AGH UNIVERSITY OF SCIENCE AND TECHNOLOGY

| | | | | | |
|---|---|---|---|---|---|
| Code: | UBPJO-232 | Module name: | Functional programming with Scala | | |
| Academic year: | 2017/2018 | Semester: | Spring, Fall | ECTS credits: | 6 |
| Programme: | Physics and Applied Computer Science | | | | |
| Course homepage: | | Lecture language: | English | | |
| Responsible teacher: | dr hab. inż. Bołd Tomasz (tomasz.bold@fis.agh.edu.pl) | | | | |
| Academic teachers: | dr hab. inż. Bołd Tomasz (tomasz.bold@fis.agh.edu.pl) | | | | |

### Module summary

The functional programming regains popularity in the context of the big data and high-reliability applications. One of mainstream programming languages in this domain is Scala. It is a language from the family rooted in Java. Scala offers a complete support for all functional concepts, modern object-oriented mechanisms and well thought over generics. A comprehensive overview of the Scale is given in the course.

## Description of learning outcomes for module

| MLO code | Student after module completion has the knowledge/ knows how to/is able to | Method of learning outcomes verification (form of completion) |
|---|---|---|
| Social competence | | |
| M_K001 | Student is able to provide a compelling explanation as to why he/she choose a certain approach | Activity during classes |
| Skills | | |
| M_U001 | Student is able to compile Scala programs and solve the syntactical and semantical issues on the way to the solution | Execution of laboratory classes |
| Knowledge | | |
| M_W001 | Student understands language concepts | Execution of laboratory classes |
| M_W002 | Student is able to structure larger projects and choose between functional and object-oriented approaches | Execution of a project |

## FLO matrix in relation to forms of classes

| MLO code | Student after module completion has the knowledge/ knows how to/is able to | Form of classes | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Lectures | Auditorium classes | Laboratory classes | Project classes | Conversation seminar | Seminar classes | Practical classes | Fieldwork classes | Workshops | Others | E-learning |
| Social competence | | | | | | | | | | | | |
| M_K001 | Student is able to provide a compelling explanation as to why he/she choose a certain approach | - | - | + | - | - | - | - | - | - | - | - |
| Skills | | | | | | | | | | | | |
| M_U001 | Student is able to compile Scala programs and solve the syntactical and semantical issues on the way to the solution | - | - | + | - | - | - | - | - | - | - | - |
| Knowledge | | | | | | | | | | | | |
| M_W001 | Student understands language concepts | + | - | + | - | - | - | - | - | - | - | - |
| M_W002 | Student is able to structure larger projects and choose between functional and object-oriented approaches | + | - | - | - | - | - | - | - | - | - | - |

## Module content

**Lectures**

Introduction to the course

The scope and organization of the lectures, organization of the labs, student projects. Functional programming introduction, Scala with the Hello World example. REPL with help.

Walk through the basics

Scala code structure, documentation, values and variables, basic types, expressions, control statements, iteration and generators. Pre-, post- and in-fix operators.

Functions

Declaration and definition, return value, return statement and implicit return value, function invocation, function arguments, named function arguments, anonymous functions.

Advanced topics functions

Nested functions, partially applied functions, higher order functions, recursions, head- and tail- recursions. Functions as a custom control statements. Partial functions.

Object oriented programming

Classes and objects. Class attributes, methods and procedures, access control, construction. Company objects. Class properties. Functional classes.

Polymorphism
Inheritance and Scala class hierarchy, abstract classes. Polymorphic methods. Multiple inheritance and traits. Rich interfaces. Tailored behaviors through anonymous classes. Traits in stacking modifications.

Packages and implicit conversions
Declaring implicit conversions. Use cases for conversions. Packaging and importing code. Configurable packages.

Pattern matching
Case classes: definition and uses, sealed class hierarchies.  Matching: constants, variables, case classes, types. Extractors.

Generics
Functions, classes, variance definition, abstract attributes.

Scala collections library
Mutable and immutable containers, rich collections interface. Views.

Code testing

GUI
Swing library: application structure, GUI elements overview, arrangement. Handing user interactions through events and reactors.

Selected small subjects
Filling the gaps: exceptions, XML, objects comparison. Monads. Scala in big data.

Summary and outlook
Reliability of Scala programs. Applications in big data market.

**Laboratory classes**

Laboratories
During laboratories, students shall complete two exercises. First, covering material that was already covered in the previous lab. This task should be performed without looking at any additional material than student’s own exercises. This part, when completed successfully, results in 6 points. About 30-40 minutes is allotted for this exercise. The second exercise will cover new material discussed in the last lecture. Successful completion of this exercise provides 4 points.  Students can look up any material they wish during this exercise except forums, chats etc. while the language documentation, lectures are accepted.
During some labs (3-4) the first part will be omitted for the sake of more interesting problem to be solved.
Larger student projects are possible as a way to improve the final grade or compensate for absences.

## Method of calculating the final grade
The final grade is just a grade from the laboratories. Additional student projects are possible to increase the grade. They will be also needed to compensate for absences.

## Prerequisites and additional requirements
Some experience in object oriented programming and design is an advantage, though not required. During labs we will use a Linux, therefore, elementary user skills will be required.

### Recommended literature and teaching resources
Scala is well documented online. A language overview books would be useful for students to read though.  For self-education a computer with any operating system is good enough.

### Scientific publications of module course instructors related to the topic of the module
Additional scientific publications not specified

### Additional information
Advertisments:
https://ileriseviye.wordpress.com/2014/12/22/how-and-why-is-scala-used-in-aerospace-industry/
https://www.indeed.com/q-Scala-jobs.html

## Student workload (ECTS credits balance)

| Student activity form | Student workload |
|---|---|
| Participation in lectures | 30 h |
| Participation in laboratory classes | 30 h |
| Completion of a project | 45 h |
| Preparation for classes | 45 h |
| Summary student workload | 150 h |
| Module ECTS credits | 6 ECTS |