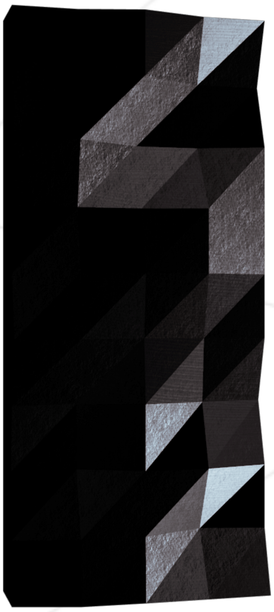


# 42 Roma Luiss x Leonardo | Java Piscine

## Module 00 - Management structures and arrays

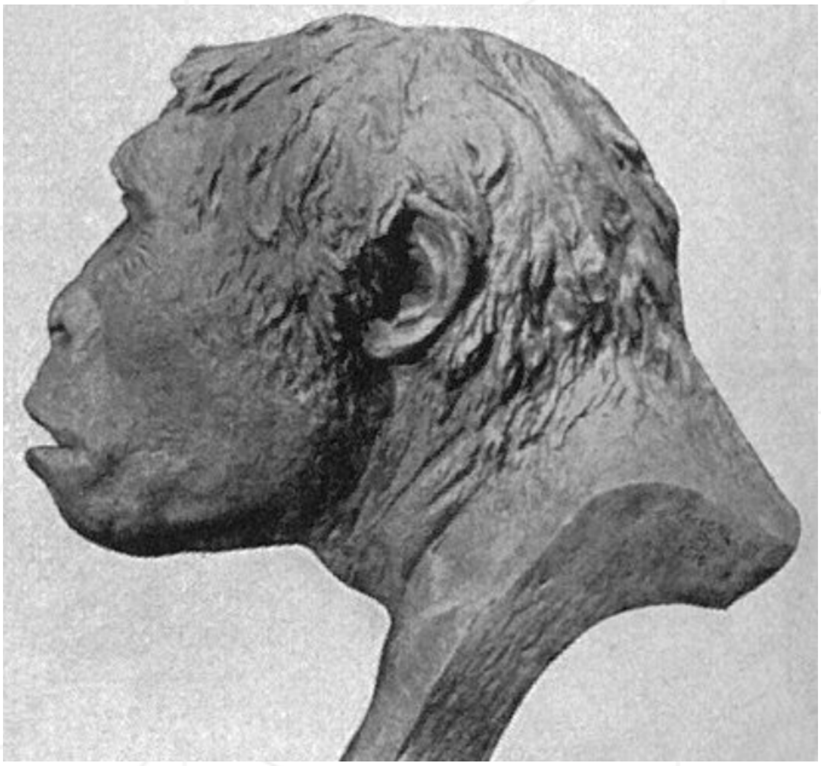
Summary: Today, you will learn the basics of solving both trivial and more challenging business tasks using basic Java language constructs.

Version: 1.0

# Contents

# Chapter I

# Foreword



Java Man, or Homo erectus erectus

# Chapter II

# General Rules

- Use this page as the only reference. Do not listen to any rumors and speculations about how to prepare your solution.
- Now there is only one Java version for you, 1.8. Make sure that compiler and interpreter of this version are installed on your machine.
- You can use IDE to write and debug the source code.
- The code is read more often than written. Read carefully the document where code formatting rules are given. When performing each task, make sure you follow the generally accepted Oracle standards
- Comments are not allowed in the source code of your solution. They make it difficult to read the code.
- Pay attention to the permissions of your files and directories.
- To be assessed, your solution must be in your GIT repository.
- Your solutions will be evaluated by your piscine mates.
- You should not leave in your directory any other file than those explicitly specified by the exercise instructions. It is recommended that you modify your .gitignore to avoid accidents.
- When you need to get precise output in your programs, it is forbidden to display a precalculated output instead of performing the exercise correctly.
- Have a question? Ask your neighbor on the right. Otherwise, try with your neighbor on the left.
- Your reference manual: mates / Internet / Google. And one more thing. There's an answer to any question you may have on Stackoverflow. Learn how to ask questions correctly.
- Read the examples carefully. They may require things that are not otherwise specified in the subject.
- Use "System.out" for output

- And may the Force be with you!
- Never leave that till tomorrow which you can do today ;)

# Chapter III

# Rules of the day

- User-defined methods and classes are prohibited for all tasks of the day, except for user-defined static functions and procedures in the main class file of the solution.
- All tasks contain a list of ALLOWED language constructs for the specific task.
- System::exit may be used for all tasks.
- All tasks contain an example of how the application operates. The implemented solution must be identical to the specified output example for current input data.
- For illustration purposes, the data entered by the user in task examples are preceded by an arrow (->). Do not take account of these arrows when implementing a solution!

P.S. Some tasks require a non-trivial approach because of the above-mentioned limitations. These limitations will teach you how to find solutions for automating actual business processes.

# Chapter IV

## Exercise 00 : Sum of Digits

| | |
|---|---|
| | Exercise 00 |
| Sum of Digits | |
| Turn-in directory : *ex*00/ | |
| Files to turn in : Program.java | |
| Allowed functions :<br>Input/Output : System.out<br>Types : Primitive types<br>Operators : Standard operations of primitive types | |

Java is a strictly typed programming language. Fundamental data types (boolean, character, integer, floating point number) are represented in Java by eight primitive types: boolean, char, byte, short, int, long, float, double.

Work with integer type.

- Calculate the sum of digits of a six-digit int number (the number value is set directly in the code by explicitly initializating the number variable).

Example of the program operation for number 479598:

```
$ java Program
42
```

# Chapter V

# Exercise 01 : Really Prime Number

| | |
|---|---|
| | Exercise 01 |
| Really Prime Number | |
| Turn-in directory : *ex*01/ | |
| Files to turn in : Program.java | |
| Allowed functions :<br>Input/Output : System.out, System.err, Scanner(System.in)<br>Types : Primitive types,<br>Operators : Standard operations of primitive types, conditions, loops | |

According to Böhm-Jacopini theorem, any algorithm can be written using three statements: sequence, selection, and iteration.

- Using these statements in Java, you need to determine if the input number is a prime. A prime is a number which has no dividers other than the number itself and 1.
- The program accepts the number entered from the keyboard as input and displays the result of checking whether that number is a prime. In addition, the program shall output the number of steps (iterations) required to perform the check. In this task, an iteration is a single comparison operation.
- For negative numbers, 0 and 1, display theIllegalArgument message and shut down the program with the -1 code.

Example of program operation:

```
$ java Program
-> 169
   false 12

$ java Program
-> 113
   true 10

$ java Program
-> 42
   false 1
```

```
$ java Program
-> -100
    IllegalArgument
```