UNIVERSITY OF L'AQUILA

Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica

Università degli Studi dell'Aquila

1st Cycle Degree in COMPUTER SCIENCE

Laurea in INFORMATICA

Course Catalogue

Academic year starts the last week of September and ends the first week of June.
1st Semester - *Starting date:* last week of September, *end date*: 3rd week of January
2nd Semester - *Starting date:* last week of February, *end date:* 1st week of June
Exams Sessions: I) from last week of January to 3rd week of February, II) from 2nd week of June to end of July, III) from 1st to 3rd week of September

Comprehensive Scheme of the First Cycle Degree in COMPUTER SCIENCE				
YEAR	**CODE**	**COURSE**	**Credits (ECTS)**	**Semester**
I	F0035	Computer Architecture	6	1
	DT0002	Mathematical Analysis	9	1
	F0050	Introduction to Programming: Foundations and Languages	12	1
	F0633	Physics	6	2
	F0143	Discrete Mathematics	12	2
	F0123	Computer Architecture Laboratory	6	2
	F0127	*English Language*	3	2
II	F0130	Algorithms and Data Structures with Lab	12	1
	F0133	Operating Systems with Lab	12	1
	DT0003	Probability and Statistics	6	1
	F0136	Data Bases with Lab	12	2
	F0139	Operations Research and Combinatorial Optimization	12	2
	F0142	Programming Principles and Paradigms	6	2
III	F0150	Computability and Complexity Theory	6	1
	F0144	Computer Networks	6	1
	F0149	Web Technologies	6	1
	F0151	Programming Languages and Compilers	6	2
	F0146	Software Engineering with Lab	9	2
		Optional Courses within a selected set	12	1/2
		Free choice Courses	12	1/2
		Other Activities	6	1/2
		Thesis	6	2

List of Optional Courses of First Cycle Degree in COMPUTER SCIENCE				
YEAR	**CODE**	**COURSE**	**Credits (ECTS)**	**Semester**
III	F0057	Algorithms for Distributed System	6	1
	F0159	Advanced Computer Networks: Architecture	6	1
	F0156	Software Engineering II	6	1
	F0162	Web Engineering	6	2
	F0072	Artificial Intelligence	6	1
	F0157	Models and Algorithms for Financial Management I	6	1
	F0993	Models and Algorithms for Financial Management I	6	1
	F0164	Neural Networks	6	1
	F0083	Advanced Computer Networks: Internetworking	6	2
	DT0043	Object Oriented Software Design	6	2
	F0158	Information Theory	6	1
	F1081	Mobile applications development	6	2
	DS9001	Bioinformatics	6	2
	DS9003	Information systems and network Security	6	2

Programme of "ARCHITETTURA DEGLI ELABORATORI" ***"COMPUTER ARCHITECTURE"***		
F0035, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 1st Year, 1st Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Michele FLAMMINI		
1	Course objectives	Provide the students with knowledge of the computer architecture, ability to analyze and devise combinatorial and sequential modules, capacity to individuate and dimension the fundamental computer components, knowledge of the basic set of machine instructions and their execution modalities.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Integer and real numbers. Integer and real numbers arithmetic. - Boole's Algebra. Boolean functions and expressions. Minimization of boolean functions and Karnaugh's maps. Analysis and synthesis of combinatorial networks. Fundamental operators and combinatorial modules. - Synchronous and asynchronous flip-flops. Registers and counters. Sequential networks: state diagrams and flow tables. Analysis and synthesis of synchronous sequential networks. - Machine and assembly languages. Instructions set and architectural models. RISC and CISC instructions sets. Addressing of data and control. - CPU: internal structure and components. Fetch and execute phases. Cabled and microprogrammed realization of the control unit. CPU performances. System bus: mechanical, electrical and logic-functional characteristics. Bus allocation. - Memory: classification criteria. Main memory, cache and magnetic disks. Locality principle and hierarchical organization. I/O subsystem: interfaces, I/O ports and their addressing. Peripherical devices management methods: programmed, interrupt driven and with direct memory access (DMA). On successful completion of this module, the student should : o have knowledge about (i) how computers represent data and information, (ii) fundamentals of logic and digital systems; (iii) fundamental components of computer architectures such as CPUs and memory devices. o understand the main issues arising from the design of efficient computing systems, including programming aspects.

		o be capable of: (i) understanding and representing information of computers; (ii) analyzing and devising combinatorial and sequential modules; (iii) solving practical problems related to the design process of the different components of a standard computer architecture; (iv) estimating the global performances of a standard computing architecture; (v) understanding basic aspects of computer programming, from bottom (machine coding) to top (instructions and programming languages) level. o acquire skills to deal with real world computer architectures, to identify problems and to, independently, choose the corresponding most efficient solution, as known from the literature. o know how to design and program basic computer architectures. o explain and illustrate the fundamental notions studied in this course o demonstrate ability in solving concrete computer architectures related problems, focusing on their main features and discarding the inessential ones. o acquiring competencies and abilities necessary in their future studies, especially with respect to studies on operating systems, computer networks and complex architectures topics.
3	Prerequisites and learning activities	Ability to integrate classroom and homework study, ability to interact with the teacher during the class for originating discussion. Fundamentals of mathematics.
4	Teaching methods and language	Lectures, seminars, lab Language: Italian, English Ref. Text books - Giacomo Bucci, *Calcolatori elettronici. Architettura e organizzazione*. McGraw-Hill, 2009.
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website. Formative Assessment The formative assessment is performed via interaction between teacher and students during lectures. Students are involved in questioning and discussion, by means of open oral questions to the entire class. Summative Assessment consists of written test followed by an oral exam. An optional mid-term written test will be also provided, which is meant to cover the first part of the course, in order to help the students to split the workload. If a student passes the mid-term written exam, she will take a final-term written exam concerned with the second part of the course content only. The mid-term written exam (lasting 2 hours) consists of exercises and open questions concerning the first part of the course content. The final-term written exam is split into two parts (each lasting one hour and half), each consisting of exercises and open questions, concerning the first and the second part of the course content, respectively. Students who passed the mid-term part will have to take only the second part. The final result of the written exam will be given by the average result of the two parts. The oral exam will occur within the same exam session of the written test, and it will typically cover the areas of the written answers that need clarification, plus a subject of one's choice. The oral exam (max 1 hour) will test the student's ability to engage in discussion of issues relevant to the topics discussed during the course. Criteria of evaluation will be the level of knowledge and the fluency in the technical language of computer architecture.

Programme of "ANALISI MATEMATICA" "MATHEMATICAL ANALYSIS"		
DT0002, COMPULSORY First Cycle Degree in COMPUTER SCIENCE, 1st Year, 1st Semester		
Number of ECTS credits: 9 (workload is 225 hours; 1 credit = 25 hours)		
Teachers: Klaus ENGEL, Marta MACRÌ		
1	Course objectives	To give students a rigorous understanding of the theory of real- and vector-valued functions. Students will acquire an understanding of basic properties of the field of real numbers, concepts of infinity, limits of functions and methods for calculating them, continuity, differentiation, integration and Taylor series.
2	Course content and Learning outcomes (Dublin	Topics of the module include: - Set theory (notations, basic concepts), Real numbers (basic properties, interval notation),

	descriptors)	mathematical induction, - Sequences and series (convergence, divergence and irregularity, convergence criteria), - Functions (injectivity, surjectivity, invertibility, composition), - Limits (basic definitions, the Sandwich Rule, boundedness), - Continuity (basic definitions, the Intermediate Value Theorem, numerical methods for solving equations), - Differentiation (basic definitions, rules and properties, Rolle's Theorem, the Mean Value Theorem), L'Hopital's Rule (techniques and applications), Taylor's Theorem (generalisation of the Mean Value Theorem, polynomial approximations to functions, convergence criteria), - Integration (basic properties, the Riemann definition, the Fundamental Theorem of Calculus, integration by parts and substitution, calculation of improper integrals), - Limits and continuity of functions of several real variables (basic techniques, polar coordinates), - Differentiation of real- and vector-valued functions of several real variables (partial derivatives, gradient, differential, Jacobi matrix), - Integration of real functions of several real variables (simple domains, Fubini-Tonelli's Theorem, integration by substitution). Exercises and application of methods for solving problems, support the learning process throughout the theoretical course. On successful completion of this module, the student should : o have a good knowledge and understanding of basic properties of real numbers, functions, and finite and infinite sets; o demonstrate an understanding of basic topics in the analysis of functions, including limits, continuity, differentiation, Taylor-MacLaurin series, and integration; o be able to apply his knowledge and understanding to tackle basic problems from applied mathematics and engineering; o understand formal mathematical definitions and theorems, and apply them to prove statements about functions; o demonstrate skills in mathematical reasoning and ability to conceive a proof; o be able to explain the main notions and results of mathematical analysis; o demonstrate capacity to read and understand advanced texts.
3	Prerequisites and learning activities	Basic mathematical notions and methods as learnt at high school
4	Teaching methods and language	Lectures and exercise classes Language: Italian Reference textbooks - Klaus Engel, *Appunti del Corso di Analisi Matematica*. *http://univaq.it/~engel/ana1.pdf* - A.Marson, P.Baiti, F.Ancona, B.Rubino, *Corso di Analisi Matematica 1*. Carocci. - P.Marcellini, C.Sbordone, *Esercitazioni di Matematica*. Liguori. - S.Salsa, A.Squellati, *Esercizi di Matematica*. Zanichelli. (vol. 1) - M.Bramanti, C.D.Pagani, S.Salsa, *Matematica*. Zanichelli.
5	Assessment methods and criteria	<u>Pre-Assessment</u> The course prerequisites are stated on the module website and are verified by an entrance test. <u>Formative Assessment</u> The formative assessment is performed via interaction between the teacher and the students during the lectures. Students are involved in questioning and discussions by means of open questions to the entire class. <u>Summative Assessment</u> The summative assessment consists of a written test followed by an optional oral exam. The written test consists of a set of two questions and five exercises in order to assess (i) whether the student knows the basic concepts and results of Mathematical Analysis; (ii) the student's practical skills in selecting and applying suitable methods for problem solving. Criteria of evaluation are the level of knowledge, the correctness of solutions, the proper use of mathematical language and the clarity and completeness of explanations. The oral exam can be asked either by the student to improve the mark or by the teacher in presence of serious mistakes or misunderstandings in the written test. The oral exam takes place within the same exam session of the written test. It typically covers the parts of the written test that need clarification and, eventually, additional subjects proposed by the teacher.

Programme of "FONDAMENTI DI PROGRAMMAZIONE CON LABORATORIO"
"INTRODUCTION TO PROGRAMMING: FOUNDATIONS AND LANGUAGES"

This course is composed of two modules:
1) Foundations of Programming Languages, 2) Introduction to Programming

F0050, COMPULSORY
First Cycle Degree in COMPUTER SCIENCE, 1st Year, 1st Semester

Number of ECTS credits: 12 (workload is 300 hours; 1 credit = 25 hours)

1) FOUNDATIONS OF PROGRAMMING LANGUAGES (6 ECTS)

Teacher: Paola INVERARDI e Marco AUTILI

1	Course objectives	This course introduces essentials of a high-level programming language. Students, applying rules of Syntax and Semantics, develop the skills in program design, implementation and debugging to solve computational problems in the programming language. The course focuses in particular on the Operational Semantics of a subset of the Java Programming Language.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the Module include: - Programming Languages Syntax: Context-Free Grammars, Derivation Trees, Ambiguity - Programming Languages Semantics: Transition System - Operational Semantics for +/- Java (a subset of the Java language) Upon successful completion of the course, the student will: o know and understand the importance of structured programming concepts as well as good programming practice; o understand the role of primary data structures and algorithms and have the ability to perform operations involving various data structures and implement some simple examples of them; o be able to utilize the basic elements of programming character-based I/O, assignment, loops, conditionals, vectors, functions and parameter-passing in programming practice such as reading, writing and debugging a program; o be able to apply techniques for expression evaluation, and role of operator precedence and associativity in expression evaluation in programming practice; o be able to apply syntax of the programming language to design, implement, test and debug a non-trivial program that solves a practical problem.
3	Prerequisites and learning activities	Knowledge of basic math functions and sets, and logical expressions. Strong integration with the "Introduction to Programming" module.
4	Teaching methods and language	Lectures, Seminars Language: Italian Reference textbooks: -R. Barbuti, P. Mancarella e F. Turini, *Elementi di Semantica Operazionale*. 2004/2005. *https://informatica.di.univaq.it/getres.php?resid=1171* -R. Barbuti, P. Mancarella, D. Pedreschi, F. Turini, *Elementi di Sintassi dei Linguaggi di Programmazione*. Corso di Laurea in Informatica Università di Pisa a.a. 2004/05. *https://informatica.di.univaq.it/getres.php?resid=746* -R. Barbuti, P. Mancarella e C. Montangero, *Semantica Operazionale*. *https://informatica.di.univaq.it/getres.php?resid=747* -M. Autili, P. Inverardi, *Semantica Operazionale di +/- Java - 03 Dicembre 2010* . 2010. *http://informatica.di.univaq.it/getres.php?resid=1053*
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website. Fulfillment of such pre-requisites is verified by formative assessment. Additional lectures or short seminars or individual homework might be provided by the teacher in case significant problems are detected. Formative Assessment The formative assessment is performed via interactive interaction between teacher and students during lectures. Students are aware since the beginning of the Course that they will be involved (in turns) in: - Questioning and discussion, by means of open oral questions to the class or to single students. - Exit Slips: students are assigned written questions or exercises to be answered in 10 minutes, and a student is then selected for oral presentation of her/his solution to the class. Summative Assessment: Written test followed by a mandatory oral exam.

		In order to help the students to split the workload, an optional mid-term written test will also be provided, which is meant to cover the first part of the course. The written test (2 hours) consists in: (a) Two or three short essays, to cover point (i) and (ii), 50% of total marks; (b) One or two exercises, to cover point (iii), 50% of total marks. All parts can result in negative marks if the answer is omitted or seriously flawed. The written test includes exercises on (i) programming languages syntax, i.e., context-free grammars, derivation trees, ambiguity, (ii) programming languages semantics, i.e., transition system, and (iii) operational semantics of +/- Java, i.e., the operational semantics of a subset of the Java programming language. The former aims at verifying the level of knowledge of the foundational concepts underpinning programming languages. It is organized in terms of well-structured and formally-defined questions or open questions to which the student has to answer in detail. The latter is meant to verify the student's ability to apply the foundational concepts on the more practical setting of the Java programming language. Criteria of evaluation will be: (i) the level of knowledge of the course topics and the ability to reason on them in order to devise possible solutions to related problems, hence following a creative approach; (ii) the proper use of technical terminology and of the mathematical concepts underlying the course topics, e.g., functions, algebra, and logic; and (iii) the clarity and completeness of explanations. The oral test (max 1 hour) consists of one question for each serious mistake in the written test (the answer compensates the negative marks obtained therein) and one question for at most 3 extra points that the student intends to add to the written test marks. Possibly, for a complete evaluation of the competence acquired, additional subjects are proposed by the teacher. Assessment breakdown: 100% mid-term plus end-of-semester summative assessment. The final mark of the "Foundations of Programming Languages" 6 ECTS Module are obtained as the average among the marks of the written and oral tests. The final mark of the 12 ECTS Course "Introduction to Programming: Foundations and Languages" is obtained as the average among the marks of the two 6 ECTS Modules "Foundations of Programming Languages" and "Introduction to Programming" .
2) *INTRODUCTION TO PROGRAMMING (6 ECTS*)		
Teacher: Monica NESI		
1	Course objectives	The goal of this course is to introduce the basic notions of imperative and object-oriented programming. On successful completion of this module, the student should be able to solve simple problems, implement the related algorithms in a structured programming language, and use a computer to run simple programs.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of this module include: - Algorithms, programs and programming languages. Flow charts, structure of a program. - Basic data types. Constants, variables, arithmetic expressions, Boolean expressions. Assignment. Input/Output primitives. Control structures: sequence, conditional, iteration and loop. - Structured data types: array, string. - Methods. Block structure and scoping rules. Parameter passing. Side-effects of methods. Recursion and recursive methods. - Classes and objects: basic notions, object creation and their manipulation. Static methods. Array of objects. - Inheritance and hierarchies. Polymorphism and late binding. Exceptions and their handling. On successful completion of this module, the student should : o have profound knowledge of the basic concepts of imperative and object-oriented programming; o know and understand algorithms for solving simple problems; o know and understand the core of the programming language used for coding the algorithms and have familiarity with a simple environment to run programs; o understand and be able to apply definitions as given by the syntax and the semantics of the programming language; o be able to analyse and discuss different algorithms for solving a problem and different implementations of the same algorithm in the given language; o be able to explain and illustrate the fundamental notions of basic and structured data types, control structures, methods definition and invocation, parameter passing,

		recursive methods, classes and objects, inheritance and exceptions; o demonstrate skills in problem-solving, ability to use a (subset of a) programming language to code algorithms and test the solutions on computer, capacity of abstraction and modularity.
3	Prerequisites and learning activities	Basic notions of mathematics (in particular, sets and functions). No knowledge on programming and specific programming languages is required.
4	Teaching methods and language	Lectures and exercises. Language: Italian Ref. Text books: -Cay Horstmann, *Concetti di informatica e fondamenti di JAVA*. Apogeo. 2007. -Marco Bertacca e Andrea Guidi, *Programmare in Java*. McGraw-Hill. 2007.
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but the pre-requisites of the course are clearly stated on the module website. Fulfillment of such pre-requisites is verified by formative assessment. Additional lectures or individual homework might be provided by the teacher in case significant problems are detected. Formative Assessment The formative assessment is performed via interactive interaction between teacher and students during lectures. Students are aware since the beginning of the course that they will be involved (in turns) in: - Questioning and discussion by means of open oral questions to the class or single students. - Assignment of written exercises to be solved during a lecture and a student is then selected for oral presentation of her/his solution to the class. Summative Assessment: Written test followed by an optional oral exam. In order to help the students to split the workload, an optional mid-term written test will also be provided, which is meant to cover the first part of the course. The written test includes exercises on (i) evaluation of Java code and understanding of its syntax and semantics, (ii) defining static methods (iterative and recursive) using data structures and objects and possibly raising exceptions and (iii) defining classes, subclasses and instance methods using inheritance, information hiding, code reuse, late binding. The written test aims at verifying (1) the level of knowledge of the constructs of the subset of the programming language introduced during the course and (2) the skills in problem solving and in implementing the proposed algorithms with correct code. This in order to verify the ability of analyzing simple problems, of applying the techniques learnt during the course, of implementing the proposed solutions correctly, and of evaluating alternative solutions. Criteria of evaluation will be: (i) the level of knowledge of the course topics and the ability of reasoning on them; (ii) the proper use of technical terminology and of the mathematical concepts underlying the course topics, e.g. functions and logic; and (iii) the clarity and completeness of solutions. The written test (3 hours) consists in: (a) Three or four exercises to cover point (i) and (ii), 50% of total marks; (b) Three or four exercises to cover point (iii), 50% of total marks. All parts can result in negative marks if the answer is omitted or seriously flawed. The oral examination is optional, but it becomes compulsory in certain cases, e.g. when the answers to exercises on defining methods are omitted or seriously flawed. The oral test (max 1 hour) consists of one question for each serious mistake in the written test (the answer compensates the negative marks obtained therein) and one question for at most 3 extra points that the student intends to add to the written test marks. Possibly, for a complete evaluation of the competence acquired, additional subjects are proposed by the teacher.

Programme of "FISICA" ***"PHYSICS"***
F0633, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 1st Year, 2nd Semester**
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)
Teachers: Massimo VELLANTE

1	Course objectives	The goal of the course is to provide the students with knowledge of basic principles of the mechanics with particular regard to the concepts of force, energy and wave propagation; and with the ability (through classroom exercises and homework) to schematize a natural phenomenon, to apply the proper physical laws, and to critically evaluate the results.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Kinematics of a point particle: motion in one and more dimensions, relative motions, vectorial composition of velocities. - Dynamics of a point particle: Newton's laws, concept of force, weight force, constraint reaction forces, sliding friction, viscous friction, elastic force, harmonic motion, the simple pendulum, inertial and non-inertial frames of reference, fictitious forces. - Work, energy, momentum: definition of work, theorem of kinetic energy, work of the weight, work of the elastic force, conservative forces, potential energy, mechanical energy, momentum and impulse, torque, angular momentum. - Dynamics of a system of particles: external and internal forces, momentum of the system, first cardinal equation of the dynamics, isolated system and conservation of momentum, motion of the center of mass, collisions. - Gravitation: Kepler's laws, law of universal gravitation, Cavendish experiment, gravitational potential energy, gravitational field, motion of a satellite in circular orbit, escape velocity. - Waves in elastic media: longitudinal and transverse waves, mathematical description of wave propagation, harmonic waves, D'Alembert equation, vibrating string, elastic waves in a bar and in a gas, interference, beats, standing waves, Doppler effect. On successful completion of this module, the student should : o have acquired a good knowledge and understanding of the fundamental concepts in Physics and a method of logical and rigorous approach to a physical problem. o be able to schematize a natural phenomenon and to apply correctly basic physical laws to solve simple problems in mechanics. o acquire skills and reasoning attitude for approaching problems in other fields of science and outlining solving methods. o be able to choose the proper model to describe a physical phenomenon and be aware of the degree of approximation adopted. o be able to communicate in a simple but rigorous way the acquired knowledge in Physics to other colleagues as well as to non-experts. o be able to understand other topics in General Physics not covered in this course.
3	Prerequisites and learning activities	Algebraic calculus, trigonometry, mathematical analysis (functions, derivatives and integrals).
4	Teaching methods and language	Lectures and exercises on blackboard. Learning checks with exercises to be solved in class or at home. Language: Italian Reference textbooks -P. Mazzoldi, M. Nigro, C. Voci, *Elementi di Fisica, Meccanica e Termodinamica*. EdiSES. -Serway & Jewett, *Principi di Fisica, Volume 1* . EdiSES.
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website. Fulfillment of such pre-requisites is verified by formative assessment. Formative Assessment The formative assessment is performed via interaction between teacher and students during lectures and exercises. Students are involved in questioning, in discussion and in proposing strategies to solve exercises, by means of open oral questions to the entire class. Summative Assessment: Written test followed by an oral examination. An optional mid-term written test will be also provided, which is meant to cover the first part of the course, in order to help the students to split the workload. If a student passes the mid-term written exam, he/she will take a final-term written test concerned with the second part of the course content only. The written test (to be solved within 2 hours) consists in 2-3 exercises each composed by several questions. It is aimed to verify the ability in schematizing simple physical problems, and using proper laws and correct formalism for their solution. A large number of previous written tests and corresponding detailed solutions are available on the Module website. This allows the students to be well aware of the kind of test he/she will have to deal with and how the required output should look like.

		The oral examination will take place within the same or the next exam session of the passed written test. It will typically last less than 1 hour and will cover the areas of the written answers that need clarification, and additional subjects (belonging to the whole program of the course) proposed by the teacher. Criteria of evaluation will be: the level of knowledge of the basic physical laws presented in the course, as well as the ability to apply them; the property of use of the scientific language; the clarity and completeness of explanations.

Programme of "MATEMATICA DISCRETA" ***"DISCRETE MATHEMATICS"*** **This course is composed of two Units: 1) Discrete Mathematics I, 2) Discrete Mathematics II**		
F0035, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 1st Year, 2nd Semester**		
Number of ECTS credits: 12 (workload is 300 hours; 1 credit = 25 hours)		
1) DISCRETE MATHEMATICS I (6 ECTS)		
Teacher: Carlo SCOPPOLA		
1	Course objectives	This Unit is composed of two Modules:1) Algebra, 2) Linear Algebra. The general goal is to provide the students with the motivations, definitions and techniques in support of the usefulness of algebra and linear algebra in the effective and efficient modeling of data and knowledge. The students will understand the fundamental concepts of algebra and linear algebra and will be able to analyze and explain precisely the process of solving an equation. They will learn the terminology specific to polynomials and understand that polynomials form a system analogous to the integers and will be able to apply the basic algebraic structures, to such topics as the finite Fourier transform, coding, complexity, and automata theory.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of Module "Algebra" include: - Sets: functions, equivalence relations, products, elementary combinatorics. - Permutations. - Groups: subgroups, quotients, isomorphism theorems, factor groups, permutation groups, cyclic groups. - Arithmetic: divisibility theory in the ring of integers and of polinomials over a field. - Congruences. Chinese remainder theorem. - Rings: subrings, ideals, quotients, isomorphism theorem, ring of polynomials, domains, euclidean rings, PID, UFD. - Fields: simple field extensions, finite fields. Topics of Module "Linear Algebra" include: - Matrices and systems of linear equations: Gauss reduction, determinants. - Vectors, vector spaces, independence, bases. - Inner-product, cross product and their applications and extensions. - Eigenvalues, eigenvectors. Diagonalization and canonical forms of matrices. - Application: systems of differential equations. On successful completion of this Unit, the student should be able to: o recognize and describe various examples of groups, including groups of numbers, permutations, matrices, and transformations, and examples of subgroups and quotient groups, as well as homomorphisms and isomorphisms between them; o derive basic properties of groups, subgroups, quotient groups, homomorphisms, and isomorphisms from their definitions; o recognize and describe various examples of rings, integral domains and fields and examples of subrings, ideals, and quotient rings of such rings, as well as homomorphisms and isomorphisms between them; o discuss and prove the unique factorization property of the ring of polynomials over a field, and perform related operations on polynomials such as Euclidean algorithm, irreducibility tests and factorization; o recognize and describe the process of construction of field extensions, o perform operations with complex numbers and matrices to apply them to solve problems, including the n-th root of a complex number by using the Polar form and De

		Moivre's formula; o recall the definition and properties of the six basic trigonometric functions and their inverse functions; o recognize, state, and use vector spaces definition and properties; linear algebraic substructures, inner product spaces; o provide written proof to problems and theorems in Linear Algebra, spectral mapping theorem for linear transformations and whether a linear operator is diagonalizable or not; o do independent learning and problem solving.
3	Prerequisites and learning activities	Numerical Structures (natural numbers, prime numbers, numerical fractions, rational numbers, basics of real numbers, inequalities, absolute value, powers and roots)
4	Teaching methods and language	Lectures, Exercises Language: Italian Reference textbooks: -Benedetto Scimemi, ***Algebretta***, Ed. Decibel -Michael Artin, ***Algebra,*** Ed. Boringhieri
5	Assessment methods and criteria	Formative assessment: Tutoring, small working groups for simple assignments, commented homework, open oral questions to the classroom and comments. Summative assessment: Written and oral exam (60:40). The written test is a 2-hours paper and consists of the answer to 3 out of 4 questions of which one 2 in Algebra and 2 in Linear Algebra, contributing to 60% of the total mark. It is intended as a propaedeutic evaluation of the acquired knowledge of basic algebraic structures and capacity to use appropriately the transformation rules for the equivalence of matrices. The oral exam consists of the answer to 3 questions aiming to evaluate the level of knowledge of the algebraic structures (20% of total mark), the capacity to provide examples of groups and vector spaces and discuss the properties of subgroups, subspaces and related morphisms (20% of the total mark), the ability to calculate the dimension of a space and provide counterexamples (20% of the total mark), the ability to make basic operations with matrices (20% of total mark) the capacity to face and solve problems and situations not encountered during the course (20%of the total mark) and the capacity to make a proof of simple properties (20% of total mark).
2) DISCRETE MATHEMATICS II (6 ECTS)		
Teacher: Anna TOZZI		
1	Course objectives	This Unit is composed of two Modules:1) Elements of Mathematical Logic, 2) Geometry. The goal of the Module 1) is to provide the motivations, definitions and techniques in support of the usefulness of logic in the effective and efficient modeling of data and knowledge. This Module is an introduction to mathematical logic and covers elementary discrete mathematics for computer science. The student is expected to understand the fundamental concepts of mathematical logic and to be aware of potential applications in computing, including the limitations of algorithms. The goals of Module 2) are to introduce students to the terminology and theorems of plane and solid geometry, and to apply algebraic, spatial, and logical reasoning to solve geometry problems. This Module covers the fundamental concepts of Linear Algebra and its role in describing geometric settings. On successful completion of this module, the student will develop spatial sense, visualize and represent geometric figures , explore transformations of geometric figures, understand and apply geometric properties and relationships, synthesize geometric concepts into algebraic, functional, and problem-solving activities.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of Module "LOGIC" include: - Propositional Logic: Logical formulae, valuations, truth tables, logical equivalence of formulae, satisfaction and logical implication. - Deductive Logic: Formal axiom schemes, the structure of formal proofs, Sequent Calculus, Natural Deduction, the Deduction Theorem, and connections between truth and proof (the Soundness and Completeness Theorems). Topics of Module "GEOMETRY" include: - Euclidean plane geometry, angles, radians, notion of geometric place, - Properties of triangles, parallelograms, circles, symmetry and similarity, - Transformations in the plane, Cartesian coordinates and equations of simple geometric places, elements of trigonometry, - Elements of spatial Euclidean geometry, volumes. On successful completion of this Unit, the student should : o have knowledge and understanding of the meaning of mathematical definitions of

		sets/propositions and perform set/logic operations; o understand and explain the meaning of complex statements using mathematical notation and language; o understand the fundamental concepts of mathematical logic and be aware of potential applications in computing; o be able to create algorithmic methods of real-world problems, and to develop and present the solutions; o demonstrate capacity for finding rigorous proofs of small problems; o be able to use logic to demonstrate the equivalence of statements and test the validity of arguments; o have knowledge and understanding of geometric relationships within the axiomatic structure of Euclidean geometry; o be able to understand and apply geometric properties and relationships; o understand and explain the relation of geometry to algebra and trigonometry by using the Cartesian coordinate and recognize geometric relationships in the world; o be able to use Euclid's fifth postulate to deduce the measure of angles between parallel lines; o be able to use vectors and trigonometry concepts to solve problems related to geometry, or to solve problems related to parametric equations or polar coordinates and physics. o demonstrate skill in mathematical reasoning, manipulation and calculation by synthesizing geometric concepts into algebraic, functional, and problem-solving activities;.
3	Prerequisites and learning activities	For LOGIC the student must have the basic mathematical notions and methods as acquired in the secondary Schools. For GEOMETRY the student must know: Set Theory, Numerical Structures; Elementary algebra; Algebraic Structures (Groups, homeomorphisms, rings); Linear Algebra.
4	Teaching methods and language	Lectures and exercises. Language: Italian Ref. Text books: -Paola Favro e Andreana Zucco, *Appunti di Geometria Analitica*. Quaderni Didattici del Dipartimento di Matematica-Università di Torino. 2004. -A. Asperti, A. Ciabattoni, *Logica a Informatica*. McGraw Hill, 1997. *Texts are available on the Department web site*.
5	Assessment methods and criteria	Formative assessment: Tutoring, small working groups for simple assignments, commented homework, questioning and discussion (open oral questions to the classroom and comments), written question to be answered in 5 minutes and presentation to the classroom. Summative assessment: Written and oral exam (60:40). The written test is a 2-hours paper and consists of the answer to at least 3 of 4 questions of which one in Logic and two in Geometry, contributing to 60% of the total mark. It is aimed to ascertain the capacity to solve simple logic problems and the knowledge of the basic geometrical properties in the plane and in the space. The oral exam consists of the answer to 3 questions aiming to evaluate the level of knowledge of the logic derivations rules (20% of total mark), of the relationships within the axiomatic structure of Euclidean Geometry (20% of the total mark), the ability to explain the meaning of complex statements using mathematical notation and language (20% of total mark) the capacity to face and solve problems and situations not encountered during the course (20%of the total mark) and the capacity to make a proof of simple properties (20% of total mark).

Programme of "INGLESE – Livello B1" ***"ENGLISH LANGUAGE – Level B1"***		
F0123, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 1st Year, 2nd Semester**		
Number of ECTS credits: 3 (workload is 75 hours; 1 credit = 25 hours)		
Teacher: Silvia MAROTTOLI		
1	Course objectives	The goal of the course is to provide the students with a good level of language competences. In addition to consolidation of intermediate linguistic structures (equivalent to C.E.F.R. B1) students will be introduced to specific language for IT as well as related idioms and appropriate register.

2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - IT RELATED TOPICS : Computers today Basic computer terms. Basic hardware and software terminology. Units of memory. Input/output devices . Describing input and output devices. Groups of keys and mouse actions. Describing scanners and cameras. Basic software. Describing and comparing different operating systems. Learn about the features of a GUI. Basic functions and features of word processors, spreadsheets and databases. Faces of the Internet. Study vocabulary related to the Internet and email. Learn about the basic features of the Web. Learn and use vocabulary related to e-commerce, online banking, online chatting and videoconferencing. Creative software. Basic principles of web page design. Jobs in ICT (unit 26). Discuss the personal qualities and professional skills needed for a job in ICT. Learn how to write a letter of application for a job. - PRONUNCIATION. Word stress and the sound /?/ (schwa). The sounds /æ/ and /e/ . Minimal pairs (/I/ and /i/, /æ/ and /e/ etc...). Rising and falling intonation. "-ed" forms with /t/, /d/ and /Id/ - STANDARD GRAMMAR STRUCTURES REQUIRED AT AN INTERMEDIATE (B1) LEVEL: Present simple and continuous. Present perfect simple and continuous. Past simple and continuous. Past perfect simple. Future with will and shall, be going to, present continuous and present simple. –ing forms after verbs and prepositions. Modals: can / would / will / shall / should / may / might / have to / ought to / must / mustn't / need / needn't / used to. Phrasal verbs / verbs with prepositions. Passive forms: present and past simple. Have/get something done. Conditional sentences: Type O, Type 1, Type 2. Reported speech. Question words. Nouns: Singular and plural (regular and irregular forms), countable and uncountable nouns. Possessive case ('s) and double genitive. Personal pronouns (subject, object, possessive). Reflexive and emphatic pronouns. Demonstrative pronouns: this, that, these, those. Quantitative pronouns: one, something, everybody. Indefinite pronouns: some, any, something, one. Relative pronouns: who, which, that, whom, whose. Adjectives: colour, size, shape, quality, nationality. Cardinal and ordinal numbers; possessive; quantitative; order of adjectives; participles as adjectives; compound adjectives; comparative and superlative forms (regular and irregular). Adverbs. Prepositions. Connectives. - STANDARD FUNCTIONS REQUIRED AT AN INTERMEDIATE (B1) LEVEL. Asking and answering questions about personal possessions. Asking for and giving information about routines and habits. Asking for and giving information about travel and places. Asking for and giving personal details. Asking for and giving the spelling and meaning of words. Asking the way and giving directions. Describing education, qualifications and skills. Describing people (personal appearance, qualities). Following and giving simple instructions. Greeting people and responding to greetings. Identifying and describing accommodation. Introducing oneself and other people. Starting and maintaining a conversation. Talking about food and ordering meals. Talking about one's health. Talking about past events and states in the past, recent activities and completed actions. Talking about the weather. Talking about what people are doing at the moment. Understanding and completing forms giving personal details. Understanding and writing diaries and letters, giving personal details. Understanding simple signs and notices. - STANDARD VOCABULARY REQUIRED AT AN INTERMEDIATE (B1) LEVEL. Clothes. Daily life. Education. Entertainment and media. Environment. Food and drink. Free time. Health, medicine and exercise. Hobbies and leisure. House and home. Jobs. Languages. People. Personal feelings, opinions and experiences. Personal identification. Places and buildings. Relations with other people. Shopping. Social interaction. Sport. Transports. Travel and holidays. Weather. On successful completion of this module, the student should : o have acquired a good knowledge and understanding of technical language. o be able to communicate with correct and complete sentences. o acquire skills for writing clear scientific reports in English. o be able to make conversation with English mother tongue people on both scientific and general topics.
3	Prerequisites and learning activities	A2 level.
4	Teaching methods and language	Lectures and exercises. Language: English Reference textbooks

		- R. Murphy, *English Grammar in Use*. CUP. (vol. ISBN: 9780521189064) -C. Walsh, L. Warwick, *Gold Preliminary*. Pearson Longman. (vol. Coursebook) 2013. -A. Gallagher, F. Galuzzi, *Grammar and vocabulary trainer*. Pearson Longman. (vol. ISBN: 9788883390227) -Santiago Remacha Esteras, *Infotech English for Computer Users, Student's Book, 4th ed.*. Cambridge University Press. 2008.
5	Assessment methods and criteria	Written examination including the P.E.T. listening paper as well as reading and writing activities (related to IT topics) - dictionaries are not allowed. Oral test.

Programme of "LABORATORIO DI ARCHITETTURA DEGLI ELABORATORI" ***"COMPUTER ARCHITECTURE LABORATORY"***		
F0123, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 1st Year, 2nd Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Luca FORLIZZI		
1	Course objectives	This Course provides the students with the intermediate knowledge of imperative programming, of memory management techniques, of assembly language programming and related technologies, and of machine languages. At the end of the course the students will understand non-elementary programs written with an imperative programming language, will be able to realize constructs of a high level programming language with machine language instructions, and to understand and use the technology underlying a programming language. The students are expected to develop capacities for being attentive to details in algorithms implementation, to program portability, to the respect of programming languages standards and rules, as well as for looking for efficient implementations of algorithms.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Structured computed organization, computational models, programming languages and their standards ; - Imperative programming with high level languages oriented towards system programming ; - Basic and derived data types; data types representation - Memory management; low-level programming ; - Assembly language programming; translation of the main constructs of a high level programming language in an assembly language; - Machine languages; assembly and disassembly processes. On successful completion of this module, the student should : o have acquired intermediate-level notions about imperative programming; o have in-depth knowledge about memory management techniques, assembly language programming and related technologies, and about machine languages; o be able to interpret non-elementary programs written with an imperative programming language; o be able to implement constructs of a high level programming language with machine language instructions and interpret technical specifications of high-level programming languages, assembly languages and machine languages; o be able to select, among different ways to implement programs, those leading to better performances; o be able to discuss, with appropriate vocabulary, languages and computer architectures; o be able to use an imperative language oriented towards system programming in subsequent courses about computer architectures; o be able to see links between real computers and computational models as presented in theoretical computer science courses.
3	Prerequisites and learning activities	Knowledge: elementary knowledge of imperative programming, basics of computer systems architecture, elementary knowledge of mathematics, reading and understanding English language. Abilities: implementing correctly elementary algorithms, compiling and running simple programs on a computer
4	Teaching methods and language	Lectures, exercises Language: Italian, English

		Ref. Text books - Kim N. King, *C Programming: a Modern Approach*, W.W.Norton & Company. 2008 (second edition). *http://knking.com/books/c2/index.html* -Andrew S. Tanenbaum, *Structured Computer Organization*. Prentice Hall. 2005 (fifth edition). -David A. Patterson, John L. Hennessy, *Struttura e progetto dei calcolatori*. Zanichelli. 2010 (terza edizione, condotta sulla quarta edizione americana).
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the website. Formative Assessment The formative assessment is performed via interaction between teacher and students during lectures. Students are involved in questioning and discussion, by means of open oral questions to the entire class. Summative Assessment The course grade is determined by a written test followed by an oral exam. It will be also provided an optional mid-term written test, which is meant to cover the first part of the course, in order to help the students to split the workload. The written tests consist in a series of exercises and open questions, to which the student has to provide a detailed extended answer, aimed to verify the practical abilities and level of knowledge acquired. The oral exam will occur within two weeks of the written test and will typically cover the areas of the written answers that need clarification, plus, possibly, additional subjects proposed by the teacher. Criteria of evaluation will be: level of knowledge and practical ability; the property of use of the technical/mathematical language; the clarity and completeness of explanations; capacity to make sensible decisions in the writing and translation of programs.

Programme of "ALGORITMI E STRUTTURE DATI CON LABORATORIO" ***"ALGORITHMS AND DATA STRUCTURES WITH LAB"*** **This course is composed of two modules:** **1) Algorithms and Data Structures, 2) Laboratory of Algorithms and Data Structures**		
F0130, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 2nd Year, 1st Semester**		
Number of ECTS credits: 12 (workload is 300 hours; 1 credit = 25 hours)		
1) ALGORITHMS AND DATA STRUCTURES (6 ECTS)		
Teacher: Guido PROIETTI		
1	Course objectives	The aim of this module is to give the student the knowledge and competence to deal with algorithms in the context of problem solving and computer programming. Starting with simple algorithmic constructs the students are introduced to more advanced data structures such as stacks, queues, and linked lists, and to problems that require use of these data structures. A range of algorithms is studied that give the student an appreciation of their applicability in many areas of computing. The aim is to make the student capable of abstracting models and formal algorithmic problems from real computational problems, and designing efficient algorithmic solutions.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the Module include: - Algorithms and problems. Complexity analysis of an algorithm. Lower and upper bound. - Sorting algorithms: insertion-sort, selection-sort, merge-sort, quick-sort - Priority queues: binary heaps, binomial heaps, heap-sort. - The dictionary problem: searching, inserting, deleting. Hast tables and AVL trees. - Graphs: definitions, memory representations, DFS and BFS. - Elementary graph algorithms: shortest paths and minimum spanning trees. Upon successful completion of the course, the student will: o known efficient algorithms for basic computational problems (sorting, searching, graph problems, etc.). o understand the importance of designing efficient algorithms; o be able to describe the properties a good algorithm should have; o be able to describe the basic algorithms used to represent the data in databases and how to manipulate it; o be able to analyze the resources (space and time) needed by an algorithm;

		o be able to formally present and model concrete problems, focusing on their main features and discarding the inessential ones; o be able to present and compare different solutions to a given problem, and to identify independently the most efficient solution; o acquire competencies and abilities necessary in their future studies, especially with respect to advanced algorithmic courses.
3	Prerequisites and learning activities	A basic background in discrete mathematics are necessary prerequisites to this course.
4	Teaching methods and language	Lectures, Seminars Language: Italian Reference textbooks: - C. Demetrescu, I. Finocchi, G.F. Italiano, *Algoritmi e Strutture Dati*. Ed. McGraw-Hill.
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website. Formative Assessment The formative assessment is performed via interaction between teacher and students during lectures. Students are involved in questioning and discussion, by means of open oral questions to the entire class. Summative Assessment: Written test followed by an oral exam. An optional mid-term written test will be also provided, which is meant to cover the first part of the course, in order to help the students to split the workload. If a student passes the mid-term written exam, she will take a final-term written exam concerned with the second part of the course content only. A written test (40 minutes) consists of 10 multiple-choice tests. It is aimed at verification of theoretical competences, and in particular of knowledge and comprehension of course's content. A correct answer to a test provides 3 points, a missing answer counts 0, while a wrong answer provides a -1 penalization. The final result will be given by summing up all the obtained points. The oral exam will occur within the same exam session of the written test, and it will typically cover the areas of the written answers that need clarification, plus a subject of one's choice. The oral exam (max 1 hour) will test the student's ability to engage in discussion of issues relevant to the topics discussed during the course. Criteria of evaluation will be the level of knowledge and the fluency in the technical language of algorithms. The final marks of the Algorithms and Data Structures with Laboratory 12 CFU Course are roughly obtained as the average among the marks of the 6 CFU Modules "Algorithms and Data Structures" and the "Laboratory of Algorithms and Data Structures" .
2) *LABORATORY OF ALGORITHMS AND DATA STRUCTURES (6 ECTS*)		
Teacher: Giovanna MELIDEO		
1	Course objectives	This module invites students to study and implement the most important algorithms and data structures for information processing. The module focuses on techniques for the design and analysis of efficient algorithms, emphasizing methods useful in practice. Topics include lists, stacks, queues, trees, heaps and priority queues, binary search trees (including red-black), union-find for disjoint sets and graphs (including both array-based and linked representations) and evaluation of classic algorithms that use these structures for tasks such as sorting and searching. The module encourages students to develop implementations using the Java language, understand their performance characteristics, and estimate their potential effectiveness in applications using lectures and exercises.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of this module include: - Java review. Interfaces and abstract classes. Generics. The Java Collection Framework Interfaces and classes. - Elementary data structures and their implementation in Java: lists, stacks and queues. Array and singly linked list implementations of lists, stacks and queues. Use of the standard classes from a client code perspective. - Searching and sorting algorithms and their implementation in Java. Java implementation of sequential and binary search. Java implementation of bubble-sort, insertion-sort, selection-sort, merge-sort, quick-sort. - Advanced data structures and their implementation in Java: Binary trees, breadth-first search and depth-first search; Binary search trees (BST), balanced BST, red-black trees; Disjoint sets. - Priority queues and their implementation in Java. Binary heaps and use of a heap to

		implement a priority queue. Use of a priority queue to develop a sorting method. Java implementation of heap-sort. - Graphs: representations for both directed/undirected and weighted/unweighted models, Java implementation of breadth-first search and depth-first search, Java implementation of classical graph algorithms for solving the single-source shortest path problem and the minimum spanning tree problem. On successful completion of this module, the student should : o Know classical algorithms and data structures for information processing and their implementations, with particular focus on computational complexity aspects, and be aware of the effects of data organization and algorithms on program efficiency; o Be familiar with standard techniques for designing algorithms, including the techniques of recursion, divide-and-conquer, and greedy and understand how apply them to design efficient algorithms for different kinds of problems; o Be able to apply their knowledge of data structures and algorithmic techniques to design efficient algorithms that correctly satisfy a given specification and write more efficient programs in Java; o Be able to rigorously analyze the relative time and space efficiency of competing algorithms and carry out a comparative evaluation of merits in terms of efficiency and applicability of standard data structure; o Be able to identify efficient solutions to a given problem and choose appropriate data structures that effectively model the information in a problem; o Be able to judge efficiency tradeoffs among alternative data structure implementations or combinations; o Demonstrate the capability of formally presenting and modeling concrete problems and explain the most important characteristics concerning the standard data structures, their analyses, and their Java implementations; o Have an understanding of the role and characteristics of data structures and of the importance of time and space efficiency in designing algorithms and writing programs; o Be able to recognize and look up variations of the material studied in the literature.
3	Prerequisites and learning activities	A basic understanding of programming in Java and a background in discrete mathematics are necessary prerequisites to this course.
4	Teaching methods and language	Lectures and exercises. Language: Italian Ref. Text books: - W.J.Collins, *Algoritmi e strutture dati in Java*. Ed. Maggioli, Apogeo Education. -C.Demetrescu, U.Petrillo, I.Finocchi, G.Italiano , *Progetto di Algoritmi e Strutture Dati in Java*. McGraw-Hill. Teaching material available on the external website http://www.di.univaq.it/melideo/lezioni_labalg2014.html
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website. Formative Assessment The formative assessment is performed via interaction between teacher and students during lectures. Students are involved in questioning and discussion, by means of open oral questions to the entire class. Summative Assessment consists of written test followed by an optional oral exam. An optional mid-term written test will also be provided, which is meant to cover the first part of the course, in order to help the students to split the workload. The written exam consists of a set of exercises to assess: (i) whether the student knows, can motivate and can compare the concepts, the implementation and the time and memory requirements of the data structures and algorithms; (ii) a student's general knowledge and understanding on the module material; (iii) a student's practical skills in selecting and applying suitable data structures and algorithms in software development and general problem solving. This involves writing program code with pen and paper. Criteria of evaluation will be the level of knowledge and understanding, and practical ability. The oral exam can be required either by the student, to improve grades, or by the teacher, in presence of significant mistakes/misunderstandings in the written exam. The oral exam will occur within the same exam session of the written test and will typically cover the areas of the written answers that need clarification plus, possibly, additional subjects proposed by the teacher.

		The final mark of the 12 CFU Course "Algorithms and Data Structures with Laboratory" is roughly obtained as the average among the marks of the two 6 CFU Modules "Algorithms and Data Structures" and "Laboratory of Algorithms and Data Structures" .

Programme of "SISTEMI OPERATIVI CON LABORATORIO" ***"OPERATING SYSTEMS WITH LAB"*** **This course is composed of two modules: 1) Operating Systems, 2) Laboratory of Operating Systems**		
F0133, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 2nd Year, 1st Semester**		
Number of ECTS credits: 12 (workload is 300 hours; 1 credit = 25 hours)		
1) *OPERATING SYSTEMS (6 ECTS*)		
Teacher: Vittorio CORTELLESSA		
1	Course objectives	This Module provide the students with the knowledge of basic concepts common to all the operating systems, mechanisms and policies of operating systems, system overhead vs solution efficiency tradeoff and the ability to relate different topics and to solve problems never faced in classroom, but solvable through logic deductions and reasoning. Students are continuously stimulated to work during the course time, to pose questions in the classroom and to originate discussion. In this way they will acquire interest for an integrated knowledge of different aspects of computer science, awareness of relationships among computer subsystems, hence awareness of the fact that a satisfactory behavior of a computer may derive from the combination of very different (sometimes unexpected) factors and ability to analyze and synthesize concepts.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of this module include: - General concepts, computer system and operating system structures - Processes and CPU scheduling - Process synchronization and deadlock management - Memory management - The virtual memory - The file system On successful completion of this module, the student should : o know and understand the basic concepts that are common to all general-purpose operating systems, in particular the ones related to the management of CPU and central memory; o be able to relate these concepts in order to synthesize the intrinsic tradeoffs that underlie a virtual machine (i.e. a computer plus its operating system); o be able to solve complex problems related to the synchronization among concurrent processes.; o be able to apply the operating system policies studied in the course (such as CPU schedulers, pagers, etc.) to specific examples; o be able to select the best solutions (among those studied in the course) for specific examples; o be able to critically explain why the existing operating systems operate in their ways, basing also on the historical notions that they have received in the course and that help to understand the current state-of-art; o on the basis of the knowledge and capacities acquired in this course, be able in the future to tackle any actual operating system, by just studying its handbook. This is expected because their knowledge is independent on any specific existing system in this course module, with the goal of providing general instruments suitable for a continuous learning in this domain.
3	Prerequisites and learning activities	Knowledge of fundamentals of programming, algorithms and data structures, computer architecture, reading and understanding English language, and ability to integrate classroom and homework study, to pose questions in the classroom and to originate discussion.
4	Teaching methods and language	The module includes 54 hours of frontal lectures plus 30 hours of on-demand clarifications in the teacher's office. The frontal lectures are partitioned in theory and exercises.. Language: Italian Ref. Text books: -A. Silberschatz, P.B. Galvin, G. Gagne, *Operating System Concepts*. John Wiley & Sons.

5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website. Fulfilment of such pre-requisites is verified by formative assessment. Formative Assessment The formative assessment is performed via interactive interaction between teacher and students during lectures. Students are aware since the beginning of the Course that they will be involved (in turns) in: - Questioning and discussion, by means of open oral questions to the class or to single students. Summative Assessment: Written test followed by an optional oral exam. An optional mid-term written test is also be provided, which is meant to cover the first part of the course, in order to help the students to split the workload. The written test is aimed at: (1) verification of theoretical competences, and in particular of knowledge and comprehension of Course contents; (2) verification of skills in understanding and solving significant exercises, and in explaining the proposed solutions. This in order to verify the ability of application of techniques learnt during the Course, of analysis of problems and synthesis of suitable solutions, and of evaluation of alternative solutions. Criteria of evaluation will be: the level of knowledge and practical ability; the property of use of the technical/mathematical language; the clarity and completeness of explanations. The oral exam will occur within one week of the written test and will typically cover the areas of the written answers that need clarification plus additional subjects proposed by the teacher. The oral test can be required: (i) by the student, to improve final marks; (ii) by the teacher, in presence of significant mistakes/misunderstandings in the written test. Assessment breakdown: 100% mid-term plus end-of-semester summative assessment. The written test (2 hours) consists, in general, in: (a) Three exercises to solve; (b) A list of about 3-4 questions to answer. The oral test (max 1 hour) consists of questions on the written exam and extra ones. The final marks of the Operating Systems with Laboratory 12 CFU Module are obtained as the average among the marks of the Operating Systems and Operating Systems Laboratory 6 CFU Modules.
2) LABORATORY OF OPERATING SYSTEMS (6 ECTS)		
Teacher: Salvatore CUSENZA		
1	Course objectives	KNOWLEDGE : principles of unix/linux system programming, basic notions about concurrent a/o distributed programming. CAPABILITIES: adequacy in using unix/linux system calls. competence in facing and solving simple problems by applying basic unix/linux system-programming techniques. EXPECTED BEHAVIOUR: interest in design and pragmatic implementation aspects of operating-system technology.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the Module include: - UNIX Architecture - UNIX Bash shell - System programming - processes - System programming - I/O and File System - Process communication and synchronization. Threads - Distribute applications Upon successful completion of the course, the student will: o known how to Download and Install Linux o know and understand the computer architecture and UNIX architecture; o be able to write programs using the system calls of UNIX operating system (Fork, exec, getpid, exit, wait, close, stat, opendir, readdir); o be able to write programs using the I/O system calls of UNIX operating system (open, read, write, etc); o be able to write C programs to simulate UNIX commands; o be able to write Shell scripts and manage Linux files and directories; o demonstrate competence in administering ownership, permissions, and quotas; o be able to setup a VM (Virtual Machine) and install Linux on the VMs.
3	Prerequisites and learning activities	Knowledge of C programming language, algorithms and data structures, computer architecture, basics operating system concepts, English language. Competence about basic analysis techniques, design, and implementation of simple solutions for elementary problems.
4	Teaching methods and language	Lectures, exercises Language: Italian

		Ref. Textbooks: -Michael Kerrisk, *The Linux Programming Interface A Linux and Unix System Programming Handbook*. No Starch Press. 2010.
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website. Fulfillment of such pre-requisites is verified by formative assessment. Additional lectures or short seminars or individual homework are provided by the teacher in case significant problems are detected. Formative Assessment The formative assessment is performed via interactive interaction between teacher and students during lectures. Students are aware since the beginning of the Course that they will be involved (in turns) in: - Questioning and discussion, by means of open oral questions to the class or to single students. - Exit Slips: students are assigned written questions or exercises to be answered in 10 minutes, and a student is then selected for oral presentation of her/his solution to the class. - Short seminars: students may be assigned personalized homework, that they will have to illustrate to the class by means of 20 minutes' short seminars. Summative Assessment: Written test. An optional mid-term written test is also be provided, which is meant to cover the first part of the course, in order to help the students to split the workload. The written test is aimed at: (1) verification of theoretical competences, and in particular of knowledge and comprehension of Course contents (2) verification of skills in understanding and solving significant exercises, and in explaining the proposed solutions. This in order to verify the ability of application of techniques learnt during the Course, of analysis of problems and synthesis of suitable solutions, and of evaluation of alternative solutions. Criteria of evaluation will be: the level of knowledge and practical ability; the property of use of the technical/mathematical language; the clarity and completeness of explanations. Assessment breakdown: 100% mid-term plus end-of-semester summative assessment. The *written test* (2 hours) consists in: (a) Twelve multiple-choice questions, to cover point (12), 40% of total marks; (b) Tree short essays (max 600 words) to cover point (9), 30% of total marks; (c) Two exercises, to cover point (9), 30% of total marks. The final marks of the 12 CFU course "Operating Systems with Laboratory" are obtained as the average among the marks of the 6 CFU Modules "Operating Systems" and "Laboratory of Operating Systems".

Programme of "CALCOLO DELLE PROBABILITA' E STATISTICA" ***"PROBABILITY AND STATISTICS"***		
DT0003, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 2nd Year, 2nd Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Davide GABRIELLI		
1	Course objectives	This is a first course on probability theory. It will give a general introduction to the basic ideas constructions and techniques of modern probability theory. Students will acquire an understanding of the basics combinatorial tools, of the central inotions of conditional probability and independence and of the law of large numbers. In the final part of the course there will be also an introduction to the theory of Markov chains.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Elements of combinatorial calculus: Permutations, Combinations - Axioms of Probability theory: probability spaces, sigma algebras - Conditional probabilities and independence - Discrete random variables and their distributions: Bernoulli, Binomial, Poisson, geometric - Sums of random variables, functions of random variables - Expected value, variance, covariance moments and correlations. - Joint and marginal distributions, couplings - Generating functions and their applications - Elementary inequalities: Markov and Chebisev

		- Convergence of random variables, weak law of large numbers - Introduction to Markov chains, classification of states, invariant measures relaxation to equilibrium - All the topics will be discussed and illustrated using a large number of examples and exercises On successful completion of this module, the student should : o Have a good knowledge and understanding of basic combinatorial tools of probability theory, of the notions of conditional probability and independence, of the law of large numbers and of the basic tools of Markov chains, o Be able to apply his knowledge and understanding to tackle basic problems and exercises, o Understand the importance of probability and statistics in modern theoretical and applied science, o Be able to construct suitable probability spaces associated to concrete specific problems and deduce predictions, o Demonstrate capacity to read and understand probabilistic texts, o Be able to explain the main notions and results, o Demonstrate skills in mathematical reasoning and ability to conceive a proof.
3	Prerequisites and learning activities	The student must know Discrete Mathematics
4	Teaching methods and language	Lectures, exercises Language: Italian, English Ref. Text books - G.Grimmett, D. Stirzaker, *"Probability and Random Processes",* Oxford University Press - S. Ross, "Calcolo delle Probabilità", Apogeo
5	Assessment methods and criteria	Formative assessment: The formative assessment is performed via interaction between the teacher and the students during the lectures. Students are involved in questioning and discussions by means of open questions to the entire class. Problems to be solved as homework are also assigned. Summative assessment: Written exam followed by a short oral examination based on a discussion of the written part (80:20) . The written exam contains some theoretic questions, in order to assess the general knowledge of the subject and some exercises to be solved in order to assess the ability to apply the knowledge. The oral exam consists of questions aimed at assess the degree of understanding of the subject. Criteria of evaluation are the level of knowledge, the correctness of solutions, the proper use of mathematical language and the clarity and completeness of explanations. Particular importance is given to the creativity of the methods of solution.

Programme of "BASI DI DATI CON LABORATORIO" ***"DATA BASES WITH LAB"*** **This course is composed of two modules: 1) Data Bases, 2) Laboratory of Data Bases**		
F0136, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 2nd Year, 2nd Semester**		
Number of ECTS credits: 12 (workload is 300 hours; 1 credit = 25 hours)		
1) DATA BASES (6 ECTS)		
Teacher: Stefania COSTANTINI		
1	Course objectives	Following this course, the students will: understand what is a database, from the conceptual, mathematical and practical points of view; become able to develop the conceptual and logical design of a relational database, including both the structure and the operations; understand the basic functionalities of a DBMS (Data Base Management System); acquire notions about advanced aspects and future directions of the field.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the Module include: - Introduction, data models, the relational model (RL). - Relational algebra and calculus, Datalog. - Database Conceptual and Logical Design: Entity Relationship (E-R) diagrams, restructuration and translation into relational schemas. - Normalization of relational schemas.

		- Advanced databases: object-oriented databases, hints on data wharehouses and cloud-computing. - Database Technology: concurrency and fault-tolerance Upon successful completion of the course, the student will: o know and understand the basic concepts of the relational model and understand its mathematical foundation; o understand query processing and optimization, transaction and security management in a relational database management system; o be able to apply conceptual database modelling methods such as entity-relationship model to design a relational database; o be able to identify fundamental concepts and techniques of related database management, database technology, why databases are used, and the basic components of a database; o be able to recognize the relational model and define key relational terminology and principles; o be able to use Structured Query Language, an international standard for creating and processing relational databases; o be able to discuss the database design process and the Entity-Relationship Model and to transform a data model into a relational database design; o be able to recognize and discuss the components and responsibilities of database o management; o demonstrate capacity to identify problems that occur when a database is concurrently processed by more than one user; o be able to survey and discuss important advanced database concepts, including the use of databases to support web sites and Geographical Information Systems using spatial database software.
3	Prerequisites and learning activities	The student should be acquainted with some programming language, so as to understand what a file is, and to have experimented some basic operations on files. The student should possess some basic knowledge of operating systems, namely about the file-system and concurrency. It is mandatory to possess basic notions of mathematical Logic and Set Theory.
4	Teaching methods and language	Lectures and interactive exercice sessions. Language: Italian Ref. Textbooks: -Atzeni, Ceri, Paraboschi, Torlone, *Basi di Dati: Concetti, Linguaggi e Architetture,*. McGraw-Hill, (English version available)
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but the above mentioned pre-requisites are fundamental and their fulfillment is verified throughout formative assessment. Additional lectures or short seminars or individual homework are assigned by the teacher in case significant problems are detected. Formative Assessment The formative assessment is performed via interactive interaction between teacher and students during lectures. Students are aware since the beginning of the Course that they will be involved (in turns) in: - Questioning and discussion, by means of open oral questions to the class or to single students. - Exit Slips: students are assigned written questions or exercises to be answered in 10 minutes, and a student is then selected for oral presentation of her/his solution to the class. - Short seminars: students may be assigned personalized homework, that they will have to illustrate to the class by means of 20 minutes' short seminars. Summative Assessment that consists of a written test followed by an optional oral exam. An optional mid-term written test is also foreseen and is meant to assess the first part of the course, in order to help the students to split the workload. The written test is aimed at: (1) verification of theoretical competences, and in particular of knowledge and comprehension of Course contents (2) verification of skills in understanding and solving significant exercises, and in explaining the proposed solutions. This in order to verify the ability of application of techniques learnt during the Course, of analysis of problems and synthesis of suitable solutions, and of evaluation of alternative solutions. Criteria of evaluation will be: the level of knowledge and practical ability; the property of use of the technical/mathematical language; the clarity and completeness of explanations.

		The oral exam will occur within one week after the written test and will typically cover the areas of the written answers that need clarification plus, possibly, additional subjects proposed by the teacher. The oral test can be required: (i) by the student, to improve final marks; (ii) by the teacher, in presence of significant mistakes/misunderstandings in the written test. Assessment breakdown: 100% mid-term plus end-of-semester summative assessment. The *written test* (2 hours) consists in: (a) Six multiple-choice questions, to cover point (1), 30% of total marks; (b) Two short essays (max 600 words) to cover point (1), 30% of total marks; (c) Two exercises, to cover point (2), 30% of total marks. All parts can result in negative marks if the answer is omitted or seriously flawed. The *oral test* (max 1 hour) consists of one question for each serious mistake in the written test (the answer compensates the negative marks obtained therein) and one question for each 3 extra points that the student intends to add to the written test marks. The final mark of the 12 CFU Course "Artificial Intelligence" is obtained as the average among the marks of the 6 CFU Modules "Data Bases" and "Laboratory of Data Bases"
2) *LABORATORY OF DATA BASES (6 ECTS*)		
Teacher: Pierluigi PIERINI		
1	Course objectives	Following this course, the students will integrate and complete their knowledge of the formal models used in the database design process (entity-relationship, relational, etc), viewing them applied to real case studies. They will learn how to interact with the most common DBMS systems through the SQL language and the interfaces supplied by the programming languages. Finally, the students will be involved in the entire development process of a complete database, starting from the stakeholder's specifications and going through the conceptual and logical design phases to the implementation of the requested functionalities and of the user interface
2	Course content and Learning outcomes (Dublin descriptors)	Topics of this module include: - Requirement analysis and conceptual database design - Logical database design - Data definition in SQL. Integrity constraints - Data insertion, modification and deletion with SQL - Base SQL queries. Advanced queries: subqueries, inner and outer join between tables, results grouping and ordering, union queries. - Advanced SQL concepts: views, procedures, triggers. SQL and programming languages On successful completion of this module, the student should be able to: o Design and model relational databases. o Document database structures and rules. o Maintain and retrieve data. o Perform basic administrative functions. o Perform security Administration to protect data integrity. o Develop databases with a variety of current industry software applications, such as MySQL, Oracle, DB2 and SQL Server, on a variety of operating system platforms like Linux and Windows. o Provide and support client interfaces for a database with a variety of programming languages (java, php, etc.). o Identify and utilize sources of information to research technical specifications and solve technical problems.
3	Prerequisites and learning activities	The course requires some basic programming skills (variables and data types, functions, iteration constructs, conditional statements, etc.)
4	Teaching methods and language	Lectures and exercises. The course final project requires the development of a complete, working database from a specification given during the course. The project can be carried out individually or in small workgroups. Language: Italian Ref. Text books: -Atzeni, Ceri, Paraboschi, Torlone, *Basi di Dati: Concetti, Linguaggi e Architetture,*. McGraw-Hill
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but pre-requisites are clearly stated. Fulfillment of such pre-requisites is verified by formative assessment. Formative Assessment

		The formative assessment is performed via interaction between teacher and students during lectures and exercises. Students are involved in questioning, in discussion and in proposing strategies to solve exercises, by means of open oral questions to the entire class. Summative assessment: Written test and a DB project. An optional mid-term written test will also be provided, which is meant to cover the first part of the course, in order to help the students to split the workload. If a student passes the mid-term written exam, he/she will take a final-term written test concerned with the second part of the course content only. The written test (2 hour) is structured as a small and simplified DB project, droved by a set of open questions, in order to check the student knowledge and the solving capability in a stressed environment (i.e. short time to answer). The DB project starts from a set of given requirements. The student, or a small team of up to 3 students, will develop the project as homework. A technical document reporting the methodological steps followed to design the DB (requirement analysis, conceptual and logical design and SQL implementation) must be provided for a discussion with the teacher. The aim of the examination is to evaluate the knowledge of the methodology and the SQL language, as well as the problem solving, work organization and structuring a written technical document capabilities. The two tests allow for a cross evaluation of the student in term of: the level of knowledge, design and implementation skill; the proper use of technical language; the clarity and completeness of explanations. The student should collect up to 18 points with the written test and up to 12 point with the project, the final mark is obtained as the sum of the two evaluations. During the project discussion, the teacher take the chance to slightly modify the mathematically computed result to better reflect the student profile and skillfulness.

Programme of "RICERCA OPERATIVA E OTTIMIZZAZIONE" ***"OPERATIONS RESEARCH AND COMBINATORIAL OPTIMISATION"*** **This course is composed of two modules: 1) Combinatorial Optimisation I, 2) Operations Research**		
F0139, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 2nd Year, 2nd Semester**		
Number of ECTS credits: 12 (workload is 300 hours; 1 credit = 25 hours)		
1) COMBINATORIAL OPTIMISATION (6 ECTS)		
Teacher: Claudio ARBIB		
1	Course objectives	Learn algorithmic techniques for some combinatorial optimization problems. Being able to formulate and solve combinatorial optimization problems using integer linear programming. Understand the complexity of the problems studied.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the Module include: - Prerequisites: graphs, definitions and basic properties. Elements of computational complexity: classes P, NP; polynomial-time reduction and the class NP-complete. Examples. Combinatorial optimization problems. Basic definitions. Examples: Transversal (vertex-cover), stable set, dominating set, edge-cover, (perfect) matching, knapsack, graph colouring (chromatic number and index), shortest path, spanning subgraph etc. Formulation as 0-1 linear programming. - Unimodular and totally unimodular matrices. Necessary conditions, sufficient conditions. Hoffmann-Kruskal's Theorem. Convex hull and ideal formulation of an integer LP. Examples: shortest path, min-cost flow, bipartite matching, interval graphs. - Recursion in CO: paths on directed acyclic graphs (DAG) and dynamic programming. Examples of application: optimal paths in a DAG, 0-1 Knapsack. [Projection of polyhedra, systems of linear inequalities, Fourier-Veronese's Theorem, Fourier-Motzkin's Method. Fundamentals of duality theory in LP. The Primal-Dual Method. Example of application: Dijkstra's Algorithm]. - Matroids and the Greedy Algorithm. Definition and characterization of a matroid (Rado's Theorem). Basics of linear algebra: linear/affine dependence-independence, bases, uniqueness of representation, Steinitz's (or exchange) Theorem. Examples: trivial matroid, graphical matroid, partition matroid, vector matroid, [matching matroid]. Matroid representation. [An industrial application]. [Rank function, submodular and supermodular set functions. Rank of a matroid. Support function of a convex set. Submodular

		polyhedron. Lòvasz's extension. Matroid polyhedron, matroid intersection, intersection of two submodular polyhedra]. - Guaranteed approximation algorithms for CO. Definitions. Polynomial approximation schemes (PTAS, EPTAS, FPTAS). Examples: TSP: double tree and Christofides; Knapsack 0-1. - Matching (perfect/non-perfect, weighted/unweighted, bipartite/non-bipartite). Weak dual inequalities: transversal, stability number, edge-cover and matching. The bipartite case: Gallai's and König's Theorem. [Bi-stochastic matrices and perfect bipartite matching. Non-bipartite matching: matching polyhedron, Edmonds' Theorem]. - Linear relaxation of a (mixed) integer linear program. Implicit enumeration: the Branch-and-Bound Method. Linear bounds, example: integer Knapsack, 0-1 Knapsack. Dichotomy on a fractional variable. Combinatorial bounds, example: TSP. - [Separation of an integer polyhedron. Tips on the Ellipsoid Method. Separation and optimization: Finding valid inequalities. Solution of LP with exponentially many variables, example: Min-cut, TSP, 0-1 Knapsack, Stable Set]. Upon successful completion of the course, the student will: o know primal-dual relations, elementary matroid theory, dynamic programming; o know and understand the main properties of unimodular matrices; o have basic notions on algorithm and problem complexity and know standard algorithms for spanning tree, bipartite matching, shortest path, heuristics for the TSP and Knapsack, general implicit enumeration, LP based algorithms (branch-and-bound); o be able to evaluate the general complexity of combinatorial optimization problems and to formulate them as 0-1 linear programming; o understand the difference between an "easy" and a "hard" problem and be able to evaluate the complexity of an algorithm and, for simple cases, of a problem; o be able to recognize whether a matrix is, or is not, totally unimodular, decide when it is convenient to use dynamic programming and realize the limits of heuristics; o be able to prove rigorously a simple theorem, prove or disprove (by counterexample) a simple conjecture and understand the role played by combinatorial optimization in applications; o be able to address submodular set functions, polyhedral combinatorics, advanced integer linear programming.
3	Prerequisites and learning activities	Notion on basic algorithms and linear algebra, linear programming
4	Teaching methods and language	Lectures and exercices. Language: Italian Ref. Textbooks: - W.J. Cook, W. H. Cunningham, W. R. Pulleyblank, A. Schrijver, *Combinatorial Optimization*. Wiley. 1997. -C.H. Papadimitriou, K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Dover Books on Computer Science. 1998. *http://www.amazon.com/Combinatorial-Optimization-Algorithms-Complexity-Computer/dp/0486402584* . -A. Sassano, *Modelli e Algoritmi della Ricerca Operativa*. Franco Angeli Editore.
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but the above mentioned pre-requisites are fundamental and their fulfillment is verified throughout formative assessment. Additional lectures or short seminars or individual homework are assigned by the teacher in case significant problems are detected. Formative Assessment The formative assessment is performed via interactive interaction between teacher and students during lectures. Students are aware since the beginning of the Course that they will be involved (in turns) in: - Questioning and discussion, by means of open oral questions to the class or to single students. - Exit Slips: students are now and then assigned written questions or exercises to be answered within the next lecture, with a possibility of selecting a student for oral presentation of her/his solution to the class. Summative Assessment that consists of mandatory written test followed by an mandatory oral exam. An optional mid-term written test is also foreseen and is meant to assess the first part of the

		course, in order to help the students to split the workload. The written test is aimed at: (1) verification of theoretical competences, and in particular of knowledge and comprehension of Course contents (2) verification of skills in understanding and solving significant exercises, and in explaining the proposed solutions. This in order to verify the ability of application of techniques learnt during the Course, of analysis of problems and synthesis of suitable solutions, and of evaluation of alternative solutions. Criteria of evaluation will be: the level of knowledge and practical ability; the property of use of the technical/mathematical language; the clarity and completeness of explanations. The oral exam will occur within one week after the written test and will mainly cover theoretical areas (definitions, implications, proofs of theorems) plus, possibly, additional subjects and exercises proposed by the teacher. Assessment breakdown: 100% mid-term plus end-of-semester summative assessment. The *written test* (2 hours and a half) consists in: (a) Six to eight multiple-choice questions, to cover point (1), 10-20% of total marks; (b) Four to seven exercises, to cover point (2), 90-80% of total marks. The *oral test* (max 1 hour) consists of one question for each serious mistake in the written test and one question in order to assess the level of formal reasoning. The final mark of the 12 CFU Course "Operations Research and Optimization" is obtained as the average among the marks of the 6 CFU Modules "Operations Research" and "Combinatorial Optimization".
2) *OPERATIONS RESEARCH (6 ECTS*)		
Teacher: Stefano SMRIGLIO		
1	Course objectives	Aim of this Module is to introduce the student to the formulation of basic Optimization problems, particularly Linear Optimization problems, and train him/her to the related solution algorithms.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of this module include: - Optimization problems: decision variables, objectives and constraints; modeling techniques and model classification - Convex optimization problems; local and global optima - Geometry of Linear Programming - The Simplex method - Duality theory in Linear Programming and its applications - Dual interpretation of the simplex method and the dual simplex method On successful completion of this module, the student should: o acquire the knowledge of Optimization problems and of the mathematical modeling techniques for complex decisions. Acquire the knowledge of some solution algorithms for Linear Programming problems. o be able to recognize optimization problems and develop mathematical models of decision-making problems. o acquire the ability of computing solutions of linear programming problems. o acquire autonomy in modeling and algorithmic choices for problems related to complex decision-making. o be able to hold a conversation and to read texts on topics related to the modeling of decision problems and Linear Programming. o acquire the ability of upgrading flexible knowledge and skills in the field of Optimization and related problems that arise in various areas, such as mathematics, computer science and management science.
3	Prerequisites and learning activities	Knowledge of vector space, scalar product, matrix product, inverse matrix.
4	Teaching methods and language	Lectures and exercises. Language: Italian Ref. Text books: -Dimitris Bertsimas and John N. Tsitsiklis, *Introduction to Linear Optimization*. Athena Scientific. 1997. -Matteo Fischetti, *Lezioni di Ricerca Operativa*. Progetto Libreria Padova. 1995. -Antonio Sassano, *Modelli e Algoritmi della Ricerca Operativa*. Franco Angeli. 1992.
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but the above mentioned pre-requisites are fundamental and additional lectures are provided in order to keep the presentation self-consistent. Formative Assessment performed via interaction between teacher and students during lectures. Students are assigned methodological questions or exercises to be answered in a

		few minutes and the solutions are collegially discussed. Some questions deserve more investigation from the students and they are assigned as homework and discussed either in class or individually. Summative Assessment consists of a written test followed by an optional oral exam. An optional mid-term written test is also foreseen and is meant to assess the first part of the course, in order to help the students to split the workload. The mid-term test only concerns the written part of the exam, not the oral assessment. The written test (2-3 hours) consists in 3 to 6 exercises concerning with different topics: - formulating mathematical programming problems - understanding the basic mathematical tools, the geometry of Linear Programming and the associated algebraic notions - solving Linear Programming problems by the Simplex Method or the Dual Simplex Method - understanding Duality theory and its applications It is designed to verify the ability of the students in the application of methods and algorithms presented during the Course. Criteria of evaluation will be: the level of knowledge and practical ability; the property of use of the technical/mathematical language; the clarity and completeness of the presentation. Each exercise is assigned with a number of points contributing to the final mark.

Programme of "PRINCIPI E PARADIGMI DI PROGRAMMAZIONE" ***"PROGRAMMING PRINCIPLES AND PARADYGMS"***		
F0142, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 2nd Year, 2nd Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Massimo TIVOLI		
1	Course objectives	The aim of the course is to present three main programming paradigms: imperative, functional and object-oriented; one main objective is to highlight different characteristics and common aspects of such paradigms. These aspects will be analyzed by accounting for both the abstractions provided by a specific language and its design principles. Giving an overview of some real languages conforming different paradigms will carry on the analysis.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Introduction and background notions - Functional languages: Lisp, Scheme - Functions, recursion, and lists - Hybrid languages: the family of Algol languages and ML - Type system - Type inference in ML - Polymorphism - Scope and Memory management - Control structures in sequential languages - Base concepts about object-oriented languages: C++ and Java On successful completion of this module, the student should : o know the fundamental principles of the different programming paradigms and the main programming techniques to be adopted by using a specific programming paradigm. o be aware of the constructs that distinguish a specific programming paradigm. o know and understand the main common and distinctive characteristics of the several existing programming paradigms o be able to describe the normal steps in developing a program using a common high level programming language and have capacity of studying in detail specific languages conforming to specific programming paradigms. o be able to declare and process appropriate data structures and to apply selection constructs in conditional statements. o be able to employ repetition constructs in programs, including nested loops. o be able to declare and apply one-dimensional arrays. o be able to implement input and output statements. o be able to construct programs which employ procedures and functions. o be able to explain how to identify program errors and how to test programs in a

		systematic manner. o be able to complete the detailed documentation of one or more programs. o be able to complete a programming project and demonstrate the skills required in the management of a software development project.
3	Prerequisites and learning activities	Background notions about the imperative programming paradigm (as gained within the course of "Fondamenti di Programmazione con Laboratorio").
4	Teaching methods and language	Lectures, exercises Language: Italian, English Ref. Text books - J. Mitchell, *Concepts in Programming Languages*. Cambridge University Press . 2003.
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website. Fulfillment of such pre-requisites is verified by formative assessment. Additional lectures or short seminars or individual homework might be provided by the teacher in case significant problems are detected. Formative Assessment The formative assessment is performed via interactive interaction between teacher and students during lectures. Students are aware since the beginning of the Course that they will be involved (in turns) in: - Questioning and discussion, by means of open oral questions to the class or to single students. - Exit Slips: students are assigned written questions or exercises to be answered in 10 minutes, and a student is then selected for oral presentation of her/his solution to the class. Summative Assessment Written test followed by an optional oral exam. In order to help the students to split the workload, an optional mid-term written test will also be provided, which is meant to cover the first part of the course. The written test includes (i) a theoretical part and (ii) a practical part. The former aims at verifying the level of knowledge of the concepts underpinning the course topics. It is organized in terms of open questions to which the student has to answer in detail. The latter is meant to verify the student's ability to reason on and solve practical problems hence providing a detailed explanation of the applied solution. Criteria of evaluation will be: (i) the level of knowledge of the course topics and the ability to reason on them in order to devise possible solutions to related problems, hence following a creative approach; (ii) the proper use of technical terminology and of the mathematical concepts underlying the course topics, e.g., functions, algebra, and logic; and (iii) the clarity and completeness of explanations. The oral exam will typically cover the areas of the written answers that need clarification plus, possibly, additional subjects proposed by the teacher. The oral test can be required either by the student, to improve grades or by the teacher, in presence of significant mistakes/misunderstandings in the written test. Assessment breakdown: 100% mid-term plus end-of-semester summative assessment. The written test (2 hours) consists in: (a) Two or three short essays, to cover point (i), 50% of total marks; (b) Two or three exercises, to cover point (ii), 50% of total marks. All parts can result in negative marks if the answer is omitted or seriously flawed. The oral test (max 1 hour) consists of one question for each serious mistake in the written test (the answer compensates the negative marks obtained therein) and one question for each 3 extra points that the student intends to add to the written test marks. The final marks of the "Programming Principles and Paradigms" 6 CFU Module are obtained as the average among the marks of the written and oral tests.

Programme of "TEORIA DELLA CALCOLABILITA' E COMPLESSITA'" ***"COMPUTABILITY AND COMPLEXITY THEORY"***
F0150, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 1st Semester**
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)
Teacher: Filippo MIGNOSI

1	Course objectives	The goal of this module is to provide an insight into the fundamental capabilities and limitations of computers. This module will expose students to two central areas of the theory of computation: computability and complexity. A number of different models of computation will be studied, and it will be shown that these are all of equivalent computational power, giving rise to a “robust” notion. The boundaries of this notion will be also analyzed. The main classes of languages and their main properties both in theory of Computability and in Complexity Theory will be studied, together with the main techniques used in both theories, such as diagonalization, the notion of reduction and of polynomial reduction.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - The Theory of computability. Countability, models of computation and Church's thesis, Turing machine, nondeterministic computation. Counters machine. - Calculable and non-calculable functions and languages, recursive and recursively enumerable sets. Languages and problems, acceptability and decidability of languages. - Diagonalization techniques, reductions, universal language, Rice's theorem. - Elements of the theory of complexity : static and dynamic measures, time and space complexity classes, the classes P and NP. The conjecture P = NP? NP-completeness. - Polynomial reductions. Theorems concerning NP-completeness. Statement of the Cook's theorem. CSAT, 3SAT and other NP-complete problems. - Class PS and NPS, Savitch's theorem. Elements of modular arithmetic and cryptography. On successful completion of this module, the student should : o have acquired motivation, definitions and basic techniques of the Theory of Computability and of the Complexity Theory. The student should be able to understand and read literature concerning the basics of both theories. The student should be able to use the language adopted by the scientific community that concerns these two theories. o be able to recognize and organize autonomously basic topics of both theories treated in the course in related applications. In particular the student should know the notion of algorithm and several of its equivalent formulations. The student should also know the boundaries of this notion. The student should know the main classes of languages and their main properties both in Theory of Computability and in Complexity Theory. The student should also know the main techniques used in these theories, such as the techniques of diagonalization, the notion of reduction and of polynomial reduction. The student should be able to apply this knowledge mainly in order to avoid to spend time in trying to find algorithms for problems that are known not to be feasible or to find polynomial algorithms for problems that are known to be hard (such as "create a program that checks if another generic program performs the desired task" or "create a polynomial time program for an NP complete problem”) and to direct his energy in other directions that can be valid under stronger assumptions (such as model checking or approximation techniques). o understand that the choice of a particular programming language depends mainly on the productivity and not on its computational power. The student should also be able to assess the relevance the main topics of both theories and be able to link the theoretical aspects of both theories with the practical aspects. o be able to describe the main results of the Theory of Computability and of the Complexity Theory to other non-specialist people in the scientific community. o be able to read and understand books and papers concerning the arguments treated in the course. The student should also be able, using also the knowledge acquired in this course, to follow more advanced studies in Computer Science (such as masters or second-level university degrees) and also other more complex courses and/or seminars related to both theories studied in the course.
3	Prerequisites and learning activities	The student must have had prior experiences in programming and at least an intuitive notion of what is an algorithm. The student must have a good knowledge of recursive algorithms and a good knowledge on how to visit trees and graphs by using recursive algorithms and by using classic data structures such as stacks and queues.
4	Teaching methods and language	Lectures, exercises Language: Italian, English Ref. Text books -Hopcroft, Motwani, Ullman, *AUTOMI LINGUAGGI E CALCOLABILITA'* . Pearson, Addison Wesley. -Arora Barak, *Computational Complexity: a modern approach*. Cambridge University

		press.
5	Assessment methods and criteria	The exam will be both written and oral. The written represent a filter in order to allow to make the oral exam only to the students that have a sufficient knowledge of the basic definitions and of the main theorems. During the oral exam the questions to the student will cover the whole program of the course.

Programme of "RETI DI CALCOLATORI" ***"COMPUTER NETWORKS"***		
F0144, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 1st Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Luigi VETRANO		
1	Course objectives	The course is designed to teach students fundamentals of analysis and design of computer networks. Introduction to the basic concepts of computer and communication networks, like flow control, congestion control, end-to-end reliability, routing, framing, error-recovery, multiple access and statistical multiplexing. In-depth presentation of the different networking layers, with emphasis on the Internet reference model. Protocols and architectures such as the TCP, IP, Ethernet, wireless networks etc. are described in order to illustrate important networking concepts.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Computer Networks, the Internet and the World Wide Web - Application Layer and network applications - Transport Layer: TCP and UDP - Network Layer and Routing - Data Link Layer and Local Area Networks - Wireless Networking - Security - Network management On successful completion of this module, the student should : o know the network architecture, both hardware and software and understand various network protocols; o understand one or more routing algorithms; o be able to write software to implement a client-server application using the socket programming API; o demonstrate an understanding of public-private key encryption and how it can be o used in an authentication scheme; o be aware of quality of service; o be able to install, maintain, and troubleshoot computer hardware., Microsoft-based desktop and server operating systems; o Install and maintain physical cabling infrastructure of a network; o be able to create and maintain a secure network. o be able to identify some of the factors driving the need for network security. o be able to identify and classify particular examples of attacks; o be able to define the terms vulnerability, threat and attack; o be able to identify physical points of vulnerability in simple networks.
3	Prerequisites and learning activities	The student must have had prior experiences in programming and at least an intuitive notion of what is an algorithm.
4	Teaching methods and language	Lectures, exercises Language: Italian, English Ref. Text books -James F. Kurose, Keith W. Ross, *Computer Networking: a top-down approach, featuring the Internet*. Addison Wesley. 2007. *http://www.aw.com/kurose-ross/* -Andrew S. Tanenbaum, *Computer Networks*. Prentice Hall. 2002. *http://www.cs.vu.nl/~ast/* -Behrouz A. Forouzan, *Computer Networks and Internet* .
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website.

		Formative Assessment The formative assessment is performed via interaction between teacher and students during lectures. Students are involved in questioning and discussion, by means of open oral questions to the entire class. Many hours are spent in laboratory activities where students will learn how to capture sensitive traffic data, how to read them and how to prevent attacks to the network infrastructure. Summative Assessment The summative assessment consists of a one hour written test (mandatory) followed by either an oral test or an additional individual project (both optional). The oral exam will occur within the same exam session of the written test, and it will typically cover the areas of the written answers that need clarification, plus a subject of one's choice. The oral exam (max half an hour) will test the student's ability to engage in discussion of issues relevant to the topics discussed during the course. Criteria of evaluation will be the level of knowledge and the fluency in the technical language of computer networking. This part will be weighted 33 % compared to the written test. Optionally, to increase their score, the students might provide a well documented individual project on interesting topics about computer networking that are not covered during the lectures. This kind of activity will contribute with an additional score from 1 to 5.

Programme of "TECNOLOGIE DEL WEB" ***"WEB TECHNOLOGIES"***		
F0149, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 1st Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Alfonso PIERANTONIO		
1	Course objectives	The main objectives is to provide the students with the insights of the Internet programming and how to design and implement complete realistic-scale distributed applications on the web. At the end of the course, the students will be familiar with design-methodologies necessary for managing the problem complexity, client-side programming, server-side programming, database connectivity. Moreover, they will be proficient in using the following languages, systems, and techniques: HTML/CSS, DOM, JavaScript, jQuery, PHP, MySQL, Templating, beContent, etc
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Introduction. Three-tier architecture. Graphics and Communication. - Client-side: HTML/CSS, DHTML, JavaScript/ECMAScript, DOM, jQuery. - Server-side: PHP, MySQL, templating and separation of concerns (presentation, business logics, presentation logics). - Sessions and their management. User management: authentication, authorization and permissions. - Modelling and designing web application with beContent. - Case study (eg. ecommerce, news portal) On successful completion of this module, the student should : o know and understand design and implementation of computing programs; o understand the methodologies and the technologies necessary for the development of web applications; o be able to apply the most recent techniques and technologies to the design and development of web applications including the client and server-side in order to achieve a higher degree of usability and trust; o be able to evaluate and detect the right technologies and the best interfaces to achieve the design goals; o be able to assess and choose optimal methods and approaches for specification, design and implementation etc.; o be able to choose the right tool and to access information in written and electronic form, including library and Internet search etc. ; o be able to explain in detail the concepts relating to information discovery and to optimise web pages for searching; o be able to continue learning all the evolving technologies related to the development of web applications.

3	Prerequisites and learning activities	The main prerequisites for this course are: the ability to program with an object oriented language (eg. Java or C++), how to design a database, entity/relationship diagrams, SQL language, finally a familiarity with the web ecosystem is important.
4	Teaching methods and language	Theoretical and practical lectures where the techniques will be first illustrated and then demonstrated. Language: Italian Ref. Text books -D Goodman, *Dynamic HTML The Definitive Guide*. O'Reilly. -D Sklar, *Learning PHP 5*. O'Reilly. -R Nixon, *Learning PHP, MySQL, JavaScript, and CSS: A Step-by-Step Guide to Creating Dynamic Websites*. O'Reilly. -S Ceri, P Fraternali et al, *Progettazione di Dati e Applicazioni per il Web*. McGraw-Hill. -H E Williams, D Lane, *Web Database Applications with PHP and MySQL*. O'Reilly.
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website. Formative Assessment The formative assessment is performed via interaction between teacher and students during lectures. Students are involved in questioning and discussion, by means of open oral questions to the entire class. Summative Assessment The summative assessment consists in realizing a final project (in team) followed by an oral exam. The final project consists in designing and implementing a complete web application (whose specification is made available during the course), in order to asses: a) whether the student knows the basic concepts of web programming and b) the student's practical skills in - selecting and adopting the most convenient navigation and presentation model (20% of total marks) - designing and implementing the client-side and server-side components (40% of total marks) - managing the separation of logics (20% of total marks) The oral exam consists of discussing the project with a specific emphasis in assessing the knowledge of the methodologies and technologies presented during the course (20% of total marks).

Programme of "LINGUAGGI DI PROGRAMMAZIONE E COMPILATORI" ***"PROGRAMMING LANGUAGES AND COMPILERS"***		
F0151, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 2nd Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Sergio OREFICE		
1	Course objectives	The course is an introduction in compiler design and presents the main principles and techniques used during the compilation process, focusing on the analysis phases of a compiler
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Formal language basic notions. Introduction to compiling: the analysis-synthesis model, the phases of a compiler, interpreters - Lexical analysis: tokens-patterns-lexemes, input buffering, Thompson's algorithm. - LEX: software architecture and syntax - Syntax analysis: parsing top-down, parsing bottom-up. Nonrecursive predictive parsing - LR parsers: methods SLR, LALR, canonical LR - LR grammars and ambiguous grammars. - YACC: software architecture and syntax - Syntax-directed translation. Syntax-directed definitions: synthesized and inherited attributes, S-attributed definitions and L-attributed definitions - Translation schemes. Semantic analysis and type checking - Intermediate code generation. Three-address code: syntax and examples of generation On successful completion of this module, the student should : o acquire knowledge of the structure and the role of compilers; o acquire knowledge and understanding of the principles and techniques used in the

		compilation process; o be able to apply the methodologies learnt by the course to the development of small software projects regarding the construction of parts of compiler through tools like Lex and Yacc; o be able to assess the features of the lexical and syntactic components of a syntax-directed translation specification and choose the most suitable solution for its development; o understand and explain the relationships between tools and results from the formal language theory and their application to the analysis phases of compilers; o demonstrate capacity for reading and understanding other texts on related topics.
3	Prerequisites and learning activities	The main prerequisite for this course is the knowledge of the basic concepts of the formal language theory and programming.
4	Teaching methods and language	Lectures and exercises. Language: Italian Ref. Text books - A.V. Aho, R. Sethi, J.D. Ullmann, *Compilers, principles, techniques and tools*. Addison-Wesley, Reading, Mass.
5	Assessment methods and criteria	Formative assessment: the formative assessment is performed via interaction between teacher and students during lectures. The students are encouraged to actively participate to the lectures by making questions and discussing the theoretical concepts presented and the solutions adopted in the developed examples. Summative assessment: written test on the subjects treated in the course. An optional mid-term written test will also be provided, which is meant to cover the first part of the course, in order to help the students to split the workload. The written test (lasting 2 hours) consists of a set of questions for the verification of theoretical/formal competences and for the verification of skills in understanding and solving significant exercises. Criteria of evaluation will be: the level of knowledge of the principles and techniques presented in the course, as well as the ability to apply them; the clarity and completeness of explanations.

Programme of "INGEGNERIA DEL SOFTWARE CON LABORATORIO" ***"SOFTWARE ENGINEERING WITH LAB"***		
F0146, COMPULSORY **First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 2nd Semester**		
Number of ECTS credits: 9 (workload is 225 hours; 1 credit = 25 hours)		
Teacher: Antinisca DI MARCO		
1	Course objectives	This Module provides the students with an excellent education incorporating industry-relevant, applied laboratory-based instruction in both the theory and application of software engineering. The deep theoretical knowledge and the practical experience in the Lab sessions aim to prepare the students for productive careers in industry and government and to enable them to meet current and future industrial challenges and emerging software trends.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: THEORY - Software Process Models (waterful, incremental delivery). Agile Development - Requirement Engineering: Requirement Definition and Requirement Specification Process. System Models - Software Architecture Design. Architectural Patterns - Design Patterns and Antipatterns. Software Design: Object-Oriented Design - Verification and Validation. Testing: testing process, planning, testing strategies for test case design, Black-box and White-box testing - Project Management: project scheduling and risk analysis LABORATORY - UML Language as formalism for modelling software systems. UML Meta-Model. - UML for requirement analysis and specification, for project design, implementation and documentation. - UML in practice: MagicDraw Tool. - Models' transformation: ACCELEO

		On successful completion of this module, the student should : o Acquire deep knowledge of the core areas of software engineering: requirement engineering, architectural design, object-oriented design and implementation, verification and validation, testing, project management (,). (data structures, theory of computation, operating systems, compilers, programming languages, computer architecture). o Be able to apply the principles dealt with in the course, such as i) to interpret and properly use UML diagrams during requirement engineering, software architecture design and low design; ii) to specify functional, non functional requirements iii) apply testing techniques on simple cases iv) define ACCELEO model-to-code transformation starting from UML models. o Be able: i) to identify, formulate, and solve software engineering problems; ii) to explain, argue and defend his/her design decisions o Be able to: i) organize, develop and manage a project and the relative documentation ii) compose a report documenting the developed project iii) to convey technical material through oral presentation and written reports. o Develop capacity to regularly engage in exploring, learning and applying state-of-the-art software technologies to the solution of software engineering problems. o Have capacity to be an effective software development team member who contributes innovative software design solutions to the resolution of IT problems. Be able to communicate effectively and successfully, both individually and within multi-disciplinary teams
3	Prerequisites and learning activities	The main prerequisite for this course is the knowledge of the basic concepts of the formal language theory and programming.
4	Teaching methods and language	Lectures and exercises. Language: Italian Ref. Text books -Ian Sommerville, *Software Engineering*. Addison-Wesley. -Braude, Bernstein, *Software Engineering: Modern Approaches , 2nd Edition*. -H.E. Eriksson e altri, *UML 2 Toolkit*. Wiley. 2004. -Patrick Grässle, Henriette Baumann, Philippe Baumann, *UML 2.0 in Action, A Project-Based Tutorial*. Packt Publishing . 2005.
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website. Fulfilment of such pre-requisites is verified by formative assessment. Additional lectures or short seminars or individual homework are provided by the teacher in case significant problems are detected. Formative Assessment The formative assessment is performed via interactive interaction between teacher and students during lectures. Students are aware since the beginning of the Course that they will be involved (in turns) in: - Questioning and discussion, by means of open oral questions to the class or to single students. - Exit Slips: students are assigned written questions or exercises to be answered in 10 minutes, and a student is then selected for oral presentation of her/his solution to the class. Summative Assessment: Written test, Project followed by a project presentation. An optional mid-term written test is also be provided, which is meant to cover the first part of the course, in order to help the students to split the workload. The written test is aimed at: (1) verification of theoretical competences, and in particular of knowledge and comprehension of Course contents (2) verification of skills in understanding and solving simple exercises, and in explaining the proposed solutions. This in order to verify the ability of application of techniques learned during the Course, of analysis of problems and synthesis of suitable solutions, and of evaluation of alternative solutions. The project and its presentation are aimed at: verification of skills in understanding and solving significant exercises, and in explaining the proposed solutions. This in order to verify the ability of application of techniques learned during the Course, of analysis of problems and design of suitable solutions, and of evaluation of alternative solutions. Criteria of evaluation will be: the level of knowledge and practical ability; the property of use of the technical language; the clarity and completeness of explanations. The oral exam will occur within one week of the written test and will typically focus on a technical presentation of the solutions and decisions taken in the project. Additional questions on theoretical aspects can be raised by the teacher during the presentation. The oral exam is mandatory.

		Assessment breakdown: 100% mid-term plus end-of-semester summative assessment. The written test (1 hour and half) covers the 1/3 of the total mark and consists in: (a) short essays (max 600 words) on theoretical aspects; (b) One or Two exercises. The project and its related documentation (corresponding to 1/3 of the total marks) has to be sent by email to the teacher before the written exam date. Criteria of its evaluation will be the correct usage of theoretical topics in the projects and the quality of the presentation of the related documentation. The oral exam (corresponding to 1/3 of the total marks) focuses on the presentation of the project by using slides. Criteria of its evaluation will be the ability in proposing and defending the solutions devised in the project.

Programme of "ALGORITMI PER SISTEMI DISTRIBUITI" ***"ALGORITHMS FOR DISTRIBUTED SYSTEMS"***		
F0057, OPTIONAL **First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 1st Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Guido PROIETTI		
1	Course objectives	The course provides the foundations for designing and analyzing (distributed) algorithms for both cooperative (reliable, faulty, concurrent), and non-cooperative distributed systems (elements of cryptography, equilibria in strategic distributed systems, algorithmic mechanism design).
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Algorithms for COOPERATIVE Distributed Systems (DS) 1. Leader Election 2. Minimum Spanning Tree 3. Maximal Independent Set - Algorithms for UNRELIABLE DS: The consensus problem - Algorithms for CONCURRENT DS: Mutual exclusion - Security aspects of DS: Elements of cryptography - Algorithms for NON COOPERATIVE (i.e., strategic) DS 1. Equilibria in networks 2. Algorithmic mechanism design On successful completion of this module, the student should : o be able to: 1) understand the difference between a centralized and a distributed algorithm; 2) analyze the resources (space and time) needed by a distributed algorithm; 3) known efficient algorithms for basic computational distributed problems (leader election, consensus, etc.); 4) understand the difference between a canonical and a strategic distributed system. o Acquire the capacity of abstracting models and formal algorithmic problems from real distributed computational problems, and designing efficient algorithmic solutions. o Be able to develop, through the presentation and the comparison of different solutions to a given problem, capacity to identify independently their most efficient solution. o Acquire the capability of formally presenting and modelling concrete problems, focusing on their main features and discarding the inessential ones. o Be able to develop competencies and abilities necessary in future studies, especially with respect to doctoral studies on algorithmic topics.
3	Prerequisites and learning activities	Knowledge of basic courses of discrete mathematics and algorithms.
4	Teaching methods and language	Lectures, Exercises. Language: English Ref. Text books - P. Ferragina e F. Luccio, *Crittografia*. Bollati Boringhieri. -H. Attiya e J. Welch, *Distributed Computing*. Wiley. -C. Montet e D. Serra, *Game Theory & Economics*. Palgrave.
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website. Formative Assessment The formative assessment is performed via interaction between teacher and students during lectures. Students are involved in questioning and discussion, by means of open oral questions to the entire class or to single students.

		Summative Assessment: Oral exam. An optional mid-term written exam will be also provided, which is meant to cover the first part of the course, in order to help the students to split the workload. If a student passes the mid-term written exam, she will take a final-term oral exam concerned with the second part of the course content only. The mid-term written exam (lasting 2 hours) consists of 10 multiple-choice tests, plus an open-answer question, plus the design of a distributed algorithm. It is aimed at verification of theoretical competences, and in particular of knowledge and comprehension of course's content. A correct answer to a test provides 3 points, a missing answer counts 0, while a wrong answer provides a -1 penalization. The final result will be given by summing up all the obtained points and is normalized on a 10 basis. The open-answer question provides with a mark between 0 and 10. Finally, the design of a distributed algorithm provides with a mark between 0 and 10, with a maximum of 5 points for the correctness, 3 points for the efficiency, and 2 points for the analysis, respectively. The oral exam will typically cover the areas of the written answers that need clarification (if any), plus a subject of one's choice. The oral exam (max 1 hour) will test the student's ability to engage in discussion of issues relevant to the topics discussed during the course. Criteria of evaluation will be the level of knowledge and the fluency in the technical language of distributed algorithms.

Programme of "RETI DI CALCOLATORI EVOLUTE" ***"ADVANCED COMPUTER NETWORKS: ARCHITECTURE"***		
F0159, OPTIONAL **First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 1st Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Giuliano PARIS		
1	Course objectives	The module presents the architectures, protocols and services of the current and future telecommunication networks and ICT applications. After introducing the requirements (bandwidth, real time, etc.) of voice, data and video and a brief description of the legacy PSTN (TDM) infrastructures (originally designed only for phone services), the main features of an integrated multiservice IP-based backbone are described. Such IP based architecture is an essential element for the growing digital services and applications (i.e. web 2.0, cloud computing, big data, etc.) Among the various access networks the fixed access (i.e. ADSL, NGAN-fiber based, etc.) and mobile and wireless access are covered. Regarding the mobile technologies, the course presents the evolution from GSM/GPRS/EDGE to 3G systems (UMTS/HSPA) up to 4G-LTE architectures, services and applications. For the local environments wired (LAN) and wireless (WiFi) standards are described including the upcoming wifi-mobile integration. Finally Voice over IP and audio/video streaming architectures and protocols are described.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Services requirements and Network architectures (access technologies, switching nodes, etc.) - Basics on transmission and information theory - Circuit and packet switching - Legacy PSTN/ISDN infrastructures - The fixed access (Fiber to the X architectures) towards the UltraBroadband Services - Signaling protocols (SS7) and call scenarios (PSTN services, IN services) - Intro to mobile networks and GSM architecture and services - Mobile procedures (authentication, location update, handover, call scenario, roaming) - TCP/IP architecture fundamentals - Local Area Networks (LANs) standards and protocols - Wi-Fi networks and security issues - Mobile data networks evolution: from GPRS/EDGE to UMTS/HSPA and 4G-LTE systems; Voice over IP (VoIP) architectures and protocols (H.323, SIP) and audio/video streaming. On successful completion of this module, the student should : o understand the main features of current and future IP multimedia network architectures and protocols,

		o be able to analyse and evaluate the performance of fixed, mobile and wireless access networks, o be able to analyse and critically assess the design of advanced computer networks o be able to understand the potentialities and challenges of new technologies and architectures such as Cloud Computing, Big Data, Internet of Things, etc. o understand and explain the design and application issues on mobile and wireless networks and services with particular emphasis on 3G-4G systems and IEEE 802.11 standards, o be able to identify and assess possible research opportunities and difficulties within the course scope.
3	Prerequisites and learning activities	Basics on mathematics and physics
4	Teaching methods and language	The course consists of a main part and a series of seminars on topics such as Fiber Optics and Next Generation Access Networks. Language: Italian, English Ref. Text books -A.R. Prasad, N.R. Prasad, *802.11 WLANs and IP Networking: Security, QoS, and Mobility.* Artech House. -C. Cox, *An Introduction to LTE: LTE, LTE-Advanced, SAE, VoLTE and 4G Mobile Communications*. WILEY. -D. Collins, *Carrier Grade Voice Over IP.* McGraw-Hill. -D. E. Comer, *TCP/IP principles, protocols, and architectures*. Prentice Hall. -E. Brynjolfsson, A. McAfee, *The Second Machine Age: Work, Progress, and Prosperity in a Time of Brilliant Technologies.*
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, Course pre-requisites are clearly stated on the Module website. Formative Assessment The formative assessment is performed via interaction between teacher and students during lectures. Students are periodically involved in discussions, by means of open oral questions and exercises held during the Course duration. Summative Assessment consists of a written test (50 multiple choice questions) followed by an oral exam, both concerning the whole course contents.

Programme of "INGEGNERIA DEL SOFTWARE II" ***"SOFTWARE ENGINEERING II"***		
F0156, OPTIONAL **First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 1st Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Vittorio CORTELLESSA		
1	Course objectives	This course aims, on one hand, at in-depth enhancing the students' knowledge acquired in the Software Engineering course, on the other end at introducing new topics. The latter ones can be summarized as: dealing with functional and non-functional properties of component-based systems, studying quantitative properties of software systems like reliability and performance, introducing advanced notions on UML and related technologies. The course is intended to develop students' capabilities for modeling and analyzing software systems, as well as capabilities of getting use to different tools that support such activities. This experience shall make them able, in future, to fully exploit the functionalities of any tool they will be exposed to.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: o Software Architectures o Model-Driven Engineering o UML profiling o Non-functional Validation of Software o Performance Analysis o Reliability Analysis On successful completion of this module, the student should : o Be able to use UML profiles for tailoring software architectures to specific domains.

		o Be able to analyze a software architecture from a non-functional viewpoint, with particular emphasis for reliability and performance aspects. o Be aware of potential of software models as primary artifacts in the whole software engineering process. o Be experienced in the integration of multiple tools for the development and analysis of software systems. o Be able to identify and define the computing requirements appropriate to its solution. o Be able to design models that reflect abstract architectures of software systems. o Have effectively worked on team to deliver some group homework.
3	Prerequisites and learning activities	Students should have attended the "Software Engineering I" course.
4	Teaching methods and language	The module includes 54 hours of frontal lectures plus 30 hours of on-demand clarifications in the teacher's office. The frontal lectures are partitioned in theory and exercises. Language: Italian, English Ref. Text books - C.U.Smith, L.Williams, *Performance Solutions*. Addison Wesley. 2002. - Ian Sommerville, *Software Engineering*. Addison-Wesley.
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website. Fulfilment of such pre-requisites is verified by formative assessment. Formative Assessment The formative assessment is performed via interactive interaction between teacher and students during lectures. Students are aware since the beginning of the Course that they will be involved (in turns) in: - Questioning and discussion, by means of open oral questions to the class or to single students. Summative Assessment: Group project followed by an optional oral exam. The group project is aimed at: (1) verification of theoretical competences, and in particular of knowledge and comprehension of Course contents; (2) verification of skills in understanding and solving significant problems, and in explaining the proposed solutions, (ii) capability of collaborative work. This in order to verify the ability of application of techniques learnt during the Course, of analysis of problems and synthesis of suitable solutions, and of evaluation of alternative solutions. Criteria of evaluation will be: the level of knowledge and practical ability; the property of use of the technical/mathematical language; the clarity and completeness of explanations. The oral exam will occur within one week of the project delivery and will typically cover the areas of the project that need clarification plus additional subjects proposed by the teacher. The oral test takes place for all students. Assessment breakdown: 100% end-of-semester summative assessment.

Programme of "INGEGNERIA DEL WEB" ***"WEB ENGINEERING"***		
F0162, OPTIONAL **First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 2nd Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Giuseppe DELLA PENNA		
1	Course objectives	The course aims to provide basic knowledge about all the kinds of web application and the technologies used to implement them. After an in-depth study of base technologies such as XML, markup languages (HTML 4 and HTML 5) and style sheets, the course will focus on server-side and client-side programming languages, in particular Java and JavaScript. Finally, we will discuss accessibility and validation issues for web applications.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Web Standards. Accessibility and Usability: what they mean and how to achieve them. - Web Content Accessibility Guidelines (WCAG). XML and markup languages. - Structure of web contents: HTML4, XHTML1, HTML5. Correct use of HTML and crossbrowser compatibility techniques. Accessible HTML. Rich User Applications (RIA) accessibility. Web content validation. - Presentation of web contents: CSS2 and CSS3. Correct use of style sheets and crossbrowser compatibility techniques. Graceful degradation of style sheets. CSS stylesheets and web page layout: base techniques. Responsive design.

		- Logic of web contents, server side: Java Servlets. Sessions in Java web applications. Databases in Java web applications: JDBC, connection pooling. Dynamic content generation: Java template engines. Web application security: server-side basic techniques. - Logic of web contents, client side: Javascript. Javascript as an object-oriented programming language. The HTML Document Object Model. The CSS Document Object Model. The HTML event model. DOM manipulation with Javascript. Gracefully degrading Rich User Interaction with Javascript e CSS. AJAX. Introduction to JQuery. On successful completion of this module, the student should be able to: o understand all the basic web development technologies, o apply all the latest technologies to the development of web applications, o develop server-side web applications in Java and create dynamic, accessible, versatile client-side interfaces, o assess the accessibility level of a web site and choose the most suitable technologies for its development o continue learning all the evolving technologies related to the development of web applications. o be able to analyze a problem, identify and define the computing requirements appropriate to o its solution. o be able to design, implement and evaluate a computer-based system, process, component, or program to meet desired needs. o be able to function effectively on teams to accomplish a common goal. o be able to use current techniques, skills, and tools necessary for computing practice.
3	Prerequisites and learning activities	Knowledge of basic math functions and sets, and logical expressions. Basics of object-oriented programming can be acquired with the integrated module Programming Laboratory.
4	Teaching methods and language	Lectures, Exercises. Language: Italian, English Ref. Text books -R. Barbuti, P. Mancarella e F. Turini, *Elementi di Semantica Operazionale*. 2004/2005. *https://informatica.di.univaq.it/getres.php?resid=1171* -R. Barbuti, P. Mancarella, D. Pedreschi, F. Turini, *Elementi di Sintassi dei Linguaggi di Programmazione*. Corso di Laurea in Informatica Università di Pisa a.a. 2004/05. *https://informatica.di.univaq.it/getres.php?resid=746* -R. Barbuti, P. Mancarella e C. Montangero, *Semantica Operazionale*. *https://informatica.di.univaq.it/getres.php?resid=747* -M. Autili, P. Inverardi, *Semantica Operazionale di +/- Java - 03 Dicembre 2010* . 2010. *http://informatica.di.univaq.it/getres.php?resid=1053*
5	Assessment methods and criteria	Formative assessment: the students are encouraged to actively participate to the lectures by making questions and discussing the solutions adopted in the developed examples. Summative assessment: project development and presentation (in team) and oral exam (individual) (80:20). The project to be developed consists of a complete website, whose specifications are given by the teacher and are valid for the entire academic year of publication. The project evaluation aims to verify its level of completion and documentation (15% of total mark), the proper use of the basic web development technologies (30%), the level of client-side and server-side (Java) programming (20%), the ability to exploit the latest technologies (5%), and the crossbrowser compatibility, usability and accessibility features (30%). The oral exam starts from the discussion of the project, and aims to verify the achieved level of teamwork (20% of total mark) as well as the individual contribution to the project, with strong emphasis on the knowledge of the main technologies presented in the course (40%), the ability to apply them where and as appropriate (20%), as well as the ability to design, implement and properly present a complex web application (20%).

Programme of "INTELLIGENZA ARTIFICIALE"
"ARTIFICIAL INTELLIGENCE"

F0072, OPTIONAL
First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 1st Semester

<table>
<tr><td colspan="3">Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)</td></tr>
<tr><td colspan="3">Teacher: Pasquale CAIANIELLO</td></tr>
<tr><td>1</td><td>Course objectives</td><td>This module introduces the fundamentals of Artificial Intelligence by exploring the basis of its Logic and Knowledge Representation techniques. The course modules aims at equipping students with the fluent skills in Prolog and Python programming leading to exploring real didactic applications in the paradigms in knowledge representation and reasoning, in order to enable them to understand contemporary techniques in machine learning and to hybridize and evaluate different successful artificial intelligence algorithmic strategies.</td></tr>
<tr><td>2</td><td>Course content and Learning outcomes (Dublin descriptors)</td><td>Topics of the module include:
- Heuristic problem solving
- Search Algorithms
- Prolog and Python
- Planning and Control
- Natural Language Processing
- Probabilistic and Information Theoretic Reasoning
- Basics of Machine Learning

On successful completion of this module, the student should be able to:
o understand the history, development and various applications of Artificial Intelligence;
o familiarize with propositional and predicate logic and their roles in logic programming;
o write code in Prolog and Python;
o learn the knowledge representation and reasoning techniques in rule-based systems, case-based systems, and model-based systems;
o appreciate how information uncertainty is being tackled within knowledge representation and reasoning, in particular, with probability, possibility theory, and fuzzy logic;
o master the skills and techniques in heuristic exploration in its applications to planning, machine learning, artificial neural networks, and genetic algorithms;
o realise an implemented solution for a project with intelligent skill;
o apply and integrate various artificial intelligence techniques for developing and maintaining intelligent systems;
o explore the nature of human intelligence and its role in problem solving;
o deepen thoughts and understanding of human abilities in learning, reasoning, and planning;
o appreciate the rooted philosophical arguments in logic and its impact on human thoughts.</td></tr>
<tr><td>3</td><td>Prerequisites and learning activities</td><td>Competence in programming and data structures.</td></tr>
<tr><td>4</td><td>Teaching methods and language</td><td>Lectures, Exercises.
Language: Italian, English
Ref. Text books
-D. Poole, A. Mackworth, R. Goebel, Computational Intelligence a logical approach. Oxford Univercity Press. 1998.
-S. Russell, T. Norvig, Intelligenza Artificiale un approccio moderno. Prentice Hall. (vol. 1) 2005</td></tr>
<tr><td>5</td><td>Assessment methods and criteria</td><td>Summative Assessment:
One written test, about five homework, facultative class presentations, and a final written technical report.
Written test: Student may take a midterm exam or a final exam
Homework: a minimum of five assignments suggested in class
Final presentation: a pdf

<table>
<tr><th>Learning Outcome Assessed</th><th>Testing method adopted</th></tr>
<tr><td>Baseline theoretical and pragmatic knowledge provided through lessons and guided online exploration</td><td>Multiple Choice tests
Open Answer questions</td></tr>
<tr><td>Programming capabilities</td><td>Homework reports throughout the course</td></tr>
<tr><td>Project design for complex tasks</td><td>Final Presentation
Implementation Test</td></tr>
</table>

<table>
<tr><th>Formative and Summative Assessment towards the definition of a final grade</th><th>% weight on the final grade</th></tr>
<tr><td>In class participation</td><td>10</td></tr>
</table>
</td></tr>
</table>

		Homeworks	30	
		Final presentation	10	
		In class written Tests	30	
		Project Evaluation	20	

Programme of "MODELLI E ALGORITMI PER LA FINANZA AZIENDALE I" ***"MODELS AND ALGORITHMS FOR FINANCIAL MANAGEMENT I"***		
F0157, OPTIONAL **Second Cycle Degree in COMPUTER SCIENCE, 1st Year, 1st Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Giuseppe ALESII		
1	Course objectives	This module is an introduction to programming applied to quantitative modeling for financial management and asset pricing. The main objective of the course is to introduce students to programming – on Spreadsheets and matrix languages such as Aptech-Gauss™ or MatLab – applied to models and algorithms most frequently used in finance.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Introduction to finance from three points of view: (a) macro-finance approach: flow of funds matrix; (b) micro-finance approach: Irving Fisher's Separation Theorem; (c) financial accounting approach: financing deficit with different sources - Fixed income securities valuation under the assumption of certainty: interest rates risk sensitivity: (a) Macaulay duration (b) term structure of interest rates, spot, forward and short rates; (c) Fisher Weil duration. - Shares valuation under the assumption of certainty, dividend discount model: (a) Myron Gordon's dividend growth model; (b) Modigliani Miller 1961, growth model; (c) fundamental indexes and dividend discount models, cross sectionals and longitudinal evidences; - Capital Budgeting, choosing real investments in an industrial firm under the assumption of certainty: (a) criteria: i. Payback Period, ii. Internal Rate of Return, iii. Net Present Value, iv. Profitability Index, v. Economic Value Added (as an extension of Modigliani Miller 1961); (b) methods: i. capital rationing, individual and multiple periods cases, linear programming applied to multiple periods cases, geometric approach and Excel Solver Tool application; ii. optimal harvesting, Faustmann problem individual and repeated cycles problems solutions. (c) comparative static and dynamic optimization: Richard Bellman's Dynamic Programming in a deterministic framework: i. continuous and discrete control variables ii. application to the choice and optimal dynamic management of investment project for renewable and exhaustible resources. - Risky assets (non derivatives) valuation : (a) Markowitz's portfolio selection, analytic solutions to the following portfolio selection problems: • efficient portfolios; • Minimum variance opportunity set; • global minimum variance portfolio; • tangency portfolio; • orthogonal portfolio; (b) Single index model, Market model; (c) Sharpe Lintner Mossin CAPM; • analytic derivation; • capital budgeting application: risk adjusted discount rate, Certainty equivalent approach; • financial leverage and its influence on hurdle rates, Hamada 1972; (d) multifactor models; • Ross APT; • Three Factor Model di Fama French; (e) which asset pricing is most suited for capital budgeting decisions: Jeremy Stein 1996. On successful completion of this module, the student will: o have a thorough and deep knowledge of capital budgeting / securities pricing models under certainty and portfolio selection and basic asset pricing models. Moreover, she/he must be knowledgeable with the general themes of finance. o be able to use her/his programming skills in simple Excel spreadsheets and/or in high programming languages such as Gauss or MatLab, not only for financial models and algorithms dealt with at lesson but also for other similar problems. o have acquired general skills in the field of algorithms and applied programming for financial modeling which enable him/her to make educated choices in a problem solving practice framework. The student should be able to retrieve financial data, compute main descriptive statistics, estimate parameters of main asset pricing models (Information procurement and analysis). The student should be able to apply integer and non integer linear programming and numerical non linear optimization algorithms in both capital

<table>
<tr><td></td><td></td><td>budgeting and asset pricing (algorithm choice).
o be capable to give a presentation both in front of a general practitioners' audience and a more academic one about the models dealt within the course.
o have acquired a method of study both thanks to a wide knowledge of the main streams in which financial modeling is evolving, theoretical continued learning, and a confident practice with respect to the main high level programming languages, GAUSS and MatLab, which are continually evolving, best practice continued learning.</td></tr>
<tr><td>3</td><td>Prerequisites and learning activities</td><td>Pre-Assessment Prerequisites are the knowledge of Numerical Analysis, Calculus, Stochastic Calculus, Mathematical Statistics and good programming ability for the following applications: A) any spreadsheet, e.g. Excel, Calc; B) any matrix oriented language, e.g. MatLab, Gauss, Ox, Octave, Scilab. In the computer lab classes, Gauss will be used. Univariate and multivariate calculus is applied in most of the models. A solid background in probability theory is required.
Formative Assessment is performed during the whole teaching period through
- class participation during lessons: students may be asked to answer questions about topics dealt with at lesson; students may ask instructor questions during lessons both about the very topic dealt with at lesson and about correlated topics they are particularly interested in;
-summary of previous week lessons: a student is randomly selected to sum up topics dealt with in the previous sessions, actually introducing extant session;
-short seminars: students are required to apply their skills in Calculus, Stochastic Calculus, Numerical Analysis and Mathematical Statistics to specific problems in finance, proposing their own solutions previously prepared as assigned homework.</td></tr>
<tr><td>4</td><td>Teaching methods and language</td><td>Lectures, Exercises.
Language: Italian, English
Ref. Text books
-Thomas E. Copeland, J. Fred Weston, and Kuldeep Shastri, Financial Theory and Corporate Policy. Addison-Wesley 2005. (4th Edition).
-Luenberger, D, Investment Science. Oxford University Press. 1998.
-Edwin J. Elton, Martin J. Gruber, Stephen J. Brown, William N. Goetzmann, Modern Portfolio Theory and Investment Analysis. Wiley. 2006.</td></tr>
<tr><td>5</td><td>Assessment methods and criteria</td><td>Summative Assessment consists of written tests, homework and oral exam.
Written test: Students can decide to take two (mid-term and final) written tests or a comprehensive final test, that can be taken also in case of failing the two tests.
<table>
<tr><th>Learning Outcome Assessed</th><th>Testing method adopted</th></tr>
<tr><td>Baseline theoretical knowledge provided through lessons and suggested reading list</td><td>Open questions to be answered through short essays</td></tr>
<tr><td>Problem solving involving symbolic calculus and stochastic calculus capabilities</td><td>Questions about model building and algorithms tuning for specific formal problems</td></tr>
<tr><td>Programming capabilities</td><td>Small problems in class assignments to be programmed in a high level language, e.g. MatLab, Gauss, Ox, Scilab</td></tr>
</table>
Homework and projects: Students are required to do some compulsory homework on specific topics and some optional home projects in applying quantitative methods to finance problems.
Oral exams: Students who achieved an average pass grade in written (two partial or one final) tests can take an oral exam made up of: questions about mistakes in written tests and on one topic on their choice.
<table>
<tr><th>Formative and Summative Assessment towards the definition of a final grade</th><th>% weight on the final grade</th></tr>
<tr><td>In class participation</td><td>5</td></tr>
<tr><td>Summary of previous week lessons</td><td>10</td></tr>
<tr><td>Short seminars</td><td>5</td></tr>
<tr><td>In Class written tests</td><td>70</td></tr>
<tr><td>Assigned Homework And Project</td><td>5</td></tr>
<tr><td>Oral Exam</td><td>5</td></tr>
</table></td></tr>
</table>

Programme of "MODELLI E ALGORITMI PER LA FINANZA AZIENDALE II"
"MODELS AND ALGORITHMS FOR FINANCIAL MANAGEMENT II"

F0993, OPTIONAL
Second Cycle Degree in COMPUTER SCIENCE, 2nd Year, 2nd Semester

Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Giuseppe ALESII		
1	Course objectives	This module deals with programming applied to derivatives pricing. The main objective of the course is to introduce students to programming – on Spreadsheets and matrix languages such as Aptech-Gausstm or MatLab – applied to models and algorithms most frequently used to price a specific class of derivative contracts, namely options.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Introduction to derivatives: options contracts description payoff/position diagrams - A hands on introduction to the stochastic processes most used to price Derivatives: (a) modeling time series with additive shock or multiplicative shocks, MA(1) and AR(1) representation of a time series; the Wiener process as a limit case of a time series random walk; Ito process as a generalization of the Wiener process; (b) Geometric Brownian Motion: i. Univariate GBM: derivation through Ito's Lemma application, PDE Monte Carlo simulation, parameters estimation, binomial discrete approximation Cox, Ross, Rubinstein 1979, Brownian Bridge Monte Carlo simulation; ii. Multivariate GBMs with correlated Wiener Processes: Monte Carlo simulation, parametric Monte Carlo on correlation coefficients estimates; binomial discrete approximation Boyle, Evnine, Gibbs 1989. (c) Ornstein Uhlenbeck (OU) Process: arithmetic and Schwartz 1997, model # 1 geometric version, derivation through Ito's Lemma application, PDE Monte Carlo simulation, parameters estimation, binomial discrete approximation Sick 1995, (d) Inverting Black, Scholes 1973 model to derive a volatility surface; (e) Volatility modeling: equally weighted estimates, ARCH(m), exponentially weighted moving average, (f) GARCH(1,1): clustering and leverage problems, Maximum Likelihood Estimation, volatility term structure, average volatility, Options Pricing with GARCH() volatility: local risk neutrality, Duan 1995; (g) Variance Covariance matrix modeling: VEC GARCH, BEKK GARCH - Derivatives Martingale Pricing: (a) valuation of option contracts with American optionalities: Cox, Ross, Rubinstein 1979, Sick 1995, Boyle, Evnine, Gibbs 1989 (b) Valuation of option contracts with European optionalities, Black e Scholes 1973, Stultz 1982, Johnson 1987, Monte Carlo simulations, (c) Least Squares Monte Carlo , Longstaff, Schwartz 2001 RFS. - Real Options: parallelism with decision tree analysis, multi-period securities markets a' la Harrison e Kreps 1979, real and financial options, a short taxonomy of real options, Mickey mouse examples, asset substitution / underinvestment moral hazard, Kulatilaka Trigeorgis general real options pricing model. On successful completion of this module, the student will: o have a thorough and deep knowledge of derivatives pricing models. In particular, he/she must be able to model any payoff of a simple and multiple underlying derivative. He/she must be able to provide a discrete time approximation of stochastic processes dealt with at lesson both in view of a lattice approximation and a Monte Carlo simulation. Finally, the student should be able to price any financial or real option within the stochastic processes and pricing algorithms provided in the course. In any case, she/he must be knowledgeable with the general themes of martingale pricing. o be able to use her/his programming skills in simple Excel spreadsheets and/or in high programming languages such as Gauss or MatLab, not only for financial models and algorithms dealt with at lesson but also for other similar problems. o have acquired general skills in the field of algorithms and applied programming for option pricing which enable him/her to make educated choices in a problem solving practice framework. To be specific, the student should be able set up Excel spreadsheets and/or high level language, GAUSS or MatLab, codes implementing univariate and multivariate lattice models, Monte Carlo Simulations as discrete time approximations of the stochastic processes deal with at lesson. Moreover, the student should be able to program univariate and multivariate underlying European Options Derivatives closed formulas. Finally, the student should be able to program backward induction algorithms both on a lattice and in a Monte Carlo simulation framework, Least Squares Monte Carlo, choosing the algorithm which suits best to the application (algorithm educated choice) The student should be able to apply methods mentioned above both to financial derivatives and to real options, Bellman's dynamic programming in a stochastic framework, impulse control.

<table>
<tr><td></td><td></td><td>o be capable to give a presentation both in front of a general practitioners' audience and a more academic one about the models dealt within the course.
o have acquired a method of study both thanks to a wide knowledge of the main streams in which financial modeling is evolving, theoretical continued learning, and a confident practice with respect to the main high level programming languages, GAUSS and MatLab, which are continually evolving, best practice continued learning.</td></tr>
<tr><td>3</td><td>Prerequisites and learning activities</td><td>Pre-Assessment The first module of Models and Algorithms for Financial Management or an equivalent course is a necessary prerequisite.
Formative Assessment is performed during the whole teaching period through
- class participation during lessons: students may be asked to answer questions about topics dealt with at lesson; students may ask instructor questions during lessons both about the very topic dealt with at lesson and about correlated topics they are particularly interested in;
-summary of previous week lessons: a student is randomly selected to sum up topics dealt with in the previous sessions, actually introducing extant session;
-short seminars: students are required to apply their skills in Calculus, Stochastic Calculus, Numerical Analysis and Mathematical Statistics to specific problems in finance, proposing their own solutions previously prepared as assigned homework.</td></tr>
<tr><td>4</td><td>Teaching methods and language</td><td>Lectures, Exercises.
Language: Italian, English
Ref. Text books
Thomas E. Copeland, J. Fred Weston, and Kuldeep Shastri, Financial Theory and Corporate Policy (4th Edition). Addison-Wesley. 2005.
Luenberger, D., Investment Science. Oxford University Press. 1998.
John C. Hull, Options, Futures and Other Derivatives (7th edition). Prentice Hall. 2008.</td></tr>
<tr><td>5</td><td>Assessment methods and criteria</td><td>Summative Assessment consists of written tests, homework and oral exam.
Written test: Students can decide to take two (mid-term and final) written tests or a comprehensive final test, that can be taken also in case of failing the two tests.
<table>
<tr><th>Learning Outcome Assessed</th><th>Testing method adopted</th></tr>
<tr><td>Baseline theoretical knowledge provided through lessons and suggested reading list</td><td>Open questions to be answered through short essays</td></tr>
<tr><td>Problem solving involving symbolic calculus and stochastic calculus capabilities</td><td>Questions about model building and algorithms tuning for specific formal problems</td></tr>
<tr><td>Programming capabilities</td><td>Small problems in class assignments to be programmed in a high level language, e.g. MatLab, Gauss, Ox, Scilab</td></tr>
</table>
Homework and projects: Students are required to do some compulsory homework on specific topics and some optional home projects in applying quantitative methods to finance problems.
Oral exams: Students who achieved an average pass grade in written (two partial or one final) tests can take an oral exam made up of: questions about mistakes in written tests and on one topic on their choice.
<table>
<tr><th>Formative and Summative Assessment towards the definition of a final grade</th><th>% weight on the final grade</th></tr>
<tr><td>In class participation</td><td>5</td></tr>
<tr><td>Summary of previous week lessons</td><td>10</td></tr>
<tr><td>Short seminars</td><td>5</td></tr>
<tr><td>In Class written tests</td><td>70</td></tr>
<tr><td>Assigned Homework And Project</td><td>5</td></tr>
<tr><td>Oral Exam</td><td>5</td></tr>
</table></td></tr>
</table>

Programme of "RETI NEURALI"
"NEURAL NETWORKS"

F0164, OPTIONAL
First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 1st Semester

Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)

Teacher: Pasquale CAIANIELLO

<table>
<tr><td>1</td><td>Course objectives</td><td>This Module is an introduction of the fundamentals in Neural Nets and Machine Learning. By studying well established neural network models, structures, and training techniques the student will develop a skill in approaching problems of identification, recognition, classification, clustering, and learning.</td></tr>
<tr><td>2</td><td>Course content and Learning outcomes (Dublin descriptors)</td><td>Topics of the module include:
- Outline of neural physiology
- Perceptron and linearly separable sets
- Multilayer perceptron and error backpropagation
- Supervised learning, recognition, and multicategorical classification
- Unsupervised learning, temporal sequences, topological maps, clustering
- Recurrent nets, Sensor information analysis

On successful completion of this module, the student will be able to:
o understand the functional Sample-Train-Test protocol;
o explain and contrast the most common adapting and learning architectures that address complex sensorial perception analysis and classification tasks;
o explain the difference between supervised and unsupervised learning;
o identify and acquire the training set for complex recognition and mapping tasks, from online repositories, or by in house expert observation sampling.
o implement hybrid ANN systems for sensor signal processing, optimization, classification, and modeling;
o understand the technical potential, the advantage and limitation of state of the art learning, adaptive, and self organizing systems;
o apply the methods and produce applications in their working life;
o describe the relation between real brains and simple artificial neural network models;
o discuss the main factors involved in achieving good learning and generalization performance in neural network systems;
o identify the main implementational issues for neural networks and sensor analysis;
o give a presentation to professional and academic audience about the evaluation a real implemented model, its functioning and its testing;</td></tr>
<tr><td>3</td><td>Prerequisites and learning activities</td><td>Knowledge of basic concepts of linear algebra and discrete mathematics. Ability to develop an experimental or implementation project with open source resources.</td></tr>
<tr><td>4</td><td>Teaching methods and language</td><td>Lectures
Language: Italian, English
Ref. Texbooks
- Online resources
-Teacher’s Notes</td></tr>
<tr><td>5</td><td>Assessment methods and criteria</td><td>Summative Assessment consists of one written test, homework, class presentations, and final written technical report.
Written test: Student may take a midterm exam or a final exam
Homework: a minimum of three assignments suggested in class
Final presentation: a pdf

<table>
<tr><th>Learning Outcome Assessed</th><th>Testing method adopted</th></tr>
<tr><td>Baseline theoretical and pragmatic knowledge provided through lessons and guided online exploration</td><td>Multiple Choice tests</td></tr>
<tr><td>Programming capabilities</td><td>Homework reports throughout the course</td></tr>
<tr><td>Project design for complex tasks</td><td>Final Presentation
Implementation Test</td></tr>
</table>

<table>
<tr><th>Formative and Summative Assessment towards the definition of a final grade</th><th>% weight on the final grade</th></tr>
<tr><td>In class participation</td><td>10</td></tr>
<tr><td>Homework</td><td>30</td></tr>
<tr><td>Final presentation</td><td>10</td></tr>
<tr><td>In class written Tests</td><td>30</td></tr>
<tr><td>Project Evaluation</td><td>20</td></tr>
</table>
</td></tr>
</table>

Programme of "RETI DI CALCOLATORI EVOLUTE: INTERNETWORKING"
"ADVANCED COMPUTER NETWORKS: INTERNETWORKING"

F0160, OPTIONAL
First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 2nd Semester

Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)

Teacher: Dajana CASSIOLI

1	Course objectives	This module invites students to explore the networking, routing, transport and application protocols that are used in the Internet. The module encourages students to understand the key architectural issues in the design, development and implementation of Internet protocols using lectures, seminars and tutorials. Protocol specifications and standards such as IPv6 as necessary evolution of IPv4, IntServ and DiffServ to manage the quality of service (QoS), IPsec to ensure security, etc. will be examined and the use of techniques such as MPLS to improve the performance of forwarding, and protocols such as Mobile IP to manage mobility, and RTP/RSTP for Video transmission will also be investigated. Finally, in the light of the knowledge gained on internetworking protocols, we present some IP-based applications, such as VoIP and 6LoWPAN.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - The routing IP in Internet - IP version 6 - IP and mobility: Mobile IP - The DNS (Domain Name System) - The DHCP (Dynamic Host Configuration Protocol) - Quality of Service in InternetIP switching and Multi-Protocol Label Switching (MPLS) - Security in Internet - IP network management and Simple Network Management Protocol (SNMP) - Some IP-based applications: - Voice over IP (VoIP) - Internet Protocol Television - (IPTV) - Sensor Networks (IPv6 -6lowPAN) On successful completion of this module, the student should: o demonstrate the knowledge of the essential features and operation of Internet Protocols; o understand the principle and operation of a set of protocols in the TCP/IP suite Protocols; o be able to analyze a range of network protocols; o be able to design a protocol with a finite state structure to meet pre-specified requirements; o be able to analyze and evaluate performance of complex networks; o be able to implement or analyze new networking applications; o be able to explain the use of network monitoring and performance tools.
3	Prerequisites and learning activities	It is strongly recommended to have passed the exam of Computer Networks in order to have a knowledge of the problems of networking and internetworking.
4	Teaching methods and language	The course consists of a main part and a series of seminars on the following topics: - IP Routing in the Internet - Security on the Internet Language: English Ref. Text books -M. Baldi, P. Nicoletti, *Internetworking.* McGraw-Hill Milano. 1999. -Fred Halsall, *Networking e Internet*, 5ed.. Pearson Education – Addison Wesley.2006. -W. Stallings, *Sicurezza delle reti*–Applicazioni e standard, 3ed Pearson–Prentice Hall, 2007
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website. Formative Assessment The formative assessment is performed via interaction between teacher and students during lectures. Students are involved in questioning and discussions, by means of open oral questions and written exercises to the entire class done periodically during the Course duration. Summative Assessment consists of a written test, lasting two hours and consisting in six questions, which could be either numerical exercises or open-ended questions, concerning the whole course content. An optional mid-term written test will be also provided, which is meant to cover the first part of

		the course, in order to help the students to split the workload. The mid-term written exam (lasting one hour and a half) consists of three questions, either exercises or open questions, concerning the first part of the course content. If a student passes the mid-term written exam, s/he will take a final-term written exam concerned with the second part of the course content only, lasting one hour and half and consisting again of three questions, either exercises or open questions. In this examination's mode, the result of each written test is obtained by adding up the result achieved on each of the three questions. The final result of the written exam will be given by the sum of results of the two parts, i.e. of the total six questions. In the other mode, the result of the written exam is given by the sum of the results achieved on each of the six questions. Criteria of evaluation will be the level of knowledge and the fluency in the technical language of network architectures and protocols.

Programme of "PROGRAMMAZIONE A OGGETTI" ***"OBJECT ORIENTED SOFTWARE DESIGN"***		
DT0043, OPTIONAL **First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 2nd Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Romina ERAMO		
1	Course objectives	The focus of this course is on achieving advanced knowledge of the Object Oriented Programming paradigm and experience with the Java language. This course introduces students to the Object Oriented design and development of software applications. It introduces students to advanced features of Java programming language. Students will learn how to use inheritance, interfaces, exception handling, file input and output, and generic types, how to incorporate graphical user interfaces into their programming applications, how to use design patterns.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - The Object Oriented paradigm - Introduction to the Object Oriented software engineering - Introduction to the language JAVA - Java: Types, variables, expressions. - Java: Classes, objects, inheritance, scoping; Interfaces and exceptions. - Advanced aspects of JAVA: execution, documentation, threads, AWT, JDBC. - Design Patterns On successful completion of this module, the student should: o understand the basic principles of the object-oriented programming; o demonstrate an introductory understanding of graphical user interfaces, multi-threaded programming, and event-driven programming; o show competence in the use of the Java programming language in the development of small to medium-sized application programs that demonstrate professionally acceptable coding and performance standard; o be able to design, write, and test documented Java programs and servlets; o be able to use advanced graphic functions; o be able to create applications with database connectivity along with client server architecture; o be able to apply object-oriented design and programming principles to their programs; o be able to develop programs using fundamental concepts of structured programming; o be able to use software development methodology in program problem solving; o be able to code programs using data types, control structures, functions and arrays. o demonstrate the ability to run, test, and debug programs.
3	Prerequisites and learning activities	Basic notions of imperative and object-oriented programming.

4	Teaching methods and language	The course consists of Lectures and Seminars. Language: Italian, English Ref. Text books -Bruce Eckel, *Thinking in Java 4 ed. - Concorrenza e interfacce grafiche*. Pearson (vol.3) -Bruce Eckel, Thinking in Java 4 ed. - I fondamenti. Pearson. (vol. 1) -Bruce Eckel, Thinking in Java 4 ed. - Tecniche avanzate. Pearson. (vol. 2) -Gamma, Helm, Johnson, Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, *Addison-Wesley* -Bernd Bruegge, Allen H. Dutoit ,*Object-Oriented Software Engineering Using UML, Patterns, and Java, Prentice Hall*
5	Assessment methods and criteria	Pre-Assessment: Course pre-requisites are clearly stated on the Module website. Formative Assessment: The formative assessment is performed via interaction between teacher and students during lectures. Students are involved in questioning and discussion, by means of open oral questions to the entire class. Summative Assessment: Written test, lasting two hours and consisting in programming exercises and open-ended questions, concerning the whole course content. The oral exam can be required either by the student, to improve grades, or by the teacher, in presence of significant mistakes / misunderstandings in the written exam. The written test is aimed at: verification of theoretical competence, and in particular of knowledge and comprehension of Course contents, verification of skills in understanding and solving significant exercises, and in explaining the proposed solutions. The final result of the written exam will be given by the sum of results of the questions and corresponds to 1/2 of the total marks. Project consisting in the design and the development of a Java application. It is followed by a project discussion. The project and its presentation are aimed at: verification of skills in understanding and applying methodologies and techniques learned during the Course, Criteria of evaluation will be: the level of knowledge and practical ability; the property of analyze the system requirements and provide an object-oriented design and development of the application; the use of the technical language; the clarity and completeness of explanations. The project and its related documentation correspond to 1/2 of the total marks.

Programme of "TEORIA DELL'INFORMAZIONE" ***"INFORMATION THEORY"***		
F0158, OPTIONAL **First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 1st Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Filippo MIGNOSI		
1	Course objectives	This Module provides the students with: Knowledge of the basic concepts of Information Theory and ability to manipulate them formally, Deep understanding of common-sense concepts like "information", "representation", "model" and Ability to translate intuitive solutions constructed with such concepts into concrete applications in different technological areas (in particular, the methods of data compression), Acquisition of tools for reading the basic aspects of the literature of the discipline.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Information and entropy: joint, conditional, and mutual. - Relative measures, AEP and its consequences. - Representation and codes: uniform, variable length, and adaptive. - Kraft-McMillan Inequality and its consequences. - Souce Coding and Compression algorithms (Shannon, Arithmetic, Huffmann, Ziv and Lempel coding, integer encoding, gamma and omega Elias coding) - Basic concepts of channel coding, channel capacity. - Basic concepts of modern Cryptography. On successful completion of this module, the student should be able to: o describe the general principles of information theory; o understand and explain fundamental concepts such as entropy, mutual information, capacity, compression, coding theorem, coding theory, coding and Cryptography; o understand and apply fundamental concepts in information theory such as probability, entropy, information content and their inter-relationships, AEP, data compression; o compute entropy and mutual information of random variables;

		o implement and analyze basic coding and compression algorithms; o formulate and prove The main theorems treated such as: i) AEP and its consequences, ii) Optimality of Huffmann coding, and of arithmetical coding, iii) the entropy is a lower bound for the expected length of a u.d block code, iv) if P is different from NP then there exists no perfectly secret encryption scheme with key shorter than the message. o explain how information theory and coding contributes to modern communications technology; o be able to describe the main results of information theory to other non-specialist people in the scientific community. o be able to read and understand books and papers concerning the arguments treated in the course. o solve advanced problems in the area.
3	Prerequisites and learning activities	Basic probability and discrete mathematics. Ability to develop software applications.
4	Teaching methods and language	The course consists of Lectures and Seminars. Language: Italian and English Ref. Text books: -T.M. Cover and J.A. Thomas, *Elements of Information Theory*. John Wiley & Sons 2006. - S. Arora and B. Barak, Chapter 9 of *Computational Complexity: A Modern Approach*, Cambridge University press 2009.
5	Assessment methods and criteria	Written and oral examination

Programme of "APPLICAZIONI PER DISPOSITIVI MOBILI" ***"MOBILE APPLICATIONS DEVELOPMENT"***		
F1081, OPTIONAL **First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 2nd Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Ivano MALAVOLTA		
1	Course objectives	This Module provides the students with knowledge and understanding of the mobile applications development problem space, how to effectively design a business-ready mobile app, and how to correctly implement it. The module topics cover the mobile ecosystem, platforms and strategies, mobile information architecture and UI Design, Mobile app distribution and monetization , Web technologies for mobile app development (HTML5, CSS3, jQuery, and other frameworks), Data management (local data storage, REST APIs, server-side data storage), Geolocalization and mapping, Accessing the device capabilities (camera, accelerometer, contacts, messaging, etc.), Security and user authentication and Mobile app Debugging.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Understanding of the mobile ecosystem, platforms and strategies - Mobile information architecture and UI Design - Mobile app distribution and monetization - Web technologies for mobile app development (HTML5, CSS3, JQuery, and other microframeworks) - Data management (local data storage, REST APIs, server-side data storage) - Geolocalization and mapping - Accessing the device capabilities (camera, accelerometer, contacts, messaging, device rotation) - Debugging Mobile apps - Security and user authentication - Intro to mobile app testing On successful completion of this module, the student should be able to: o understand the technical challenges posed by current mobile devices and wireless communications; o understand the unique aspects of mobile user interfaces and context awareness, and their use in achieving quality user experiences; o apply user experience and user interface design principles;

		o design the information architecture and navigation infrastructure of a mobile app; o implement and deploy cross-platform mobile apps; o evaluate, select and use appropriate tools in the development and marketing of mobile apps; o develop HTML and JavaScript code for mobile devices; o work collaboratively in a small apps development team and discuss user interface design considerations; o appreciate the need to keep up with rapid changes and new developments and identify current trends in mobile communications technologies and systems; o select and evaluate suitable software tools and APIs for the development of mobile apps and understand their strengths, scope and limitations; o use an appropriate application development to design, write and test small interactive programs for mobile devices.
3	Prerequisites and learning activities	The course does not impose any specific prerequisites on programming or design . The only requirement is a basic knowledge of JavaScript and W3C web standards like HTML5 and CSS3.
4	Teaching methods and language	The course consists of Lectures and Seminars. Language: English Ref. Text books: -Wesley Hales, *HTML5 and Javascript Web Apps*. O'Reilly Media. (vol. 1) 2012. http://www.amazon.it/dp/1449320511 -Brian Fling, *Mobile Design and Development.* O'Reilly Media. (vol. 1) 2009. http://www.amazon.it/dp/0596155441
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website. The only course pre-requisite is a basic knowledge of JavaScript and W3C web standards like HTML5 and CSS. Formative Assessment The formative assessment is performed via interactive interaction between teacher and students during lectures. Students are aware since the beginning of the Course that they will be involved (in turns) in questioning and discussion, by means of open oral questions to the class or to single students. Summative Assessment During the course students organize themselves into teams, and each team must work on a given project. The project is divided into three main parts: 1. Elevator Pitch: project teams have to prepare a 5-slide presentation describing the overall idea that they want to develop, why it is new, and why it is relevant. For more information about what an elevator pitch is: http://bit.ly/xqzzgR, http://buswk.co/xxDcXo 2. Design: project teams have to prepare a document describing in details the app they want to develop, its potential users, usage scenarios, the context in which it will be used, its information architecture, UI design with wireframes. 3. Implementation: project teams have to submit the source code of the implemented mobile app, together with a brief technical documentation (at most 10 pages). Clearly, the source code of the app must run on any kind of simulator (depending on the chosen development platform). As a final step, each project team discusses with the instructor the various implementation choices made while realizing the app, and an extension of the app will be performed in a 2-hours interval. The developed app must be shown while running on a real mobile device that the team must bring for the discussion. For each type of assignment, a suitable template is provided during the course.

Programme of "BIOINFORMATICA" ***"BIOINFORMATICS"***		
DS9001, OPTIONAL **First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 2nd Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Antinisca DI MARCO		
1	Course objectives	The course introduces the bioinformatics by identifying the principal problems and solutions the algorithms can deal with and provide, respectively. Moreover, it provides an overview of the main on-line Data bases on biology data and presents their structures and the services

		they provide to end users. Finally, an introduction to system biology is given, focusing on computer science formal tools (such as, petri nets) that can be used to model biology phenomena or systems.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Biological databases of genes, polimorfisms e mutations. - Biological databases of metabolic pathways. - Alignment algorithms and substitution matrices. - Pattern search, phylogenetic trees. Genome wide character association. - Computational models for System Biology. - BioPhyton. On successful completion of this module, the student should be able to: o demonstrate detailed knowledge of: i) alignment algorithms and of the phylogenetic analysis; ii) the functioning and regulation of prokaryotic and eukaryotic cells; iii) Phyton language and BioPhyton libraries; iv) computational modeling for simple biological systems represented as pathways; o use and organize databases of genomics, proteomics and metabolomics data; o apply phylogenetic analysis to simple genomics data: Use BioPhyton libraries for bioinformatics aims, Use of Petri Nets to model biological pathways. o evaluate and interpret current literature in areas of bioinformatic practice for analysis of DNA sequence; o obtain quantitative results from computational methods and interpret quantitative results from computational methods; o report on the bioinformatics experiments and studies conducted on DNA sequences and pathways and discuss the theoretical basics of DNA sequence analysis; o demonstrate capacity to select programs for problem solving and write programs.
3	Prerequisites and learning activities	Basic knowledge of imperative and object-oriented programming techniques.
4	Teaching methods and language	The course will be composed by around 40 hours of theory and 20 hours of (computer science!) laboratory during which the theoretical concepts are showed on concrete examples. Language: English Ref. Text books: -Volker Sperschneider, Bioinformatics.Problem Solving Paradigms . Springer. *(items 3,4 of syllabus)* -selected scientific publications. -Teacher Notes. -Tutorial of on-line data base.
5	Assessment methods and criteria	Pre-Assessment There is no formal pre-assessment, but Course pre-requisites are clearly stated on the Module website. Fulfilment of such pre-requisites is verified by formative assessment. Additional lectures or short seminars or individual homework are provided by the teacher in case significant problems are detected. Formative Assessment The formative assessment is performed via interactive interaction between teacher and students during lectures. Students are aware since the beginning of the Course that they will be involved (in turns) in questioning and discussion, by means of open oral questions to the class or to single students. Summative Assessment: Project followed by a project discussion. A mid-term assay is also be provided, which is meant to cover the first part of the course, in order to help the students to split the workload. The assay is aimed at verification of theoretical competences, and in particular of deep knowledge and comprehension of a specific Course content. The student will write an assay of max 10 pages. The project and its presentation are aimed at: verification of skills in understanding and solving significant exercises, and in explaining the proposed solutions. This in order to verify the ability of application of techniques learned during the Course, of analysis of problems and design of suitable solutions, and of evaluation of alternative solutions. Criteria of evaluation will be: the level of knowledge and practical ability; the property of use of the technical language; the clarity and completeness of explanations. The oral exam will occur within one week of the written test and will typically focus on a technical presentation of the solutions and decisions taken in the project. Additional questions on theoretical aspects can

		be raised by the teacher during the presentation. The oral exam is mandatory. Assessment breakdown: 100% mid-term plus end-of-semester summative assessment. The assay (max 10 pages) covers the 1/4 of the total mark The project and its related documentation (corresponding to 2/4 of the total marks) has to be sent by email to the teacher before the exam date. Criteria of its evaluation will be the correct usage of theoretical topics in the projects and the quality of the presentation of the related documentation. The oral exam (corresponding to 1/4 of the total marks) focuses on the presentation of the project by using slides. Criteria of its evaluation will be the ability in proposing and defending the solutions devised in the project.

Programme of "SISTEMI DI INFORMAZIONE E SICUREZZA DI RETI" ***"INFORMATION SYSTEMS AND NETWORK SECURITY"***		
DS9003, OPTIONAL **First Cycle Degree in COMPUTER SCIENCE, 3rd Year, 2nd Semester**		
Number of ECTS credits: 6 (workload is 150 hours; 1 credit = 25 hours)		
Teacher: Gianpiero MONACO		
1	Course objectives	It has become nearly impossible to live in today's society and to be not dependent on information systems of one type or another. The goal for students in this course is to learn the fundamentals of Information Systems and Security. The main objective is to provide students with an overall understanding of the main concepts of information systems, to highlight their increasing importance in modern organizations and societies and to emphasise the ethical, social and security implications.
2	Course content and Learning outcomes (Dublin descriptors)	Topics of the module include: - Introduction to information systems. - Information Technology infrastructure and Support: business issues, challenges and information technology solutions, Cloud computing, Business transactions in real time. - Data and network infrastructure: network management. - Information Systems Security: - Information technology security, crime, Compliance, Continuity. - Threats, Vulnerabilities, and Risk Exposure. - Defense: Information technology Defense. Basics of cryptology - Social impacts of technology. - Introduction to cloud computing. - Algorithmic issues arising in information systems. On successful completion of this module, the student should: o Acquire knowledge on: Information systems; Information technologies that support organizations; Networks and Data management; Information Systems Security; Cryptology; Cloud computing; Algorithmic issues arising in information systems. o Be able to solve general issue arising in information systems at the enterprise level. o Be able to develop secure computer networks and information systems. o Be autonomous in solving general issue arising in information systems and network security. o Be able to assess the level of security of an Information systems and computer network. o Be able to understand issues arising in information systems and network security at the enterprise level. o Be able to improve its knowledge in the field of information systems and network security in his future studies and/or works.
3	Prerequisites and learning activities	Fundamentals of algorithms, data structures, and discrete mathematics. General knowledge of Operating Systems, Computer Networks and Database Systems.
4	Teaching methods and language	Lectures and Exercises. Language: English Ref. Text books: -Efraim Turban, Linda Volonino., *Information Technology for Management*, 8th Edition. Wiley, 2011. Moreover extra didactic material is provided by the teacher (lectures slides and papers).
5	Assessment methods and criteria	Pre-Assessment: Course pre-requisites are clearly stated on the Module website. Formative Assessment: The formative assessment is performed via interaction between teacher and students during lectures. Students are involved in questioning and discussion, by means of open oral questions to the entire class.

		Summative Assessment: Written test followed by an optional oral exam. The oral exam can be required either by the student, to improve grades, or by the teacher, in presence of significant mistakes / misunderstandings in the written exam. An optional mid-term written test is also be provided, which is meant to cover the first part of the course, in order to help the students to split the workload. The written test is aimed at: (1) verification of theoretical competence, and in particular of knowledge and comprehension of Course contents (2) verification of skills in understanding and solving significant exercises, and in explaining the proposed solutions. This in order to verify the ability of application of techniques learnt during the Course, of analysis of problems and synthesis of suitable solutions, and of evaluation of alternative solutions. Criteria of evaluation will be: the level of knowledge and practical ability; the property of use of the technical/mathematical language; the clarity and completeness of explanations. The written test (about 2 hours) consists in: (i) Multiple-choice questions, to cover point (1), 15-20% of total marks; (ii) Short essays to cover point (1), 35-40% of total marks; (iii) Exercises, to cover point (2), 40-50% of total marks. All parts can result in negative marks if the answer is omitted or seriously flawed. The oral exam (max 1 hour) will occur within the same exam session of the written test and will typically cover the areas of the written answers that need clarification plus, possibly, additional subjects proposed by the teacher.