Testi del Syllabus

Resp. Did.	**TOMAIUOLO Michele**	Matricola:	**006085**

Anno offerta:	**2015/2016**
Insegnamento:	**1002526 - FONDAMENTI DI INFORMATICA + LABORATORIO DI PROGRAMMAZIONE**
Corso di studio:	**3050 - INGEGNERIA INFORMATICA, ELETTRONICA E DELLE TELECOMUNICAZIONI**
Anno regolamento:	**2015**
CFU:	**9**
Anno corso:	**1**
Periodo:	**Primo Semestre**
Sede:	**PARMA**

Testi in italiano

Tipo testo	*Testo*
Lingua insegnamento	Italiano
Contenuti	1. Introduzione alla programmazione 1.1. Programmazione strutturata 1.2. Collezioni e flussi di dati 1.3. Funzioni e ricorsione 1.4. Oggetti e astrazioni 1.5. Interfacce grafiche 2. Introduzione all'informatica 2.1. Rappresentazione dei dati 2.2. Concetto di computazione 2.3. Sistemi di elaborazione 2.4. Sviluppo del software
Testi di riferimento	* P. Wentworth, J. Elkner, A.B. Downey, C. Meyers. How to Think Like a Computer Scientist. http://openbookproject.net/thinkcs/ * B. Stroustrup. Programming: Principles and Practice Using C++. Addison-Wesley (2009). 978-0321543721 * B. Eckel. Thinking in C++. Prentice-Hall (2000) * D. Mandrioli, S. Ceri et al. Informatica arte e mestiere. McGraw-Hill (2008)
Obiettivi formativi	L’obiettivo del corso è fornire allo studente la capacità di comprendere i principi dell'informatica e della programmazione: * Rappresentazione dei dati * Computazione, linguaggi e macchine astratte * Architettura dei sistemi di elaborazione * Paradigma di programmazione ad oggetti * Introduzione all'ingegneria del software Le capacità di applicare le conoscenze elencate riguardano lo sviluppo del cosiddetto "pensiero computazionale": * Scomposizione di problemi complessi * Soluzione di problemi tramite ricorsione * Composizione di oggetti in sistemi complessi * Modellazione con diversi livelli di astrazione

Tipo testo	*Testo*
Prerequisiti	Nessuna propedeuticità. Si suppone comunque che lo studente conosca le basi dell'uso del computer e di Internet - l'equivalente dei moduli 1, 2, 3 e 7 del Syllabus ECDL (European Computer Driving Licence).
Metodi didattici	Lezioni in aula, con l'ausilio di slide rese disponibili in anticipo agli studenti. Soluzione guidata di esercizi in aula. Esercizi di programmazione in laboratorio. Le esercitazioni in laboratorio sono centrali per il corso. Gli esercizi proposti vertono sugli stessi argomenti generali delle lezioni in aula. L'obiettivo è di introdurre i principi della programmazione orientata agli oggetti, guidando lo studente alla soluzione di problemi con un livello di complessità crescente.
Altre informazioni	Testi alternativi * J.C. Lusth. The Art and Craft of Programming. http://beastie.cs.ua.edu/cs150/book/ * A. Koenig, B.E. Moo. Accelerated C++: Practical Programming by Example. Addison-Wesley (2000) * E. Clementini. Fondamenti di Informatica – Programmazione strutturata in C++. Carocci (2006) * H. Schildt. C++: A Beginner's Guide. McGraw-Hill (2003) * S. Prata. C++ Primer Plus. Addison-Wesley (2011) * L.J. Aguilar. Fondamenti di programmazione in C++. McGraw-Hill (2008) * B. Eckel. Thinking in C++, vol. 2. Prentice-Hall (2003). 978-0131225527 * M. Dawson. Beginning C++ Through Game Programming. Course Technology (3rd ed., 2010). 978-1435457423 * M. Dawson. Python Programming for the Absolute Beginner. Course Technology (3rd ed., 2010). 978-1435455009 * J. Blanchette, M. Summerfield. C++ GUI Programming with Qt 4. Prentice Hall (2nd ed., 2008). 978-0132354165 * A. Lorenzi, V. Moriggia. Programmazione ad oggetti e linguaggio C++. Atlas (2004). 978-8826811956. (Testo per le scuole medie superiori) * J.G. Brookshear. Informatica. Una panoramica generale. Pearson (2012) * U. Avalle, L. Console, M. Ribaudo. Introduzione all'informatica. UTET (2010)
Modalità di verifica dell'apprendimento	L'esame consiste di una prova sui fondamenti dell'informatica (brevi esercizi e quiz, a cui rispondere in circa mezz'ora), una prova di programmazione (un programma ad oggetti, da sviluppare in laboratorio in circa 3 ore) e un colloquio. La partecipazione continua e particolarmente proficua alle esercitazioni potrebbe esonerare dalla prova di programmazione finale.
Programma esteso	1. Introduzione alla programmazione (24 ore, in aula + 24 ore, in lab) 1.1. Programmazione strutturata 1.1.1. Struttura di un programma 1.1.2. Variabili e tipi, espressioni 1.1.3. Condizioni 1.1.4. Cicli 1.1.5. Annidamento 1.2. Collezioni e flussi di dati 1.2.1. Vettori 1.2.2. Matrici 1.2.3. Mappe 1.2.4. Operazioni di I/O su console e file 1.3. Funzioni e ricorsione 1.3.1. Passaggio dei parametri 1.3.2. Ambito delle variabili 1.3.3. Stack e record di attivazione 1.3.4. Ricorsione 1.4. Oggetti e astrazione 1.4.1. Incapsulamento 1.4.2. Composizione 1.4.3. Allocazione dinamica 1.4.4. Ereditarietà e polimorfismo

Tipo testo	*Testo*
	1.4.5. Principio di sostituibilità 1.5. Interfacce grafiche 1.5.1. Elementi di base 1.5.2. Disposizione degli elementi 1.5.3. Segnali ed eventi 1.5.4. Animazioni 2. Introduzione all'informatica (24 ore, in aula) 2.1. Rappresentazione dei dati 2.1.1. Numeri in complemento a due ed in virgola mobile, algebra booleana 2.1.2. Rappresentazione del testo, codifica ASCII ed Unicode 2.1.3. Documenti strutturati ed HTML 2.1.4. Rappresentazione di immagini e suoni 2.2. Concetto di computazione 2.2.1. Automi a stati finiti 2.2.2. Espressioni regolari 2.2.3. Macchina di Touring, architettura di von Neumann 2.2.4. Linguaggi e paradigmi di programmazione 2.2.5. Complessità computazionale, algoritmi di ricerca e di ordinamento 2.3. Sistemi di elaborazione 2.3.1. Architettura dei calcolatori 2.3.2. Sistemi operativi 2.3.3. Sistemi informativi e basi di dati 2.3.4. Reti di calcolatori e World Wide Web 2.4. Sviluppo del software 2.4.1. Metodologie di sviluppo e qualità del software 2.4.2. Contratti 2.4.3. Collaudo 2.4.4. Gestione delle versioni

Testi in inglese

Tipo testo	*Testo*
Lingua insegnamento	Italian
Contenuti	1. Introduction to programming 1.1. Structured programming 1.2. Data collections and flows 1.3. Functions and recursion 1.4. Objects and abstractions 1.5. Graphical interfaces 2. Introduction to computer science 2.1. Data representation 2.2. Concept of computation 2.3. Computing systems 2.4. Software development
Testi di riferimento	* P. Wentworth, J. Elkner, A.B. Downey, C. Meyers. How to Think Like a Computer Scientist. http://openbookproject.net/thinkcs/ * B. Stroustrup. Programming: Principles and Practice Using C++. Addison-Wesley (2009). 978-0321543721 * B. Eckel. Thinking in C++. Prentice-Hall (2000) * D. Mandrioli, S. Ceri et al. Informatica arte e mestiere. McGraw-Hill (2008)
Obiettivi formativi	The main goal of the course is to provide students with the ability to understand the principles of informatics and programming: * Data representation * Computation, languages and abstract machines * Architecture of computer systems

Tipo testo	*Testo*
	* Object oriented programming paradigm * Introduction to software engineering The ability to apply the listed knowledge elements regards the development of so-called "computational thinking": * Decomposition of complex problems * Solution of problems through recursion * Composition of objects into complex systems * Modelling with various levels of abstraction
Prerequisiti	None. However, the student is supposed to know the basic computer and Internet operations - the equivalent of modules 1, 2, 3 and 7 of ECDL (European Computer Driving Licence) Syllabus.
Metodi didattici	Lessons in classroom, presenting slides which are provided in advance to students. Guided solution of exercises in classroom. Programming exercises in laboratory. The laboratory exercitations are central for the course. The proposed exercises deal with the same general matters of classroom lessons. Their objective is introducing the principles of object-orientd programming, leading the student to the solution of problems with a growing level of complexity.
Altre informazioni	Alternative textbooks * J.C. Lusth. The Art and Craft of Programming. http://beastie.cs.ua.edu/cs150/book/ * A. Koenig, B.E. Moo. Accelerated C++: Practical Programming by Example. Addison-Wesley (2000) * E. Clementini. Fondamenti di Informatica – Programmazione strutturata in C++. Carocci (2006) * H. Schildt. C++: A Beginner's Guide. McGraw-Hill (2003) * S. Prata. C++ Primer Plus. Addison-Wesley (2011) * L.J. Aguilar. Fondamenti di programmazione in C++. McGraw-Hill (2008) * B. Eckel. Thinking in C++, vol. 2. Prentice-Hall (2003). 978-0131225527 * M. Dawson. Beginning C++ Through Game Programming. Course Technology (3rd ed., 2010). 978-1435457423 * M. Dawson. Python Programming for the Absolute Beginner. Course Technology (3rd ed., 2010). 978-1435455009 * J. Blanchette, M. Summerfield. C++ GUI Programming with Qt 4. Prentice Hall (2nd ed., 2008). 978-0132354165 * A. Lorenzi, V. Moriggia. Programmazione ad oggetti e linguaggio C++. Atlas (2004). 978-8826811956. (Testo per le scuole medie superiori) * J.G. Brookshear. Informatica. Una panoramica generale. Pearson (2012) * U. Avalle, L. Console, M. Ribaudo. Introduzione all'informatica. UTET (2010)
Modalità di verifica dell'apprendimento	The examination consists of a test about the basics of computer science (brief exercizes and quizzes to answer in about half an hour), a programming test (an object-oriented program to develop in lab, in about 3 hours) and a talk. A constant and particularly effective participation to exercitations could exonerate from the final programming test.
Programma esteso	1. Introduction to programming (24 hours, in classroom + 24 hours, in lab) 1.1. Structured programming 1.1.1. Program structure 1.1.2. Variables and data types, expressions 1.1.3. Conditions 1.1.4. Cycles 1.1.5. Nesting 1.2. Data collections and flows 1.2.1. Vectors 1.2.2. Matrices 1.2.3. Maps 1.2.4. I/O operations on console and text files 1.3. Functions and recursion

Tipo testo	*Testo*
	1.3.1. Parameter passing 1.3.2. Scope of variables 1.3.3. Stack and activation records 1.3.4. Recursion 1.4. Objects and abstraction 1.4.1. Encapsulation 1.4.2. Composition 1.4.3. Dynamic allocation 1.4.4. Inheritance and polimorphism 1.4.5. Substitution principle 1.5. Graphical interfaces 1.5.1. Basic elements 1.5.2. Layout of elements 1.5.3. Signals and events 1.5.4. Animations 2. Introduction to computer science (24 hours, in classroom) 2.1. Data representation 2.1.1. Two’s complement and floating point notations, boolean algebra 2.1.2. Text representation, ASCII and Unicode encoding 2.1.3. Structured documents and HTML 2.1.4. Image and sound representation 2.2. Concept of computation 2.2.1. Finite state automata 2.2.2. Regular expressions 2.2.3. Touring machine, von Neuman architecture 2.2.4. Programming languages and paradigms 2.2.5. Computational complexity, searching and sorting algorithms 2.3. Computing systems 2.3.1. Computer architecture 2.3.2. Operating systems 2.3.3. Information systems and databases 2.3.4. Computer networks and World Wide Web 2.4. Software development 2.4.1. Development methodologies and software quality 2.4.2. Contracts 2.4.3. Testing 2.4.4. Versioning