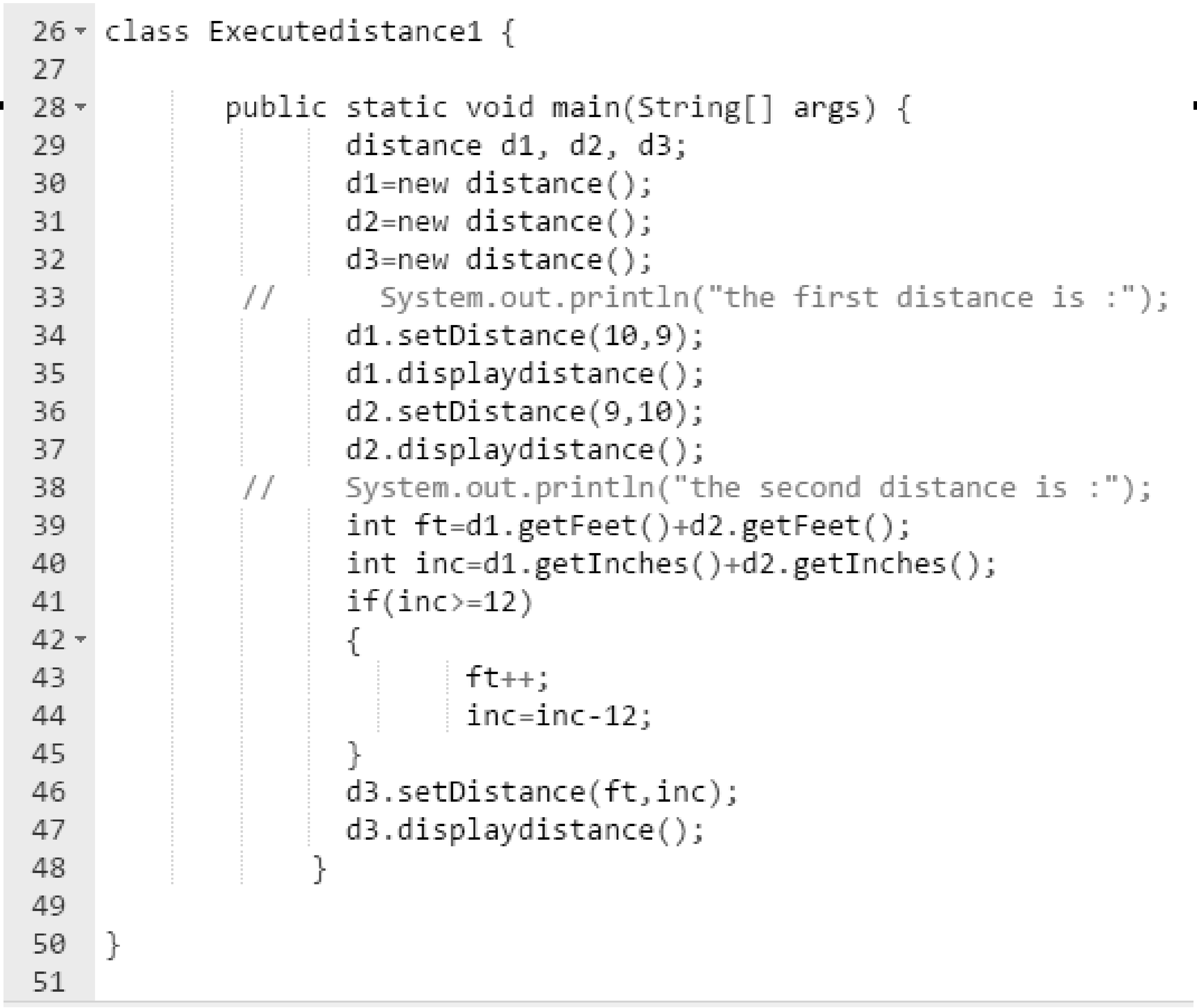
Terminal

```
sh-4.3$ javac Counter2.java
sh-4.3$ java Counter2
1
2
3
sh-4.3$
```

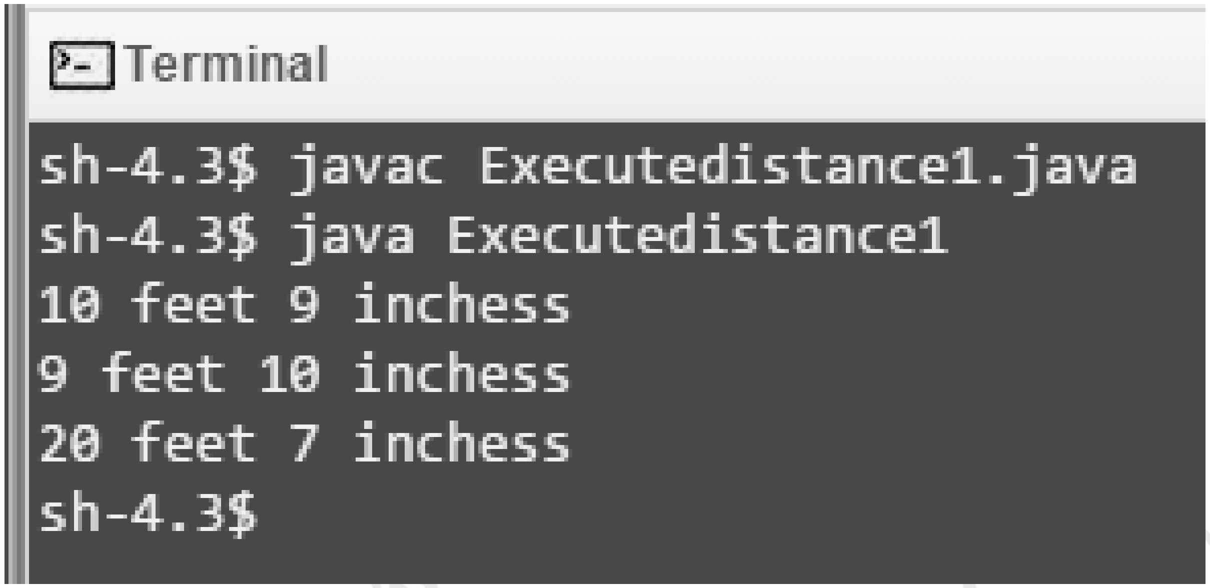
Addition of Two distances



```
class distance{
        int feet;
        int inches;

        void setDistance(int x , int y)
        {
                feet=x;
                inches=y;
        }
        void displaydistance()
        {
                System.out.println(feet+" feet" + " " +inches+" inchess");
        }

        int getFeet()
        {
                   return feet;
        }

        int getInches()
        {
                   return inches;
        }
}

```



```
class Executedistance1 {

        public static void main(String[] args) {
                distance d1, d2, d3;
                d1=new distance();
                d2=new distance();
                d3=new distance();
        //        System.out.println("the first distance is :");
                d1.setDistance(10,9);
                d1.displaydistance();
                d2.setDistance(9,10);
                d2.displaydistance();
        //      System.out.println("the second distance is :");
                int ft=d1.getFeet()+d2.getFeet();
                int inc=d1.getInches()+d2.getInches();
                if(inc>=12)
                {
                        ft++;
                        inc=inc-12;
                }
                d3.setDistance(ft,inc);
                d3.displaydistance();
        }

}
```



```
sh-4.3$ javac Executedistance1.java
sh-4.3$ java Executedistance1
10 feet 9 inchess
9 feet 10 inchess
20 feet 7 inchess
sh-4.3$
```