# Information Technology - <u>advanced</u> - syllabus

**Course period:** 4th module (April - May 2018)
**Course description:** This will be an advanced course, where some notions of Information Theory and Computer Science will also be studied, extending the study of Information Technology.
The course will assume some working knowledge of computers and the internet.
It will start with brief theoretical general definitions about computers hardware, software and networks.
It will then give some theoretical notions for sound code design, illustrating the different coding approaches. It will give knowledge on programming, with both theory and applications. The programming will be done mostly in the Python language, with some examples of C language. The programming will be done in practical sessions in the computer room.
There will be theoretical classes on Probability Theory and Statistics, Information Theory, Complexity Theory.

**The aim of this course:** To give to the students a theoretical base of concepts to correctly understand what is a computer, how it works, and what are its potentials and its limits. To give the knowledge, the skills, and hands-on experience, to be able to write good computer code, for applications in Social Sciences and Humanities.
To give some theoretical knowledge of Probability Theory and Statistics, Information Theory and Complexity Theory.
**Objectives:**

- General concepts and definitions of computers hardware, software and networking, for understanding and troubleshooting.
- Discrete knowledge of programming techniques, with emphasis on applications on Social Sciences and Humanities.
- Python language programming, theory and practice.
- Notions of Probability and Statistics.
- Notions of Information Theory and Complexity Theory.

The course structure (number of lectures, seminars): The course will develop over 8 weeks, with **32 classes** of 90', of which **18** will be in the form of lectures (frontal teaching) and **14** will be in the form of Computer Lab Practice. There will be ? lectures and ? seminars per week. Most of the seminars will be conducted in a computer room, where each student will be seating in front of a computer.
**Author of the syllabus:** Fabio Grazioso.
**Lecturer (name and surname):** Fabio Grazioso.
Seminar's professor (name and surname): Fabio Grazioso.
**Course language:** English.
**Course requirements:** Intermediate knowledge of computer usage MS-Windows operations are assumed. Intermediate mathematics (algebra, analysis) knowledge is required.
**Ground rules:** Attendance to 85% of the lectures is required. 1 intermediate assignment (homework) and 1 final assignment will be required.
Sample exam's questions: What is a general definition of an Operative System? What is the meaning of "data packet" talking of computer network protocols? What is the name of the graphic library used for data plotting in Python? What is a regular expression? What is the difference between direct methods and iterative methods in Numerical Analysis? What is the bayesian approach to probability theory? What is the definition of Information? What is the definition of Computaitonal Complexity?
**Reference Textbooks and other material:**

1) Fox - Information Technology: An Introduction for Today's Digital World (2013).
2) Vermaat, Sebok, Freund - Discovering Computers 2017 (2016).
3) Tanenbaum, Bos - Modern Operating Systems (2014).
4) Harris, Harris - Digital Design and Computer Architecture (2nd Ed) (2012).

5) Mueller - Windows command line administration instant reference (2010).
6) Gillam - Unicode Demystified (2002).
7) Hunt - TCP-IP network administration (2002).
8) Martin - Clean Code- A Handbook of Agile Software Craftsmanship (2008).
9) Cormen - Algorithms unlocked (2013).
10) Sauer - Numerical Analysis (2011).
11) Atkinson - An Introduction to Numerical Analysis (1989).
12) Burden, Faires, Burden - Numerical Analysis (2016).
13) Papoulis, Pillai - Probability, Random Variables and Stochastic Processes (2002).
14) Jaynes - Probability Theory The Logic of Science (2003).
15) Lutz - Learning Python (2013).
16) Langtangen - A primer on scientific programming with Python (2012).
17) Friedl - Mastering Regular Expressions (2006).
18) Baecker - Readings in human-computer interaction (1995).
19) Norman - The Design of Everyday Things (2002).
20) Mackay - Information Theory, Inference, and Learning Algorithms (2004)
21) Sipser - Introduction to the Theory of Computation [3rd Ed] (2012).

(note: the books list is very long, but from most of the books only small parts, sometimes just few pages, will be considered.)

In the following you can find a summary table of the main topics and the detailed schedule.

**Information Technology - Advanced course summary**

| |
|---|
| **General introduction to computers and networks.**<br>General architecture of a computer. The machine representation of numbers and errors. The operating system. MS-DOS, Microsoft Windows, Apple MacOS. Main productivity software, Scientific Typesetting, Graphic editors. Audio editors.<br>Networking protocols, layers, packets. TCP/IP protocol. HTML. Protocols: email, FTP. |
| **Introduction to programming.**<br>Software design techniques. Algorithms. Methods for code development: top-down and bottom-up. Main programming statements and pseudo-code. Procedural programming, structured programming and object-oriented programming. |
| **Programming languages.**<br>Main programming language types: machine code, low-level, high-level, interpreted, compiled. Syntax and semantics. |
| **Elements of the C language.**<br>Keywords and syntax. Control flow statements. Data types. Pointers. Code examples. |
| **The Python language.**<br>Keywords and syntax. Control flow statements. Data types. Libraries: NumPy, SciPy, Matplotlib. Examples of data processing. Examples of data plotting and representation: 2D plots, 3D plots, annotations. Datafiles input and output. Strings manipulation. Regular Expressions. |
| **Numerical analysis.**<br>Direct methods and iterative methods. Precision. Example algorithms: sorting, searching, root-finding, linear equation systems solving, functions interpolation, functions integration. |
| **Statistics and Probability.**<br>Fundamentals of statistics. Fundamentals of probability theory. Bayes theorem. |
| **Data analysis and Artificial Neural Networks.** |

| |
|---|
| Classification problems. Regression Problems. Elements of neural networks. |
| **Information theory.**<br>Definition of information. Information compression: typical set, source coding theorem. Information transmission: the noisy channel, error correction, noisy-channel coding theorem. |
| **Complexity theory.**<br>Definition of computational complexity. Complexity models: oracle model, circuit complexity, time complexity. Some example of complexity classes. Description of the “P vs NP” problem |

**Curse classes**
(type: L= lecture, P = practice)

| # | type | topics |
|---|---|---|
| 1 | L | General architecture of a computer. The Von Neumann machine. The representation of information in a computer (hardware). |
| 2 | L | The machine representation of numbers and errors (logical). The binary numbers, the hexadecimal numbers. Encodings. |
| 3 | L | The operating system in general. MS-DOS, Linux/Unix. Graphical Operating Systems: MS Windows, Apple MacOS, KDE, Gnome. Programming environments: editor, debugger, compiler, interpreter. |
| 4 | L | Networking protocols, layers, packets. TCP/IP protocol. World Wide Web, HTML, browsers. Search engines. Other protocols: email, FTP, servers, clients. |
| 5 | P | Computer Lab practice on internet resources: academic resources, advanced search engines practice. |
| 6 | L | Main programming language types: machine code, low-level, high-level, interpreted, compiled. Syntax and semantics. |
| 7 | L | Programming environments: editor, debugger, compiler, interpreter. Software design techniques. Algorithms. |
| 8 | L | Elements of the C language: keywords and syntax, control flow statements, data types. |
| 9 | L | The Python language. Keywords and syntax. Control flow statements. Data types. |
| 10 | P | Computer Lab practice on Python programming: first simple algorithms. |
| 11 | L | Python Libraries: NumPy, SciPy. Examples of mathematical functions, mathematical data generation and manipulation. Examples of simple scientific plots. |
| 12 | P | Computer Lab practice on Python programming: first scientific applications. |
| 13 | P | Computer Lab practice on Python programming: more scientific applications. |
| 14 | L | Python data processing. Examples of data plotting and representation: 2D plots, 3D plots, annotations. |
| 15 | P | Computer Lab practice on Python programming: scientific data plotting. |
| 16 | L | Python datafiles input and output. Text files and binary files. Strings |

| | | |
|---|---|---|
| | | Manipulation. Regular Expressions. |
| 17 | P | Computer Lab practice on Python programming: data files input and output. |
| 18 | L | Numerical analysis. Direct methods and iterative methods. Precision. Example algorithms: sorting, searching. |
| 19 | L | More numerical analysis. Root-finding, linear equation systems solving, functions interpolation, functions integration. Discrete Mathematics. |
| 20 | P | Computer Lab practice on Python programming: applications on Numerical analysis and discrete mathematics. |
| 21 | P | Computer Lab practice on Python programming: applications on Numerical analysis and discrete mathematics. |
| 22 | L | Statistics and Probability. Fundamentals of statistics. Fundamentals of probability theory. Bayes theorem. |
| 23 | L | Data analysis. Classification problems. Regression Problems. Elements of neural networks. Examples of applications to classification and regression Problems. |
| 24 | L | Definition of information. Information compression: typical set, source coding theorem. |
| 25 | L | Information transmission: the noisy channel, error correction, noisy-channel coding theorem. |
| 26 | L | Complexity theory. Definition of computational complexity. Complexity models: oracle model, circuit complexity, time complexity. Some example of complexity classes. Description of the “P vs NP” problem |
| 27 | P | Computer Lab practice on Python programming: Examples from the course. |
| 28 | P | Computer Lab practice on Python programming: Examples from the course. |
| 29 | P | Computer Lab practice on Python programming: Examples from the course. |
| 30 | P | Computer Lab practice on Python programming: Examples from the course. |
| 31 | P | Computer Lab practice on Python programming: Examples from the course. |
| 32 | P | Computer Lab practice on Python programming: Examples from the course. |