Course title: Computational Empowerment - OPT  
Course year: 2nd  
Semester: 2nd  
Course Code: 54129  
Scientific sector: INF/01  
Lecturer: Rosella Gennari  
gennari@inf.unibz.it  
Module: No  
Lecturer other module: /  
Credit Points: 3  
Total lecturing hours: 30  
Office hours: from Monday to Friday on request  
Attendance: according to the regulations  
Teaching Language: English  
Propaedeutic course:  
Course description: The type of course is caratterizzante, in the areas of computer science and engineering.  

By following the latest European Commission and national recommendations and guidelines on computational thinking and computing education, the course gives a general overview of scientific contents and computing technologies, which are relevant for tomorrow's educators and communication experts. The overall goal of the course is to empower different students to tackle a simple computational problem and develop a solution for it, critically and collaboratively.  

The specific objectives to achieve the goal are as follows.  

1) First, the course aims to provide participants with basic knowledge of computing to understand a basic computational problem, that is, to analyse it and abstract away what needed for developing a basic computing solution for it.  

2) Second, the course aims to enable students to develop basic computing solutions for different problems, which requires them to specify and program them.  

3) Third the course aims to enable students to collaborate in the analysis of problems and development of solutions, with peers and Artificial Intelligent (AI) agents/chatbots (e.g.,
**chatGPT**, and to critically reflect on what they are using and doing.

### List of topics covered:

1. Introduction to: different computing devices, their hardware and software; computer organisation; data hierarchy; machine languages, assembly languages, high-level programming languages; programming paradigms, with a focus on the structured programming paradigm.
2. Introduction to a lean and efficient implementation of the Python programming language that is optimised to run on different and constrained computing devices.
3. Basic data types, variables, constants, operators and expressions; standard input/output handling; control flow structures; error handling; if time allows for it, basic data structures and subroutines.

The above is tackled for covering the basics of computing to critically understand a computational problem and develop a resolution in a Python-based programming language, in collaboration with peers and existing AI technologies.

### Teaching format:
In-presence, workshop-based.

### Learning outcomes:

- **Knowledge and understanding:**
  - Know fundamental principles of computing.
  - Know different models of computation and computing devices.
  - Have a basic knowledge of programming for different computing devices.
  - Have an understanding of how to efficiently interact with AI chatbot technologies to co-develop computing solutions.

- **Applying knowledge and understanding**
  - Be able to analyse basic computational problems.
  - Be able to specify one among many computational solutions.
  - Be able to program computational solutions.
  - Be able to understand computational solutions, developed in collaboration with other peers or AI technologies.

- **Making judgements**
  - Be able to collect and interpret useful data and to judge computational solutions and their applicability.
  - Be able to identify critical aspects in the development process and take a critical stance towards what is developed.

- **Communication skills**
  - Be able to describe and motivate their choices.
  - Be able to properly document a computing solution.

- **Learning skills**
  Be able to learn how to critically approach AI technologies and techniques.
  Be able to deepen their computing knowledge in autonomy and understanding the relevant literature.

### Assessment:
Project work, defined during the course.

### Evaluation criteria and criteria for awarding marks:
The criteria are as follows:

- Ability to understand a computational problem, abstract away what needed, specify, and develop a possible computational solution.
- Ability to present the solution and critically assess it.
- Ability to critically reflect in the development process.

<table>
<thead>
<tr>
<th>Required readings:</th>
<th>Material provided by the lecturer.</th>
</tr>
</thead>
</table>