KTH ROYAL INSTITUTE
OF TECHNOLOGY

EH2745 – Lecture 2

Introduction to Programming in Java

The lecture slides are heavily based on the MIT OpenCourseware course 6.092 – Introduction to Java

Outline

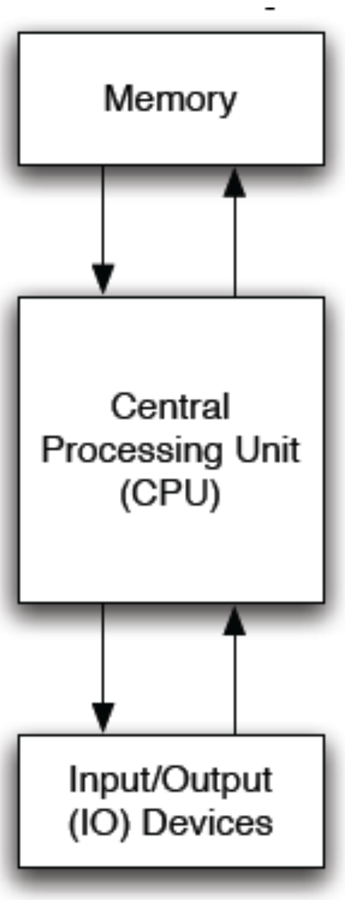
Intro to Java

Types and variables

Operators

Methods

A computer (slightly simplified)



The CPU exectues machine code instructions stored in memory

The memory is used both to store the program and the data needed

Communication with the real world through I/O units (keyboard, monitor, speaker, harddrive, ethernet,…)

Machine code (abstracted to human level)

MOV A,128	Move the value 128 into register A
MOV B, 127	Move the value 127 into register B
ADD A,B	Overwrite value of A with A+B

Takes a lot of programming at that level to achieve this

Man meets machine

- Programming languages (there are several) are all the result of mans attempt to create a language which is sufficiently specific so that computer can understand it, yet understandable by humans.

- Different trade offs between flexibility, efficiency, ease of use, hardware performance result in different programming languages.

What’s the thing with Java?

“Most popular” language – debatable these days with Python a strong contender

Runs on a “virtual machine” (JVM)

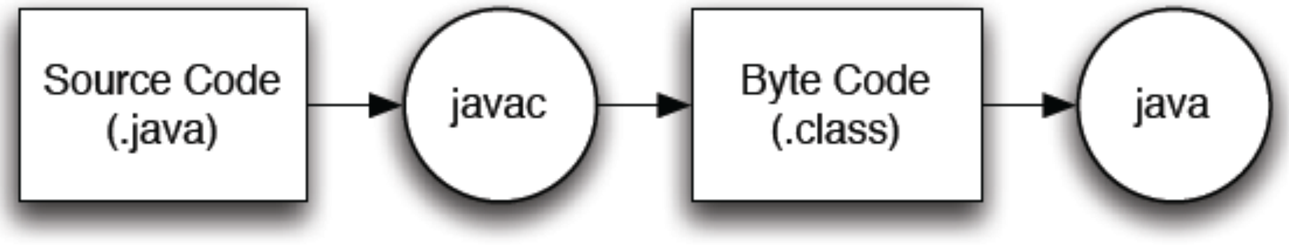
Allows platform independence – used to be a big thing

More complex than some (eg. Python)

Simpler than others (eg. C++)

We do not have to worry so much about the internals of the computer (e.g. Memory allocation)

From woman to machine



A person ”programs” the computer by writing Java Source code

This language is ”sort of OK” for the human, kind of strange with ***[}*** *and* ***;*** *apparently necessary.*

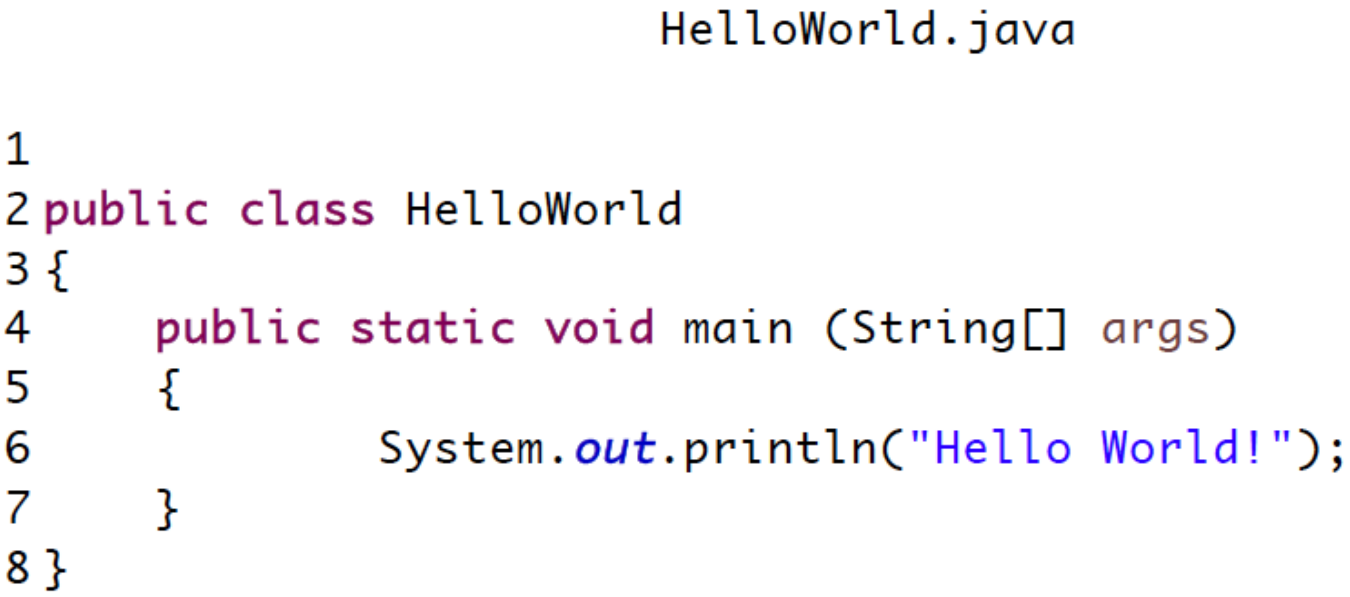
The Java compiler converts the source code into Java Byte code

Checking for errors and converting the source code halfway into machine code.

The Java byte code can run on any Java Virtual Machine

The Java VM is specific for different computer Operating systems (MacOS, Linux, Windows, …)

Java source code



HelloWorld.java

```java

public class HelloWorld
{
    public static void main (String[] args)
    {
            System.out.println("Hello World!");
    }
}
```

Outline

Intro to Java

Types and variables

Operators

Methods

What here follows is the rules of the Java language that we people have to learn to make computers do their thing

Types of Data in Java

Kinds of values that can be stored and manipulated.

boolean: Truth value (**true** or **false**).

int: Integer (0, 1, -47).

double: Real number (3.14, 1.0, -2.1).

String: Text (“hello”, “example”).

The eight basic types of types

Note!
Types do **not** start with a capital letter

boolean, True or False

char, 16-bit unicode characters

Integers

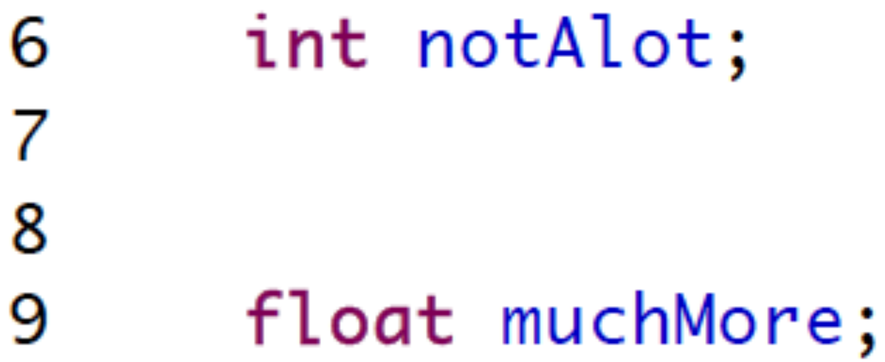
- **byte**
- **short**
- **int**
- **long**

Floating-point types (real numbers)

- **float**
- **double**

Variables in Java

A variable is a named location in memory that stores data of a specific type.



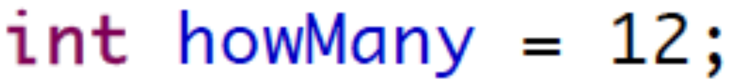
```java
int notAlot;


float muchMore;
```

Assigning values to variables

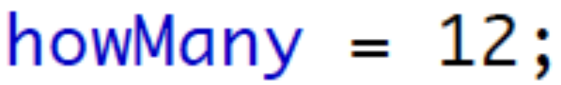
To assign a variable a value use the ”=” sign

Either at declaration



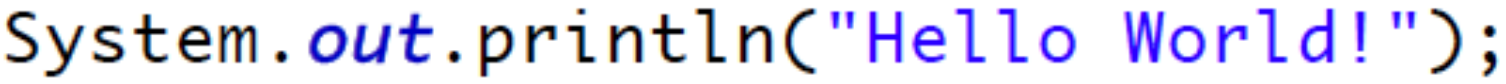
```
int howMany = 12;
```

Or when needed in the program



```
howMany = 12;
```

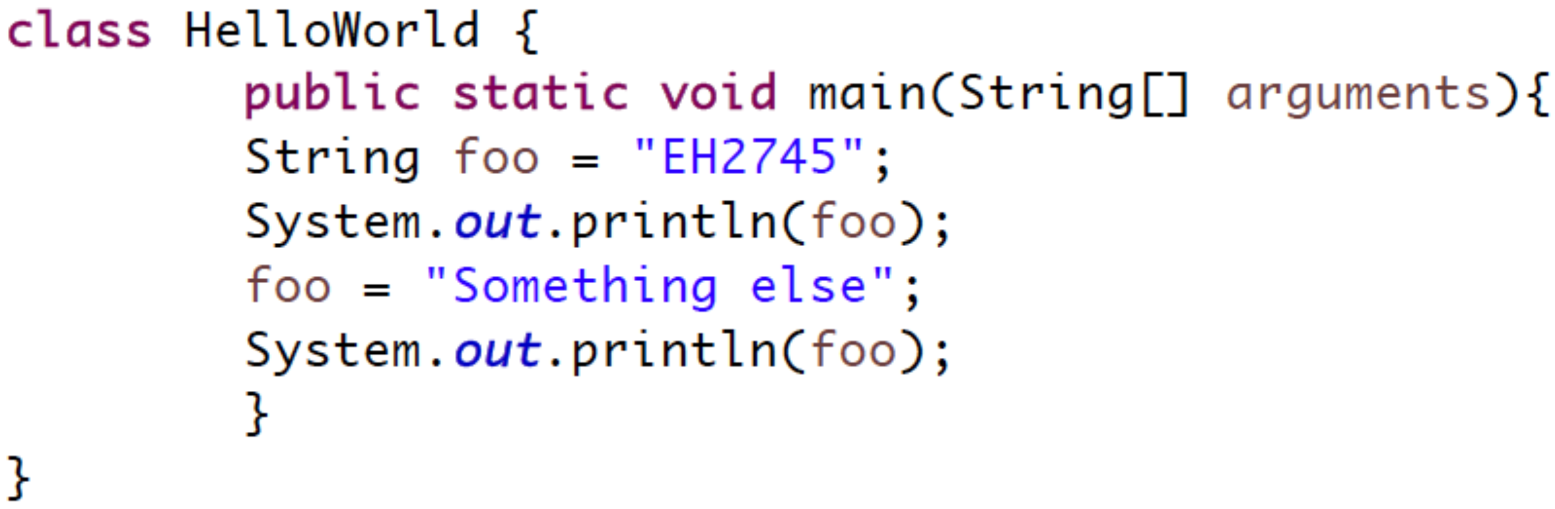
Output from the program



```
System.out.println("Hello World!");
```

System.out.println – calls the method `println` in the Class System and sends the string within ”” to the concole.

Putting it all together



```
class HelloWorld {
        public static void main(String[] arguments){
        String foo = "EH2745";
        System.out.println(foo);
        foo = "Something else";
        System.out.println(foo);
        }
}
```

What appears on the console?

Outline

Intro to Java

Types and variables

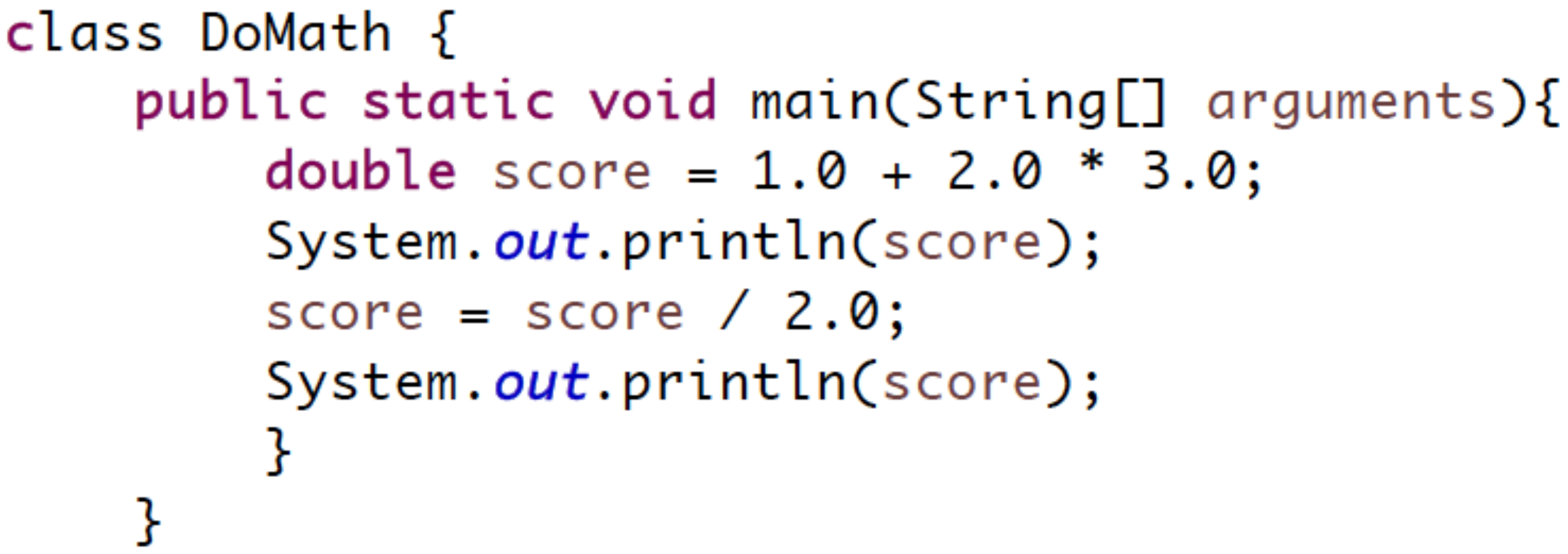
Operators

Methods

Operators

- Symbols that perform simple computations
  - Assignment: =
  - Addition: +
  - Subtraction: -
  - Multiplication: *
  - Division: /

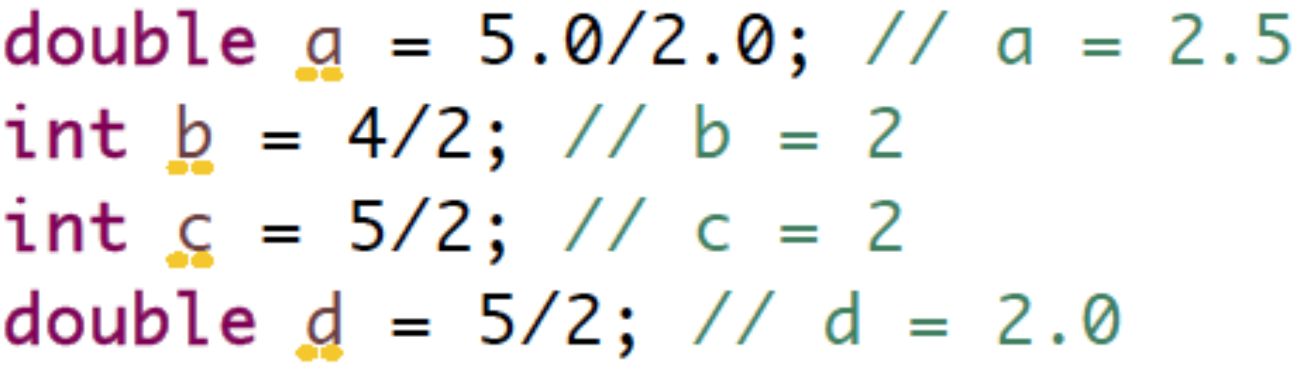
Math example



```
class DoMath {
    public static void main(String[] arguments){
        double score = 1.0 + 2.0 * 3.0;
        System.out.println(score);
        score = score / 2.0;
        System.out.println(score);
        }
    }
```

A note on division!

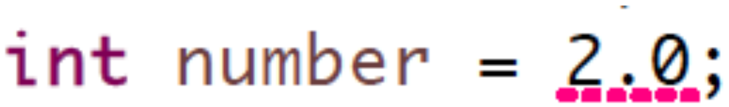
Division ("/") operates differently on variables of integer and on floating point type



```
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
```

Java is strongly typed

Java checks already at compilation time that you are using types in the rightway. Eclipse helps you further by providing warnings.

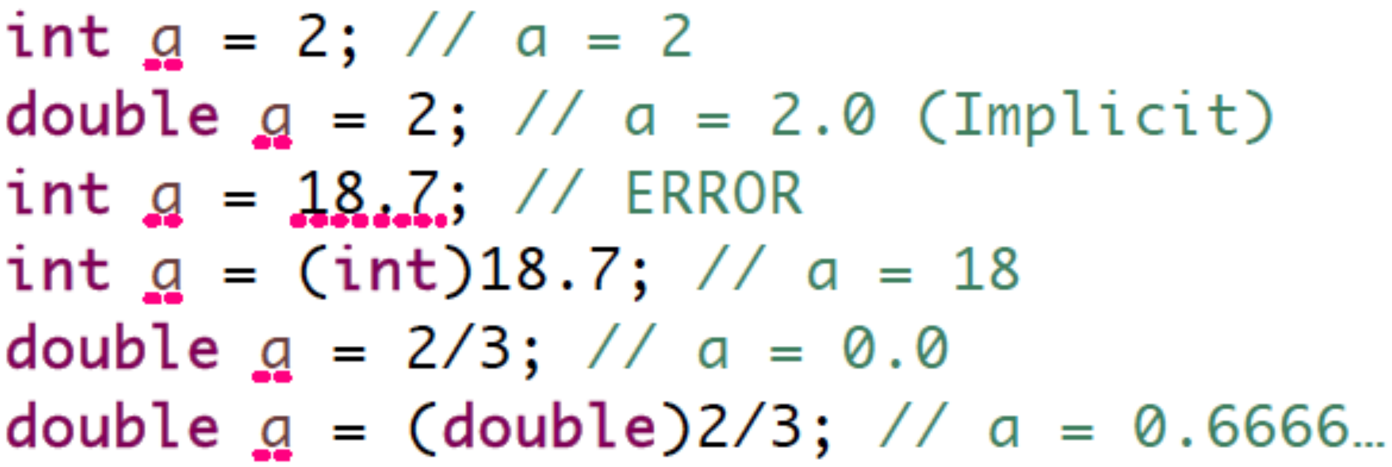


```
int number = 2.0;
```

Generates an error at compile time, i.e.g when sourcecode is converted to bytecode.

Type casting

Converting type by explicitly assigning values and types.



```
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…
```

Outline

Intro to Java

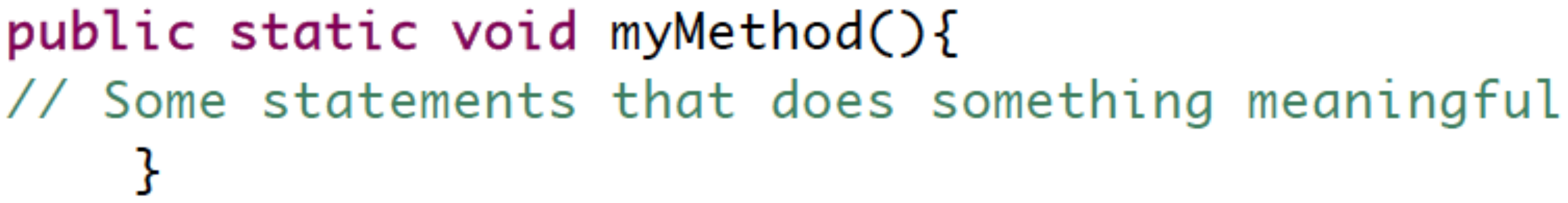
Types and variables

Operators

Methods

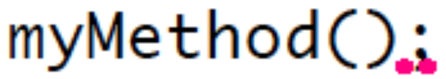
Methods in Java

Methods are in Java what in other programminglanguages are called functions or procedures, or sub-routines or….



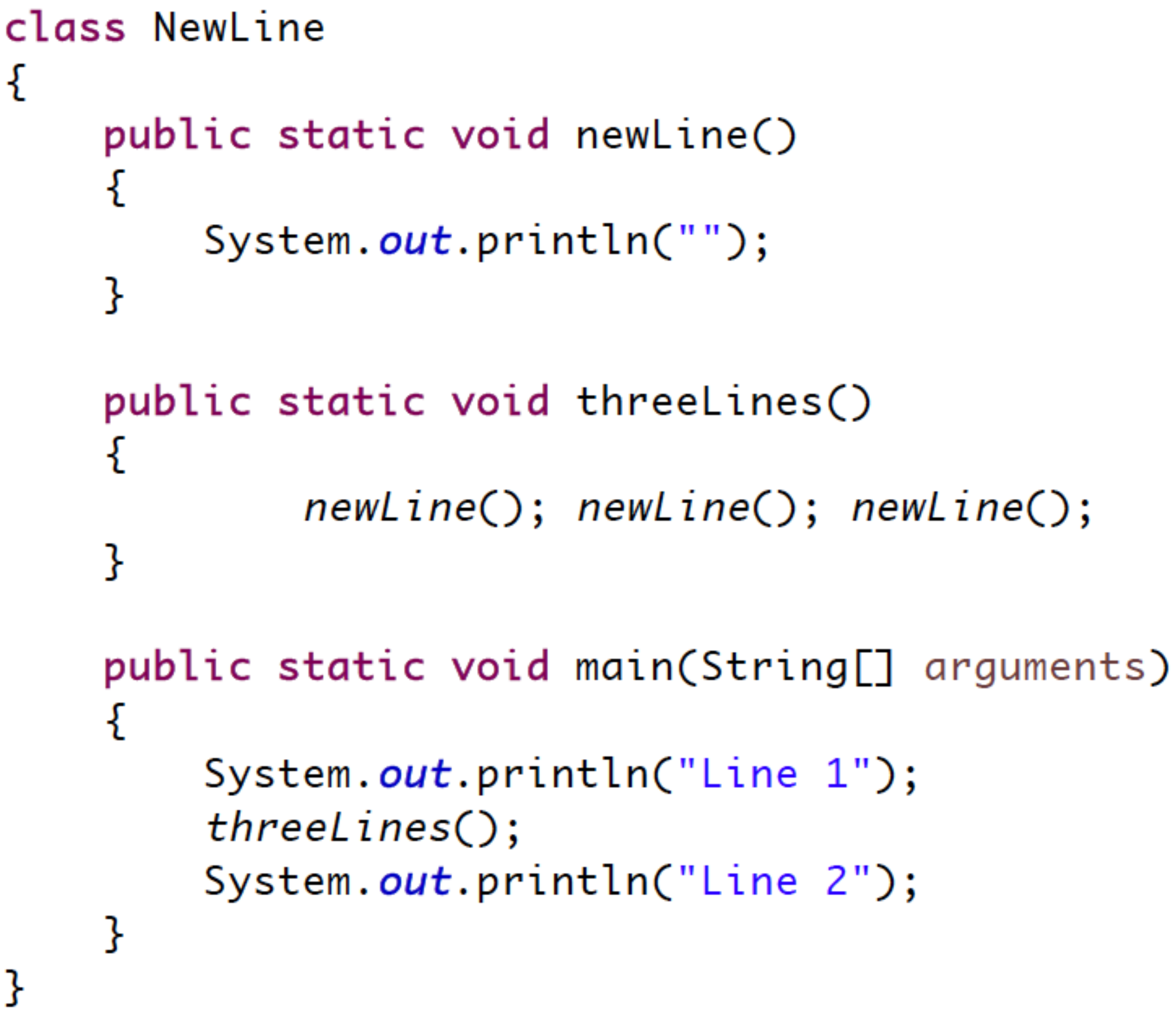
```
public static void myMethod(){
// Some statements that does something meaningful
    }
```

Calling a method is done by calling its name (!?)



```
myMethod();
```

NewLine example



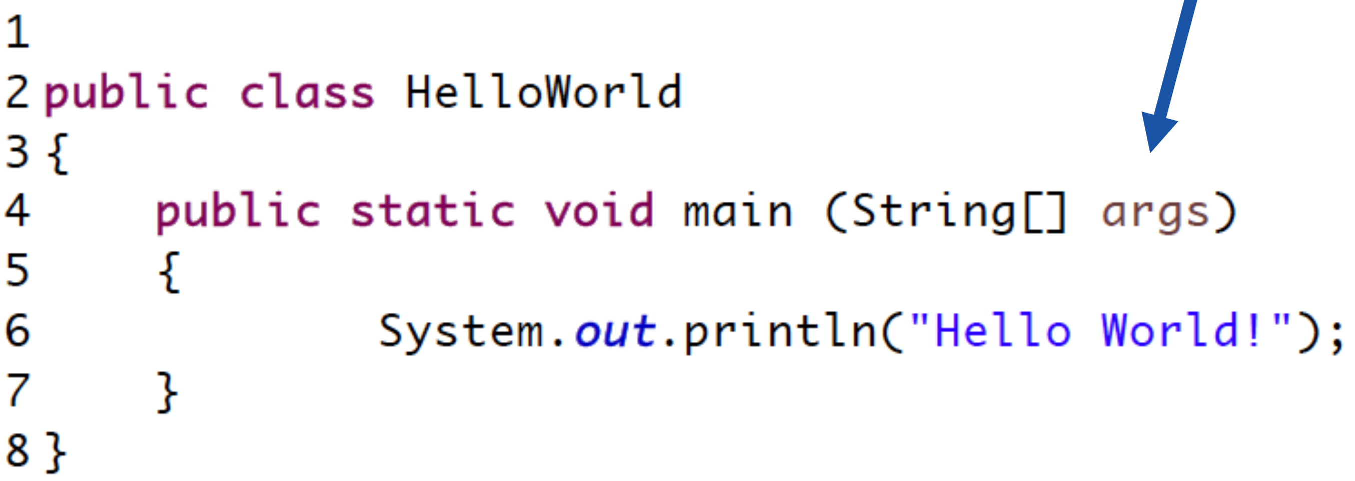
```
class NewLine
{
    public static void newLine()
    {
        System.out.println("");
    }

    public static void threeLines()
    {
            newLine(); newLine(); newLine();
    }

    public static void main(String[] arguments)
    {
        System.out.println("Line 1");
        threeLines();
        System.out.println("Line 2");
    }
}
```

Calling methods with parameters

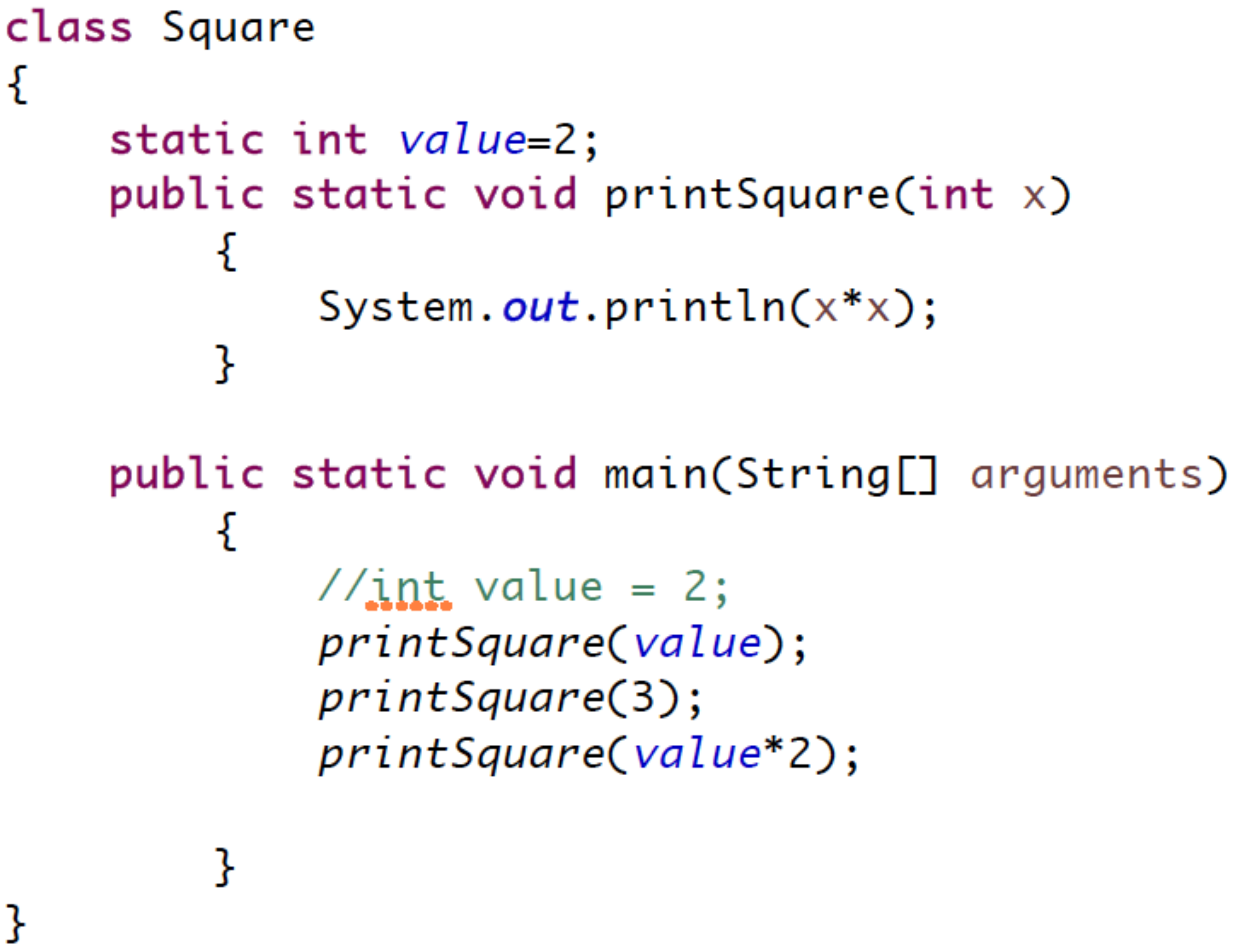
If you want to send input to a method – add it as arguments



```

public class HelloWorld
{
     public static void main (String[] args)
     {
             System.out.println("Hello World!");
     }
}
```

Squares method

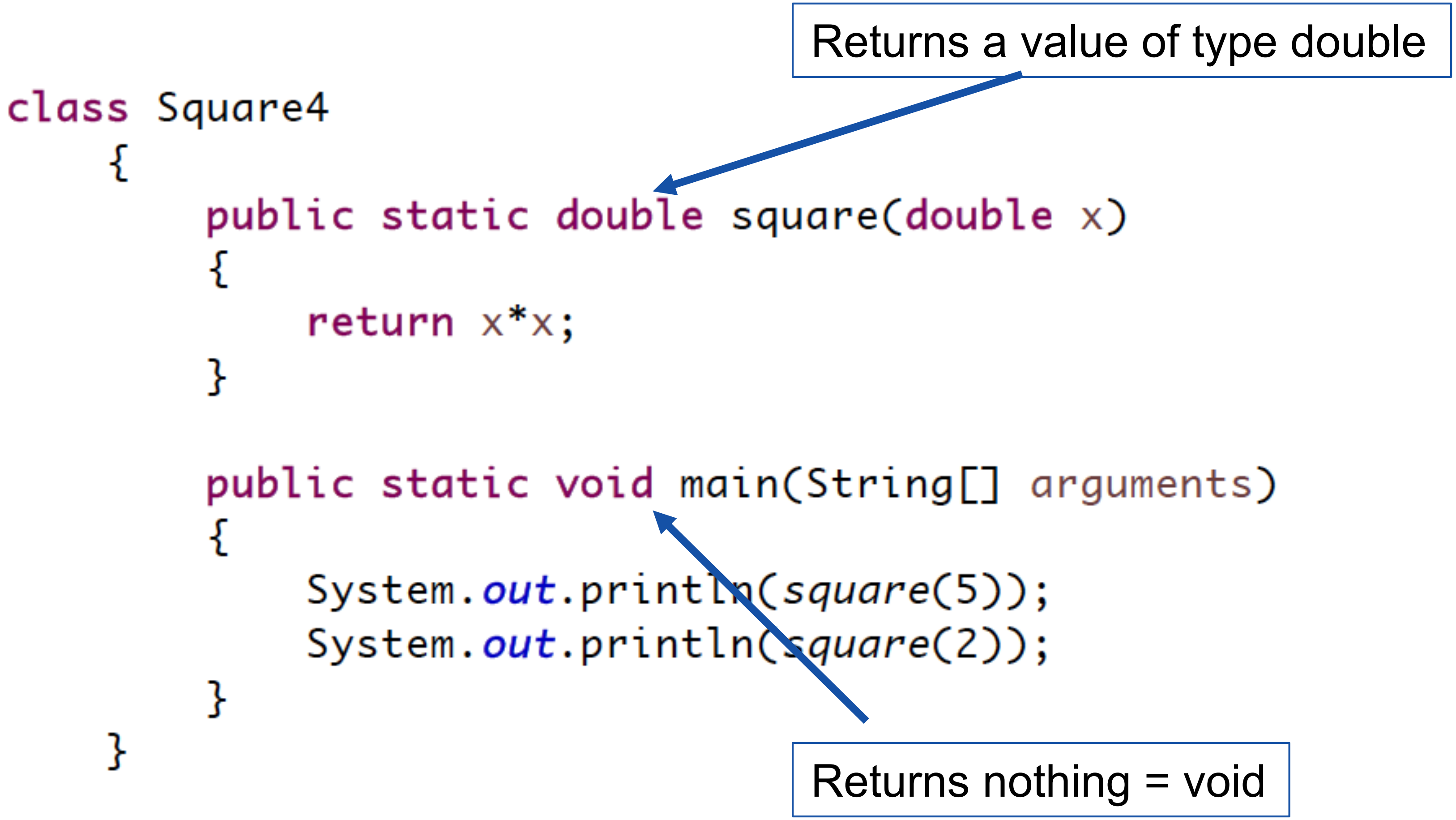


```
class Square
{
    static int value=2;
    public static void printSquare(int x)
        {
            System.out.println(x*x);
        }

    public static void main(String[] arguments)
        {
            //int value = 2;
            printSquare(value);
            printSquare(3);
            printSquare(value*2);

        }
}
```

Return values from methods



```
class Square4
    {
        public static double square(double x)
        {
            return x*x;
        }

        public static void main(String[] arguments)
        {
            System.out.println(square(5));
            System.out.println(square(2));
        }
    }
```

Returns a value of type double

Returns nothing = void

Using libraries of methods

```
Math.sin(x)
Math.cos(Math.PI / 2)
Math.pow(2, 3)
Math.log(Math.log(x + y))
```

Etc… etc....

Type conversion by method

int to String:

```
String five = 5; // ERROR!
String five = Integer.toString (5);
```

String to int:

```
int foo = “18”; // ERROR!
int foo = Integer.parseInt (“18”);
```