



UNIVERSITÀ  
DEGLI STUDI  
DI PADOVA



**Bollettino Notiziario - A.A. 2013/2014**

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

### **Curriculum: Corsi comuni**

### **ALGEBRA E GEOMETRIA**

**Titolare:** Prof.ssa ELOISA MICHELA DETOMI

**Periodo:** I anno, 1 trimestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 48A+32E; 10,00

**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, l'algebra lineare e la geometria.

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

Lezioni frontali ed esercitazioni.

**Contenuti:**

Numeri naturali, interi, razionali, reali, complessi. Relazioni di equivalenza. Cardinalità: insiemi finiti e infiniti. Massimo comun divisore e algoritmo di Euclide; anelli di classi resto. Induzione; definizioni e conti per induzione. Richiami sui polinomi: divisione, zeri, fattorizzazione in irriducibili (sui reali e sui complessi). Vettori nel piano e nello spazio ordinario; rappresentazione cartesiana di rette e piani. 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. Cenni a forme quadratiche.

**Modalità di esame:**

Lo studente deve superare uno scritto; l'eventuale orale e' a discrezione del docente.

**Criteri di valutazione:**

Lo scritto prevede nella prima parte una serie di domande volte a verificare che lo studente conosca e sappia applicare in casi semplici i contenuti teorici del corso (conoscenze minime di base). La seconda parte dello scritto prevede esercizi volti a valutare il livello di apprendimento delle nozioni impartite durante il corso e la capacità di elaborarle ed applicarle.

**Testi di riferimento:**

Marco Abate e Chiara de Fabritiis,, Geometria analitica con elementi di algebra lineare. : McGraw-Hill, J.F. Humphreys e M. Prest, Numbers, Groups and Codes. : Cambridge University Press,

**Eventuali indicazioni sui materiali di studio:**

Vengono rese disponibili dispense sulla prima parte del corso. Per la seconda parte, come alternativa al libro di testo, si veda anche il W.K. Nicholson, Algebra lineare, McGraw-Hill.

**ALGORITMI E STRUTTURE DATI**

**Titolare:** Prof. LIVIO COLUSSI

**Periodo:** Il anno, 1 trimestre

**Indirizzo formativo:** Corsi comuni

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

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

**Aule:** LuM250

**Prerequisiti:**

Programmazione.

**Conoscenze e abilità da acquisire:**

Il corso ha come finalità quella di fornire un'introduzione agli algoritmi ed 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 consente inoltre di sviluppare nello studente una sensibilità per la realizzazione di programmi efficienti e corretti.

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

Lezioni ed esercitazioni in aula.

**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, RadixSort e BucketSort. Algoritmo di selezione in tempo medio lineare. - 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. - Strutture Dati Avanzate B-alberi. Heap binomiali. Strutture dati per insiemi disgiunti.

**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, C.E.Leiserson, R.L.Rivest, C.Stein., Introduzione agli Algoritmi e Strutture Dati. (terza edizione). : McGraw-Hill Italia.,

**Eventuali indicazioni sui materiali di studio:**

I lucidi delle lezioni e una dispensa di esercizi svolti sono disponibili nel sito del docente.

**ANALISI MATEMATICA**

**Titolare:** Dott. FRANCESCO PAOLO MONTEFALCONE

**Periodo:** I anno, 2 trimestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 48A+32E; 10,00

**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.

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

Lezioni ed esercitazioni in classe

**Contenuti:**

Numeri reali. Successioni in 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. Alcune generalizzazioni dell'Analisi per funzioni di più variabili. Serie numeriche e di funzioni. Equazioni differenziali del primo ordine.

**Modalità di esame:**

Scritto e Orale

**Criteri di valutazione:**

Comprensione degli argomenti teorici e capacità di risolvere esercizi

**Testi di riferimento:**

CONTENUTO NON PRESENTE

**Eventuali indicazioni sui materiali di studio:**

Dispense del docente

## ARCHITETTURA DEGLI ELABORATORI

**Titolare:** Prof. ALESSANDRO SPERDUTI

**Periodo:** I anno, 1 trimestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 40A+20E+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.

**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:**

William Stallings, Architettura e organizzazione dei calcolatori. : Pearson Education, 2010

**Eventuali indicazioni sui materiali di studio:**

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

## AUTOMI E LINGUAGGI FORMALI

**Titolare:** Prof.ssa FRANCESCA ROSSI

**Periodo:** Il anno, 1 trimestre

**Indirizzo formativo:** Corsi comuni

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

**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, mostrando la loro applicazione ai compilatori. Inoltre, introduce le nozioni di 'indecidibilità' e 'intrattabilità'.

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

Lezioni frontali ed esercizi in classe.

**Contenuti:**

Gli argomenti principali del corso sono: Parte 1: linguaggi regolari e analisi lessicale (3 cfu) -- automi a stati finiti -- espressioni e linguaggi regolari -- analisi lessicale Parte 2: linguaggi liberi da contesto e analisi sintattica (3 cfu) -- grammatiche e linguaggi liberi dal contesto -- automi a pila -- analisi sintattica: parsers top-down (LL) e bottom-up (LR) Parte 3: 'indecidibilità' e 'intrattabilità' (2 cfu) -- macchine di Turing -- concetto di 'indecidibilità' -- problemi intrattabili -- classi P e NP

**Modalità di esame:**

Scritto e orale.

**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:**

J. E. Hopcroft, R. Motwani, J. D. Ullman, *Automi, Linguaggi e calcolabilità*. : Addison Wesley, 2003 Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman., *Compilers: Principles, Techniques, and Tools (2nd Edition)*. : Addison-Wesley, 2006

**Eventuali indicazioni sui materiali di studio:**

Vengono rese disponibili le trasparenze usate a lezione.

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

**Titolare:** Prof. PAOLO BALDAN

**Periodo:** Il anno, 3 trimestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 52A+8E+20L; 10,00

**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) e della relativa interfaccia web. 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 ed interfacce web in PHP. Sono propedeutiche alla realizzazione del progetto, nel quale lo studente individua autonomamente un caso di studio, progetta ed implementa una base di dati e realizza un'applicazione PHP che la utilizza.

**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. - Il linguaggio SQL Il data definition language di SQL. Il data manipulation language di SQL (interrogazione e aggiornamento). Conoscenza procedurale: procedure e trigger. Transazioni. Controllo degli accessi (basato sui privilegi). Il DBMS MySQL. - SQL per le applicazioni SQL embedded. Call level interface (JDBC, ODBC). Accesso tramite web: PHP e pagine web dinamiche. Mantenimento dello stato. Autenticazione.

**Modalità di esame:**

Lo studente deve superare uno scritto e realizzare un progetto nel quale mettere in pratica le nozioni acquisite nel corso. Il progetto è poi discusso in forma orale.

**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:**

A. Albano, G. Ghelli, R. Orsini, *Fondamenti di basi di dati*. 2a Edizione. : Zanichelli, 2005 R. Elmasri, S. Navathe, *Sistemi di basi di dati*. Fondamenti. 5a Edizione. : Pearson/Addison Wesley, 2007

**Eventuali indicazioni sui materiali di studio:**

Vengono rese disponibili, come riferimento, le trasparenze utilizzate a lezione. Pagina web: <http://www.math.unipd.it/~baldan/BD>

**C.I. DI INGEGNERIA DEL SOFTWARE**

**Titolare:** Prof. TULLIO VARDANEGA

**Indirizzo formativo:** Corsi comuni

**Prerequisiti:**

L'insegnamento assume e richiede familiarità con i principali linguaggi e tecniche di programmazione presentati nel triennio (C, C++; programmazione imperativa e a oggetti), e con le funzionalità fondamentali delle base di dati e delle più comuni tecnologie basate su SQL. La criticità di tali conoscenze preliminari è elevata al punto da formare vincolo di propedeuticità per gli insegnamenti Programmazione a oggetti e Basi di dati.

**Conoscenze e abilità da acquisire:**

Operando in stretto coordinamento con il modulo B dello stesso insegnamento, il corso si propone di erogare la formazione essenziale che consenta agli studenti di aspirare al ruolo professionale di "software engineer". La formazione sarà erogata sia sul piano teorico-fondazionale che su quello pratico-esperienziale, tramite lo svolgimento di un impegnativo progetto di gruppo attraverso tutte le fasi principali del suo ciclo di vita, dalla risposta a una gara di appalto fino alla revisione di accettazione e collaudo.

**Modalità di esame:**

L'esame finale si tiene alla fine del modulo B del corso e consta di una prova scritta individuale che incide per il 40% della valutazione finale. Il rimanente 60% della valutazione risulta dalla media di voto conseguita in revisioni intermedie di progetto sostenute dal gruppo di appartenenza dello studente (da gara di appalto a collaudo).

**Criteri di valutazione:**

Lo scritto individuale consiste di 3 domande pratiche di tipo "problem solving" e 3 domande teoriche che mirano a valutare il livello complessivo di apprendimento raggiunto dallo studente delle nozioni impartite durante il corso. Le attività di progetto didattico, svolte in gruppi, e sviluppate nell'arco di un periodo ampio da un minimo di un trimestre a un massimo di un semestre, consentono di verificare la maturazione individuale degli studenti in relazione alle problematiche tecniche, gestionali e relazionali proprie delle competenze richieste a un software engineer.

**Moduli del C.I.:**

Ingegneria del Software (Mod. A)

Ingegneria del Software (Mod. B)

**INGEGNERIA DEL SOFTWARE (MOD. A)**

**Titolare:** Prof. TULLIO VARDANEGA

**Periodo:** III anno, 1 trimestre

**Indirizzo formativo:** Corsi comuni

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

**Contenuti:**

- Genesi e obiettivi del software engineering. Ciclo di vita del software e processi di ciclo di vita. - Gestione di progetto: pianificazione, analisi e mitigazione dei rischi, gestione del tempo e delle risorse. - Amministrazione di progetto: pianificazione, installazione e manutenzione dell'infrastruttura tecnica di supporto al lavoro di progetto, sia per la parte di gestione che per quella di sviluppo. - Elementi di UML: diagrammi dei casi d'uso; diagrammi delle classi e dei package; diagrammi di attività e di sequenza. - Metodi di sviluppo consapevole: come la programmazione deve essere conseguenza di analisi e progettazione; come contenere i costi di verifica. - Obiettivi e tecniche di verifica e di validazione.

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

L'attività formativa del modulo A si svolge prevalentemente mediante una serie di lezioni d'aula nelle quali vengono illustrate le principali aree di conoscenza della disciplina del software engineering. Tali lezioni sono accompagnate da attività di esercitazione, inizialmente a volte in aula, alla presenza del docente, e successivamente svolte in autonomia dagli studenti, incentrate sulla esplorazione pratica di alcune delle problematiche illustrate a lezione. Nel corso del trimestre il docente pubblica un bando di appalto formale per lo svolgimento di alcune tipologie di progetto software. Il bando prevede condizioni tecniche, temporali ed economiche. Gli studenti sono chiamati a costituirsi in gruppo di progetto ("impresa virtuale") e formulare una proposta per la partecipazione alla gara di appalto. Il trimestre si chiude con la valutazione e selezione delle proposte e la formalizzazione dei conseguenti impegni contrattuali per i gruppi partecipanti. La transizione tra il modulo A e il modulo B viene sancita con una revisione formale delle proposte di progetto preparate dagli studenti costituitisi in gruppo. L'esito di tale revisione formale produce punteggio per la valutazione finale.

**Eventuali indicazioni sui materiali di studio:**

Il docente pubblica regolarmente tutte le diapositive utilizzate a lezione e anche materiale supplementare utile per l'approfondimento dei temi trattati in aula.

**Testi di riferimento:**

Ian Sommerville, Software Engineering. : Pearson Education | Addison-Wesley, 2010 E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns. : Pearson Education | Addison-Wesley, 1994

**INGEGNERIA DEL SOFTWARE (MOD. B)**

**Titolare:** Dott. RICCARDO CARDIN

**Periodo:** III anno, 2 trimestre

**Indirizzo formativo:** Corsi comuni

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

**Contenuti:**

L'attività formativa del modulo B si svolge prevalentemente con una serie di lezioni d'aula nelle quali vengono illustrate tecniche e metodologie utili ad affrontare le attività pratiche dello sviluppo di progetto software. Vengono approfonditi i Design Pattern e presentati alcuni strumenti concreti che aiutano nello sviluppo software di tipo ingegneristico (Spring framework, Hibernate, strumenti di Continuous Integration). Una parte significativa di tali lezioni si svolge in forma di esercitazione nella quale gli studenti si raccolgono nei rispettivi gruppi di progetto e affrontano le problematiche proposte dal docente, dialogando ripetutamente con esso. Alcune lezioni sono infine dedicate allo svolgimento di revisioni formali sullo stato di avanzamento del progetto didattico nella realizzazione dei vari gruppi partecipanti

**Eventuali indicazioni sui materiali di studio:**

Il corso consta di lezioni frontali nelle quali si introducono strumenti tecnici e metodologici per la realizzazione del progetto didattico, alternate con lezioni pratiche nelle quali il docente discute con i gruppi di progetto, specifici aspetti del lavoro assegnato al gruppo.

**Testi di riferimento:**

E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns.. : Addison-Wesley, Pearson Education Italia, 2000 Ian Sommerville, Software Engineering (8th edition). : Pearson Education, Addison-Wesley, 2007

## CALCOLO NUMERICO

**Titolare:** Prof. MARCO VIANELLO

**Periodo:** Il anno, 3 trimestre

**Indirizzo formativo:** Corsi comuni

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

**Sede dell'insegnamento:** Padova.

**Prerequisiti:**

Analisi matematica, Algebra e geometria

**Conoscenze e abilità da acquisire:**

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

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

Lezioni frontali ed esercitazioni in 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 - Complessità computazionale per esempi: schema di Horner per polinomi, calcolo rapido di una potenza tramite codifica binaria dell'esponente, calcolo della funzione exp, calcolo del determinante con il metodo di eliminazione gaussiana - 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 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 della matrice inversa, fattorizzazione QR, soluzione ai minimi quadrati di sistemi sovradeterminati - Laboratorio: implementazione e applicazione di codici numerici

**Modalità di esame:**

Prova scritta ed eventuale prova orale.

**Criteri di valutazione:**

Prova orale per risultati nell'intervallo 18-23 nello scritto, o per scelta dello studente con voto > 23 nello scritto.

**Testi di riferimento:**

A. Quarteroni, F. Saleri, Introduzione al calcolo scientifico. : Springer, A. Quarteroni, F. Saleri, Scientific computing with Matlab and Octave. : Springer,

## FISICA

**Titolare:** Prof. GIOVANNI ZANELLA

**Periodo:** Il anno, 3 trimestre

**Indirizzo formativo:** Corsi comuni

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

**Sede dell'insegnamento:** Padova

**Prerequisiti:**

Analisi Matematica.

**Conoscenze e abilità da acquisire:**

Scopo del corso è quello di fornire allo studente, dopo un propedeutico richiamo dei principi fondamentali della fisica, conoscenze di base della fisica dei dispositivi elettronici e delle loro principali applicazioni circuitali, nonché della teoria dei segnali e del rumore.

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

Lezioni in aula.

**Contenuti:**

- Dimensioni delle grandezze fisiche. Unità di misura. Errori di misura. Elementi di calcolo vettoriale. - Cinematica del punto materiale. Forze e riferimenti. Dinamica dei sistemi di più punti materiali. - Elettrostatica. Conduzione elettrica. Magnetostatica. Elettrodinamica. Elementi di elettrotecnica. Fenomeni ondulatori. - Elementi di fisica dei semiconduttori e delle loro principali applicazioni circuitali. - Teoria dei segnali e del rumore.

**Modalità di esame:**

Esame scritto e successivo orale.

**Criteri di valutazione:**

L'esame consiste di domande aperte ed esercizi finalizzati a valutare il livello di preparazione dello studente e la sua capacità di applicare i concetti acquisiti a problemi reali.

**Testi di riferimento:**

CONTENUTO NON PRESENTE

**Eventuali indicazioni sui materiali di studio:**

Dispense del docente.

**LINGUA INGLESE**

**Titolare:** Prof. MASSIMO MARCHIORI

**Periodo:** I anno, 3 trimestre

**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/>

**Prerequisiti:**

CONTENUTO NON PRESENTE

**Conoscenze e abilità da acquisire:**

CONTENUTO NON PRESENTE

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

CONTENUTO NON PRESENTE

**Contenuti:**

CONTENUTO NON PRESENTE

**Modalità di esame:**

CONTENUTO NON PRESENTE

**Criteri di valutazione:**

CONTENUTO NON PRESENTE

**Testi di riferimento:**

CONTENUTO NON PRESENTE

**Eventuali indicazioni sui materiali di studio:**

CONTENUTO NON PRESENTE

**LOGICA**

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

**Periodo:** I anno, 3 trimestre

**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 la matematica. In particolare, lo studente dovrà 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 dei seguenti concetti: alcuni di carattere generale come linguaggio, espressione, proposizione, asserzione, metalinguaggio; alcuni specifici della matematica quali derivazione, dimostrazione, sistema assiomatico, induzione, indipendenza, interpretazione. Lo studente raggiungerà padronanza di tali concetti e sarà quindi in grado di riconoscerli ed applicarli nella matematica ed 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 cenni sulla storia della logica e sulle 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:**

Contenuti formativi: 1. Linguaggio, segni e espressioni, simboli e proposizioni, asserzioni e dichiarazioni, metalinguaggio, livelli di riferimento, iterazione infinita. 2. Concetto di macchina o robot, significato dei connettivi e loro regole di deduzione, logiche delle risorse, regole strutturali, logica intuizionistica e logica classica, tavole di verità, funzioni proposizionali e sottoinsiemi, quantificatori e loro regole di deduzione. 3. Metodi di decisione per calcoli dei sequenti proposizionali (classico e intuizionista). 4. Analisi dettagliata di un esempio pratico, ordini parziali, numeri naturali e teoria assiomatica dell'aritmetica di Peano, strutture dell'algebra astratta. 5. Definizioni e dimostrazioni per induzione, termini e formule, interpretazione delle formule, validità. 6. Cenni ai teoremi di completezza e incompletezza (Goedel) e di indecidibilità (Church) e loro significato.

**Modalità di esame:**

Esame Scritto

**Testi di riferimento:**

Maria Emilia Maietti, Manuale pratico di Logica. Padova: , 2013

**Eventuali indicazioni sui materiali di studio:**

Verranno fornite dispense con teoria ed esercizi riguardanti ogni argomento del corso.

**MATEMATICA DISCRETA E PROBABILITA'**

**Titolare:** Dott.ssa CARLA DE FRANCESCO

**Periodo:** I anno, 3 trimestre

**Indirizzo formativo:** Corsi comuni

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

**Prerequisiti:**

Lo studente deve essere familiare con i contenuti del corso di Analisi Matematica. L'insegnamento non prevede propedeuticità.

**Conoscenze e abilità da acquisire:**

Conoscere i principali concetti di teoria dei grafi. Essere in grado di risolvere elementari problemi di conteggio su strutture discrete. Conoscere le nozioni di base del Calcolo delle Probabilità, nel caso discreto. Possedere una conoscenza operativa delle leggi limite classiche: legge dei grandi numeri e del teorema limite centrale.

**Contenuti:**

- Teoria dei grafi: introduzione ai grafi, isomorfismi tra grafi, grafi planari, cicli euleriani e circuiti hamiltoniani, alberi. - Enumerazione: permutazioni e combinazioni semplici, permutazioni e combinazioni con ripetizione, distribuzioni, identità binomiali e triangolo di Pascal Tartaglia, relazioni di ricorrenza. - Probabilità: spazi di probabilità discreti, probabilità condizionata, formule delle probabilità totale e di Bayes, variabili aleatorie discrete e continue, valore atteso e varianza, v.a. notevoli: Bernoulli, binomiale, geometrica, Poisson, uniforme, esponenziale, normale. Disuguaglianza di Markov e di Chebyshev, legge dei grandi numeri, teorema del limite centrale e approssimazione normale.

**Modalità di esame:**

Prova scritta a libro chiuso a fine corso: 100%. La prova scritta consiste di domande teoriche (a risposta chiusa o aperta) e di esercizi.

**Criteri di valutazione:**

L'esame contiene esercizi e domande di teoria volte a valutare il livello di apprendimento e di elaborazione personale dei concetti illustrati durante il corso.

**Testi di riferimento:**

Ross, S., Calcolo delle Probabilità (2<sup>a</sup> edizione). : Apogeo, 2008 Alan Tucker, Applied Combinatorics. : Wiley and Sons, 2007

**Eventuali indicazioni sui materiali di studio:**

L'insegnamento prevede lezioni frontali, esercizi per casa e successiva discussione in aula degli stessi.

**PROGRAMMAZIONE**

**Titolare:** Prof. GILBERTO FILE'

**Periodo:** I anno, 2 trimestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 60A+20L; 10,00

**Sede dell'insegnamento:** Edificio Paolotti

Aule: Aula Lum 250, via Luzzatti 5

**Prerequisiti:**

Qualche conoscenza 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 in C++ 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: comandi semplici, puntatori, array, funzioni, passaggio dei parametri per valore e riferimento, tipi ad-hoc, eccezioni.

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

Il corso consiste principalmente di lezioni classiche in aula. Comunque circa 24 ore del corso (su 80 ore totali) vengono spese nel laboratorio informatico. I primi appuntamenti, concentrati nei primi 10 giorni del corso, permettono agli studenti di accrescere la loro capacità di usare in modo consapevole un PC. Nel resto dei laboratori gli studenti ricevono esercizi di difficoltà crescente e cercano di risolverli fornendo anche prove di correttezza delle loro soluzioni. La modalità di queste esercitazioni è completamente analoga a quella degli esami e questo facilita l'esecuzione degli esami. Il corso ha un sito di elearning moodle che permette un nutrito scambio di informazioni tra docenti e studenti.

**Contenuti:**

I contenuti del corso si pongono su due piani diversi: 1) Da una parte vengono insegnati alcuni concetti della 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. 2) Contemporaneamente alla parte (1) le nozioni di base della programmazione imperativa sono insegnate. In particolare, tipi predefiniti, istruzioni semplici, puntatori, array, funzioni, funzioni ricorsive, memoria dinamica, liste concatenate ed alberi binari.

**Modalità di esame:**

L'esame consiste fondamentalmente di una prova scritta. In quest prova ci sono domande teoriche e si richiede di sviluppare un programma iterativo ed uno ricorsivo. Si richiede anche qualche passaggio relativo alla correttezza dei programmi prodotti. L'esame si svolge in laboratorio informatico e gli studenti ricevono l'assegnamento e lavorano direttamente sul PC. Un esame orale può venire richiesto in casi speciali.

**Criteri di valutazione:**

L'esame è fatto per mettere in rilievo la capacità di ragionare dello studente e di esprimere in forma chiara il proprio ragionamento. In particolare si valuta la capacità di specificare il problema da risolvere e di realizzare un programma semplice per risolvere il problema specificato. Viene valutata anche la capacità di spiegare perché la soluzione proposta effettivamente è giusta per risolvere il problema proposto. Sono apprezzate semplicità e chiarezza.

**Testi di riferimento:**

Gilberto Filè, Programmazione consapevole. Padova: Progetto, 2012

**Eventuali indicazioni sui materiali di studio:**

Il corso usa un sito moodle di elearning che raccoglie molto materiale utile per il corso: le slide delle lezioni, gli esercizi in laboratorio, esami passati con e senza soluzioni, ecc.

**PROGRAMMAZIONE AD OGGETTI**

**Titolare:** Prof. FRANCESCO RANZATO

**Periodo:** Il anno, 2 trimestre

**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 e lo sviluppo di un progetto software di laboratorio in modo indipendente.

**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. 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, esame orale, sviluppo di un progetto software orientato agli oggetti.

**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:**

## PROGRAMMAZIONE CONCORRENTE E DISTRIBUITA

**Titolare:** Prof.ssa SILVIA CRAFA

**Periodo:** III anno, 1 trimestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 30A+18E; 6,00

**Sede dell'insegnamento:** Padova

**Prerequisiti:**

Conoscenze di programmazione e di programmazione ad oggetti.

**Conoscenze e abilità da acquisire:**

Il corso presenta il linguaggio Java e la programmazione a diversi livelli di astrazione: la programmazione ad oggetti, la programmazione concorrente, e la programmazione distribuita.

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

Il corso prevede lezioni teoriche, lezioni di esercitazioni, e la realizzazione di un progetto che prevede l'uso di tutte le caratteristiche del linguaggio Java viste nel corso.

**Contenuti:**

1. Programmazione ad oggetti: - classi, oggetti, ereditarietà, polimorfismo - organizzazione delle classi: classi astratte, interfacce, classi interne (statiche, di istanza, anonime, innestate in interfacce) - grafica e gestione degli eventi. 2. Programmazione concorrente: thread, scheduling, accesso sincronizzato a dati condivisi, comunicazione tra thread. 3. Programmazione distribuita: stream e serializzazione, socket, RMI

**Modalità di esame:**

L'esame consiste in una prova scritta seguita da una discussione orale del progetto obbligatorio. Il progetto consiste in un'applicazione distribuita.

**Criteri di valutazione:**

La prova scritta valuta l'apprendimento del linguaggio java e la capacità di realizzare soluzioni corrette per problemi di natura concorrente. La prova orale valuta non solo la correttezza e la funzionalità dell'applicazione distribuita realizzata, ma anche la capacità dello studente di illustrare il programma e giustificare le scelte di base.

**Testi di riferimento:**

Silvia Crafa, Programmazione Concorrente e Distribuita.. Padova: Edizioni Cortina, 2011

## PROVA FINALE

**Titolare:** da definire

**Periodo:** III anno, 3 trimestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** ; 3,00

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

**Aule:** Informazioni in lingua non trovate

**Prerequisiti:**

CONTENUTO NON PRESENTE

**Conoscenze e abilità da acquisire:**

CONTENUTO NON PRESENTE

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

CONTENUTO NON PRESENTE

**Contenuti:**

CONTENUTO NON PRESENTE

**Modalità di esame:**

CONTENUTO NON PRESENTE

**Criteri di valutazione:**

CONTENUTO NON PRESENTE

**Testi di riferimento:**

CONTENUTO NON PRESENTE

**Eventuali indicazioni sui materiali di studio:**

CONTENUTO NON PRESENTE

## RETI E SICUREZZA

**Titolare:** Prof. MASSIMO MARCHIORI

**Periodo:** Il anno, 2 trimestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 72A+8L; 10,00

## RICERCA OPERATIVA

**Titolare:** Prof. LUIGI DE GIOVANNI

**Periodo:** III anno, 1 trimestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 28A+12E+8L; 6,00

**Sede dell'insegnamento:** Padova

**Prerequisiti:**

Conoscenze di base di analisi matematica e algebra.

**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:**

Scritto, con eventuali orale e/o discussione di un mini-progetto.

**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: - M. Fischetti, Lezioni di Ricerca Operativa, 1999, Libreria Progetto Padova. - 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, 3 trimestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 32A+24E+8L; 8,00

**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, rilocazione, 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:**

È 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 - 3rd Edition. : Prentice Hall, 2008

**Eventuali indicazioni sui materiali di studio:**

Vengono rese disponibili le trasparenze utilizzate a lezione

**STAGE**

**Titolare:** da definire

**Periodo:** III anno, 3 trimestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** ; 11,00

**TECNOLOGIE WEB**

**Titolare:** Prof.ssa OMBRETTA GAGGI

**Periodo:** III anno, 2 trimestre

**Indirizzo formativo:** Corsi comuni

**Tipologie didattiche:** 40A+8E+15L; 8,00

**Prerequisiti:**

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

**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 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 e la tecnologia XML. 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. Il linguaggio XHTML 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. Accessibilità e legislazione. Tecniche per garantire l'accessibilità. Search Engine Optimization. 4. Il linguaggio XML. EXtensible Markup Language (XML), i linguaggi per la definizione di uno schema (DTD e XMLSchema), cenni al reperimento dati (XPath) e introduzione ai fogli di trasformazione di stile per XML (XSLT). 5. I linguaggi per il web dinamico (Programmazione su Internet). Il linguaggio Javascript. Il modello DOM per la gestione delle pagine via JavaScript. Il linguaggio Perl. Il modulo Common Gateway Interfaces (CGI). Le librerie LibXML e LibXSLT.

**Modalità di esame:**

Lezioni frontali, esercitazioni in laboratorio, 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, in particolare relativamente alle tecnologie XML. 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:**

Joel Sklar, Principi di Web Design. : Apogeo, 2012 Patrick Griffiths, XHTML & CSS. Il web secondo HTML Dog. : Pearson Education, 2007

**Eventuali indicazioni sui materiali di studio:**

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