



UNIVERSITÀ  
DEGLI STUDI  
DI PADOVA



**Bollettino Notiziario - A.A. 2021/2022**

## **LAUREA IN INFORMATICA (ORD. 2011)**

### **Curriculum: Corsi comuni**

### **ALGEBRA E MATEMATICA DISCRETA**

**Titolare:** Prof.ssa GEMMA PARMEGGIANI

**Periodo:** I anno, 2 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 58A+40E; 12,00

**Prerequisiti:**

Abilità analitiche (abilità di ragionamento logico); conoscenze e abilità come specificato nel Syllabus della pagina del Corso di laurea in informatica. In particolare: -strutture numeriche (numeri naturali, numeri primi, frazioni numeriche, numeri razionali, elementi dei numeri reali, disuguaglianze, valore assoluto, potenze, radici); -algebra elementare (calcolo letterale, polinomi e operazioni fra polinomi, identità, equazioni di primo e secondo grado, sistemi lineari); -insiemi e funzioni (linguaggi degli insiemi, nozione di funzione, grafici di funzioni notevoli, concetto di condizione sufficiente, necessaria); -geometria (geometria euclidea piana, angoli, radianti, aree e figure simili, nozione di luogo geometrico, proprietà dei triangoli, dei parallelogrammi, dei cerchi, simmetrie, similitudini e trasformazioni nel piano, coordinate cartesiane ed equazioni di semplici luoghi geometrici, elementi di trigonometria, elementi di geometria euclidea nello spazio, volumi).

**Conoscenze e abilità da acquisire:**

Obiettivo del corso è quello di permettere allo studente di sviluppare le proprie capacità analitiche e di acquisire alcune conoscenze di base riguardanti l'algebra, la geometria e la matematica discreta.

**Attività di apprendimento previste e metodologie di insegnamento:**

Lezioni frontali ed esercitazioni.

**Contenuti:**

Massimo comun divisore e algoritmo di Euclide; anelli di classi resto. Richiami sui polinomi: divisione, zeri, fattorizzazione in irriducibili (sui reali e sui complessi). Equazioni lineari e matrici: matrici, operazioni sulle matrici, sistemi di equazioni lineari, metodo di eliminazione di Gauss, sistemi omogenei, matrice inversa. Spazi vettoriali, sottospazi, basi. Funzioni lineari, nucleo e immagine. Autovalori, autovettori e diagonalizzazione di matrici. Prodotti scalari, ortogonalità e procedimento di Gram-Schmidt. Teoria dei grafi: introduzione ai grafi e nozioni di base, connettività, cammini, tagli, alberi, grafi planari, cicli euleriani e circuiti hamiltoniani. Enumerazione: permutazioni e combinazioni semplici, permutazioni e combinazioni con ripetizione, distribuzioni, identità binomiali e triangolo di Pascal, relazioni di ricorrenza.

**Modalità di esame:**

Esame scritto. Lo scritto prevede domande ed esercizi volti a valutare il livello di apprendimento delle nozioni impartite durante il corso e la capacità di elaborarle ed applicarle.

**Criteri di valutazione:**

I criteri per una valutazione positiva sono: - la correttezza e la completezza delle soluzioni degli esercizi - la proprietà del linguaggio matematico utilizzata

**Testi di riferimento:**

Alan Tucker, Applied Combinatorics. : Wiley and Sons, 2007 Marco Abate e Chiara de Fabritiis, Geometria analitica con elementi di algebra lineare. : McGraw-Hill,

**Eventuali indicazioni sui materiali di studio:**

Materiale didattico predisposto dai docenti e reso disponibile sulla piattaforma Moodle.

## ALGORITMI E STRUTTURE DATI

**Titolare:** Prof. PAOLO BALDAN

**Periodo:** Il anno, 1 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 56A+16E; 9,00

**Sede dell'insegnamento:** Plesso Paolotti, Via Luzzatti.

**Aule:** LuM250

**Prerequisiti:**

Elementi di programmazione Matematica discreta Analisi matematica

**Conoscenze e abilità da acquisire:**

Il corso intende fornire un'introduzione agli algoritmi e alla loro analisi. Lo studente apprende alcuni algoritmi e strutture dati fondamentali che sono alla base dello sviluppo dei sistemi software. L'analisi di tali algoritmi aiuta lo studente a sviluppare una sensibilità per la realizzazione di programmi efficienti e corretti.

**Attività di apprendimento previste e metodologie di insegnamento:**

Lezioni ed esercitazioni.

**Contenuti:**

- Fondamenti Analisi dettagliata di InsertSort: Pseudocodice. Calcolo diretto del tempo calcolo in funzione di n. Tasso di crescita e notazione asintotica. L'algoritmo MergeSort e la tecnica divide et impera. Analisi della complessità di MergeSort. Soluzione delle ricorrenze. Il teorema dell'esperto. QuickSort. Complessità media di QuickSort e analisi probabilistica. Randomizzazione di QuickSort. - Ordinamento e Statistiche d'Ordine HeapSort e sua analisi. Limite inferiore per la complessità degli algoritmi di ordinamento. Implementazione di code con priorità mediante heap. Ordinamento in tempo lineare. Algoritmi CountingSort e RadixSort. - Strutture Dati Tavole hash. Alberi binari di ricerca. Alberi rosso-neri. Aumento di strutture dati. Teorema dell'aumento per alberi rosso-neri. Alberi di intervalli. - Tecniche avanzate di progettazione e analisi Programmazione dinamica. Algoritmi golosi. Analisi ammortizzata.

**Modalità di esame:**

Prova scritta ed esame orale.

**Criteri di valutazione:**

Gli esercizi della prova scritta mirano a valutare la capacità dello studente di utilizzare le nozioni apprese per l'individuazione di soluzioni algoritmiche efficienti a problemi assegnati. La prova orale verifica la conoscenza degli argomenti teorici discussi a lezione.

**Testi di riferimento:**

T.H.Cormen, Introduzione agli Algoritmi e Strutture Dati. : McGraw-Hill Italia, 2010

**Eventuali indicazioni sui materiali di studio:**

Esercizi e materiale aggiuntivo sono resi disponibili tramite la pagina web del corso e la pagina moodle.

## ALTRI PARADIGMI DI PROGRAMMAZIONE

**Titolare:** da definire

**Mutuato da:** Laurea in Informatica (Ord. 2011)

**Periodo:** III anno, 1 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 48A; 6,00

**Prerequisiti:**

Concetti di programmazione Object Oriented. Interazione con la linea di comando. È particolarmente utile aver seguito il corso di Programmazione ad Oggetti.

**Conoscenze e abilità da acquisire:**

Riconoscere le principali difficoltà legate alla costruzione di programmi e sistemi concorrenti e/o distribuiti; capire le limitazioni intrinseche dei vari approcci ed interpretare le garanzie fornite dai diversi strumenti in funzione dei requisiti del problema affrontato.

**Attività di apprendimento previste e metodologie di insegnamento:**

Lezioni frontali. Discussione in aula di un problemi e relative soluzioni, suddivise in passi e affrontate gradualmente, con interazione diretta insieme agli studenti.

**Contenuti:**

Il panorama dei problemi in cui l'informatica è usata come soluzione è ormai estremamente vasto; questo consente di individuare alcune larghe categorie le cui caratteristiche principali sono state studiate e su cui si è raccolta una cospicua esperienza. Usando il linguaggio Java e l'ecosistema della JVM come supporto, il corso illustra le difficoltà che si incontrano realizzando programmi e sistemi concorrenti e/o distribuiti, quali soluzioni si possono trovare in letteratura e sul mercato, e quali sono i criteri per selezionarle. Temi trattati: -Presentazione della JVM e del linguaggio Java; -collezioni ed operazioni sulle liste; -streams; -definizioni e problemi della concorrenza; threads; -definizioni e problemi della distribuzione; socket e channels; -cenni di stato distribuito; - reactive manifesto e tecnologie correlate.

**Modalità di esame:**

Esame scritto composto da domande a risposta multipla sugli argomenti teorici esposti a lezione e su problemi di valutazione immediata del comportamento di parti di codice. Svolgimento di esercizi completi su temi proposti a lezione.

**Criteri di valutazione:**

Chiarezza di intenti nel suddividere un problema in parti per risolvere ciascuna con il corretto metodo. Attenzione alla produzione di codice leggibile e in grado di comunicare chiaramente l'intento dell'autore.

**Testi di riferimento:**

Urma, Fusco, Mycroft, Modern Java in Action. : Manning, 2018

**Eventuali indicazioni sui materiali di studio:**

Nelle prime lezioni verranno indicati i software necessari per seguire il codice discusso a lezione. Questi comprenderanno sicuramente almeno una versione della piattaforma OpenJDK (17 o successive). Un IDE è consigliato (in ordine di preferenza: VSCode, Eclipse) ma non indispensabile.

**ANALISI MATEMATICA**

**Titolare:** Dott. FRANCESCO PAOLO MONTEFALCONE

**Periodo:** I anno, 1 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 64A+32E; 12,00

**Prerequisiti:**

Matematica di base (disequazioni, coordinate cartesiane, funzioni trigonometriche, logaritmiche ed esponenziali).

**Conoscenze e abilità da acquisire:**

Il corso si propone di illustrare i concetti e gli strumenti dell'Analisi per funzioni di una variabile reale, dando particolare rilievo agli aspetti di base del calcolo integro-differenziale e ai suoi aspetti logico-deduttivi.

**Attività di apprendimento previste e metodologie di insegnamento:**

Il metodo di insegnamento si basa su classiche lezioni frontali. Tutto il materiale didattico presentato nelle lezioni frontali è reso disponibile sulla piattaforma MOODLE.

**Contenuti:**

Numeri (naturali, interi, razionali, reali, complessi. Cardinalità: insiemi finiti e infiniti). Piano e Spazio euclideo (vettori nel piano e nello spazio ordinario; equazioni cartesiane di rette e piani). Successioni in  $\mathbb{R}$ . Limiti di funzioni di una variabile reale. Derivate di funzioni di una variabile reale. Teoremi fondamentali del calcolo integro-differenziale. Formula di Taylor. Massimi e minimi locali. Grafici di funzioni di una variabile. Integrale definito. Integrale indefinito e metodi di integrazione. Integrali generalizzati. Serie numeriche. Equazioni differenziali del primo ordine. Cenni su alcune generalizzazioni dell'Analisi per funzioni di più variabili.

**Modalità di esame:**

Le modalità di esame saranno illustrate durante la prima lezione e rese poi disponibili sulla pagina MOODLE del corso.

**Criteri di valutazione:**

Comprensione degli argomenti teorici e capacità di risolvere esercizi. In particolare allo studente viene richiesto: 1) di usare il linguaggio matematico correttamente, 2) di dimostrare in modo rigoroso un certo numero di teoremi, 3) di sviluppare un approccio critico che gli permetta di individuare eventuali errori di un ragionamento matematico, 4) di risolvere gli esercizi proposti.

**Testi di riferimento:**

CONTENUTO NON PRESENTE

**Eventuali indicazioni sui materiali di studio:**

Le Dispense del Corso e gli altri materiali didattici (fogli di esercizi etc.) saranno forniti dal docente all'inizio del corso sulla pagina Moodle. In classe saranno dati consigli per ulteriori libri di testo, in particolare testi di esercizi.

**ARCHITETTURA DEGLI ELABORATORI**

**Titolare:** Prof. ALESSANDRO SPERDUTI

**Periodo:** I anno, 1 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 40A+16E+10L; 8,00

**Sede dell'insegnamento:** Padova

**Aule:** LUM 250, Paolotti

**Prerequisiti:**

Non sono richieste conoscenze particolari, se non quelle di base della matematica. L'insegnamento non prevede propedeuticità.

**Conoscenze e abilità da acquisire:**

Obiettivo dell'insegnamento è quello di permettere allo studente di acquisire alcune conoscenze di base funzionali e tecnologiche riguardanti l'architettura degli elaboratori limitatamente al loro utilizzo in ambito locale. Sono previste esercitazioni in laboratorio informatico che consentono allo studente di approfondire le conoscenze acquisite mediante l'utilizzo di semplici simulatori di CPU, Cache, e Pipeline.

**Attività di apprendimento previste e metodologie di insegnamento:**

L'insegnamento prevede lezioni frontali ed esercitazioni in laboratorio informatico. Le esercitazioni in laboratorio informatico consistono nella sperimentazione da parte degli studenti di simulatori di CPU, Cache e Pipeline, sotto vari scenari operativi. In questo modo gli studenti possono verificare sperimentalmente i concetti appresi a lezione e acquisire sia capacità di applicazione dei concetti appresi che di giudizio critico. In caso di emergenza sanitaria sia le lezioni che le esercitazioni saranno tenute online.

**Contenuti:**

La struttura e le tematiche dell'insegnamento saranno le seguenti: - Introduzione: Evoluzione dei calcolatori; visione ad alto livello della struttura di un calcolatore; struttura e funzione della Cpu. - Gestione della Memoria: Memorie e Gerarchie di Memorie. Cache: tecniche di associazione, politiche di rimpiazzo. Simulatore Cache. - Dispositivi e Gestione dell'Input/Output: Input/Output: dispositivi esterni, modulo I/O, gestione da programma, gestione tramite interruzioni, DMA. - Cenni di Circuiti Combinatori e Sequenziali, Microprogrammazione: Algebra di Boole. Porte logiche. Circuiti Combinatori. Circuiti sequenziali. Microprogrammazione. - Aritmetica dei Calcolatori: Livello Macchina, Rappresentazione Binaria, Aritmetica. - Linguaggio Assembler e Livello Instruction Set: Linguaggio assembler. Caratteristiche istruzioni macchina. Tipi degli operandi, dati, operazioni. Indirizzamento. Formato istruzioni. Simulatore CPU. - Livello Instruction Set: Architetture CISC e RISC, Processori Multicore - Valutazione e Miglioramento delle Prestazioni: Pipeline: principi generali, prestazioni ideali, dipendenze, tecniche per la riduzione delle dipendenze, MIPS. Simulatore pipeline MIPS.

**Modalità di esame:**

Lo studente deve superare un esame scritto e, se ritenuto necessario dal docente, un esame orale.

**Criteri di valutazione:**

Il testo dell'esame scritto contiene alcune domande che consentono di valutare il livello di apprendimento delle nozioni impartite durante l'insegnamento e la capacità dello studente nell'analizzarle criticamente. Sono poi presenti esercizi in cui si richiede allo studente di ricostruire il funzionamento o il dimensionamento di alcune componenti dell'elaboratore. Tali esercizi hanno lo scopo di valutare se lo studente ha sviluppato la capacità di applicare le nozioni apprese durante l'insegnamento. Nel caso in cui la valutazione dello scritto risulti appena sotto la sufficienza, il docente può decidere di integrare l'esame scritto con un esame orale per meglio verificare la preparazione dello studente.

**Testi di riferimento:**

Stallings, William, Computer organization and architecture designing for performance. Upper Saddle River: Pearson education, 2015 Patterson, David A.; Hennessy, John L.; Borghese, Alberto, Struttura e progetto dei calcolatori. Bologna: Zanichelli, 2015

**Eventuali indicazioni sui materiali di studio:**

Vengono rese disponibili, come riferimento, i lucidi utilizzati a lezione.

<b>AUTOMI E LINGUAGGI FORMALI</b>
-----------------------------------

**Titolare:** Prof. DAVIDE BRESOLIN

**Periodo:** Il anno, 2 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 32A+16E; 6,00

**Sede dell'insegnamento:** Plesso Paolotti, via Paolotti.

**Aule:** LUM 250.

**Prerequisiti:**

Nozioni di logica.

**Conoscenze e abilità da acquisire:**

Questo corso fornisce i concetti fondamentali della teoria degli automi e dei linguaggi formali. Inoltre, introduce le nozioni di indecidibilità e intrattabilità.

**Attività di apprendimento previste e metodologie di insegnamento:**

Lezioni frontali ed esercizi in classe.

**Contenuti:**

I principali contenuti del corso sono i seguenti: Parte 1: linguaggi regolari -- automi a stati finiti -- espressioni e linguaggi regolari -- pumping lemma -- proprietà dei linguaggi regolari Parte 2: linguaggi liberi da contesto -- grammatiche e linguaggi liberi da contesto -- automi a pila -- pumping lemma -- proprietà dei linguaggi liberi da contesto Parte 3: indecidibilità e intrattabilità -- macchine di Turing -- indecidibilità -- problemi intrattabili -- classi P e NP

**Modalità di esame:**

Scritto

**Criteri di valutazione:**

Lo scritto contiene alcune domande che consentono di valutare il livello di apprendimento delle nozioni impartite durante il corso. Sono poi presenti esercizi di costruzione di automi e di grammatiche formali.

**Testi di riferimento:**

Sipser, Michael, Introduction to the theory of computation. : Cengage Learning, 2013 Sipser, Michael, Introduzione alla teoria della computazione. : Maggioli Editore, 2016

**Eventuali indicazioni sui materiali di studio:**

Un sito moodle con tutto il materiale del corso. Sono disponibili anche e le registrazioni delle lezioni

<b>BASI DI DATI</b>
---------------------

**Titolare:** Prof. MASSIMILIANO DE LEONI

**Periodo:** Il anno, 2 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 48A+8E+16L; 9,00

**Prerequisiti:**

È opportuno avere familiarità con gli elementi di base della programmazione, così come forniti nel corso di "Programmazione" L'insegnamento non prevede propedeuticità.

**Conoscenze e abilità da acquisire:**

Obiettivo del corso è quello di permettere allo studente di acquisire alcune conoscenze di base riguardanti le funzionalità dei DBMS (Data Base Management Systems – sistemi per la gestione di basi di dati), la progettazione delle basi di dati mediante l'uso di modelli concettuali, il progetto logico mediante il modello relazionale dei dati e l'uso del linguaggio SQL per la definizione e la realizzazione di basi di dati. Sono previste esercitazioni in laboratorio che consentono allo studente di sperimentare le conoscenze acquisite mediante l'utilizzo di un DBMS concreto. Infine è richiesto lo sviluppo di un progetto che consiste nella progettazione e realizzazione di una base di dati (analisi, progetto concettuale, progetto logico, implementazione, individuazione e implementazione di query di interesse). Oltre ad incrementare la capacità di applicare le conoscenze teoriche acquisite, le varie fasi del progetto espongono lo studente a scelte autonome e ragionate, che richiedono una rielaborazione personale delle conoscenze acquisite.

**Attività di apprendimento previste e metodologie di insegnamento:**

L'insegnamento prevede lezioni frontali, esercitazioni in laboratorio e la realizzazione di un progetto. Le esercitazioni in laboratorio consistono nell'implementazione di semplici database, relative interrogazioni SQL. Sono propedeutiche alla realizzazione del progetto, nel quale lo studente individua autonomamente un caso di studio, progetta ed implementa una base di dati.

**Contenuti:**

La struttura e le tematiche del corso saranno le seguenti: - Introduzione Basi di dati e sistemi di gestione di basi di dati. Architettura di un sistema di gestione di basi di dati. - La progettazione concettuale di una base di dati Il modello concettuale a oggetti. Progettazione concettuale di basi di dati mediante il modello a oggetti. - La progettazione logica di una base di dati Il modello relazionale. L'algebra relazionale e il calcolo relazionale. Progettazione logica di basi di dati mediante traduzione di schemi a oggetti in schemi relazionali. - La normalizzazione di schemi relazionali Dipendenze funzionali. Scomposizione di relazioni, con preservazione di dati e/o dipendenze. Forme normali per gli schemi di relazione (1NF, 2NF, 3NF e BCNF) e decomposizioni in forma normale. - Transazioni e le proprietà ACID. - Il linguaggio SQL Il data definition language di SQL. Il data manipulation language di SQL (interrogazione e aggiornamento). Il DBMS PostgreSQL. - Accesso a Basi di Dati da Applicazioni Software

**Modalità di esame:**

Lo studente deve superare uno scritto e realizzare un progetto (nel quale mettere in pratica le nozioni acquisite nel corso).

**Criteri di valutazione:**

Lo scritto contiene alcune domande che consentono di valutare il livello di apprendimento delle nozioni impartite durante il corso. Sono poi presenti esercizi di progettazione, formulazione di query SQL e normalizzazione che richiedono allo studente un'elaborazione personale di concetti e tecniche viste nel corso. La valutazione del progetto considera la capacità, da parte dello studente, di individuare un caso di studio adeguato, di svolgere in modo autonomo un'attività di progettazione qualitativamente appropriata e di realizzare una implementazione disciplinata.

**Testi di riferimento:**

Paolo Atzeni, Stefano Ceri, Stefano Paraboschi, Riccardo Torlone, Basi di Dati. : McGraw Hill,

**Eventuali indicazioni sui materiali di studio:**

Verranno rese disponibili, come riferimento, le slide utilizzate a lezione.

<b>CALCOLO NUMERICO</b>
-------------------------

**Titolare:** Prof. MARCO VIANELLO

**Periodo:** Il anno, 2 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 40A+16L; 6,00

**Sede dell'insegnamento:** Padova.

**Prerequisiti:**

Conoscenze di base di analisi matematica e algebra lineare.

**Conoscenze e abilità da acquisire:**

Apprendere le basi del calcolo numerico in vista delle applicazioni in campo scientifico e tecnologico, con particolare attenzione ai concetti di errore, discretizzazione, approssimazione, convergenza, stabilità, costo computazionale.

**Attività di apprendimento previste e metodologie di insegnamento:**

Lezioni in aula ed esercitazioni di laboratorio.

**Contenuti:**

Sistema-floating point e propagazione degli errori: errore di troncamento e di arrotondamento, rappresentazione floating-point dei reali, precisione di macchina, operazioni aritmetiche con numeri approssimati, condizionamento di funzioni, propagazione degli errori in algoritmi iterativi per esempi, il concetto di stabilità. Costo computazionale degli algoritmi numerici per esempi: schema di Hoerner per polinomi, potenza rapida tramite codifica binaria dell'esponente, calcolo della funzione esponenziale con la formula di Taylor via scalatura della variabile e potenza rapida. Soluzione numerica di equazioni non lineari: metodo di bisezione, stima dell'errore col residuo pesato; metodo di Newton, convergenza globale, velocità di convergenza, convergenza locale, stima dell'errore, altri metodi di linearizzazione; iterazioni di punto fisso. Interpolazione e approssimazione di funzioni e dati: interpolazione polinomiale, interpolazione di Lagrange, errore di interpolazione, il problema della convergenza (controesempio di Runge), interpolazione di Chebyshev, stabilità dell'interpolazione; interpolazione polinomiale a tratti, interpolazione spline; approssimazione polinomiale ai minimi quadrati. Integrazione e derivazione numerica: formule di quadratura algebriche e composte, convergenza e stabilità, esempi; instabilità dell'operazione di derivazione, calcolo di derivate tramite formule alle differenze; il concetto di estrapolazione. Elementi di algebra lineare numerica: norme di vettori e matrici, condizionamento di matrici e sistemi; metodi diretti: metodo di eliminazione gaussiana e fattorizzazione LU, calcolo del determinante, calcolo della matrice inversa, fattorizzazione QR, soluzione ai minimi quadrati di sistemi sovradeterminati. Laboratorio: implementazione e applicazione di codici numerici in Matlab.

**Modalità di esame:**

Esame scritto e prova di laboratorio.

**Criteri di valutazione:**

La prova scritta mira a verificare la comprensione dei fondamenti teorici dei metodi numerici. La prova di laboratorio mira a verificare la capacità di implementazione e applicazione degli algoritmi numerici.

**Testi di riferimento:**

Rodriguez, Giuseppe, Algoritmi numerici. Bologna: Pitagora, 2008

**Eventuali indicazioni sui materiali di studio:**

Lezioni estese scritte (su Moodle). Dispense sintetiche online ([www.math.unipd.it/~marcov/studenti.html](http://www.math.unipd.it/~marcov/studenti.html)).

**CYBERSECURITY: PRINCIPLES AND PRACTICES**

**Titolare:** Prof. MAURO CONTI

**Mutuato da:** Laurea in Informatica (Ord. 2011)

**Periodo:** III anno, 1 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 48A; 6,00

**Prerequisiti:**

Il corso non prevede propedeuticità.

**Conoscenze e abilità da acquisire:**

Acquisire concetti di base di sicurezza (e.g., Access Control, User Authentication, Malware, Attacchi DoS, Intrusion Detection/Prevention, Software and OS security, Trusted Computing) e conoscenze di sicurezza di sistema in ambiente Linux/Windows/Android, sicurezza di reti wireless/wired, web-application security. Al termine del corso gli studenti avranno acquisito le conoscenze di base della sicurezza informatica e saranno in grado di analizzare un sistema, identificandone le vulnerabilità.

**Attività di apprendimento previste e metodologie di insegnamento:**

Lezioni frontali; esercitazioni in laboratorio.

**Contenuti:**

Teoria: Crittografia: cifrari classici, crittografia simmetrica e asimmetrica, autenticazione dei messaggi, confidenzialità di messaggi, hash, cifrari di stream e cifrari a blocchi. Sicurezza Web: struttura di un'applicazione Web, sicurezza dei database, vulnerabilità XSS, vulnerabilità dei linguaggi di programmazione web. Analisi forense: analisi del traffico di rete, steganografia, text watermarking. Sicurezza del software: buffer overflow, stack overflow, sicurezza dei sistema operativo, antidebug, patching, mitigazioni (stack canaries, ASLR, W ^ X, RELRO), sfruttamento di heap di base non orientato ai metadati, sfruttamento di heap di base orientato ai metadati. Laboratorio: sfide pratiche su ciascuno degli argomenti sopra menzionati

**Modalità di esame:**

Scritto.

**Criteri di valutazione:**

Conoscenza dei concetti studiati nel corso.

**Testi di riferimento:**

Stallings, William; Brown, Lawrie, Computer security principles and practice. Boston: Pearson, 2015 Bishop, Matthew, Introduction to computer security. Boston: Addison-Wesley, 2005

**Eventuali indicazioni sui materiali di studio:**

Slide e materiale fornito durante il corso. Libro (testo principale Computer Security: Principles and Practice 2/E). Il corso sarà tenuto in Inglese. Il sito web del corso offrirà tutte le informazioni e materiale ulteriore: <http://www.math.unipd.it/~conti/teaching.html>

**CYBERSECURITY: PRINCIPLES AND PRACTICES**

**Titolare:** Prof. MAURO CONTI

**Periodo:** Il anno, 1 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 48A; 6,00

**Prerequisiti:**

Il corso non prevede propedeuticità.

**Conoscenze e abilità da acquisire:**

Acquisire concetti di base di sicurezza (e.g., Access Control, User Authentication, Malware, Attacchi DoS, Intrusion Detection/Prevention, Software and OS security, Trusted Computing) e conoscenze di sicurezza di sistema in ambiente Linux/Windows/Android, sicurezza di reti wireless/wired, web-application security. Al termine del corso gli studenti avranno acquisito le conoscenze di base della sicurezza informatica e saranno in grado di analizzare un sistema, identificandone le vulnerabilità.

**Attività di apprendimento previste e metodologie di insegnamento:**

Lezioni frontali; esercitazioni in laboratorio.

**Contenuti:**

Teoria: Crittografia: cifrari classici, crittografia simmetrica e asimmetrica, autenticazione dei messaggi, confidenzialità di messaggi, hash, cifrari di stream e cifrari a blocchi. Sicurezza Web: struttura di un'applicazione Web, sicurezza dei database, vulnerabilità XSS, vulnerabilità dei linguaggi di programmazione web. Analisi forense: analisi del traffico di rete, steganografia, text watermarking. Sicurezza del software: buffer overflow, stack overflow, sicurezza dei sistema operativo, antidebug, patching, mitigazioni (stack canaries, ASLR, W ^ X, RELRO), sfruttamento di heap di base non orientato ai metadati, sfruttamento di heap di base orientato ai metadati. Laboratorio: sfide pratiche su ciascuno degli argomenti sopra menzionati

**Modalità di esame:**

Scritto.

**Criteri di valutazione:**

Conoscenza dei concetti studiati nel corso.

**Testi di riferimento:**

Stallings, William; Brown, Lawrie, Computer security principles and practice. Boston: Pearson, 2015 Bishop, Matthew, Introduction to computer security. Boston: Addison-Wesley, 2005

**Eventuali indicazioni sui materiali di studio:**

Slide e materiale fornito durante il corso. Libro (testo principale Computer Security: Principles and Practice 2/E). Il corso sarà tenuto in Inglese. Il sito web del corso offrirà tutte le informazioni e materiale ulteriore: <http://www.math.unipd.it/~conti/teaching.html>

**DIRITTO, INFORMATICA E SOCIETA'**

**Titolare:** Prof. ANDREA SITZIA

**Periodo:** Il anno, 2 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 48A; 6,00

**Prerequisiti:**

Nessuno

**Conoscenze e abilità da acquisire:**

Il corso si propone di introdurre gli studenti di informatica agli aspetti giuridici legati alle professioni di riferimento. Gli studenti apprenderanno come le tecnologie, ed in particolare le tecnologie digitali e la robotica, possono impattare sui diritti degli individui quali privacy, auto-determinazione, integrità personale e come la valutazione del rischio di una violazione di questi diritti debba essere tenuta in considerazione nella realizzazione di un prodotto. Verrà fornito un quadro generale in materia di governance, responsabilità, ricerca e innovazione, privacy by design, GDPR.

**Attività di apprendimento previste e metodologie di insegnamento:**

Discussione di questioni pratiche rilevanti e ricostruzione ragionata delle regole, approccio casistico, problem solving e discussion della metodologia giuridica. Lezioni frontali

**Contenuti:**

1-Introduzione alla regolamentazione: nozione di fonte, regola, diritto (soggettivo e oggettivo), ordinamento, hard e soft law (aspetti generali) 2- Introduzione ai principi giuridici fondamentali per l'informatico (informatica e diritto) I contratti per l'informatica, i contratti di cessione/gestione/manutenzione del software, la responsabilità degli Internet Service Provider, la proprietà intellettuale e il diritto d'autore, le privative industriali, i contratti per la prestazione di servizi, il contratto di lavoro subordinato e autonomo, la "consulenza" 3-Introduzione alla governance tecnologica: crisi del modello di regolamentazione command and control, nozione di attore di governance (pubblico e privato), separazione tra government e governing, teorie e modelli di governance (governance, self-governance, modelli rights-based), il modello della Responsible, Research and Innovation. Governance per la robotica, Intelligenza Artificiale e tecnologie digitali 4-Nuove tecnologie: questioni etiche e giuridiche Intelligenza Artificiale e tecnologie digitali, la circolazione dei dati personali, la tutela della privacy, la privacy nel contesto del lavoro, i limiti al controllo tecnologico

**Modalità di esame:**

Prova orale

**Criteri di valutazione:**

Verifica dell'apprendimento dei concetti chiave fondamentali, interazione tra linguaggi, verifica del corretto approccio metodologico.

**Testi di riferimento:**

CONTENUTO NON PRESENTE

**Eventuali indicazioni sui materiali di studio:**

I materiali di studio verranno forniti dai docenti.

## INGEGNERIA DEL SOFTWARE

**Titolare:** Prof. TULLIO VARDANEGA

**Periodo:** III anno, annuale

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 64A+8E+32L; 13,00

**Prerequisiti:**

L'insegnamento assume e richiede familiarità con i linguaggi e tecniche di programmazione presentati nel triennio (programmazione imperativa e a oggetti come istanziata in C, C++, Java), e con le funzionalità fondamentali delle base di dati e delle più comuni tecnologie basate su SQL. L'importanza di tali conoscenze preliminari determina vincolo di propedeuticità per l'accesso alle attività didattiche di Ingegneria del Software, nel superamento degli insegnamenti Programmazione a oggetti e Basi di dati.

**Conoscenze e abilità da acquisire:**

L'insegnamento intende fornire le conoscenze alla base della professione di "software engineer" e confronta gli studenti con attività di pratica esperienziale collaborativa, per il tramite di un sostanzioso progetto didattico collaborativo, che attraversa tutte le fasi principali del suo ciclo di sviluppo, dalla risposta a una gara di appalto fino a collaudo. Nello specifico, lo studente apprenderà: 1) metodi e tecniche per l'organizzazione e la gestione di attività collaborative (pianificazione, ripartizione dei compiti, calendarizzazione, monitoraggio del progresso, verifica degli esiti); 2) professionalità nell'interazione con "clienti" esterni (cattura, analisi e discussione di requisiti, dimostrazione di prototipi e di prodotti finiti); 3) metodi e tecniche per l'auto-apprendimento e per la condivisione di conoscenza; 4) metodi e tecniche di presentazione e comunicazione.

**Attività di apprendimento previste e metodologie di insegnamento:**

1) insegnamento d'aula, sia sincrono che asincrono; 2) lezioni "rovesciate", svolte in modalità "flipped classroom", su argomenti selezionati, centrati prevalentemente su strumenti e tecniche di sviluppo collaborativo; 3) esercitazione guidate su temi di teoria; 4) attività pratiche con feedback di progresso.

**Contenuti:**

Per una carrellata aggiornata degli argomenti di insegnamento, si veda la pagina , che include diapositive di presentazione e materiale di approfondimento.

**Modalità di esame:**

L'esame di questo insegnamento consta di due parti complementari: - una prova scritta, con quesiti e piccoli esercizi di temi pratici e teorici, il cui esito positivo conta per il 40% sul voto finale; - una serie di revisioni di avanzamento di progetto didattico, di gruppo, da candidatura all'ingresso, fino a collaudo del prodotto sviluppato, la cui media di esito conta per il rimanente 60% del voto finale individuale.

**Criteri di valutazione:**

Lo scritto individuale consiste di domande pratiche di tipo "problem solving" e domande sui concetti fondanti della disciplina del software engineering. Nel complesso, tali domande mirano a valutare il livello di apprendimento raggiunto dallo studente nelle attività di corso, in aula, nel progetto didattico, e nello studio autonomo. Per mettere alla prova le capacità di collaborazione sviluppate nel progetto didattico, alcune domande della prova scritta richiederanno risposta concordate da gruppi di due o tre studenti determinati prima della prova. La valutazione delle revisioni di progresso, nel loro complesso, misura invece il gradiente di miglioramento qualità dei prodotti realizzati nel corso del progetto, sia documentali che software, e delle corrispondenti presentazioni e dimostrazioni pubbliche.

**Testi di riferimento:**

Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software. : Pearson Education, 1995 IEEE Computer Society. Software Engineering Coordinating Committee, Guide to the Software Engineering Body of Knowledge. : , 2014 Ian Sommerville, Software Engineering. : Pearson Education, 2016

**Eventuali indicazioni sui materiali di studio:**

Tutto il materiale didattico dell'insegnamento è reso disponibile attraverso la piattaforma Moodle, per la parte di risorse online accessibile con autenticazione, e una apposita pagina-calendario statica dell'insegnamento, che fornisce materiale di apprendimento, di pubblico dominio.

## INTRODUZIONE ALL'APPRENDIMENTO AUTOMATICO

**Titolare:** Prof. LAMBERTO BALLAN

**Mutuato da:** Laurea in Informatica (Ord. 2011)

**Periodo:** III anno, 2 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 48A; 6,00

**Prerequisiti:**

Sono richieste conoscenze di base di Analisi Matematica, Probabilità e Statistica, Algebra Lineare, e di Programmazione.

**Conoscenze e abilità da acquisire:**

Il corso presenta i concetti e gli algoritmi fondamentali dell'Apprendimento Automatico, cioè quella classe di tecniche ed algoritmi che a partire dai dati consentono di acquisire nuova conoscenza. Questi sono particolarmente utili per tutti quei problemi per cui è impossibile, o molto difficile, pervenire ad una formalizzazione utilizzabile per la definizione di una soluzione algoritmica ad hoc. Sono previste varie esercitazioni in laboratorio informatico, al fine di consentire allo studente di sperimentare le conoscenze acquisite mediante la loro applicazione a piccoli esempi pratici. Il corso insegna inoltre le competenze necessarie per applicare e valutare modelli di apprendimento automatico in contesti applicativi reali.

**Attività di apprendimento previste e metodologie di insegnamento:**

Il corso consiste in lezioni e esercizi in laboratorio informatico. Gli esercizi in laboratorio informatico consentono agli studenti di sperimentare, in diversi scenari operativi, le tecniche introdotte a lezione. Gli studenti possono così verificare sperimentalmente i concetti appresi in classe, ed acquisire la capacità di applicarli criticamente di fronte ad un problema concreto.

**Contenuti:**

Il corso tratta gli argomenti elencati di seguito: - Intelligenza Artificiale ed Apprendimento Automatico, quando e perchè utilizzare tecniche di apprendimento automatico; gli ingredienti fondamentali, i principali paradigmi e le loro applicazioni. - Modelli non-parametrici ed alberi di decisione. - Modelli di regressione lineare; spazio delle ipotesi, rappresentazione e definizione della funzione di costo; ottimizzazione e discesa del gradiente. - Modelli di classificazione lineare; regressione logistica; regolarizzazione ed model selection. - Complessità dei modelli di apprendimento automatico; Bias-Variance Tradeoff, overfitting e underfitting; misure di valutazione delle performance, esempi ed applicazioni; come effettuare debugging e diagnosticare il comportamento di un sistema di apprendimento automatico. - Reti Neurali Artificiali: perceptron, reti multistrato e cenni al deep learning; apprendimento dei parametri in reti neurali, backpropagation e discesa di gradiente. - Support Vector Machines; metodi kernel e classificazione non lineare. - Apprendimento non supervisionato: clustering e riduzione di dimensionalità. - Introduzione ai servizi cognitivi, alla percezione nelle macchine (visione e riconoscimento del parlato), NLP e allo sviluppo di sistemi intelligenti.

**Modalità di esame:**

La valutazione delle conoscenze e delle capacità acquisite viene effettuata mediante due prove: 1. Una prova scritta in cui lo studente deve risolvere dei problemi e rispondere a domande inerenti i contenuti affrontati nel corso. 2. Esercitazioni assegnate ed introdotte in laboratorio informatico, rivolte all'acquisizione delle competenze, anche pratiche, necessarie per un corretto utilizzo degli strumenti di apprendimento automatico. Lo studente deve produrre una breve relazione che descriva le metodologie utilizzate per risolvere i diversi problemi assegnati, assieme ai risultati ottenuti.

**Criteri di valutazione:**

Il testo dell'esame scritto contiene domande ed esercizi che consentono di valutare il livello di apprendimento delle nozioni impartite nel corso e la capacità dello studente nell'analizzarle criticamente. Lo scopo delle esercitazioni è quello di verificare se lo studente ha acquisito le competenze necessarie ad applicare algoritmi di apprendimento automatico a problemi reali. La discussione critica di quanto realizzato e dei risultati ottenuti, costituiscono elemento fondamentale di valutazione.

**Testi di riferimento:**

CONTENUTO NON PRESENTE

**Eventuali indicazioni sui materiali di studio:**

Le presentazioni mostrate durante le lezioni sono rese disponibili su Moodle come materiale di riferimento.

## INTRODUZIONE ALL'APPRENDIMENTO AUTOMATICO

**Titolare:** Prof. LAMBERTO BALLAN

**Periodo:** Il anno, 2 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 40A+8L; 6,00

**Prerequisiti:**

Sono richieste conoscenze di base di Analisi Matematica, Probabilità e Statistica, Algebra Lineare, e di Programmazione.

**Conoscenze e abilità da acquisire:**

Il corso presenta i concetti e gli algoritmi fondamentali dell'Apprendimento Automatico, cioè quella classe di tecniche ed algoritmi che a partire dai dati consentono di acquisire nuova conoscenza. Questi sono particolarmente utili per tutti quei problemi per cui è impossibile, o molto difficile, pervenire ad una formalizzazione utilizzabile per la definizione di una soluzione algoritmica ad hoc. Sono previste varie esercitazioni in laboratorio informatico, al fine di consentire allo studente di sperimentare le conoscenze acquisite mediante la loro applicazione a piccoli esempi pratici. Il corso insegna inoltre le competenze necessarie per applicare e valutare modelli di apprendimento automatico in contesti applicativi reali.

**Attività di apprendimento previste e metodologie di insegnamento:**

Il corso consiste in lezioni e esercizi in laboratorio informatico. Gli esercizi in laboratorio informatico consentono agli studenti di sperimentare, in diversi scenari operativi, le tecniche introdotte a lezione. Gli studenti possono così verificare sperimentalmente i concetti appresi in classe, ed acquisire la capacità di applicarli criticamente di fronte ad un problema concreto.

**Contenuti:**

Il corso tratta gli argomenti elencati di seguito: - Intelligenza Artificiale ed Apprendimento Automatico, quando e perchè utilizzare tecniche di apprendimento automatico; gli ingredienti fondamentali, i principali paradigmi e le loro applicazioni. - Modelli non-parametrici ed alberi di decisione. - Modelli di regressione lineare; spazio delle ipotesi, rappresentazione e definizione della funzione di costo; ottimizzazione e discesa del gradiente. - Modelli di classificazione lineare; regressione logistica; regolarizzazione ed model selection. - Complessità dei modelli di apprendimento automatico; Bias-Variance Tradeoff, overfitting e underfitting; misure di valutazione delle performance, esempi ed applicazioni; come effettuare debugging e diagnosticare il comportamento di un sistema di apprendimento automatico. - Reti Neurali Artificiali: perceptron, reti multistrato e cenni al deep learning; apprendimento dei parametri in reti neurali, backpropagation e discesa di gradiente. - Support Vector Machines; metodi kernel e classificazione non lineare. - Apprendimento non supervisionato: clustering e riduzione di dimensionalità. - Introduzione ai servizi cognitivi, alla percezione nelle macchine (visione e riconoscimento del parlato), NLP e allo sviluppo di sistemi intelligenti.

**Modalità di esame:**

La valutazione delle conoscenze e delle capacità acquisite viene effettuata mediante due prove: 1. Una prova scritta in cui lo studente deve risolvere dei problemi e rispondere a domande inerenti i contenuti affrontati nel corso. 2. Esercitazioni assegnate ed introdotte in laboratorio informatico, rivolte all'acquisizione delle competenze, anche pratiche, necessarie per un corretto utilizzo degli strumenti di apprendimento automatico. Lo studente deve produrre una breve relazione che descriva le metodologie utilizzate per risolvere i diversi problemi assegnati, assieme ai risultati ottenuti.

**Criteri di valutazione:**

Il testo dell'esame scritto contiene domande ed esercizi che consentono di valutare il livello di apprendimento delle nozioni impartite nel corso e la capacità dello studente nell'analizzarle criticamente. Lo scopo delle esercitazioni è quello di verificare se lo studente ha acquisito le competenze necessarie ad applicare algoritmi di apprendimento automatico a problemi reali. La discussione critica di quanto realizzato e dei risultati ottenuti, costituiscono elemento fondamentale di valutazione.

**Testi di riferimento:**

CONTENUTO NON PRESENTE

**Eventuali indicazioni sui materiali di studio:**

## LINGUA INGLESE B2 (ABILITA' RICETTIVE)

**Titolare:** Prof. MASSIMO MARCHIORI

**Periodo:** I anno, 2 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** ; 3,00

**Sede dell'insegnamento:** Vedi <http://corsi.math.unipd.it/inglese/>

**Aule:** Vedi <http://corsi.math.unipd.it/inglese/>

**Contenuti:**

Tutti i Corsi di Laurea di Scienze richiedono una conoscenza della Lingua inglese pari al livello B2 (abilità ricettive ascolto e lettura) del Quadro Comune Europeo di Riferimento per le Lingue del Consiglio d'Europa. Chi è già in possesso di una certificazione di livello B2 o superiore può chiederne il riconoscimento. Tutti gli altri studenti possono sostenere presso il Centro Linguistico di Ateneo il corrispondente Test di Abilità Linguistica (TAL), il cui superamento permette il riconoscimento dei crediti formativi per la lingua straniera. Tutte le informazioni sull'idoneità, sul test di lingua e sulle certificazioni riconosciute, sono disponibili all'indirizzo [http://www.scienze.unipd.it/index.php?id=inglese\\_triennali\\_1819](http://www.scienze.unipd.it/index.php?id=inglese_triennali_1819) Per maggiori informazioni: <http://corsi.math.unipd.it/inglese/>

**Testi di riferimento:**

CONTENUTO NON PRESENTE

## LOGICA

**Titolare:** Prof.ssa MARIA EMILIA MAIETTI

**Periodo:** I anno, 1 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 32A+18E; 6,00

**Prerequisiti:**

Nessuno

**Conoscenze e abilità da acquisire:**

Lo scopo del corso è fornire un'introduzione alla logica e alla sua rilevanza per l'informatica. Lo studente dovrà quindi acquisire la capacità di esprimere un enunciato tramite una formula di un linguaggio formale, di dare una dimostrazione tramite una derivazione in un sistema assiomatico e di fornire controesempi nel caso la formula non sia derivabile. In particolare lo studente sarà condotto alla comprensione di alcuni concetti di carattere generale, come linguaggio, espressione, proposizione, asserzione, metalinguaggio, e altri specifici della matematica, come derivazione, dimostrazione, sistema assiomatico, interpretazione. Lo studente raggiungerà padronanza di tali concetti e sarà quindi in grado di riconoscerli ed applicarli nella matematica, nell'informatica e anche nella vita quotidiana. Il corso illustrerà, inoltre, come la logica possa chiarire con rigore quali siano i limiti intrinseci a quel che può essere espresso in un dato linguaggio e a quel che può essere dimostrato in un dato sistema assiomatico. Infine, il corso darà qualche cenno alla storia della logica e alle sue potenzialità e prospettive attuali.

**Attività di apprendimento previste e metodologie di insegnamento:**

Oltre alle lezioni teoriche sono previste esercitazioni in aula con risoluzione di molti esercizi. Sono previste simulazioni in aula degli esami scritti.

**Contenuti:**

1. Linguaggio, metalinguaggio, livelli di riferimento, iterazione infinita. 2. Concetto di macchina o robot, significato dei connettivi e loro regole di deduzione, calcolo dei sequenti per logica classica proposizionale, tabelle di verità, teoremi di validità e completezza. 3. Calcolo dei sequenti per la logica classica predicativa, nozione di interpretazione e modello, teoremi di validità e completezza. 4. Metodi di decisione per calcoli dei sequenti proposizionali classici. 5. Costruzione di contromodelli classici di enunciati predicativi. 6. Cenni ai teoremi di completezza e incompletezza (Gödel) e di indecidibilità (Church) e loro significato.

**Modalità di esame:**

Esame scritto

**Criteri di valutazione:**

La valutazione della prova scritta si baserà sull'assegnazione di un punteggio ad ogni esercizio presente nel testo d'esame.

**Testi di riferimento:**

Maria Emilia Maietti, Manuale pratico di Logica. : Padova, 2020 Giovanni Sambin, Per istruire un robot. : Libreria Cortina, Padova, 2007

**Eventuali indicazioni sui materiali di studio:**

Verranno fornite dispense contenenti tutti gli aspetti teorici e pratici necessari (con incluso l'assegnazione e lo svolgimento di esercizi specifici) per l'apprendimento degli argomenti trattati nelle lezioni frontali.

## METODI E TECNOLOGIE PER LO SVILUPPO SOFTWARE

**Titolare:** Dott. NICOLA BERTAZZO

**Periodo:** II anno, 2 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 48A; 6,00

**Prerequisiti:**

Nessuno.

**Conoscenze e abilità da acquisire:**

Fornire agli studenti un bagaglio di esperienza base per la gestione tecnologica di un progetto software e la definizione e l'implementazione di una continuous delivery pipeline.

**Attività di apprendimento previste e metodologie di insegnamento:**

Lezioni frontali e laboratorio.

**Contenuti:**

Saranno trattati i seguenti temi: - Issue Tracking - Source Code Management (SCM) - Testing Software - Processo di Build - Continuous integration e Continuous Delivery - Configuration Management

**Modalità di esame:**

Esercitazioni e prova orale

**Criteri di valutazione:**

La valutazione si baserà su colloqui (individuali o di gruppi), a valle di 4 assignment presentati e discussi ai laboratori.

**Testi di riferimento:**

CONTENUTO NON PRESENTE

**Eventuali indicazioni sui materiali di studio:**

Slide e materiale indicato nelle slide quando necessario.

## PARADIGMI DI PROGRAMMAZIONE

**Titolare:** Dott. MICHELE MAURO

**Periodo:** Il anno, 2 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 48A; 6,00

**Prerequisiti:**

Concetti di programmazione Object Oriented. Interazione con la linea di comando. È particolarmente utile aver seguito il corso di Programmazione ad Oggetti.

**Conoscenze e abilità da acquisire:**

Riconoscere le principali difficoltà legate alla costruzione di programmi e sistemi concorrenti e/o distribuiti; capire le limitazioni intrinseche dei vari approcci ed interpretare le garanzie fornite dai diversi strumenti in funzione dei requisiti del problema affrontato.

**Attività di apprendimento previste e metodologie di insegnamento:**

Lezioni frontali. Discussione in aula di un problemi e relative soluzioni, suddivise in passi e affrontate gradualmente, con interazione diretta insieme agli studenti.

**Contenuti:**

Il panorama dei problemi in cui l'informatica è usata come soluzione è ormai estremamente vasto; questo consente di individuare alcune larghe categorie le cui caratteristiche principali sono state studiate e su cui si è raccolta una cospicua esperienza. Usando il linguaggio Java e l'ecosistema della JVM come supporto, il corso illustra le difficoltà che si incontrano realizzando programmi e sistemi concorrenti e/o distribuiti, quali soluzioni si possono trovare in letteratura e sul mercato, e quali sono i criteri per selezionarle. Temi trattati: -Presentazione della JVM e del linguaggio Java; -collezioni ed operazioni sulle liste; -streams; -definizioni e problemi della concorrenza; threads; -definizioni e problemi della distribuzione; socket e channels; -cenni di stato distribuito; -reactive manifesto e tecnologie correlate.

**Modalità di esame:**

Esame scritto composto da domande a risposta multipla sugli argomenti teorici esposti a lezione e su problemi di valutazione immediata del comportamento di parti di codice. Svolgimento di esercizi completi su temi proposti a lezione.

**Criteri di valutazione:**

Chiarezza di intenti nel suddividere un problema in parti per risolvere ciascuna con il corretto metodo. Attenzione alla produzione di codice leggibile e in grado di comunicare chiaramente l'intento dell'autore.

**Testi di riferimento:**

Urma, Fusco, Mycroft, Modern Java in Action. : Manning, 2018

**Eventuali indicazioni sui materiali di studio:**

Nelle prime lezioni verranno indicati i software necessari per seguire il codice discusso a lezione. Questi comprenderanno sicuramente almeno una versione della piattaforma OpenJDK (17 o successive). Un IDE è consigliato (in ordine di preferenza: VSCode, Eclipse) ma non indispensabile.

## PROBABILITA' E STATISTICA

**Titolare:** Prof. MARKUS FISCHER

**Periodo:** Il anno, 1 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 32A+16E; 6,00

**Prerequisiti:**

Familiarità con concetti di base di analisi, algebra lineare e calcolo combinatorio. Gli insegnamenti "Analisi matematica" e "Algebra e matematica discreta" coprono i contenuti necessari alla comprensione dell'insegnamento.

**Conoscenze e abilità da acquisire:**

Lo studente dovrà acquisire la conoscenza degli strumenti di base del calcolo delle probabilità e della statistica inferenziale. Alla fine del corso l'allievo dovrà essere in grado di costruire semplici modelli probabilistici di fenomeni aleatori e di effettuare i relativi calcoli.

**Attività di apprendimento previste e metodologie di insegnamento:**

Lezioni frontali. Nel corso delle lezioni sono esposti gli aspetti teorici del corso, vengono illustrati esempi di applicazione e svolte esercitazioni. Vengono inoltre proposte esercitazioni settimanali da svolgere a casa con successiva illustrazione delle soluzioni.

**Contenuti:**

\*\*\* CALCOLO DELLE PROBABILITÀ \*\*\* Assiomi e conseguenze elementari degli assiomi. Esempi di spazi di probabilità: discreti, finiti, uniformi. Calcolo combinatorio elementare. Probabilità condizionata. Formule della probabilità totale e di Bayes. Eventi indipendenti. Variabili aleatorie discrete. Densità discreta di probabilità. Parametri riassuntivi: valore atteso, varianza e momenti. Esempi di variabili aleatorie discrete: Bernoulli, binomiali, geometriche e Poisson. Teorema limite di Poisson. Vettori aleatori discreti. Densità congiunte e marginali. Valore atteso di funzioni scalari di vettori aleatori. Covarianza. Variabili aleatorie discrete indipendenti. Variabili aleatorie assolutamente continue. Funzione di densità e funzione di distribuzione. Parametri riassuntivi: valore atteso e varianza. Esempi di variabili aleatorie assolutamente continue: uniformi, esponenziali e normali. Teoremi limite. Legge (debole) dei grandi numeri. Il metodo Monte Carlo. Teorema limite centrale. Approssimazione normale. \*\*\* STATISTICA \*\*\* Statistica descrittiva. Dati qualitativi e quantitativi, frequenze relative, metodi grafici di analisi dei dati. Indici di centralità, di dispersione e di forma. Statistica inferenziale. Stimatori puntuali. Correttezza e consistenza.

**Modalità di esame:**

Esame scritto (3 ore).

**Criteri di valutazione:**

Lo studente dovrà dimostrare di aver acquisito le conoscenze teoriche e di saperle applicare correttamente alla soluzione di problemi di calcolo delle probabilità e di statistica inferenziale di congrua difficoltà.

**Testi di riferimento:**

Bramanti, Marco, Calcolo delle probabilità e statistica. Teoria ed esercizi. Bologna: Progetto Leonardo, 1997 Finesso, Lorenzo, Lezioni di probabilità. Padova: Libreria Progetto, 2017 Caravenna, Francesco e Dai Pra, Paolo., Probabilità: Un'introduzione attraverso modelli e applicazioni.. Milano: Springer Milan, 2013

**Eventuali indicazioni sui materiali di studio:**

Tutti gli argomenti del corso vengono illustrati in aula. Gli appunti delle lezioni possono essere integrati dal libro di testo e dal materiale aggiuntivo reso disponibile sulla piattaforma moodle (referenze aggiuntive, appunti del docente, fogli di esercizi).

<b>PROGRAMMAZIONE</b>
-----------------------

**Titolare:** Prof. GIOVANNI DA SAN MARTINO

**Periodo:** I anno, 2 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 48A+24L; 9,00

**Prerequisiti:**

Nozioni di architettura degli elaboratori.

**Conoscenze e abilità da acquisire:**

Specificare un problema attraverso la formulazione di una pre- ed una post-condizione cui il programma che vogliamo costruire per risolvere il problema deve obbedire. La capacità di costruire un programma e dimostrare che fa quanto specificato nella sua pre- e post-condizione. Capacità di trovare anche soluzioni ricorsive ai problemi e di dimostrare la loro correttezza grazie ad una prova induttiva. Conoscenza delle nozioni di base della programmazione imperativa.

**Attività di apprendimento previste e metodologie di insegnamento:**

Si segue l'approccio secondo cui per imparare a programmare in modo consapevole, è necessario fare molti esercizi ricevendo feedback sulle soluzioni proposte. Quindi ogni settimana vengono assegnati esercizi che gli studenti sono invitati a risolvere. Gli esami si svolgono nel laboratorio informatico e seguono le stesse modalità degli esercizi settimanali.

**Contenuti:**

- Alcuni concetti sulla correttezza dei programmi alla Hoare, cioè basati sulle nozioni di pre-, post-condizione ed invarianti dei cicli. Ogni programma è accompagnato da una pre- e post-condizione e la sua correttezza rispetto ad esse va dimostrata in modo convincente. - Nozioni di base della programmazione imperativa. In particolare, tipi predefiniti, istruzioni semplici, puntatori, array, aritmetica dei puntatori, funzioni, funzioni ricorsive, memoria dinamica, liste concatenate ed alberi.

**Modalità di esame:**

L'esame consiste di una prova scritta con alcune domande teoriche ed un esercizio di programmazione. La parte di programmazione richiede di sviluppare un programma iterativo ed uno ricorsivo. Si richiede anche di specificare la correttezza delle funzioni prodotte. L'esame si svolge in laboratorio informatico e gli studenti ricevono l'assegnamento e lavorano direttamente sul PC.

**Criteri di valutazione:**

L'esame è fatto per mettere in rilievo la capacità di ragionare e di esprimere in forma chiara il proprio ragionamento, da parte dello studente. In particolare, si

valuta la capacità di specificare il problema da risolvere e di spiegare i motivi per cui la soluzione proposta effettivamente risolve il problema considerato. Sono apprezzate semplicità e chiarezza.

**Testi di riferimento:**

Filè, Gilberto, Programmazione consapevole (semplice è bello) 2015-2016. Padova: Libreria Progetto, 2015

**Eventuali indicazioni sui materiali di studio:**

Il corso usa un sito Moodle che raccoglie tutto il materiale del corso: le regole del corso, le diapositive delle lezioni, gli esercizi settimanali che gli studenti possono risolvere e testare interattivamente, e finalmente l'accesso alle registrazioni di tutte le lezioni del corso. Materiale di studio ed approfondimento, come link ad articoli e dispense, verranno distribuiti sulla piattaforma Moodle.

## PROGRAMMAZIONE AD OGGETTI

**Titolare:** Prof. FRANCESCO RANZATO

**Periodo:** Il anno, 1 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 48A+20E+12L; 10,00

**Sede dell'insegnamento:** Dipartimento di Matematica Pura ed Applicata

**Aule:** Aula LuM250

**Prerequisiti:**

Propedeuticità: Programmazione.

**Conoscenze e abilità da acquisire:**

Il corso mira ad introdurre la programmazione orientata agli oggetti in tutti i suoi aspetti, incluso lo sviluppo di un progetto software.

**Attività di apprendimento previste e metodologie di insegnamento:**

L'insegnamento prevede lezioni frontali (o in modalità telematica) e lo sviluppo di un progetto software di laboratorio.

**Contenuti:**

Il corso introduce la programmazione orientata agli oggetti utilizzando il linguaggio C++. Si tratteranno i seguenti argomenti principali. Tipi di dato astratti. Classi e oggetti. Campi dati e metodi. Parti private e pubbliche. Costruttori. Overloading. Distruttori. Metodi e classi friend. Classi collezione. Tecniche di condivisione controllata della memoria. Template di funzioni e di classe. Ereditarietà e gerarchie di classi. Metodi virtuali. Ereditarietà multipla e derivazione virtuale. Classi e gestione delle eccezioni. Cenni di design pattern. Uso di alcune librerie standard e ausiliarie: libreria STL e classi contenitore, libreria di I/O, librerie grafiche (ad esempio, Qt). Il corso prevede un laboratorio in cui gli studenti realizzeranno un progetto di programmazione ad oggetti usando gli strumenti introdotti nel corso.

**Modalità di esame:**

Esame scritto: quesiti sulla modellazione di problemi mediante programmi ad oggetti; quesiti sul comportamento dei programmi ad oggetti. Sviluppo di un progetto software orientato agli oggetti. Eventuale esame orale di discussione del progetto software.

**Criteri di valutazione:**

L'esame scritto verte su tutti gli argomenti del corso. Il progetto di laboratorio sarà sviluppato in C++ ed utilizzerà alcune librerie ad ampia diffusione. L'esame orale consiste in una discussione del progetto.

**Testi di riferimento:**

Francesco Ranzato, Appunti di programmazione ad oggetti. Padova: Libreria Progetto, Padova, 2012

## PROVA FINALE

**Titolare:** da definire

**Periodo:** III anno, 2 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** ; 3,00

**Sede dell'insegnamento:** Informazioni in lingua non trovate

**Aule:** Informazioni in lingua non trovate

**Contenuti:**

Il percorso di studi prevede la redazione di una relazione che riassume e discute in modo critico l'attività relativa allo stage. La prova finale consiste quindi nella presentazione e discussione di tale relazione di fronte alla Commissione per l'esame finale di Laurea.

**Testi di riferimento:**

CONTENUTO NON PRESENTE

## RETI DI CALCOLATORI

**Titolare:** Prof. MASSIMO MARCHIORI

**Periodo:** Il anno, 1 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 72A; 9,00

**Prerequisiti:**

E' opportuno avere familiarità con gli elementi di base della programmazione, così come forniti nel corso di "Programmazione" e con i contenuti del corso di "Sistemi Operativi".

**Conoscenze e abilità da acquisire:**

L'obiettivo principale del corso è quello di dare una panoramica a tutto tondo delle nozioni di base che costituiscono il grande mondo delle reti, cercando non solo di capire come sono fatte ma soprattutto perché funzionano così. Si analizzeranno quindi tutti i vari strati fondanti, a partire dallo strato fisico per salire via via agli strati superiori (data link, medium access control, network, transport, application). Si passerà poi al problema della sicurezza nelle reti, esponendone i concetti e le tecnologie fondamentali.

**Attività di apprendimento previste e metodologie di insegnamento:**

L'insegnamento prevede per la maggior parte lezioni frontali, ed una parte finale in laboratorio illustrativa delle problematiche applicate di sicurezza nelle reti.

**Contenuti:**

+ Introduzione al mondo delle reti Categorie e classificazioni delle reti, modelli di riferimento + Lo strato fisico Proprietà dei mezzi trasmissivi, dal cavo al wireless ai satelliti. + Lo strato data link Framing, tecniche di error-detection e di error-correction, flow control. + Il sottostrato medium access Protocolli ad accesso multiplo, ethernet, tecnologie wireless. + Lo strato network Tipi di connessioni, routing, qualità del servizio, internet. + Lo strato trasporto Protocolli di trasporto, internet. + Lo strato application Il Domain Name System. + Sicurezza Vulnerabilità, crittografia, chiavi simmetriche e pubbliche, firme digitali, communication e application security, autenticazione.

**Modalità di esame:**

Lo studente deve superare uno scritto. Sopra una certa soglia minima di punteggio lo studente può opzionalmente richiedere un ulteriore esame orale.

**Criteri di valutazione:**

Il criterio di valutazione principale è la comprensione delle tecnologie di rete e di sicurezza mostrate durante il corso. Questo significa quindi conoscere il funzionamento, i punti deboli ed i punti di forza delle tecnologie, la loro interazione nel contesto.

**Testi di riferimento:**

Andrew S. Tanenbaum, Computer Networks (4th edition) : Prentice Hall, 2002

**Eventuali indicazioni sui materiali di studio:**

Il materiale di studio è costituito principalmente dal libro di testo, con eventuali integrazioni fornite sottoforma di materiale online.

## RICERCA OPERATIVA

**Titolare:** Prof. LUIGI DE GIOVANNI

**Periodo:** III anno, 1 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 36A+12E+12L; 7,00

**Sede dell'insegnamento:** Padova

**Prerequisiti:**

Conoscenze di base di analisi matematica e algebra. E' propedeutico l'insegnamento di "Algebra e Matematica Discreta".

**Conoscenze e abilità da acquisire:**

Costruzione di modelli matematici per il supporto alle decisioni e relativi algoritmi, con particolare riferimento alla programmazione lineare nel continuo e nel discreto e all'ottimizzazione su grafi. Uso di pacchetti software per la soluzione di problemi di ottimizzazione.

**Attività di apprendimento previste e metodologie di insegnamento:**

L'insegnamento prevede lezioni frontali ed esercitazioni in laboratorio. Le esercitazioni in laboratorio consistono nell'implementazione in un linguaggio di modellazione algebrica di semplici modelli di programmazione lineare (mista intera).

**Contenuti:**

1. Problemi di ottimizzazione e modelli: modellazione e utilizzo di risolutori software in laboratorio. 2. Programmazione lineare: teoria e metodo del simplesso, teoria della dualità e applicazioni. 3. Ottimizzazione su grafi: modelli e algoritmi per il problema dell'albero di copertura di costo minimo, il problema del cammino minimo (algoritmi di Dijkstra e Bellman-Ford), il problema del flusso massimo (algoritmo di Ford-Fulkerson) e del flusso di costo minimo. 4. Elementi di Programmazione Lineare Intera e Ottimizzazione Combinatoria: metodi esatti (Branch-and-Bound), cenni su metodi euristici e metaeuristici (ricerca locale e varianti).

**Modalità di esame:**

L'esame è scritto, comprensivo di un problema da formulare con un modello di programmazione lineare, esercizi e domande di teoria. Se necessario, si terrà una discussione orale dello scritto. Il candidato può inoltre, a sua discrezione, preparare un mini-progetto facoltativo.

**Criteri di valutazione:**

L'esame scritto richiede lo svolgimento di esercizi per la valutazione del livello di apprendimento degli argomenti svolti (ad esempio, modellazione di un problema di ottimizzazione in programmazione lineare intera, applicazione dell'algoritmo del simplesso, applicazione di algoritmi di ottimizzazioni su rete, applicazione della teoria della dualità, applicazione dell'algoritmo del Branch-and-Bound, domande sui diversi argomenti svolti etc.)

**Testi di riferimento:**

CONTENUTO NON PRESENTE

**Eventuali indicazioni sui materiali di studio:**

Vengono rese disponibili delle dispense degli argomenti trattati a lezione e dei lucidi degli argomenti trattati in laboratorio, che contengono tutte le nozioni richieste all'esame. Gli studenti interessati possono approfondire gli argomenti sui seguenti testi: - D. Bertsimas, J. Tsitsiklis, Introduction to linear optimization, 1996, Athena Scientific. - R. K.Ahuja, T. L. Magnanti, J. B. Orlin "Network flows. Theory, algorithms, and applications", 1993, Prentice Hall. - L. A. Wolsey: "Integer programming", 1998, Wiley.

**SISTEMI OPERATIVI**

**Titolare:** Prof. CLAUDIO ENRICO PALAZZI

**Periodo:** I anno, 2 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 40A+24E+8L; 9,00

**Prerequisiti:**

Gli studenti dovrebbero preferibilmente avere una conoscenza generale delle Architetture dei Computer, così come fornita nel corso di "Architettura degli Elaboratori". Tuttavia, l'insegnamento non prevede propedeuticità.

**Conoscenze e abilità da acquisire:**

Questo corso introduce alle funzionalità di base dei moderni sistemi operativi. In particolare, il corso è diviso in tre parti principali. Nella prima, allo studente vengono presentati argomenti quali processi e thread, scambi di contesto, sincronizzazione, ordinamento e stallo. Nella seconda parte del corso, lo studente impara a conoscere problematiche e possibili soluzioni riguardanti la gestione della memoria quali, ad esempio, allocazione dinamica della memoria, memoria virtuale, paginazione e segmentazione. La terza parte del corso tratta i file system, inclusa la gestione di dischi e partizioni. Il corso termina con un'analisi delle scelte progettuali effettuate da sistemi operativi esistenti in commercio.

**Attività di apprendimento previste e metodologie di insegnamento:**

L'insegnamento prevede lezioni frontali, esercitazioni in aula e in laboratorio.

**Contenuti:**

Introduzione ai Sistemi Operativi. Gestione dei Processi: definizione, strutture, concorrenza, sincronizzazione, ordinamento, stallo. Gestione della Memoria: gerarchie, rilocalizzazione, strutture, memoria virtuale, paginazione, segmentazione. File System: architetture, struttura logica, modalità di accesso, directory, aspetti implementativi. Modelli e Architetture di Sistemi Operativi: discussione sulle scelte progettuali dei sistemi UNIX/Linux e dei sistemi Windows.

**Modalità di esame:**

Lo studente deve superare un esame scritto.

**Criteri di valutazione:**

Lo scritto contiene domande ed esercizi che consentono di valutare il livello di apprendimento delle nozioni discusse in classe e l'abilità dello studente nel maneggiare concetti in modo pratico.

**Testi di riferimento:**

A. S. Tanenbaum, Modern Operating Systems - 4th Edition. : Prentice Hall, 2014

**Eventuali indicazioni sui materiali di studio:**

Vengono rese disponibili le trasparenze utilizzate a lezione

**STAGE**

**Titolare:** Prof. TULLIO VARDANEGA

**Periodo:** III anno, 2 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** ; 11,00

**TECNOLOGIE OPEN-SOURCE**

**Titolare:** Dott. NICOLA BERTAZZO

**Mutuato da:** Laurea in Informatica (Ord. 2011)

**Periodo:** III anno, 2 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 48A; 6,00

**Prerequisiti:**

Nessuno.

**Conoscenze e abilità da acquisire:**

Fornire agli studenti un bagaglio di esperienza base per la gestione tecnologica di un progetto software e la definizione e l'implementazione di una continuous delivery pipeline.

**Attività di apprendimento previste e metodologie di insegnamento:**

Lezioni frontali e laboratorio.

**Contenuti:**

Saranno trattati i seguenti temi: - Issue Tracking - Source Code Management (SCM) - Testing Software - Processo di Build - Continuous integration e Continuous Delivery - Configuration Management

**Modalità di esame:**

Esercitazioni e prova orale

**Criteri di valutazione:**

La valutazione si baserà su colloqui (individuali o di gruppi), a valle di 4 assignment presentati e discussi ai laboratori.

**Testi di riferimento:**

CONTENUTO NON PRESENTE

**Eventuali indicazioni sui materiali di studio:**

Slide e materiale indicato nelle slide quando necessario.

<b>TECNOLOGIE WEB</b>
-----------------------

**Titolare:** Prof.ssa OMBRETTA GAGGI

**Periodo:** III anno, 1 semestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 40A+12E+20L; 9,00

**Prerequisiti:**

È opportuno avere familiarità con gli elementi di base della programmazione, così come forniti nei corso di "Programmazione" e "Programmazione ad oggetti". Gli studenti devono aver superato il corso di "Basi di Dati" e di "Programmazione".

**Conoscenze e abilità da acquisire:**

L'insegnamento intende presentare agli studenti il World-Wide Web e le tecnologie informatiche che lo caratterizzano. Ha lo scopo di fornire le conoscenze necessarie per la progettazione e lo sviluppo di siti web di qualità con l'uso delle tecnologie più avanzate. Gli studenti, oltre ad acquisire una conoscenza di alto livello dei vari tipi di tecnologie web esistenti, verranno formati a divenire sviluppatori di siti web basati sui i linguaggi standard. Verranno inoltre trattati aspetti dell'interattività sul web (linguaggi di script).

**Attività di apprendimento previste e metodologie di insegnamento:**

L'insegnamento prevede lezioni frontali, esercitazioni in laboratorio e la realizzazione di un progetto.

**Contenuti:**

1. Introduzione. Il concetto di ipertesto, il World Wide Web ed Internet. Gli enti di standardizzazione, le architetture Client-Server e i protocolli di Internet. 2. I linguaggi del web statico. I linguaggi XHTML e HTML5 e i fogli stile (il linguaggio CSS): formattazione del testo e la grafica su Web; links e navigazione. 3. Principi di web design. Architettura dell'informazione. Schemi Organizzativi e strutture per la navigazione. Progettazione dell'interfaccia. Emotional Design. 4. Accessibilità e legislazione. Tecniche per garantire l'accessibilità. Search Engine Optimization. 5. I linguaggi per il web dinamico (Programmazione su Internet) sia lato client che lato server. Il linguaggio Javascript. Il modello DOM per la gestione delle pagine via JavaScript. Il linguaggio PHP.

**Modalità di esame:**

Compito scritto e realizzazione di un progetto.

**Criteri di valutazione:**

Lo scritto contiene alcune domande che consentono di valutare il livello di apprendimento delle nozioni teoriche impartite durante il corso. Il progetto, svolto in gruppo, mira a valutare la capacità, da parte dello studente, di individuare un caso di studio adeguato, e di progettare e realizzare un sito web sia per quanto riguarda la parte di backend che di frontend.

**Testi di riferimento:**

CONTENUTO NON PRESENTE

**Eventuali indicazioni sui materiali di studio:**

I lucidi del corso e il materiale dei laboratori sono messi a disposizione sul sito web del corso.