2: More types, Methods,Conditionals

除法的不同之处：

```
double a = 5.0/2.0; // a = 2.5
int b = 4/2; // b = 2
int c = 5/2; // c = 2
double d = 5/2; // d = 2.0
```

一般来说，我们的结果是根据我们的类型来决定的，但是是先计算后面的值，再根据前面的类型进行输出

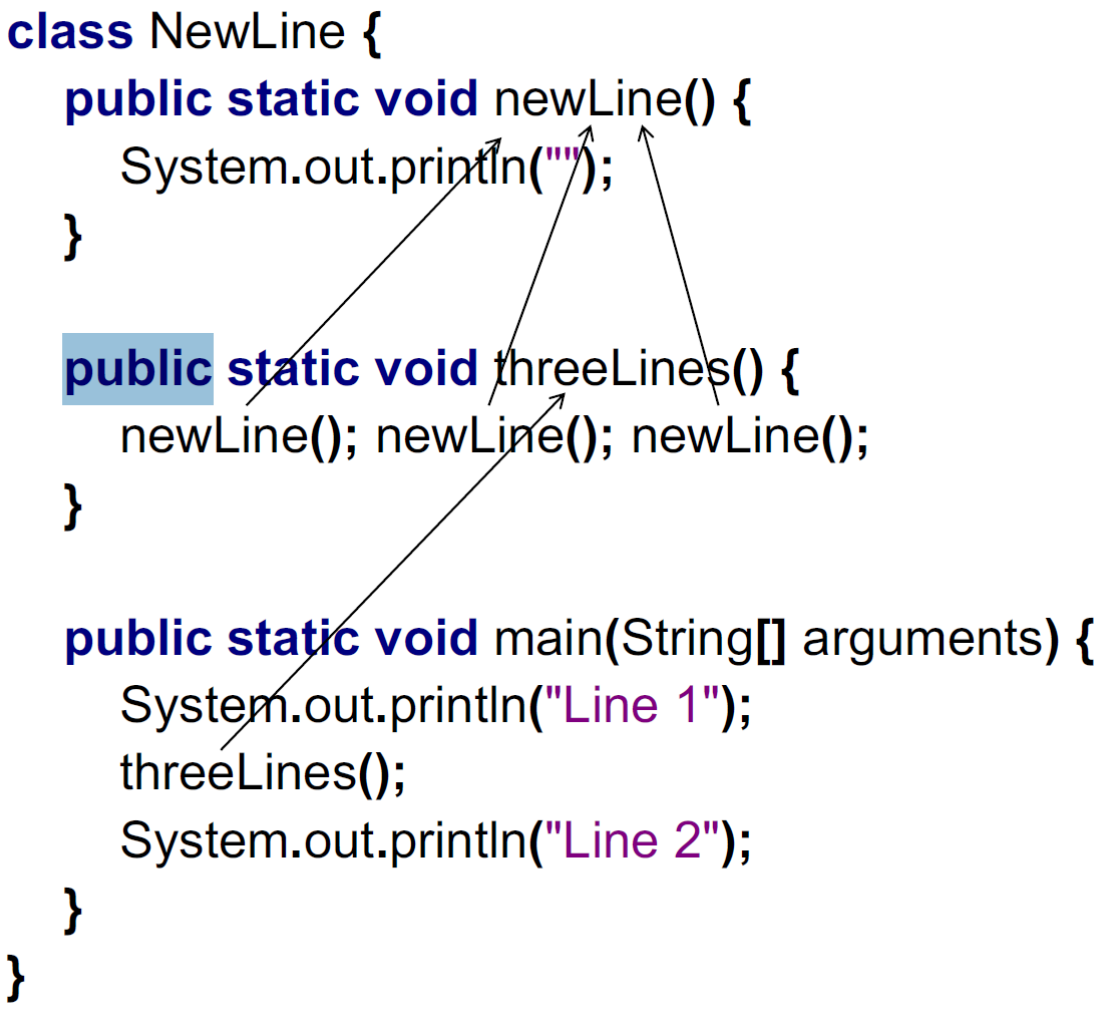
如果我们的类型与后面的数字不匹配的话，就会报错

```
String five = 5; // ERROR!

int a = 2; // a = 2
double a = 2; // a = 2.0 (Implicit)

int a = 18.7; // ERROR
int a = (int)18.7; // a = 18

double a = 2/3; // a = 0.0
double a = (double)2/3; // a = 0.6666…
```

我们通过类名来调用类函数

```
public static void NAME() {
    STATEMENTS
}

NAME();//调用我们的NAME函数
```

嵌套调用：



我们的main函数先调用我们的threelines，然后我们的threelines调用我们的newline，然后再进行输出

我们也可以在函数中加入我们的值

```
class Square {
    public static void printSquare(int x){
        System.out.println(x*x);
    }
    public static void main(String[] arguments){
        int value = 2;
        printSquare(value);
        printSquare(3);
        printSquare(value*2);
    }
}
```

但是我们在规定完我们的x的类型之后，就不能随便变动了，如果输入的不是int的话，就会报错

```
public class example {
    public static void Square(double x){
        System.out.println(x*x);
    }
    public static void main(String[] arguments){
        Square(5);
    }
}
```

这样我们输出就是25.0，输出的是double类型

传递多个参数：

```
public class example {
    public static void jisuan(int a,int b){
        System.out.println(a*b);
    }
    public static void main(String[] arguments){
        jisuan(4,5);
    }
}
```

输出的形式：

前面的都是void，我们现在可以使用int，double等形式来进行输出

```
public class example {
    public static int jisuan(int a,int b){
        return a*b;
    }
    public static void main(String[] arguments){
        System.out.println(jisuan(4,5));
    }
}
```

改变变量的值：

```
class SquareChange {
public static void printSquare(int x){
    System.out.println("printSquare x = " + x);
    x = x * x;
    System.out.println("printSquare x = " + x);
}
public static void main(String[] arguments){
        int x = 5;
        System.out.println("main x = " + x);
        printSquare(x);
        System.out.println("main x = " + x);
    }
}
```

输出：

```
main x = 5
printSquare x = 5
printSquare x = 25
main x = 5
```

可以发现最后x的值也没有发生改变

```
public class test {
    public static void main(String[] arguments){
        int x=6;
        if(x==6){
            int z=5;
            int y=72;
            System.out.println(z+y);
        }
        System.out.println(x);
    }
}
```

这就是说，我们在if语句中的赋值都是不在整体中存在的

数学方法：

```
public class test {
    public static double jisuan(double x){
        return Math.sin(x);
    }
    public static void main(String[] arguments){
        System.out.println(jisuan(2.5));
    }
}
```

if语句

```
public class test {
    public static void jisuan(int x){
        if(x>5){
            System.out.println("x>5");
        }
        else{
            System.out.println("x<=5");
        }
    }
    public static void main(String[] arguments){
        jisuan(6);
    }

}
```

if else else if

```
public class test {
    public static void jisuan(int x){
        if(x>5){
            System.out.println("x>5");
        }
        else if(x==5){
            System.out.println("x=5");
        }
        else{
            System.out.println("x<5");
        }
    }
    public static void main(String[] arguments){
        jisuan(6);
        jisuan(5);
        jisuan(4);
    }

}
```

类型之间的互相转换：

```
//int to String:
String five = 5; // ERROR!
String five = Integer.toString (5);
String five = "" + 5; // five = "5"

//String to int:
int foo = "18"; // ERROR!
int foo = Integer.parseInt ("18");
```

double类的比较

```
public class test {
```

```
    public static void bijiao(double x,double y){

        if(x==y){
            System.out.println("a=b");
        }
        else{
            System.out.println("a!=b");
        }

    }
    public static void main(String[] arguments){
        double a = Math.cos (Math.PI / 2);
        double b = 0.0;
        bijiao(a,b);
    }


}
```