Functional Programming

1. **Course number and name:** 020PFSES3/020FPRES3 Functional Programming

2. **Credits and contact hours:** 4 ECTS credits, 2x1:15 contact hours

3. **Name of course coordinator:** Youssef Bakouny

4. **Instructional materials:** Slides; Moodle Ressources, Assignments, GitHub repos with code developed in class, Massive Online Open Courses (MOOCs) on Coursera: *Functional Programming in Scala Specialization by Martin Odersky*.

5. **Specific course information**
   a. **Catalog description:**
      The goal of this course is to introduce the functional programming paradigm using, mainly, the Java programming language. It also illustrates some functional programming concepts in Python and introduces Scala as mutli-paradigm hybrid programming language.
      The course begins with an overview of functional programming followed by a gradual exposition of the evaluation model (used to reason about functional programs) alongside the explanation of the following concepts: recursion and the optimization of recursive functions, the use of functions as values, the partial application of functions, object immutability and its advantages, types and pattern matching, pairs and tuples, lists and functional collections, combinatorial search problem solving using for-expressions, lazy evaluation, functional streams, infinite sequences, the variance of polymorphism with regards to inheritance and a brief of overview of key monad such as Option, Try and Future. These concepts will be illustrated by examples and exercises in Java, Python and Scala. Finally, the course will end with an introduction to program proving using structural induction.

   b. **Prerequisites:** 020CPPES1/020OOPES1 Object-Oriented Programming

   c. **Selected Elective** for CCE students

6. **Educational objectives for the course**
   a. **Specific outcomes of instruction:**
      - Explain the difference between the functional programming paradigm and the imperative programming paradigm.
      - Implement a program using the main concepts of the functional paradigm.
      - Analyze a functional program in terms of correctness, maintainability and performance.
      - Design and implement a functional program in response to a complex problem.
      - Evaluate the quality of a code in terms of maintainability and propose an adequately refactored code.

   b. **PI addressed by the course:**

PI	1.3	2.3	2.5	6.4
Covered	x	x	x	x
Assessed	x	x	x	x

7. **Brief list of topics to be covered**
   - A comparison of programming paradigms: functional and imperative An introduction to functional programming and the evaluation model (2 lectures)
   - An introduction to IntelliJ IDEA, Visual Studio Code and PyCharm The definition and use of functions Recursion and the termination of recursive functions Tail recursive functions The use of functions as values Higher Order Functions Currying and the partial application of functions (4 lectures)
   - The definition and use of immutable objects Test Driven Development applied to functional object-oriented programming The use of build automation tools, Git and GitHub (4 lectures)
   - Types, generics, variance and pattern matching (4 lectures)
   - Immutable linked lists and higher order functions on lists Reduction on lists Pairs and tuples (3 lectures)
   - Immutable collections, the resolution of combinatorial search problems and the use of for-expressions (3 lectures)
   - Lazy evaluation, functional streams and infinite sequences (3 lectures)
   - Introduction to static analysis and proof of programs (1 lecture)