

# Language Scala - programming I

Course code: SCALA\_INTRO

The graduate will be thoroughly familiar with OOP and functional techniques and their use in a strongly typed language. The course starts with the basic constructs of the language, continues with OOP, then functional transformations, the type system of the language, and ends with asynchronous programming.

## Required initial knowledge:

- The course assumes knowledge of any other programming language and the basics of algorithmization.

## Teaching methods:

- Interpretation with presentation
- Practical demonstrations on small pieces of code, so-called "boards" or scratches in IntelliJ Idea
- Small exercises to test what has just been explained
- Large independent tasks from the thematic unit evaluated individually for each student (with advice on how to proceed)

## Study materials:

- Presentation of the subject matter in printed or online form.

## Syllabus:

Installation of necessary tools/Scala and introduction

- Scala-cli installation
- Scala in IntelliJ Idea (Scala plugin)
- What is Scala, a small sample of the aims and direction of the course, also with practical examples
- Creating a project

Basic constructions of the language and an introduction to types

- Mutable vs immutable variables - the immutable principle
- Basic types: numeric, strings, truth values
- Terms and cycles
- For cycle that returns a result, theory of expressions - what is an expression and what is not
- Functions, recursion, tail recursion
- String interpolation
- Tuple type, "breaking" into variables, pattern matching
- Type option as a one-element collection, chaining options in the for loop, Option(null)
- List/Seq/Vector/Set/Map and operations on them
- Mutable variants of containers

Operations on collections

- foreach, map, flatMap
- recursive counting with collections, foldLeft, sum, reduce
- find, headOption, filter, exists, contains, collect, groupBy, mkString

OOP in Scala

- Class and its attributes, constructor, companion object and apply
- case class, copy method, pattern matching
- Traits and multiple inheritance
- Anonymous classes
- sealed trait and enumerations

More advanced constructions of the language

- Higher functions, function as parameter and return value
- When is the code actually called/evaluated?

### GOPAS Praha

Kodaňská 1441/46  
101 00 Praha 10  
Tel.: +420 234 064 900-3  
[info@gopas.cz](mailto:info@gopas.cz)

### GOPAS Brno

Nové sady 996/25  
602 00 Brno  
Tel.: +420 542 422 111  
[info@gopas.cz](mailto:info@gopas.cz)

### GOPAS Bratislava

Dr. Vladimíra Clementisa 10  
Bratislava, 821 02  
Tel.: +421 248 282 701-2  
[info@gopas.sk](mailto:info@gopas.sk)



Copyright © 2020 GOPAS, a.s.,  
All rights reserved

# Language Scala - programming I

- Lambdas
- Wrapping primitive values into types instead of using them directly
- implicit functions, conversions
- implicit classes, method addition
- default values
- Try vs try
- Chaining of potentially unsuccessful operations in the for loop, recover, orElse
- Function with type parameter
- Delimitation of types
- Our own reducer

## Asynchronous programming

- Futures
- Await.ready/result
- Execution Context (global, fixed thread pool, cached, work stealing pool)
- Future does not mean thread, What is a thread pool
- map, flatMap, folding in the for loop
- andThen, recover, transform
- laziness

### **GOPAS Praha**

Kodaňská 1441/46  
101 00 Praha 10  
Tel.: +420 234 064 900-3  
[info@gopas.cz](mailto:info@gopas.cz)

### **GOPAS Brno**

Nové sady 996/25  
602 00 Brno  
Tel.: +420 542 422 111  
[info@gopas.cz](mailto:info@gopas.cz)

### **GOPAS Bratislava**

Dr. Vladimíra Clementisa 10  
Bratislava, 821 02  
Tel.: +421 248 282 701-2  
[info@gopas.sk](mailto:info@gopas.sk)



Copyright © 2020 GOPAS, a.s.,  
All rights reserved