

Fundamentals of Programming

Laboratory 5

Methods.

**General-purpose and user -
defined methods**



What is a method?

- Block of code that receives its own identifier
- i.e. can be executed using its name
- Different names in programming languages:
method, function, procedure
- A method receives data through parameters,
and/or operates on globally accessible
variables

Procedural Programming vs Object Oriented Programming



- In PP data is kept **separate** from functionality – procedures/functions operate on data passed to them, and have only temporary variables
- In OOP a method needs to **belong** to a class or an object, and has access to object's data
- Java is a strictly OOP language, thus all methods need to be put in classes
- However, for now we will try to avoid dealing with objects and work on **general-purpose methods**
- The goal is to get you used to the idea of dividing your program into procedures/functions

CALLING METHODS

Methods are executed using their assigned identifiers, parentheses and if needed parameters.

If the method belongs to a class or object, the name of that class or object preceeds the method and is followed by a dot (multiple dots in case of subclasses).

For example:

```
System.out.print("Text passed as parameter") ;
```

Calls the method **print()** that belongs to the class **System** and subclass **out**, passing the text as the argument (a.k.a. parameter).

EXAMPLE MATHEMATICAL METHOD – `abs()`



Consider the mathematical method named ***abs()***, which calculates the absolute value of a number.

```
a = Math.abs(number) ;
```

 method
 class argument

The items that are passed to the method through the parentheses, as we have noted previously in relation to the `println()` method, are called ***arguments*** of the method and constitute its input data.

METHOD OVERLOADING

A method is defined to work with a fixed set of parameters. However, different versions of the same method can be prepared for different parameter data types, or a different number of parameters. This is called **method overloading**.

Expression	Value Returned	Returned Data Type
<code>Math.abs (-4)</code>	4	integer
<code>Math.abs (-17.25f)</code>	17.25	floating-point
<code>Math.abs (-23456.78)</code>	23456.78	double-precision

MATHEMATICAL METHODS

The arguments that are passed to a method need not be single constants. An expression, a variable or another method can also be an argument.

For example, the following arguments are valid for the given methods:

```
a = Math.sqrt(4.0 + 5.3 * 4.0);
```

```
b = 4 * Math.sqrt(4.5 * 10.0) - 2.0;
```

```
c = Math.sqrt(Math.pow(Math.abs(num1), num2));
```




Common Java Math Methods

Method Name	Description	Returned Value
abs (x)	absolute value	same data type as argument
pow (x1 , x2)	x1 raised to the x2 power	double
sqrt (x)	square root of x	double
log (x)	natural logarithm of x	double
exp (x)	e raised to the x power	double
ceil (x)	smallest integer value that is not less than x	double
floor (x)	largest integer value that is not greater than x	double
min (x , y)	smaller of its two arguments	same data type as arguments
max (x , y)	larger of its two arguments	same data type as arguments
round (x)	rounded value	integer
random ()	random number between 0.0 incl. and 1.0 excl.	double
sin (x)	sine of x (x in radians)	double

QUESTIONS



Write valid Java statements to determine:

- a. The square root of 6.37
- b. The absolute value of $a^2 - b^2$
- c. The value of e raised to the 3rd power



ORAL
EXERCISE

EXERCISE 1

Write a program that determines the time it takes a ball to hit the ground after it has been dropped from an 800-foot tower. The mathematical formula to calculate the time, in seconds, that it takes to fall a given distance, in feet, is:

$$time = \sqrt{2 * distance / g}$$

where g is the gravitational constant equal to 32.2 ft/sec².



EXERCISE 2

A model of worldwide population, in billions of people, after the year 2000 is given by the equation

$$\textit{Population} = 6.0 e^{0.02 [\textit{Year} - 2000]}$$

Using this formula, write, compile, and execute a Java program to estimate the worldwide population in years 2012-2020.



EXERCISE 3

Create a series of 10 random numbers using Java's library method `Math.random()`

The numbers can be equal to 1,2,3,4,5 or 6

Hint: Use the `Math.ceil()` method, `Math.round()` and/or cast value to `int`

casting:

```
int x = (int) (2.7*3.5) ;
```



DECLARING METHODS

To create your own method, you need to write its header and body **inside a class** but **outside any other methods** (such as your main method).

The header contains:
keywords for scope, return type, method name, arguments

Copy the example on the next two slides to your computers to work on.

CALLING AND PASSING DATA TO A METHOD



```
import java.util.*;
public class Main
{
    public static void main(String[] args)
    {
        Scanner input = new Scanner(System.in);

        System.out.print("Enter the value of firstnum:  ");
        double firstnum = input.nextDouble();

        System.out.print("Enter the value of secnum:  ");
        double secnum = input.nextDouble();
        printMaximum(firstnum, secnum);
        // the method is called here
    } // end of main() method
    // following is the findMaximum() method
```


CALLING AND PASSING DATA TO A METHOD



HEADER

```
public static void printMaximum(double x, double y)  
{    // start of method body  
    double max;    // variable declaration  
    if (x >= y)    // find the maximum number  
        max = x;  
    else  
        max = y;  
    System.out.print("The maximum of "+x+" and "+y+" is "+max);  
}    // end of method body and end of method  
}    // end of class
```


DECLARING THE METHOD

HEADER

```
public static void printMaximum(double x, double y)  
{  
    //body of the method  
}
```


THE HEADER OF THE METHOD



```
public static void printMaximum(double x, double y)
```

KEYWORDS

*(for now all our methods will be general purpose, so **public static**)*

RETURN DATA
TYPE

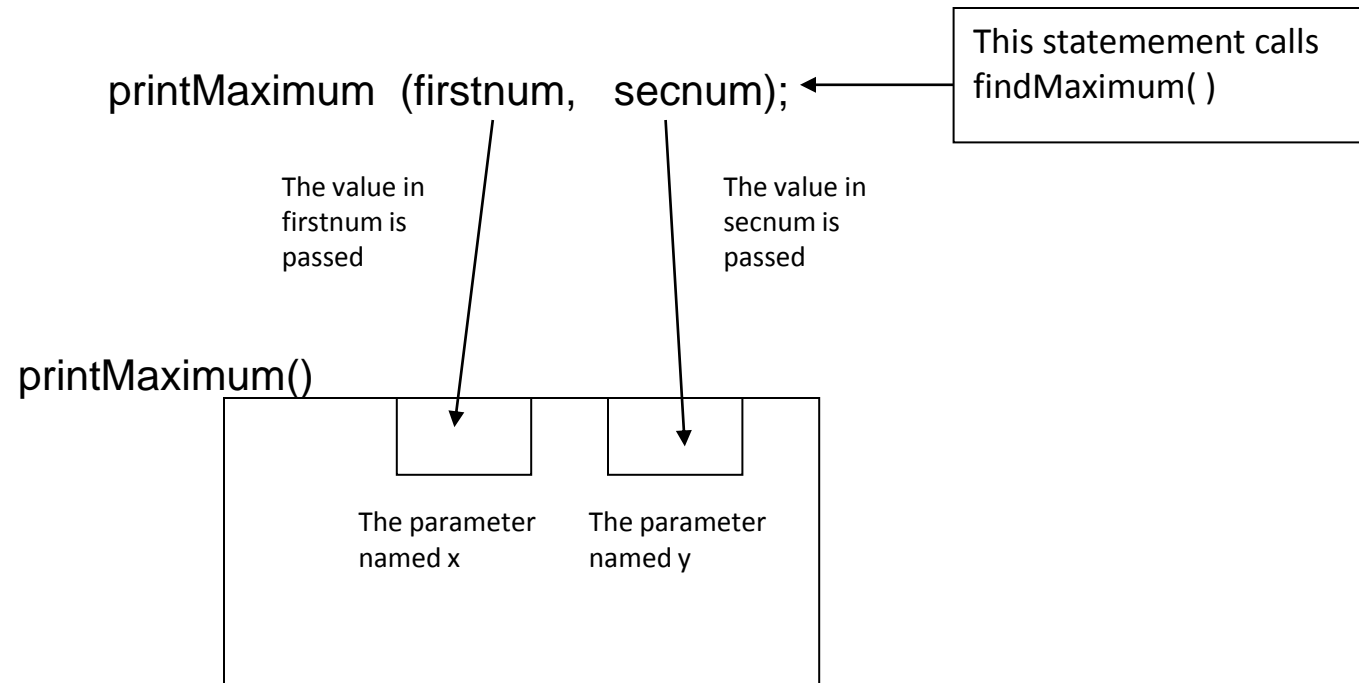
METHOD
IDENTIFIER

ARGUMENTS
AND THEIR DATA
TYPES

Note: If the method is given a return data type, then in the body it must contain a statement: `return(expression);`

Where expression is a value of that data type.

CALLING THE METHOD



QUESTIONS



For the following method headers, determine the number, type, and order (sequence) of the values that must be passed to the method:

public static void price (int type, double yield, double maturity)

public static void interest (char flag, double price, double time)



ORAL
EXERCISE

QUESTIONS



Write a general-purpose method named ***findAbs*** () that accepts a double-precision number passed to it, computes its absolute value, and displays the absolute value.



ORAL
EXERCISE

EXERCISE:

Write method headers for the following:

- a. A general-purpose method named `check ()`, which has three parameters. The first parameter should accept an integer number, the second parameter a floating-point number, and the third parameter a double-precision number. The method returns no value.
- b. A general-purpose method named `mult ()` that accepts two floating-point numbers as parameters, multiplies these two numbers, and returns the result.



EXERCISE 4

According to Plato, a man should marry a woman whose age is half his age plus seven years.

Write a program that requests a man's age as input and gives the ideal age of his wife. Use two separate methods for input and output.

Your main method should look like this:

```
public static void main(String[] args)
{
    int age;
    age = askUser();
    calculateAndPrintWifeAge(age);
}
```



EXERCISE 5

To determine the number of square centimeters of tin needed to make a tin can, add the square of the radius of the can to the product of the radius and height of the can, and then multiply this sum by 6.283.

Write a program that requests the radius and height of a tin can in centimeters as input and displays the number of square centimeters required to make the can.

In the program create a method that asks a user for a value and returns a double, and can be called like this:

```
double a = askUser(„Please give the radius”);
```



HOMework 1



Prepare a program to check whether a number is a prime number.

Inside your Main class create a method `checkPrime()` that accepts an integer argument and returns a boolean true if the argument was prime.

```
if (checkPrime(x))  
{...}
```

Repeat checking for any number entered by the user, until he enters 0.

For checking, best use the modulo operator `%` to find the remainder from division.

Bonus: Have `checkPrime()` return the number of modulo operations that were performed and look for ways to improve the algorithm to perform as few operations as possible.

Example output:

Please type a number: 7

The number is prime. The program checked 5 divisors.

Please type a number: 2

The number is not prime. The program checked 1 divisor.

Please type a number: 0



HOMework
EXERCISE