REVISIONE 1.1 – 12/10/2016: modifica codici stage e prova finale

GUIDA AL CORSO DI LAUREA IN **INFORMATICA** (d.m. 270) E MANIFESTO DEGLI STUDI

Corso di Laurea in Informatica
Dipartimento di Informatica
Università degli Studi di Torino

Via Pessinetto 12 - Torino

Anno Accademico 2016/2017

Prefazione

Gentile lettrice, gentile lettore,
questa è la guida del corso di Laurea in Informatica (secondo il DM 270, classe di laurea L31), ed è organizzata in due parti, come da indicazioni ministeriali. La prima parte è la descrizione del corso di Laurea in Informatica (DM 270, classe di laurea L31), questa descrizione costituisce il "Manifesto degli studi - Piano dell'Offerta Formativa sui tre anni per la coorte 2016", mentre la seconda parte descrive tutti gli insegnamenti offerti per il 2016/2017 (per tutte le coorti attive), elencando per ogni insegnamento i docenti di riferimento, il programma di esame, le modalità di esame ed altre informazioni utili. La prima parte è quindi di interesse precipuamente per gli studenti che si immatricolano quest'anno (coorte 2016), che trovano nel Manifesto la descrizione del loro percorso triennale, mentre la seconda parte è di interesse anche per gli studenti delle due coorti precedenti (2014 e 2015) che devono seguire gli insegnamenti del terzo e secondo anno, rispettivamente. Ricordiamo che per gli studenti immatricolati negli anni precedenti vale il manifesto degli studi della loro coorte, già pubblicato negli anni precedenti.
Le attività del Corso di Laurea (CdL) di Informatica dell'Università di Torino sono coordinate dal Consiglio di Corso di Studi (CCS), composto dai docenti che insegnano nel CdL e dai rappresentanti degli studenti, periodicamente eletti. Le attività del CCS sono normate dal ``Regolamento del Corso di Laurea in Informatica''.

Di seguito alcune note riassuntive, con le principali informazioni di interesse:

1. Tutto quanto è scritto in questa guida (e nei successivi, eventuali aggiornamenti alla guida stessa che verranno pubblicati sul sito del Corso di Laurea) è vincolante per gli studenti della coorte 2016 e per il corso di laurea e il suo corpo docente. La guida, unitamente ai regolamenti e alle scadenze di Ateneo per tasse, piano carriera e esame di laurea, costituisce la base del patto "studenti-Università": conoscerla ed averne compreso i contenuti è un obbligo dello studente.
2. Il Corso non è a numero programmato, ma tutti gli studenti devono sostenere e superare il TARM (test di verifica dei requisiti minimi) per poter sostenere gli esami dei singoli insegnamenti. Si consiglia di **effettuare il TARM prima di iscriversi**, perché tale test è utile anche come sistema di autovalutazione delle proprie capacità rispetto al corso di studio. Chi non sostiene e non supera il TARM entro dicembre si vedrà assegnato un obbligo formativo aggiuntivo. Leggete con cura la parte della guida che spiega motivazioni, modalità e sanzioni (paragrafo su "Requisiti di ammissione").
3. Per tutto quanto riguarda la procedura di immatricolazione, pagamento tasse, supporto agli studi, assegnazione login di Ateneo per l'accesso alle procedure on-line di iscrizione, e molto altro, potete far riferimento alla pagina principale del sito di Ateneo, selezionando poi il profilo "futuro studente". Tutti i servizi agli studenti (segreteria studenti, pagamento tasse e altro) sono supportati da un servizio di help-desk a cui gli studenti sono invitati a rivolgersi prima di andare allo sportello o di telefonare. L'help-desk fornisce una risposta scritta e quindi può essere uno strumento utile per risolvere eventuali incomprensioni
4. Tutte le informazioni non riportate in questa guida (come i programmi dettagliati degli insegnamenti, gli orari e la localizzazione delle aule) sono reperibili dal sito Internet del Corso di Laurea, all'indirizzo http://laurea.educ.di.unito.it/
5. Il corso di Laurea in Informatica è uno dei corsi di Laurea della Scuola di Scienze della Natura dell'Università di Torino

6. La durata della Laurea è di tre anni accademici. I primi due anni sono uguali per tutti gli studenti: gli insegnamenti dei primi due anni sono suddivisi fra corso A e corso B e gli insegnamenti di laboratorio sono poi ulteriormente suddivisi, per permettere un adeguato rapporto docente/studenti. Il terzo anno è invece articolato in tre diversi curricula, al fine di permettere allo studente una maggiore personalizzazione degli studi.
7. La Laurea in Informatica dà accesso alla Laurea Magistrale in Informatica, della durata di due anni accademici, laurea che, al momento, è anch'essa strutturata in tre curricula.
8. La Laurea dà anche accesso ai corsi di master di primo livello, normalmente della durata di un anno. I master hanno un obiettivo più spiccatamente professionalizzante e vengono organizzati in collaborazione con aziende del settore e altri enti formativi, e la loro offerta varia di anno in anno.
9. Il termine "Corso di Laurea" è usato in questa Guida per riferirsi al Corso di Laurea in Informatica (triennale) della classe L31 (Scienze e Tecnologie Informatiche) attivato dall'Università di Torino e con sede presso il Dipartimento di Informatica.
10. L'Università di Torino adotta una procedura telematica per l'iscrizione agli esami e la registrazione dei voti conseguiti (con conseguente abolizione del "libretto degli esami" in forma cartacea). Per ogni esame, nonché per la registrazione del superamento dello stage, è obbligatoria l'iscrizione.
11. La valutazione degli insegnamenti avviene con procedura telematica quando si aprono le cosiddette "finestre di valutazione", tipicamente fra la fine del semestre e la fine degli appelli di esame. La valutazione di un insegnamento da parte dello studente è condizione necessaria per l'iscrizione all'esame dell'insegnamento stesso. Gli studenti possono poi esprimere una loro valutazione sulle modalità di esame e sulla congruenza fra esame e insegnamento nelle valutazione dell'anno successivo
12. La documentazione completa del Corso di Studi, così come caricata annualmente sul sito del Ministero dell'Istruzione, l'Università e la Ricerca (MIUR), la potete trovare alla voce Scheda Unica Ateneo, Laurea L31, Università di Torino, sul sito MIUR www.universitaly.it

In ultimo, una raccomandazione ed un consiglio per gli immatricolandi: è molto importante che gli studenti che si iscrivono *a tempo pieno* seguano regolarmente le lezioni e diano gli esami con regolarità, al fine di laurearsi entro i tempi previsti. La struttura degli insegnamenti e delle prove di verifica è pensata per chi segue gli studi regolarmente, facendo gli esami a tempo debito e laureandosi nel tempo previsto. Seguire i corsi e studiare con regolarità *sin dal primo giorno*, anche se non ci sono più verifiche e interrogazioni, programmate o meno, come nella scuola secondaria di II grado, renderà il vostro percorso in questo Corso di Laurea più semplice, più interessante e più coinvolgente. Gli studenti iscritti *a tempo parziale* possono rivolgersi al proprio tutor (vedi in seguito) per mettere invece a punto un percorso personalizzato.

Per ulteriori informazioni, siete invitati a partecipare alla presentazione del Corso di Laurea che si terrà il giorno ***23 settembre 2016,*** alle ore 14.00 in aula A, Dipartimento di Informatica, con accesso da via Pessinetto 12 - Torino. La presentazione è aperta anche agli studenti non ancora immatricolati.

Infine, un augurio: spero che gli anni che passerete con noi, impegnati negli Studi Universitari di Informatica, rappresentino un momento importante, divertente e stimolante della vostra crescita personale e professionale. Spero che l'Università non sia per voi il luogo degli esami, ma il luogo dell'apprendimento e della crescita personale e sociale. Concentrarsi sull'apprendimento e sulle competenze, e sulle multiformi possibilità che l'Università vi offrirà per acquisirle, sarà il modo più facile e divertente di portare a termine i vostri Studi con successo.

Buona lettura!

Susanna Donatelli, PhD,
Professore Ordinario di Informatica,
Presidente del Corso di Laurea e Laurea Magistrale in Informatica
Tel. 011 6706711
e-mail presccs@educ.di.unito.it

INDICE

PREFAZIONE 2

PRIMA PARTE: MANIFESTO DEGLI STUDI 6

Obiettivi e sbocchi professionali 6
Obiettivi del Corso di Laurea 6
Sbocchi professionali 6
Certificazione di Qualità 7

Organizzazione della didattica 7
Introduzione 7
Requisiti di ammissione 8
Iscrizione come seconda laurea, passaggio o trasferimento. 9
Pre-corsi (corsi di riallineamento) 10
Iscrizione a tempo pieno e a tempo parziale 10
Piano Carriera e responsabili 10
Date dei periodi di lezione 11
Esami 11
Il processo della qualità del Corso di Laurea 12
Riconoscimento Certificazioni di lingua inglese 12
Il tutorato 13
Date e scadenze 13

La Laurea in Informatica 13
Struttura del Corso di Laurea 13
Biennio propedeutico comune ai tre curricula 15
Curriculum Informazione e Conoscenza 15
Curriculum Linguaggi e Sistemi 16
Curriculum Reti e Sistemi Informatici 17
Struttura dell'offerta formativa rispetto al RAD della classe 18

Supporto agli studi 19
Erasmus 19
Biblioteca 19
Supporto on-line agli insegnamenti 20
Aule e laboratori 20
Zone studio 20
Supporto per gli studenti Disabili 21
Supporto per gli studenti con DSA (Disturbi Specifici di Apprendimento) 22

Indirizzi utili 23

SECONDA PARTE: PROGRAMMI E DOCENTI DEGLI INSEGNAMENTI PER L'A.A. 16/17 24
Insegnamenti attivi nel 16/17. 24
Codici per insegnamenti esterni. 26
Codici per convalide di insegnamenti e competenze. 26
Programmi e altre informazioni per gli insegnamenti attivi nel 16/17 (syllabus degli insegnamenti) 27

Prima parte: Manifesto degli Studi

Obiettivi e sbocchi professionali

Obiettivi del Corso di Laurea

In questi anni il mondo delle imprese, dei servizi e della Pubblica Amministrazione continua a guardare con particolare interesse ai laureati in Informatica e la rapida evoluzione del settore crea sempre nuove opportunità di inserimento degli informatici nel mondo del lavoro.

Il Corso di Laurea fornisce una buona preparazione metodologica di base, coniugata con la comprensione delle più recenti tecnologie in diversi ambiti, dall'impresa al servizio pubblico e alle applicazioni individuali. Obiettivo del corso di laurea è di far acquisire le competenze necessarie sia per un rapido inserimento nel mondo del lavoro (nel settore delle tecnologie dell'informazione e della comunicazione) sia per permettere al laureato di seguire la rapida evoluzione tecnologica e di adattarsi in modo flessibile a realtà lavorative molto varie.

La laureata e il laureato in Informatica saranno in grado di utilizzare le conoscenze e le competenze acquisite nella progettazione, sviluppo e gestione di sistemi informatici; possiederanno inoltre le capacità necessarie per affrontare ed analizzare problemi in vari contesti applicativi e per sviluppare sistemi informatici per la loro soluzione.

Oltre alla preparazione nel settore matematico e informatico, integrate dalle competenze in ambito economico e del diritto, il percorso formativo prevede l'acquisizione, da parte dello studente, delle competenze necessarie per lavorare in gruppo e per comprendere e produrre rapporti e documentazione tecnica sia in Italiano che in Inglese.

Sbocchi professionali

I laureati e le laureate in Informatica svolgeranno attività professionale negli ambiti della progettazione, organizzazione e gestione di sistemi informatici, sia in imprese produttrici di hardware o software nelle aree dei sistemi informatici e delle reti, sia nelle imprese che utilizzano le tecnologie dell'informazione nel loro settore specifico (banche, assicurazioni, enti pubblici). Inoltre, le competenze acquisite durante il corso di laurea consentono di avviare attività professionali autonome, prospettiva che oggi trova sempre più spazio nel mondo del lavoro.

Il corso prepara alle seguenti **professioni ISTAT**: Analisti e progettisti di software, Analisti di sistema, Analisti e progettisti di applicazioni web, Specialisti in reti e comunicazioni informatiche, Analisti e progettisti di basi dati, Amministratori di sistemi, Tecnici programmatori, Tecnici esperti in applicazioni, Tecnici web, Tecnici gestori di basi di dati, Tecnici gestori di reti e di sistemi telematici

Il D.P.R. 328/2001 ha istituito nell'Albo professionale degli Ingegneri le Sezioni A e B, in relazione al diverso grado di capacità e competenza acquisita mediante il percorso universitario (magistrale e triennale rispettivamente). Ciascuna sezione è ripartita nei seguenti Settori: Civile e Ambientale, Industriale e dell'Informazione. La Laurea in Informatica è una delle lauree

previste per l'ammissione all'esame di stato per la professione di **Ingegnere dell'Informazione, Sez. B**.

Certificazione di Qualità

Il GRIN, che è l'organizzazione nazionale dei ricercatori di Informatica, ha istituito, dal 2004 e in intesa con AICA (Associazione Italiana per il Calcolo Automatico), un certificato che attesta la qualità dei Corsi di Laurea e Laurea Magistrale in Informatica italiani. Il nostro Corso di Laurea ha già ricevuto tale certificato nazionale di qualità negli anni scorsi ed è in corso la procedura di acquisizione per il 2016-2017.

Informazioni più dettagliate sui criteri e gli obiettivi di questa certificazione si possono ottenere direttamente sul sito GRIN.

Il nostro corso di laurea ha partecipato alle attività per l'accreditamento ministeriale dell'Università di Torino, primo grande Ateneo italiano a candidarsi per l'accreditamento. In particolare il nostro corso di laurea è stato uno dei 7 corsi selezionati dai valutatori per le visite di accreditamento che si sono tenute nel novembre 2015. La valutazione degli indicatori di qualità per il nostro corso di studi non è ancora stata resa nota, appena disponibili verranno pubblicati sul sito del Corso di Studi.

Organizzazione della didattica

Introduzione

Il ***Corso di Laurea in Informatica*** è un percorso triennale che porta al conseguimento del titolo di Dottore in Informatica; l'iscrizione richiede il possesso di un diploma di scuola secondaria di II grado. In accordo con la riforma degli studi universitari nota come "riforma 270", dal numero del decreto ministeriale che l'ha definita, si tratta di un percorso di studi a se stante, che trova però il suo naturale completamento nel ***Corso di Laurea Magistrale,*** un percorso di studi di due anni che porta al conseguimento del titolo di Dottore Magistrale in Informatica, oppure nei ***Master di primo livello*** (normalmente della durata di un anno). Il percorso Magistrale è sempre attivo, mentre l'offerta dei Master, normalmente concordata con le aziende, anche in risposta a bandi regionali od europei, non ha una cadenza, una data di inizio e un ambito tematico ricorrente, e quindi le informazioni non sono riportate in questa guida. Il percorso di studi di 5 anni, laurea triennale seguita da magistrale (anche noto come percorso "3+2") può essere completato dal ***Dottorato di Ricerca***, la cui finalità è di preparare i giovani laureati alla ricerca scientifica, o da ***Master di secondo livello***.

Il *Corso di Laurea* fornisce conoscenze di base ad ampio spettro, accanto a elementi di formazione professionalizzante, al fine di consentire il proseguimento negli studi magistrali, ma anche un inserimento diretto nel mondo del lavoro. Il *Corso di Laurea Magistrale* permette di ottenere una più spiccata specializzazione in importanti settori dell'informatica di base o di proiettarsi verso aspetti interdisciplinari o di ricerca.

Un concetto fondamentale per comprendere l'organizzazione dei Corsi di Laurea è quello di ***Credito Formativo Universitario*** (CFU o semplicemente "credito" nel prosieguo). Le norme di legge prevedono che ogni CFU equivalga a 25 ore di lavoro per lo studente. Le 25 ore comprendono sia le lezioni (e le esercitazioni), sia lo studio individuale. Un Corso di Laurea triennale è costituito da 180 CFU, mentre il percorso di una Laurea Magistrale è costituito da 120 CFU e un corso di Master da 60 CFU. Per il Corso di Laurea in Informatica il Regolamento

Didattico stabilisce che ogni CFU sia di norma equivalente a 10 ore di lezione/esercitazione in aula o in laboratorio + 15 ore di studio individuale.

Ad esempio, un insegnamento di 6 crediti quale, per esempio, Calcolo Matriciale e Ricerca Operativa (primo semestre del primo anno) corrisponderà a 60 ore di lezioni ed esercitazioni, e si assume che allo studente (che abbia una buona conoscenza delle tematiche propedeutiche per il corso, indicati nel seguito come requisiti di base) siano richieste ulteriori 90 ore per lo studio, i ripassi, la preparazione dell'esame, ecc. In tutte le tabelle che seguono, la 'durata' degli insegnamenti è espressa in CFU. È importante ricordare che questi conteggi fanno riferimento ad uno studente che abbia acquisito nella Scuola Secondaria di II grado una solida preparazione di base, e che ogni studente dovrà trovare, con l'esperienza, il numero di ore e le modalità di studio individuale che meglio gli permettono di acquisire le richieste competenze.

In questa guida, i termini *corso, insegnamento* e *unità didattica* sono usati in modo equivalente.

Requisiti di ammissione

Sono ammessi al Corso di Laurea in Informatica gli studenti in possesso di Diploma di Scuola Media Superiore o titolo ad esso equivalente. Per frequentare il Corso di Laurea in Informatica non si richiedono competenze informatiche precedenti, ma è indispensabile avere propensione al ragionamento logico e una buona preparazione nelle materie di base della scuola media superiore (in primo luogo la matematica). Tale preparazione di base sarà valutata mediante un *test di accertamento (TARM: Test di Accertamento dei Requisiti Minimi)*, al quale sono tenuti a partecipare tutti gli studenti che desiderano iscriversi ad Informatica.
Il TARM è un test organizzato dall'Università di Torino per gli studenti che intendano iscriversi ad un corso della Scuola di Scienze della Natura ed è organizzato a moduli, un modulo per ognuno dei seguenti argomenti: Matematica, Chimica, Fisica, Biologia e Comprensione del testo. Ogni corso di studi fissa i requisiti su quali siano i moduli da sostenere e con che votazione si considerino superati.
Per il Corso di Studi in Informatica i moduli richiesti sono *Matematica* e *Comprensione del testo*. Ai fini dell'iscrizione al Corso di Studi di Informatica **il TARM si considera superato** se lo studente ha risposto correttamente a 2/5 delle domande di matematica e 3/5 delle domande di comprensione del testo.
Il Corso non è a numero chiuso, ma tutti gli studenti **devono sostenere e superare** il TARM (test di verifica dei requisiti minimi) entro dicembre 2016 per poter sostenere gli esami della sessione di Febbraio. Se uno studente non avrà superato il TARM entro dicembre 2016 gli verrà assegnato un Obbligo Formativo Aggiuntivo (OFA), di norma un corso on-line con esame gestito dal Corso di Studi. Lo studente non potrà sostenere alcun esame sino a quando non avrà ottemperato all'OFA.
È molto importante che gli studenti sostengano il TARM prima dell'iscrizione, per poter adeguatamente autovalutare la propria preparazione rispetto al Corso di Studi. Gli studenti che hanno già superato le soglie nella sessione di TARM anticipato, che si è tenuta nella primavera 2016, non devono risostenere il TARM.

La procedura consigliata per chi non ha ancora sostenuto il TARM è quindi la seguente:

- Iscriversi alla sessione TARM di settembre. La scadenza per l'iscrizione è il 12 Settembre. La sessione di settembre del TARM è l'unica che assicura che lo studente possa sostenere il TARM prima della scadenza delle iscrizioni (fissata per il 5 di ottobre) e che quindi possa iscriversi dopo aver valutato le proprie capacità. Il TARM di

settembre è gestito come un "Test di Valutazione" per l'accesso a tutti i corsi di studio della Scuola di Scienze. Occorre pertanto pre-registrarsi al Portale di Ateneo (http://www.unito.it/servizi/servizi-line/istruzioni-e-supporto/istruzioni-la-registrazione-al-portale) e poi iscriversi al test di valutazione dal menù "Iscrizioni" della pagina MyUnito del portale. Iscrivetevi al test di valutazione della scuola di Scienze, anche se Informatica non è espressamente indicata fra i corsi di studio (istruzioni più precise sul sito web del CdS)

- Gli studenti che non hanno sostenuto e superato le soglie TARM prima dell'iscrizione hanno due prove di TARM (ottobre e dicembre) per il recupero.
- Gli studenti che non hanno superato il TARM entro la prova di dicembre, dovranno presentarsi ad un'apposita commissione di docenti entro la fine di Dicembre, per valutare insieme le difficoltà dello studente e consigliarlo sul proseguo dei suoi studi. A tali studenti verrà assegnato un obbligo formativo aggiuntivo - OFA (corso on-line più esame organizzato autonomamente dal Corso di Studi a gennaio e nel secondo semestre)
- La carriera è bloccata (cioè non si potranno sostenere esami) sino ad espletamento dell'OFA

Ricordiamo che l'Università mette a disposizione materiale per la preparazione al test tramite la piattaforma web orient@mente (www.orientamente.unito.it).

Lo studente che effettua passaggio/trasferimento/seconda laurea è esonerato dal sostenimento del TARM.

Lo studente *che non si è iscritto alla sessione TARM di Settembre o che non ha superato le soglie*, può comunque immatricolarsi ad Informatica, ma deve tener conto che se non ha sostenuto il TARM non ha avuto la possibilità di autovalutarsi, mentre se non ha superato le soglie deve impegnarsi nel recupero delle capacità mancanti.

Attenzione: l'Università di Torino permette alle matricole di passare ad altro corso di studi della nostra Università entro dicembre 2016, attenzione però che il passaggio non è automatico, perché le studente deve soddisfare i requisiti per l'accesso "ora per allora". Per esempio se da Informatica volete passare ad un altro corso della scuola di Scienza della Natura dovete aver sostenuto il TARM nelle sessioni della primavera o di settembre 2016, perché questi corsi *richiedono che lo studente abbia sostenuto il TARM prima della scadenza delle iscrizioni al 5 ottobre*. Se invece passate a corsi di studio di altre scuole è possibile che dobbiate sostenere il TARM su altre materie.

Iscrizione come seconda laurea, passaggio o trasferimento.

Gli studenti che intendono iscriversi come seconda laurea, passaggio da altro corsi di laurea o trasferimenti da altri Atenei devono farsi pre-valutare la carriera inviando l'elenco degli esami sostenuti, completo di codice corso, CFU attribuiti, settore scientifico/disciplinare e riferimento web al programma, ai seguenti docenti:

Docente	Telefono	E-mail
Daniele Gunetti	011 – 670 67 68	gunetti@di.unito.it
Giancarlo Ruffo	011 – 670 67 71	ruffo@di.unito.it

Per maggiori informazioni consultare il sito di Ateneo nella sezione "passaggi e trasferimenti". Ricordiamo che per questi studenti non è necessario sostenere il TARM.

Pre-corsi (corsi di riallineamento)

I pre-corsi sono uno strumento che il corso di laurea mette a disposizione degli studenti che intendano ripassare e re-inquadrare le competenze di matematica e fisica da loro acquisite nelle scuole superiori, in aggiunta a quanto fatto per il superamento del TARM. I pre-corsi sono associati ai normali insegnamenti: per Analisi Matematica si svolgeranno all'interno dell'orario dell'insegnamento del primo anno di Analisi Matematica (secondo semestre). Per Fisica il pre-corso è invece previsto al secondo anno, all'interno dell'orario dell'insegnamento di Fisica.

Iscrizione a tempo pieno e a tempo parziale

La riforma 270 ha introdotto una distinzione importante tra gli studenti universitari: quelli iscritti a tempo pieno e quelli iscritti a tempo parziale. Gli studenti "a tempo pieno" possono laurearsi nei tre anni previsti, ma devono dedicare tutte le proprie energie (lavorative) allo studio. Considerando i CFU previsti per il Corso di Laurea, si può osservare che in media lo studente deve acquisire 60 crediti all'anno, che, a 25 ore per CFU, corrispondono a 1500 ore di lavoro. Su 45 settimane, sono circa 33 ore di lavoro alla settimana.
Se però uno studente intende dedicarsi solo parzialmente allo studio (se, ad esempio, ha anche un'attività lavorativa, o se deve dedicare una parte sostanziale della sua giornata a viaggiare fra casa e università), può iscriversi a tempo parziale. Lo svantaggio è che non sarà più possibile laurearsi in tre anni; il vantaggio è che le tasse annuali sono inferiori e che lo studente non si trova costretto a seguire un ritmo di studio eccessivo, che lo porta a sentirsi sempre "indietro" rispetto agli standard. Si tratta quindi di una scelta importante, che va fatta dopo un'attenta valutazione delle due alternative. Si noti che la scelta viene effettuata ogni anno e non ci sono problemi nel cambiarla da un anno all'altro. Per informazioni sulle modalità di pagamento delle tasse universitarie, si veda http://di.unito.it/tasse. Allo stato attuale lo studente iscritto a tempo pieno potrà registrare un massimo di 80 CFU all'anno, mentre lo studente iscritto a tempo parziale potrà registrare un massimo di 36 CFU all'anno.

Piano Carriera e responsabili

Gli studenti sono tenuti a presentare il piano carriera (ex carico didattico), cioè l'elenco di tutti gli insegnamenti del proprio piano di studi, seguendo le istruzioni presenti sulla pagina pubblica e personale del sito dell'Ateneo (www.unito.it): l'applicativo per il piano carriera viene di norma reso disponibile fra ottobre 2016 e gennaio 2017. Il piano carriera si compila secondo le modalità e le **scadenze** descritte sul sito di Ateneo, e in **accordo con l'offerta formativa della propria coorte e le regole dei piani carriera ad essa associate**, che specifica gli insegnamenti obbligatori e a scelta, e la loro collocazione nei tre anni**.**
La compilazione del Piano Carriera è condizione necessaria per poter sostenere gli esami, anche quelli (tutti obbligatori) del primo anno.

Per ulteriori informazioni e per modifiche dei piani carriera al di fuori delle scelte già previste dall'offerta formativa, gli studenti devono far riferimento alla *Commissione passaggi, trasferimenti e piani di studio* del Corso di Laurea di Informatica (commpassccs@educ.di.unito.it). Gli studenti devono contattare la commissione almeno due settimane prima della scadenza per la presentazione del piano carriera. Per ulteriori informazioni sui curricula gli studenti possono rivolgersi ai relativi docenti responsabili:

- Curriculum Informazione e Conoscenza: Liliana Ardissono (liliana@di.unito.it)
- Curriculum Linguaggi e Sistemi: Viviana Bono (bono@di.unito.it)
- Curriculum Reti e Sistemi Informatici: Giancarlo Ruffo (ruffo@di.unito.it)

Date dei periodi di lezione

Per il Corso di Laurea in Informatica l'Anno Accademico è suddiviso in due periodi didattici (semestri).
Il primo semestre consiste di 12 settimane di lezione (su 13 settimane di calendario, per recuperare le festività presenti), seguite dalla sospensione natalizia e da una settimana di recupero (per le lezioni perse per motivi di forza maggiore) e da una studio individuale. Il semestre si conclude con 4 settimane di appelli di esami.
Il secondo semestre consiste di 12 settimane di lezione (su 13 settimane di calendario, per recuperare le festività presenti), seguite da una settimana di recupero (per le lezioni perse per motivi di forza maggiore) e da due di studio individuale. Il semestre si conclude con 8 settimane di appelli di esami.

Per l'Anno Accademico 2016/2017 le date di inizio e di fine del periodo di lezioni/esami sono le seguenti:

Primo semestre I anno: 26 settembre 2016 – 22 dicembre 2016 (eventuali recuperi 9-13 gennaio 2017)
Primo semestre II-III anno: 28 settembre 2016 – 22 dicembre 2016 (eventuali recuperi 9-13 gennaio 2017)
Sessione invernale di esami: 19 gennaio 2017 – 17 febbraio 2017
Secondo semestre: 20 febbraio 2017 – 19 maggio 2017 (eventuali recuperi 22-26 maggio 2017)
Sessione estiva di esami: 05 giugno 2017 – 29 luglio 2017
Sessione autunnale di esami: 04 settembre 2017 – inizio lezioni a.a. 2017-2018

Gli orari e le aule degli insegnamenti sono disponibili all'indirizzo http://di.unito.it/orarilezioni. Attenzione che gli orari sono organizzati su blocchi di due settimane, con orari diversi nelle due settimane, e prevedono, per i corsi da 6 cfu, 10 ore di lezione su due settimane (6+4 o 4+6 a seconda degli insegnamenti), mentre i laboratori da 3 cfu hanno 3 ore in orario alla settimana per il primo e il terzo anno e 6 ore in orario (4+2 o 2+4) su due settimane per i laboratori del secondo anno. Si ricorda che la frequenza ai corsi non è obbligatoria, ma è comunque fortemente raccomandata, in particolare per gli insegnamenti con laboratorio.

La coorte degli studenti viene divisa in due per le lezioni frontali (corso A, studenti con iniziale del cognome A-K e corso B, studenti con iniziale del cognome L-Z) e in quattro per le lezioni di laboratorio (corso A - turno T1, corso A - turno T2, corso B - turno T1, corso B, turno T2). La regola di suddivisione dei laboratori è descritta sulle pagine web dei singoli insegnamenti (di.unito.it/insegnamenti).

Esami

Per ogni unità didattica oggetto d'esame, sono previsti 5 appelli d'esame all'anno. Le date e gli appelli degli esami saranno disponibili a partire da fine settembre all'indirizzo http://di.unito.it/appelli. Si ricorda che per poter sostenere un esame *lo studente deve obbligatoriamente iscriversi* allo stesso dal proprio sito della bacheca degli esami, disponibile sulla pagina personale dello studente sul sito di Ateneo. Le modalità di esame variano da insegnamento ad insegnamento, ma rimangono invariate per tutti e 5 gli appelli dell'anno di uno stesso insegnamento. È possibile iscriversi all'esame di un insegnamento solo se

- si è in regola con il pagamento delle tasse *e*
- l'insegnamento fa parte del proprio piano carriera per l'anno in corso (o precedenti) *e*

- si è effettuata la valutazione dell'insegnamento stesso (procedura web Edumeter per la raccolta delle opinioni degli studenti sugli insegnamenti, si veda il punto seguente)

Sono previste tre sessioni di laurea all'anno, normalmente nei mesi di luglio, ottobre (con prolungamento a dicembre) - ultima sessione valida per l'iscrizione alla magistrale, e Marzo/Aprile. Le date per le sessioni di laurea della coorte 2016/2017 saranno rese note entro l'inizio del terzo anno accademico della coorte. Il regolamento di Ateneo prevede che uno studente debba terminare gli esami al momento della domanda di laurea. Per i soli studenti in corso, e limitatamente alle sessioni di luglio ed ottobre, è previsto di poter terminare gli esami anche dopo la scadenza della domanda, sino a 10 giorni prima della data dell'esame di laurea. Queste scadenze hanno un particolare impatto sugli studenti fuori corso (cioè iscritti da più di 3 anni) per i quali non sarà garantito che possano usufruire degli appelli di esame di giugno/luglio per potersi laureare a luglio e di settembre per potersi laureare ad ottobre.

Il processo della qualità del Corso di Laurea

La valutazione della didattica è obbligatoria per tutti gli studenti e per tutti i docenti. Per poter sostenere l'esame di un insegnamento lo studente deve aver *obbligatoriamente valutato* l'insegnamento stesso. Oggetti della valutazione saranno l'interesse per l'insegnamento, la chiarezza espositiva del docente, la regolarità nello svolgimento delle lezioni, le strutture (aule o laboratori) in cui il corso si è tenuto. Tale procedura, supportata dall'applicativo Edumeter, è obbligatoria anche per gli studenti non frequentanti (che avranno un diverso insieme di domande) e per i docenti (che potranno esprimere un parere sulle strutture e sulla preparazione e interesse degli studenti). Inoltre, una volta all'anno, gli studenti devono valutare gli insegnamenti che hanno sostenuto nell'anno precedente rispetto alle modalità di esame e alla corrispondenza fra contenuti dell'esame e programma dell'insegnamento.
Le valutazioni sono raccolte in forma anonima e costituiscono la base per le attività di monitoraggio della didattica erogata dal nostro Corso di Laurea. In particolare ogni anno, in autunno, il corso di studio effettua un'attività di riesame annuale delle attività di insegnamento: si analizzano gli eventuali punti di forza e le criticità, si identificano contromisure e le si pianificano nel tempo, inoltre si verifica che contromisure precedentemente identificate siano state correttamente implementate o siano in fase di svolgimento. Alle attività del riesame partecipano anche i rappresentanti degli studenti o un loro delegato. Eventuali problemi con l'applicativo web di valutazione possono essere risolti contattando il servizio di help-desk: www.unito.it/servizi_federati

Riconoscimento Certificazioni di lingua inglese

Gli studenti possono richiedere il riconoscimento dei certificati B1-B2 (secondo il Common European Framework) per l'esame di Lingua Inglese I compilando l'apposito modulo per il riconoscimento delle APU - Attività Pre-Universitarie. Il modulo va consegnato on-line sul proprio spazio MyUniTO oppure alla Segreteria Studenti del Polo delle Scienze della Natura, in via S. Croce 6, secondo le scadenze definite dalla Segreteria, di norma a metà giugno di ogni anno.
A seguito di apposita delibera, gli esami riconosciuti saranno direttamente caricati sulla carriera degli studenti e lo studente non dovrà più sostenere Lingua Inglese I – mfn0590 – 3 CFU, facente parte dei crediti obbligatori del piano carriera del I anno.

Il tutorato

Il tutorato è stato istituito con la legge 341/90 (riforma degli ordinamenti didattici universitari): si tratta di un'attività diretta a "orientare ed assistere gli studenti lungo tutto il percorso degli studi, a renderli attivamente partecipi del processo formativo, a rimuovere gli ostacoli ad una proficua frequenza dei corsi, anche attraverso iniziative rapportate alle necessità, alle attitudini e alle esigenze dei singoli" (art. 13).
Dall'anno accademico 2003-04, il Corso di Laurea in Informatica ha attivato il 'servizio' di tutorato individuale. Si tratta di un'attività che si propone di aiutare gli studenti a procedere con regolarità negli studi. A tale scopo, ad ogni studente viene assegnato un docente di riferimento (*tutor*). Lo studente partecipa agli incontri di tutorato nei tempi e nei modi stabiliti dalla commissione tutorato. Per il Tutorato sono previsti due incontri, uno a fine della prima sessione di esami e uno all'inizio dell'anno successivo. Il secondo incontro è particolarmente utile per pianificare il proseguo degli studi qualora lo studente abbia incontrato delle difficoltà e non abbia acquisito tutti i crediti formativi previsti per il primo anno. Il calendario degli incontri viene comunicato con mail a tutti gli studenti e pubblicato nella sezione avvisi della pagina web del corso di studi

Dal 2014/2015 è anche attivo il "tutorato di materia", previsto per i corsi di Analisi Matematica e di Fisica: un laureato sarà a disposizione degli studenti per consulenze mirate principalmente allo svolgimento di esercizi.

Date e scadenze

Per tutte le date, le scadenze e le modalità di immatricolazione e passaggi si faccia riferimento all'apposita sezione del *sito web dell'Ateneo*.

La Laurea in Informatica

Nel seguito viene descritta la struttura del corso di laurea, comune ai tre curricula, e come tale struttura sia declinata negli insegnamenti per i tre curricula attivati per l'A.A. 2016/2017. Vengono come prima cosa delineati alcuni aspetti comuni ai tre curricula, vengono poi riportate le tabelle ministeriali (RAD) relative al piano dell'offerta formativa per la classe L-31 e la corrispondente implementazione dell'offerta effettuata dall'Università di Torino, e approvata dal Ministero. Vengono infine descritti i tre curricula, con l'elenco degli insegnamenti associati e la loro distinzione in caratterizzanti, affini e liberi, e la loro natura obbligatoria o meno.

Struttura del Corso di Laurea

Il Corso di Laurea in Informatica si articola in tre diversi percorsi (*curricula*) con i primi due anni in comune, dedicati allo studio di materie propedeutiche per gli insegnamenti successivi. Nel primo e secondo anno sono presenti insegnamenti di matematica, insegnamenti di introduzione alla struttura dei calcolatori, alla programmazione, ai sistemi operativi, agli algoritmi e ai data base, e complementi di economia, diritto e fisica.

Il terzo anno lo studente dovrà scegliere tra i tre curricula: ***Informazione e Conoscenza*** (*IC*), ***Linguaggi e Sistemi*** (*LS*), oppure ***Sistemi e Reti*** (*SR*). Tutti i percorsi affiancano alla didattica svolta in aula delle ore di didattica e di esercitazioni svolte in laboratorio e prevedono la formazione anche in ambiti disciplinari di cultura generale come diritto, lavoro di gruppo e conoscenza della lingua inglese. Il percorso formativo viene concluso con lo sviluppo di un

progetto, di norma svolto presso un'azienda con la supervisione di un tutor aziendale e di un tutor accademico, che offre allo studente una prima esperienza lavorativa.

Come ricordato in precedenza, per il Corso di Laurea in Informatica è stabilito che ogni *CFU* (che, ricordiamo, corrisponde per legge a 25 ore di lavoro per lo studente) sia di norma equivalente a 10 ore di lezione/esercitazione più 15 ore di studio individuale. Per facilitare l'inserimento nel mondo universitario, si propone un primo anno più leggero, di 57 CFU, seguito da due anni di 60 e 63 CFU. Ricordiamo che, nella definizione del piano carriera occorre rispettare le propedeuticità di frequenza (cioè non è possibile inserire esami di un dato anno se non si sono già inseriti in piano carriera tutti gli esami degli anni precedenti).

Nelle tabelle che compaiono in questa sezione sono riportati solo i dati essenziali degli insegnamenti, che permettono di avere un quadro complessivo dei diversi anni. Maggiori dettagli (semestre di svolgimento, docenti) si possono trovare nelle parte II di questa guida, mentre le schede di tutti gli insegnamenti sono presenti nell'appendice (syllabus), oltre che, naturalmente, nel sito del Corso di Laurea (http://www.informatica.unito.it), nella parte relativa alla "Offerta formativa".

La laurea prevede 180 CFU. Nelle tabelle che seguono, che descrivono le unità didattiche, la somma totale dei CFU (primo e secondo anno in comune più terzo anno, specifico per ciascuno dei tre *curricula*) è pari a 156. I restanti 24 CFU sono così suddivisi: 9 o 15 CFU per lo stage e 3 per la prova finale (si veda il paragrafo successivo) e 12 CFU liberi, a scelta dello studente.

Stage e prova finale

Per completare gli studi, lo studente dovrà effettuare un periodo di stage, la cui durata è proporzionale al numero di crediti, presso un'azienda o un centro di ricerca, allo scopo di avere un contatto diretto con il mondo del lavoro (le informazioni necessarie sono disponibili alla pagina http://di.unito.it/stageccs). Lo stage può essere svolto anche all'interno dell'Università, su specifici progetti all'uopo identificati. Al termine dello stage, lo studente dovrà scrivere una relazione in cui si descriva l'attività svolta e i risultati ottenuti. Sia l'attività di stage che quella di scrittura della relazione verranno effettuate sotto la supervisione di un docente del Corso di Laurea e si concluderanno con la prova finale (esame di laurea). Lo stage terminato con successo porta al conseguimento degli associati CFU, da registrarsi sul libretto elettronico come ogni altro esame, mentre la Prova Finale, che include il lavoro di stesura della relazione del lavoro di stage e l'esame di laurea in senso stretto, è valutata in ulteriori 3 CFU.
La stage standard è da 9 cfu (corrispondenti a circa 225 ore in azienda), ma lo studente ha facoltà di aumentare l'impegno dedicato allo stage scegliendo di dedicare 6 dei 12 CFU "liberi – a scelta dello studente'' allo stage, che diventa così da 15 cfu (corrispondenti a circa 375 ore in azienda).
Per poter registrare i CFU corrispondenti allo stage, sia interno che esterno all'Università, la procedura richiede allo studente di iscriversi all'appello "stage" sulla piattaforma Esse3 relativa agli appelli d'esame e di inviare una mail al proprio relatore per comunicare l'avvenuta iscrizione. Nel caso di stage da 15 cfu ci si dovrà iscrivere anche all'appello "prolungamento stage". I 3 CFU di Prova Finale si acquisiscono invece il giorno della laurea, senza necessità di preventiva registrazione.

CFU a scelta libera dello studente

Lo studente può scegliere i 12 CFU liberi all'interno dell'offerta formativa dell'intero Ateneo. La scelta deve essere congruente con l'obiettivo formativo della classe di laurea in Informatica e pertanto la scelta dei liberi deve essere convalidata da apposita commissione. Per ogni

curricula troverete indicate le scelte per i CFU a scelta libera dello studente che sono automaticamente approvate, senza necessità di convalida. Lo studente può usare i CFU liberi anche allo scopo di acquisire crediti in settori non informatici, per esempio per soddisfare requisiti minimi di accesso ad altre lauree magistrali.

Biennio propedeutico comune ai tre curricula

Nelle tabella il numero di CFU indicato come n+x indica che il corso si compone da n CFU di lezioni e x CFU di lezioni/attività di laboratorio

I anno

Codice	Insegnamento	CFU
mfn0570	Analisi Matematica	9
mfn0578	Matematica discreta e logica	12
mfn0582	Programmazione I	6+3
mfn0585	Programmazione II	6+3
mfn0586	Architettura degli elaboratori	6+3
mfn0588	Calcolo matriciale e ricerca operativa	6
mfn0590	Lingua Inglese I	3
	Totale 1 anno	**57**

II anno

Codice	Insegnamento	CFU
mfn0597	Algoritmi e strutture dati	6+3
mfn0598	Fisica	6
mfn0600	Elementi di probabilità e statistica	6
mfn0601	Sistemi operativi	6+3+3
mfn0602	Basi di dati	6+3
mfn0603	Linguaggi Formali e Traduttori	6+3
mfn0604	Economia e gestione dell'impresa e diritto	9
	Totale 2 anno	**60**

Curriculum Informazione e Conoscenza

Il curriculum "*Informazione e conoscenza*" (IC) ha lo scopo di preparare laureati in grado di sviluppare e gestire sistemi informatici complessi, con particolare attenzione alle applicazioni Web. Il curriculum pone particolare attenzione allo sviluppo di applicazioni desktop e "mobile" di sistemi autonomi ed intelligenti, così come alla progettazione di architetture software robuste e flessibili, all'accesso efficiente ai dati ed alla loro analisi intelligente, volta a migliorare la fruizione delle grandi moli di informazioni disponibili online. Fornisce dunque una preparazione di base per rispondere alle esigenze della Società dell'Informazione, adatta alla continuazione degli studi nella Laurea Magistrale, affiancata ad una preparazione tecnica volta a preparare lo studente per un rapido inserimento nel mondo del lavoro.

III anno

Codice	Insegnamento	CFU
mfn0605	Programmazione III	4+2
mfn0606	Sviluppo delle applicazioni software	4+5

mfn0607	Sistemi Intelligenti	6
mfn0608	Interazione Uomo Macchina e Tecnologie WEB	12
mfn1362	Reti I	6
	Crediti liberi a scelta dello studente	**12**
inf0073	Stage	9
inf0074	Prova finale	3
	Totale 3 anno	63

Possibili insegnamenti liberi che rendono il Piano di Studi automaticamente approvato secondo il curriculum Informazione e Conoscenza.

Codice	**Insegnamento**	**CFU**
mfn0610	Linguaggi e Paradigmi di programmazione **se non si è già scelto mfn1354**	6+3
mfn1354	Linguaggi e Paradigmi di programmazione **se non si è già scelto mfn0610**	6
mfn0612	Calcolabilità e Complessità A	6
mfn0617	Economia e gestione dell'innovazione	6
mfn0636	Sicurezza	6
mfn0618	Sistemi Informativi	6
inf0003	Logica per l'Informatica	6
inf0004	Storia dell'Informatica	6
inf0072	Prolungamento Stage	6

Curriculum Linguaggi e Sistemi

Il curriculum "*Linguaggi e Sistemi*" (LS) ha lo scopo di preparare laureati in grado di avere una solida preparazione nell'informatica, sia da un punto di vista fondazionale, sia per lo sviluppo di applicazioni.

Questo curriculum fornisce numerosi elementi formativi di base nel settore matematico e fisico e, per quanto riguarda il settore informatico, le competenze di base relative alle metodologie di programmazione e ai sistemi di elaborazione. Approfondisce inoltre i linguaggi di programmazione e i sistemi di calcolo.

III anno

Codice	**Insegnamento**	**CFU**
mfn0610	Linguaggi e paradigmi di programmazione	6+3
mfn0633	Metodi Formali dell'Informatica	9
mfn0606	Sviluppo delle applicazioni software	4+5
mfn1362	Reti I	6
mfn0605	Programmazione III	4+2
	Crediti liberi a scelta dello studente	**12**
inf0073	Stage	9
inf0074	Prova finale	3
	Totale 3 anno	63

Possibili insegnamenti liberi che rendono il Piano di Studi automaticamente approvato secondo il curriculum Linguaggi e Sistemi:

Codice	Insegnamento	CFU
mfn0617	Economia e gestione dell'innovazione	6
mfn0608	Interazione Uomo Macchina e tecnologie web **se non si è già scelto mfn1353 o inf0002**	12
mfn1353	Interazione Uomo Macchina **se non si è già scelto mfn0608**	6
inf0002	Servizi Web **se non si è già scelto mfn0608**	6
mfn0636	Sicurezza	6
mfn0618	Sistemi Informativi	6
mfn0607	Sistemi intelligenti	6
mfn0634	Tecnologie web **se non si è già scelto inf0002 o mfn0608**	6
inf0003	Logica per l'Informatica	6
inf0004	Storia dell'Informatica	6
inf0072	Prolungamento Stage	6

Curriculum Reti e Sistemi Informatici

L'obiettivo di questo curriculum è quello di formare figure competenti nella progettazione, gestione e controllo di sistemi software. In quanto competenti in sistemi operativi, reti e sicurezza, linguaggi e ambienti di programmazione e basi dati, essi saranno in grado di contribuire alla gestione dell'informatizzazione della piccola impresa e della pubblica amministrazione, e di inserirsi attivamente nei progetti delle grandi aziende per lo sviluppo e la gestione di sistemi software complessi.

III anno

Codice	Insegnamento	CFU
mfn0606	Sviluppo delle applicazioni Software	4+5
mfn0634	Tecnologie web	6
mfn0635	Reti di elaboratori	12
mfn0636	Sicurezza	6
	Insegnamenti a scelta	**6**
mfn0612	Calcolabilità e Complessità A (6 CFU)	6
mfn0605	Programmazione III - (6 CFU)	4+2
	Crediti liberi a scelta dello studente	**12**
inf0073	Stage	9
inf0074	Prova finale	3
	Totale 3 anno	63

Possibili insegnamenti liberi che rendono il Piano di Studi automaticamente approvato secondo il curriculum Reti e Sistemi Informatici:

Codice	Insegnamento	CFU
mfn0610	Linguaggi e Paradigmi di programmazione **se non si è già scelto mfn1354**	6+3

mfn1354	Linguaggi e Paradigmi di programmazione **se non si è già scelto mfn0610**	6
mfn0617	Economia e gestione dell'innovazione	6
mfn1353	Interazione Uomo Macchina	6
mfn0618	Sistemi Informativi	6
mfn0607	Sistemi Intelligenti	6
inf0003	Logica per l'Informatica	6
inf0004	Storia dell'Informatica	6
inf0072	Prolungamento Stage	6

Struttura dell'offerta formativa rispetto al RAD della classe

Le tabelle che seguono descrivono l'impostazione dell'offerta formativa del nostro corso di Laurea (colonna "CFU") rispetto alla struttura della classe L-31 così come definita dal Ministero (Colonna "CFU RAD").

Attività di base

Ambito disciplinare	**Settore**	**CFU**	**CFU RAD**
Formazione matematico-fisica	MAT/01 Logica Matematica -- 6 CFU MAT/02 Algebra -- 6 CFU MAT/05 Analisi Matematica -- 9 CFU MAT/09 Ricerca Operativa -- 6 CFU	27	18-30
Formazione informatica	INF/01 Informatica	27	24-36

Minimo di crediti riservati all'Ateneo *(30 CFU da decreto ministeriale)*		
Totale attività di base	54	42-66

Attività caratterizzante

Ambito: Discipline Informatiche		**CFU**	**CFU Rad**
Intervallo di crediti dell'ambito *(min. 60CFU da decreto ministeriale)*		78	69-90
Gruppo	**Settore**		
C11	INF/01 Informatica	78	69-90
C12		0	0-18
Minimo di crediti riservati all'Ateneo *(60 CFU da decreto ministeriale)*			
Totale attività caratterizzanti		78	69-90

Attività affini e integrative

Ambito: attività formative affini o integrative		**CFU**	**CFU Rad**
Intervallo di crediti dell'ambito *(minimo 18CFU da decreto ministeriale)*		21	18-24
Gruppo	**Settore**		
A11	FIS/01 Fisica Sperimentale –- 6 CFU MAT/06 Probabilità e Statistica – 6 CFU	12-12	6-15
A12	IUS/02 Diritto Privato Comparato -- 3 CFU SECS-P/08 Economia e gestione delle imprese – 6 CFU	9-9	6-15
Totale attività affini		21	18-24

Altre attività

Ambito disciplinare		**CFU**	**CFU Rad**
A scelta dello studente		12	12-15
Per la prova finale e la lingua straniera (art. 10, comma 5, lettera d)	Per la prova finale	3	3-9
	Per la conoscenza della lingua straniera	3	2-6
Minimo numero di crediti riservati dall'Ateneo alle attività art. 10, comma 5, lett. c		-	
Ulteriori attività formative (art.10, comma	Ulteriori conoscenze linguistiche	0	0-6

5, lettera d)	Abilità informatiche e telematiche	0	0-6
	Tirocini formativi e di orientamento	9	0-15
	Altre conoscenze utili per l'inserimento nel mondo del lavoro	0	0-6
Minimo numero di crediti riservati dall'Ateneo alle attività art. 10, comma 5, lett. d		1	
Per stages e tirocini presso imprese, enti pubblici o privati e ordini professionali		0	0-15
Totale Altre Attività		18	18-78

Supporto agli studi

Erasmus

L'Università di Torino ha stabilito accordi bilaterali con un certo numero di Università europee per lo scambio di studenti di Informatica nel quadro del progetto Socrates/Erasmus dell'Unione Europea. Gli studenti del Corso di Laurea in Informatica possono quindi concorrere a borse annuali per un soggiorno di studio all'estero, con convalida degli esami colà sostenuti. Nel 16/17 il Dipartimento ha inoltre previsto un'integrazione di queste borse sulla base dei risultati effettivamente conseguiti.
Gli accordi attualmente esistenti riguardano:
Universidad Autonoma de Barcelona, Spagna
Universidad de Las Palmas de Gran Canaria, Isole Canarie Spagna
Universidad Autonoma de Madrid, Spagna
Universidad Politecnica de Madrid, Spagna
Universidad de La Laguna, Isole Canarie-Tenerife Spagna
Universidad de Zaragoza, Spagna
Universite de Savoie, Francia
Université de Lorraine, Nancy, Francia
Université Paris Diderot – Paris 7, Francia
Reykjavik University, Islanda
Technical University of Lodz, Polonia
Universitatea "Politehnica" din Bucuresti, Romania
Universitatea Tehnica Cluj-Napoca – Technical University of Cluj-Napoca, Romania
Université du Luxembourg, Lussemburgo
Universitat Klagenfurt, Austria
Universite Paris Nord – Paris XIII, Francia
Université Pierre et Marie Curie, Paris 6, Francia (*per Erasmus Traineeship*)
University of Helsinki, Finlandia
Sabaci University, Istanbul, Turchia
Université Montpellier 2, Francia
Wroclaw University of Technology, Polonia
Universidad Politecnica de Valencia, Spagna

Altre informazioni sono disponibili sui siti http://di.unito.it/erasmus e http://di.unito.it/erasmusccs

Biblioteca

Come molte biblioteche universitarie, anche la biblioteca del Dipartimento di Informatica (http://di.unito.it/biblio) contiene collezioni di interesse sia per gli studenti che per i ricercatori.

Per quanto riguarda l'aspetto didattico la biblioteca dispone dei testi di studio relativi ai diversi insegnamenti. Molti insegnamenti, accanto ad un testo fondamentale o a una raccolta di

dispense, consigliano altri libri complementari: la biblioteca offre la possibilità di consultare tali testi e di ottenerne il prestito per lo studio individuale.
Il patrimonio documentario è costituito da numerosi testi monografici e riviste specialistiche, in parte accessibili a scaffale aperto, in parte accessibili su richiesta del personale bibliotecario. La biblioteca dispone di una sala studio-consultazione dove è anche possibile effettuare ricerche bibliografiche online dai cataloghi accessibili da due postazioni informatiche.
Per comunicare con la biblioteca: biblioteca@di.unito.it
Per maggiori informazioni e dettagli su orari e servizi, consultare il sito http://di.unito.it/biblioteca.

Supporto on-line agli insegnamenti

Oltre ai libri di testo e ad eventuali dispense, i docenti degli insegnamenti utilizzano una piattaforma di e-learning per la distribuzione di ulteriore materiale didattico, per la consegna di eventuali esercizi, la distribuzione di esercizi di autovalutazione, e blog di discussione fra studenti dello stesso insegnamento.

Aule e laboratori

Le lezioni si svolgono nelle aule e nei laboratori situati in Via Pessinetto 12 a Torino, presso la sede del Dipartimento di Informatica dell'Università degli Studi di Torino.
I laboratori didattici (http://di.unito.it/laboratori) offrono oltre 140 postazioni di lavoro (personal computer e workstation) organizzate in cinque aree (laboratorio Turing, laboratorio Von Neumann, laboratorio Dijkstra, laboratorio Babbage e laboratorio Postel). Nei primi quattro si svolgono le lezioni degli insegnamenti di laboratorio, mentre il laboratorio Postel è principalmente dedicato allo sviluppo delle tesi di laurea. Nei periodi in cui i laboratori non sono occupati da lezioni, le postazioni di lavoro sono a disposizione degli studenti per esercitazioni individuali e per la preparazione della tesi. Ogni laboratorio è organizzato come rete locale, e queste reti sono collegate fra loro in configurazione intranet.
Ad ogni studente viene assegnato un codice personale di accesso alla macchine, indispensabile per l'uso delle postazioni di lavoro. Questo codice di accesso è differente dalle vostre credenziali di Ateneo, e serve solo per l'accesso ai laboratori del Dipartimento di Informatica. I laboratori sono a disposizione degli studenti per i soli scopi didattici, e vengono gestiti nella più stretta osservanza delle norme contro la pirateria informatica.

> Gli studenti sono invitati a richiedere il più presto possibile, dopo l'iscrizione, il codice di accesso alle macchine (login), rivolgendosi ai tecnici del Dipartimento di Informatica negli orari fissati (http://di.unito.it/login).

I laboratori sono aperti dal Lunedì al Venerdì dalle ore 8.30 alle ore 19:00.

Zone studio

Gli studenti possono utilizzare un'ampia zona studio messa a disposizione dall'Ente Diritto allo Studio Universitario della Regione (EDISU), in locali contigui alle aule. La zona studio è organizzata in una zona aperta in cui gli studenti possono lavorare in gruppo ed una zona chiusa in cui è prescritto il silenzio, dedicata allo studio individuale.

Supporto per gli studenti Disabili

L'Università degli Studi di Torino, nella prospettiva di rendere effettivo il diritto allo studio per tutti gli studenti disabili, intende garantire l'accesso fisico alle strutture di studio e di ricerca. Esiste un progetto di progressiva eliminazione delle barriere architettoniche che, ogni anno, disponendo di apposita quota di finanziamento, affronta le situazioni che sono state individuate, attraverso un censimento di tutti gli edifici, come maggiormente problematiche e gravi.

Gli studenti disabili possono usufruire di agevolazioni relative al pagamento delle tasse in relazione alla percentuale di disabilità.

Gli studenti disabili possono, inoltre, richiedere diverse tipologie di servizi:
• accompagnamento presso le strutture universitarie e gli enti di ricerca ed assistenza durante i pasti;
• tutorato didattico: aiuto per la compilazione di appunti, il reperimento testi, fotocopie, disbrigo pratiche burocratiche;
• supporto per la preparazione degli esami (rivolto esclusivamente a studenti con particolari disabilità);
• supporto di interpreti della Lingua Italiana dei Segni e di Mediatori alla Comunicazione per gli studenti non udenti;
• supporto per la richiesta di prove d'esame individualizzate;
• possibilità di utilizzo dei locali del Settore per attività di studio connesse all'Ateneo e disponibilità di postazioni informatiche accessibili;
• sostegno personalizzato attraverso progetti individuali specifici.

Gli studenti disabili possono inoltre accedere ai servizi dell'Ente per il Diritto allo Studio Universitario della Regione Piemonte e del Settore Mobilità Internazionale secondo le modalità individuate dai bandi di concorso.

L'elenco dei Delegati per gli studenti disabili delle Scuole e dei Dipartimenti è disponibile sul sito di Ateneo www.unito.it » Servizi per lo Studio » Studenti con disabilità » Delegati

Per informazioni
Direzione Risorse Umane
Settore Integrazione Studenti Disabili
Via Po, 31 – Via Po, 29 (ingresso studenti) – 10124 Torino
Tel. 011.670.4282/4283/4284 – Fax 011.670.4285 – Email: ufficio.disabili@unito.it

Orari:
• apertura dei locali dedicati agli studenti per attività di studio e di ricerca, incontri con i propri operatori e utilizzo di postazioni informatiche accessibili, preferibilmente su prenotazione, da lunedì a giovedì 9-18.45 e venerdì 9-16.30;
• ricevimento studenti per colloqui specifici e/o prima accoglienza esclusivamente su appuntamento.

La Scuola di Scienze delle Natura, a cui il nostro corso di Laurea Magistrale afferisce, ha nominato come garante per gli studenti disabili della propria Scuola la prof.ssa Anna Capietto, a cui è possibile rivolgersi per informazioni specifiche sui test d'ingresso, sulla modalità

d'esame, sui percorsi didattici specifici e particolari. Referente per il nostro Corso di Laurea Magistrale è la prof.ssa Cristina Baroglio (supplente Prof. Luca Padovani).

Prof.ssa **Capietto Anna** tel 0116702914 fax 0116702878 anna.capietto@unito.it Dip. Matematica "Giuseppe Peano" Via Carlo Alberto, 10 – 10123 Torino

Prof.ssa **Baroglio Cristina**, tel. 0116706711, cristina.baroglio@unito.it, Dipartimento di Informatica, Via Pessinetto 12, Torino
(supplente Prof. **Padovani Luca**, tel. 0116706711, luca.padovani@unito.it, Dipartimento di Informatica, Via Pessinetto 12, Torino

Tutte le informazioni sono reperibili al seguente link:
- studenti con disabilità: http://www.unito.it/servizi/lo-studio/studenti-disabili

Supporto per gli studenti con DSA (Disturbi Specifici di Apprendimento)

L'Università di Torino ha uno **sportello dedicato** agli studenti con DSA. L'Università di Torino, in sintonia con le recenti disposizioni di legge, in specifico la legge 8 ottobre 2010, n. 170, si impegna ad adottare modalità di apprendimento e di valutazione il più possibile flessibili, in grado di **garantire il pieno apprendimento** in termini di conoscenze e competenze anche per gli studenti DSA. Lo sportello predisposto offre i seguenti servizi:

- interventi di **mediazione** con i docenti in vista degli esami orali o scritti;
- **tutorato specifico** (redazione appunti, registrazione lezioni) per le attività didattiche;
- **informazioni** sulle procedure di immatricolazione e sui test d'ingresso;
- incontri individuali di **consulenza didattica**.

L'elenco dei Referenti per gli studenti con DSA è disponibile sul sito di Ateneo www.unito.it » Servizi per lo Studio » Studenti con disabilità » Delegati

In particolare, alla pagina http://www.unito.it/servizi/lo-studio/studenti-con-disabilita/supporto-agli-studenti-con-disabilita-sostenere-gli-esami segnaliamo l'apposito "Modulo di richiesta trattamento individualizzato per esame" che ogni studente deve compilare ed inviare via e-mail, con congruo anticipo, al Docente responsabile dell'insegnamento.

Per informazioni
Direzione Risorse Umane
Sportello Dislessia
Via Po, 31 – 10124 Torino
Tel. 011.670.4282/4283/4284 – Fax 011.670.4285 – Email: ufficio.dsa@unito.it
www.unito.it » Servizi per gli studenti » Studenti dislessici

Referente per il nostro Corso di Laurea è la prof.ssa Cristina Baroglio (supplente Prof. Luca Padovani).

Prof.ssa **Baroglio Cristina**, tel. 0116706711, cristina.baroglio@unito.it, Dipartimento di Informatica, Via Pessinetto 12, Torino
(supplente Prof. **Padovani Luca**, tel. 0116706711, luca.padovani@unito.it, Dipartimento di Informatica, Via Pessinetto 12, Torino)

Tutte le informazioni sono reperibili al seguente link:
- studenti con DSA: http://www.unito.it/servizi/lo-studio/studenti-con-disturbi-specifici-di-apprendimento-dsa

Indirizzi utili

Ufficio del Corso di Laurea e Laurea Magistrale in Informatica

C.so Svizzera 185, Torino (ingresso da Via Pessinetto 12 – I piano)
*da Lunedì a Venerdì solo su appuntamento**
e-mail: informatica@educ.di.unito.it
Telefono: 011 – 670 67 41; 011 – 670 68 25; fax 011 - 751603

**per appuntamento scrivere a: informatica@educ.di.unito.it con almeno 2 giorni lavorativi d'anticipo*

Segreteria Studenti (http://di.unito.it/segreteriaStudenti)

Via Santa Croce, 6 – Torino
Apertura sportelli: Lun e Ven: 9-11; Mar, Mer, Gio: 9-11 e 13.30-15
help desk: https://fire.rettorato.unito.it/helpdesk_ddss/
Telefono: 011 – 670.9900 (numero telefonico unico – selezionare Polo delle Scienze della Natura)

Job Placement (http://di.unito.it/jobplacement)

Via Michelangelo 32 – Torino
Da Lun a Ven: 09.30-12 e 13.30-16
e-mail: jp.scienzedellanatura@unito.it
Telefono: 011 – 670 6215-6216-6218-6219; fax: 011 – 670 6217

Presidente del Consiglio dei Corsi di Laurea e Laurea Magistrale in Informatica: prof. Susanna Donatelli

e-mail: presccs@educ.di.unito.it

Direttore del Dipartimento di Informatica: prof. Luca Console

e-mail: direttore@di.unito.it

Ufficio Accertamento Economico, Regolamento Tasse, inserimento Fasce di reddito (http://di.unito.it/tasse)

Vicolo Benevello 3/a, Torino
Apertura sportelli: Lun e Ven: 9-11; Mar, Mer, Gio: 9-11 e 13.30-15
help desk: https://fire.rettorato.unito.it/helpdesk_ddss/
Tel. 011 – 670 9902 (solo da lun a ven ore 9.00-12.00); fax 011 – 23 610 73

Ente Diritto allo Studio Universitario (http://www.edisu.piemonte.it)
(borse di studio, posto letto, buoni mensa)

Via Giulia di Barolo, 3/bis – 10124 Torino
Tel. 011 – 652 27 01 E_mail: edisu@edisu-piemonte.it
Lun e Ven: 9-11; Mar, Mer, Gio: 9-11 e 13.30-15

Ufficio Studenti Stranieri (http://di.unito.it/stranieri)

Vicolo Benevello, 3/a (I piano) – 10124 Torino
Lun e Ven: 9-11; Mar, Mer, Gio: 9-11 e 13.30-15
e-mail: segrstu.stranieri@unito.it
Telefono: 011 – 670 4498 oppure 011 – 670 4499

Ufficio Erasmus (http://di.unito.it/erasmus)

Vicolo Benevello 3/A – 10124 Torino
Lun e Ven: 9-11; Mar, Mer, Gio: 9-11 e 13.30-15
Per gli Studenti: relint@unito.it
Telefono: 011 – 670 4425

Rappresentanti degli Studenti: mail rappstud@di.unito.it; http://studenti.i-learn.unito.it/

Seconda Parte: Programmi e docenti degli Insegnamenti per l'A.A. 16/17

In questa seconda parte della guida vengono riportati tutti gli insegnamenti attivati nell'anno accademico 16/17 presso il Corso di Laurea di Informatica, di interesse per gli studenti immatricolati nell'anno e per gli studenti delle due coorti precedenti che devono seguire gli insegnamenti del secondo e terzo anno rispettivamente. Sono inoltre riportati gli insegnamenti che verranno attivati nei prossimi anni, nonché l'elenco degli insegnamenti esterni (vedi in seguito) e dei codici per le convalide (per gli studenti in trasferimento).

Insegnamenti attivi nel 16/17.

La tabella che segue riporta tutti gli insegnamenti che saranno attivi nell'anno accademico 16/17. Nella tabella le colonne "Codice", "Insegnamento" e "SSD" indicano, rispettivamente, il codice di Ateneo, il titolo dell'insegnamento e il suo settore scientifico disciplinare, TAF e Ambito individuano la tipologia dell'attività formativa (di base, caratterizzante, a scelta dello studente, etc.), "CFU" sono i crediti formativi associati all'insegnamento, "docenti" sono i docenti responsabili dell'insegnamento e "semestre" il periodo didattico in cui si tiene l'insegnamento. Tutti gli insegnamenti del primo e del secondo anno sono sdoppiati (corso A e corso B)*, mentre i laboratori sono quadruplicati (turni A1, A2, B1 e B2 oppure T1, T2, T3, T4). I programmi degli insegnamenti sono invece presentati più avanti, a fine guida. Ricordiamo che analoghe informazioni sono reperibili sulle pagine web del corso di laurea (da http://laurea.educ.di.unito.it/ seguire il link 'insegnamenti', nella sezione "Per chi studia con noi").

** Corso A: dalla A alla K, Corso B: dalla L alla Z*

Codice principale	INSEGNAMENTO	SSD	TAF	Ambito	CFU	DOCENTI	Semestre	Note
mfn0597	Algoritmi e Strutture Dati	inf/01	B	caratterizzante	9	**Horvath (Teoria A), de' Liguoro (Teoria B), Magro (lab T1), Esposito (lab T2), Damiani (lab T3), Pozzato (lab T4)**	2 sem	
mfn0570	Analisi Matematica	mat/05	A	di base	9	**Barutello (teoria A), Boscaggin (Eserc. A e B), Dambrosio (Teoria B), Costantini (pre-corso)**	2 sem	
mfn0586	Architettura degli Elaboratori	inf/01	A	di base	9	**Gaeta (teoria A), Aldinucci (teoria B), Baldoni (lab A1 e A2), Lucenteforte (lab B1), Donatelli (lab B2)**	2 sem	
mfn0602	Basi di Dati	inf/01	B	caratterizzante	9	**Demo (Teoria A, lab T1), Anselma (lab T2 e T3), Pensa (Teoria B), Capecchi (lab T4)**	2 sem	
mfn0612	Calcolabilità e Complessità A	inf/01	D	a scelta	6	**Ronchi della Rocca**	1 sem	Per il 16-17 mutuato su Metodi Formali dell'Informatica
mfn0588	Calcolo Matriciale e Ricerca Operativa	mat/09	A	di base	6	**Grosso (A), Aringhieri (B)**	1 sem	
mfn0604	Economia e Gestione dell'Impresa e Diritto	6 CFU secs-p/08 e 3 CFU ius/02	C	affine e integrative	9	**Pironti/Pisano (Economia A e B), Montalcini/Sacchetto (Diritto A e B)**	2 sem	
mfn061	Economia e	secs-	D	a scelta	6	**Pironti/Pisano**	1 sem	

7	Gestione dell'Innovazione	p/08						
mfn0600	Elementi di Probabilità e Statistica	mat/06	C	affine e integrative	6	**Seiler (A), Cappiello (Eserc A e B), Garello (B)**	1 sem	
mfn0598	Fisica	fis/01	C	affine e integrative	6	**Migliore (A), Gagliardi (A), Pesando (B)**	2 sem	
mfn1353	Interazione Uomo Macchina	inf/01	D	a scelta	6	**Sacco, Segnan**	1 sem	mutua su modulo di Interazione Uomo Macchina e Tecnologie Web
mfn0608	Interazione Uomo Macchina e Tecnologie Web	inf/01	B	caratterizzante	12	**Ardissono, Sacco, Segnan**	1 sem	
mfn0590	Lingua Inglese I	L-Lin/12	E	conoscenza lingua straniera	3	**Griffin (esercitatore), Radicioni, Cordero, Zacchi, Patti (responsabili)**	1 e 2 sem	
mfn0610	Linguaggi e Paradigmi di Programmazione	inf/01	B	caratterizzante	9	**Dezani**	1 sem	
mfn1354	Linguaggi e Paradigmi di Programmazione	inf/01	D	a scelta	6	**Dezani**	2 sem	mutua su modulo di Linguaggi e Paradigmi di Programmazione da 9 CFU
mfn0603	Linguaggi Formali e Traduttori	inf/01	B	caratterizzante	9	**Zacchi (Teoria A), Patti (lab T1), Sproston (lab T2), Coppo (Teoria B), Sproston (lab T3), Patti (lab T4)**	1 sem	
inf0003	Logica per l'Informatica	mat/01	D	a scelta	6	**Ronchi della Rocca, Berardi**	1 e 2 sem	
mfn0578	Matematica Discreta e logica	6 CFU mat/02 + 6 CFU mat/01	A	di base	12	**Roggero (Mat Discr A), Viale (Log A) Mori (Mat Discr B), Motto Ros (Log B)**	1 sem	
mfn0633	Metodi Formali dell'Informatica	inf/01	B	caratterizzante	9	**Ronchi della Rocca**	1 e 2 sem	
mfn0582	Programmazione I	inf/01	A	di base	9	**Cardone (teoria A), De Pierro (lab A1-A2), Roversi (Teoria B, lab B1, lab B2)**	1 sem	
mfn0585	Programmazione II	inf/01	A	di base	9	**Padovani (teoria A, lab A1-A2), Bono (teoria B, lab B1, lab B2)**	2 sem	
mfn0605	Programmazione III	inf/01	B	caratterizzante	6	**Ardissono**	2 sem	
mfn0635	Reti di Elaboratori	inf/01	B	caratterizzante	12	**Sereno, Sirovich**	1 e 2 sem	
mfn1362	Reti I	inf/01	B	caratterizzante	6	**Botta**	1 sem	
inf0002	Servizi Web	inf/01	D	a scelta	6	**Ardissono**	1 sem	mutua su modulo di Interazione Uomo Macchina e Tecnologie Web
mfn0636	Sicurezza	inf/01	B	caratterizzante	6	**Bergadano**	2 sem	
mfn0618	Sistemi Informativi	inf/01	D	a scelta	6	**Micalizio**	1	
mfn0607	Sistemi Intelligenti	inf/01	B	caratterizzante	6	**Baroglio**	2 sem	
mfn0601	Sistemi Operativi	inf/01	B	caratterizzante	12	**Gunetti (Teoria A), Baroglio (Teoria B), Radicioni (lab T1 Unix), Bini (lab T2 Unix e T4 Unix), Di Caro (lab T3 Unix), Gaeta (lab T1 C), Anselma (lab T2 C), Micalizio (lab**	1 sem	

						T3 C), Gaeta (lab T4 C)		
mfn0606	Sviluppo delle Applicazioni Software	inf/01	B	caratterizzante	9	**Bono (Teoria), da assegnare (lab 1) Torta (Teoria e lab 2), Picardi (Teoria e lab 3)**	2 sem	
inf0004	Storia dell'Informatica	inf/01	D	a scelta	6	**Gunetti e Cardone**	1 e 2 sem	
mfn0634	Tecnologie Web	inf/01	B	caratterizzante	6	**Ruffo**	1 sem	
Inf0073	Stage		F	altre attività	9			
Inf0072	Prolungamento stage		D	a scelta	6			
Inf0074	Prova Finale		E	altre attività	3			

Codici per insegnamenti esterni.

I seguenti sono codici contenitore utilizzati per convalidare attività svolte presso enti con i quali il Corso di Laurea in Informatica abbia siglato degli specifici accordi.

mfn1052	Attività Professionalizzanti	altre attività, a scelta lettera A	3
mfn1502	Attività Professionalizzanti	altre attività, a scelta lettera A	3

Codici per convalide di insegnamenti e competenze.

Gli studenti che per effetto di passaggio o trasferimento abbia ottenuto la convalida di crediti liberi con nessuna corrispondenza specifica ad insegnamenti del Corso di Laurea in Informatica, sono pregati di selezionare i codici contenitore MFN1522, MFN1409 e/o MFN1408 in base alla delibera della Commissione Passaggi e Trasferimenti, integrando eventualmente con altro insegnamento libero da selezionare dalla lista prevista per il curriculum scelto, in caso non sia ancora stato raggiunto il minimo di 12 CFU.

mfn1409	Altre Attività	altre attività, a scelta lettera A	12
mfn1408	Altre Attività	altre attività, a scelta lettera A	6
mfn1522	Altre Attività	altre attività, a scelta lettera A	6

Informazioni aggiornate al 12 ottobre 2016

Programmi e altre informazioni per gli insegnamenti attivi nel 16/17 (syllabus degli insegnamenti)

Questa parte sarà disponibile a settembre, come da regolamento didattico di Ateneo. Gli studenti possono consultare le pagine degli insegnamenti del 16/17 sul sito web del corso di studio, che sono in fase di ultimazione in questi giorni.

Insegnamento **MFN0597 - Algoritmi e Strutture Dati**

Insegnamento (inglese):	**Algorithms and Data Structures**
CFU:	**9**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**2**
Tipologia di Attività Formativa:	**B - caratterizzante**
Docenti:	**Ferruccio DAMIANI (Titolare)** **Ugo DE' LIGUORO (Titolare)** **Roberto ESPOSITO (Titolare)** **Andras HORVATH (Titolare)** **Diego MAGRO (Titolare)** **Gian Luca POZZATO (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Si presuppone che lo studente sia a conoscenza delle basi della programmazione e dei linguaggi di programmazione Java e C; che sia in possesso delle nozioni elementari della matematica del discreto, del continuo, e della logica matematica.

Eventuali corsi propedeutici

Programmazione I & Laboratorio, Programmazione II & Laboratorio, Matematica Discreta, Analisi Matematica, Logica Matematica, Sistemi Operativi (in particolare l'apprendimento del linguaggio C).

2. Obiettivi formativi:

L’insegnamento ha lo scopo di introdurre i concetti e le tecniche fondamentali per l’analisi e la progettazione di algoritmi, che sono alla base dello sviluppo del software. Gli studenti acquisiranno conoscenze circa l’analisi di correttezza e complessità computazionale degli algoritmi, sulle strutture dati per la rappresentazione dell’informazione, sulle tecniche di problem solving mediante lo sviluppo di algoritmi efficienti. L’insegnamento è supportato da un laboratorio che ne costituisce parte integrante, finalizzato alla realizzazione e sperimentazione degli algoritmi e delle strutture dati mediante un linguaggio imperativo ed uno object-oriented.

3. Risultati dell'apprendimento attesi:

Al termine del corso ci si aspetta che lo studente sia in grado di risolvere problemi algoritmici elementari mediante la progettazione degli algoritmi e l’analisi delle soluzioni adottate, sia per quanto riguarda l’efficienza, sia per quanto concerne la scelta delle strutture dati manipolate dagli algoritmi. Lo studente sarà in grado di individuare e comprendere algoritmi illustrati nella letteratura specialistica e di adattarli realizzando concretamente le proprie implementazioni, tanto producendo il codice dei programmi quanto utilizzando librerie software preesistenti. Infine lo studente dovrà essere in grado di comunicare e documentare le soluzioni adottate in modo tecnicamente rigoroso.

4. Modalità di verifica dell'apprendimento:

Al termine del corso, l’esame consisterà in una prova scritta sulla teoria, con domande e problemi simili a quelli proposti durante le lezioni, ed un esame orale in cui discuteranno un progetto di laboratorio che avranno precedentemente realizzato.

Il laboratorio deve essere discusso entro la fine della sessione in cui si è dato lo scritto, a pena di decadenza del voto dello scritto stesso. Modalità e tempi di consegna devono essere concordati

con i docenti di laborario

Per superare l’esame gli studenti dovranno dimostrare di conoscere concetti e metodi dell’analisi e dello sviluppo di algoritmi, di essere in grado di applicarli in situazioni concrete, anche ampliando le proprie conoscenze utilizzando la letteratura specialistica; infine dovranno essere in grado di illustrare in modo chiaro e tecnicamente accurato le proprie scelte realizzative nello svolgimento dei compiti assegnati.

Il voto sarà la media pesata dei voti ottenuti nelle due prove scritta ed orale, valutate in 30+1 esimi, essendo comunque necessario il raggiungimento della sufficienza in entrambe le prove. La lode sarà assegnata a quanti conseguiranno un punteggio complessivo strettamente maggiore di 30. Durante il corso gli studenti saranno coinvolti in esercitazioni sulla parte teorica e nelle attività proposte nel laboratorio, attraverso le quali riceveranno un primo riscontro del grado del proprio apprendimento.

5. Modalità d'insegnamento:

Sono previste 60 ore di lezione frontale ed esercitazioni in aula, e 30 ore di attività guidata dai docenti in laboratorio. Per le indicazioni bibliografiche, la distribuzione di note ed altro materiale didattico, e per la consegna e valutazione degli elaborati delle esercitazioni si farà uso della piattaforma e-learning Moodle, cui si richiede agli studenti l’iscrizione.

6. Attività di supporto:

Il materiale didattico sarà reso via via disponibile on line attraverso la piattaforma Moodle.

7. Programma:

Programma

- Problemi e algoritmi: risolubilità, correttezza, complessità.
- Analisi computazionale e complessità asintotica
- Algoritmi di ordinamento
    - Algoritmi elementari quadratici
    - Divide et impera: mergesort e quicksort
    - Risoluzione di relazioni di ricorrenza
    - Limiti inferiori per l’ordinamento
- Strutture dati
    - Strutture concrete: array, liste, tabelle hash
    - Strutture astratte: pile, code, dizionari
    - Code di priorità, heapsort
    - Analisi ammortizzata, cenni
- Alberi
    - Definizione e visita
    - Alberi di ricerca
    - Alberi rosso-neri
- Grafi
    - Definizione e visita
    - Ordinamento topologico e componenti fortemente connesse
    - Algoritmi greedy: alberi di copertura minima
    - Cammini minimi: algoritmo di Dijkstra

8. Testi consigliati e bibliografia:

1. Per la teoria:
    - Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein: Introduzione agli algoritmi e strutture dati, McGraw-Hill 2010.
2. Per il laboratorio:
    - Horstmann: Concetti di informatica e fondamenti di Java, Quinta Edizione (o successiva), Apogeo, 2010.

Insegnamento **MFN0570 - Analisi Matematica**

Insegnamento (inglese):	**Mathematical Analyis**
CFU:	**9**
Settore:	**MAT/05 - ANALISI MATEMATICA**
Periodo didattico:	**2**
Tipologia di Attività Formativa:	**A - di base**
Docenti:	**Vivina Laura BARUTELLO (Titolare)** **Alberto BOSCAGGIN (Titolare)** **Camillo COSTANTINI (Esercitatore)** **Walter DAMBROSIO (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Conoscenze matematiche di base fornite dalla scuola secondaria. In particolare è indispensabile la conoscenza delle funzioni elementari, delle loro proprietà e dei loro grafici e la capacità di risolvere semplici equazioni ad esse associate.

Eventuali corsi propedeutici

Precorso (prime lezioni di Analisi Matematica previste nel calendario del secondo semestre).

2. Obiettivi formativi:

L'insegnamento ha lo scopo di presentare i concetti di base dell'Analisi. La significativa presenza di teoremi, molti dei quali con dimostrazione, ha l'obiettivo di rafforzare nello studente le attitudini logico-deduttive apprese nel corso di Matematica discreta e Logica. In ottemperanza ai descrittori di Dublino, il corso accresce le capacità di comprensione dello studente e gli permette di utilizzare le sue conoscenze in nuovi problemi, al fine di comprenderli individuandone gli aspetti essenziali, sapendone determinare la soluzione e interpretandola nel contesto corretto.

3. Risultati dell'apprendimento attesi:

Lo studente dovrà essere in grado di:

- tracciare il grafico di nuove funzioni a partire da quello di funzioni note; - leggere un grafico di funzione, riconoscendovi gli elementi essenziali; - calcolare la derivata di una funzione; - calcolare semplici integrali definiti; - risolvere semplici problemi di calcolo approssimato; - risolvere semplici problemi di applicazione dei concetti dell'analisi in contesti non matematici; - studiare il carattere di una serie o di un integrale improprio; - dare le definizioni dei concetti di base introdotti nel corso; - enunciare in modo corretto i teoremi presentati a lezione; - riprodurre le dimostrazioni svolte a lezione.

4. Modalità di verifica dell'apprendimento:

L'esame è strutturato in tre prove scritte, obbligatorie per tutti gli studenti, indipendentemente dall'Anno Accademico di iscrizione, tutte da sostenere nello stesso appello. Le prove si svolgono in successione : per accedere alla seconda è necessario aver superato la prima, per accedere alla terza aver superato la seconda.

Prima prova (quiz) La prima prova consiste nella risposta a cinque domande a scelta multipla che hanno l'obiettivo di verificare la capacità dello studente di risolvere semplici esercizi su argomenti di base.

Seconda prova (teoria) La seconda prova verte sugli argomenti trattati a lezione ed esercitazioni e

mira a valutare le conoscenze di definizioni e enunciati di teoremi, nonché le capacità logico-deduttive nell’esposizione delle dimostrazioni presentate a lezione.

Terza prova (esercizi) La terza prova prevede la risoluzione di un certo numero di esercizi, in cui lo studente deve dimostrare di saper applicare le sue conoscenze in problemi maggiormente strutturati.

Informazioni per gli studenti degli anni accademici passati

Gli studenti degli anni accademici passati devono sostenere l'esame secondo le modalità dell’anno accademico corrente.

5. Modalità d'insegnamento:

Lezioni frontali.

6. Attività di supporto:

Ogni settimana i docenti inseriscono sul sito Moodle del corso un "foglio di esercizi”. Gli esercizi proposti in queste schede settimanali hanno un ruolo assai importante nello studio degli argomenti del corso: costituiscono infatti il banco di prova più affidabile per verificare se gli argomenti esposti a lezione sono stati assimilati in maniera sufficientemente profonda da riuscire a risolvere autonomamente problemi simili a quelli affrontati durante le esercitazioni e della stessa difficoltà di quelli presenti all’esame. Gli esercizi proposti possono essere svolti, in modo individuale o a piccoli gruppi, anche durante gli incontri settimanali del “tutorato”, in cui uno studente della LT in Matematica è a disposizione per aiutare nella loro risoluzione e per fornire chiarimenti e spiegazioni.

7. Programma:

Le funzioni e le loro proprietà qualitative; grafici di funzioni elementari; grafici di nuove funzioni a partire da quelli di vecchie funzioni. Il concetto di limite; principali teoremi sui limiti. Successioni e crescite. Il concetto di derivata ed i suoi significati; la funzione derivata. Principali teoremi del calcolo differenziale. Approssimazione locale di funzioni mediante polinomi. Risolubilità di equazioni; calcolo approssimato delle soluzioni. Il concetto di integrale definito ed i suoi significati; calcolo esatto e calcolo approssimato di integrali definiti. Teoremi fondamentali del calcolo integrale. Integrali impropri. Serie numeriche; principali criteri di convergenza.

8. Testi consigliati e bibliografia:

M. Bramanti, C.D. Pagani, S. Salsa "Analisi Matematica 1" © Zanichelli Editore

Insegnamento **MFN0586 - Architettura degli Elaboratori**

Insegnamento (inglese):	**Computer architecture**
CFU:	**9**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**2**
Tipologia di Attività Formativa:	**A - di base**
Docenti:	**Marco ALDINUCCI (Titolare)** **Matteo BALDONI (Titolare)** **Susanna DONATELLI (Titolare)** **Rossano GAETA (Titolare)** **Maurizio LUCENTEFORTE (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Conoscenza di base del linguaggio Java, nozioni di programmazione, Algebra di Boole.

Eventuali corsi propedeutici

È richiesta la conoscenza delle competenze fornite dai corsi di:

Programmazione I e Laboratorio (I semestre) Matematica Discreta e Logica (I semestre)

2. Obiettivi formativi:

L’insegnamento si propone di fornire allo studente: - la conoscenza delle tecniche di codifica dell’informazione all’interno dei calcolatori; - la conoscenza dell’organizzazione hardware degli elaboratori, attraverso la nozione di gerarchie di macchine virtuali; - la comprensione delle funzioni svolte dall’hardware e utilizzate dai sistemi operativi; - le nozioni di base per la comprensione del processo di traduzione da linguaggi ad alto livello a linguaggio macchina; - le conoscenze delle tecniche di microprogrammazione e di programmazione in un linguaggio assemblativo.

3. Risultati dell'apprendimento attesi:

Al termine dell’insegnamento lo studente dovrà dimostrare di essere in grado di: - codificare e decodificare qualunque tipo di informazione (numeri con e senza segno, interi e frazionari, caratteri, immagini e suoni); - conoscere e analizzare circuiti logici elementari; - saper analizzare i componenti digitali che sono alla base dell’architettura dei moderni calcolatori; - conoscere e saper utilizzare le tecniche della microprogrammazione per l’esecuzione di istruzioni a livello macchina; - conoscere le relazioni tra linguaggi ad alto livello (Java) ed equivalente linguaggio a livello macchina; - saper sviluppare piccoli programmi in un linguaggio assemblativo e conoscere le relative tecniche di traduzione in linguaggio macchina e del collegamento (linking/loading).

4. Modalità di verifica dell'apprendimento:

L’esame consiste di una prova scritta con voto finale espresso in trentesimi (con possibilità di lode). La prova scritta consiste di 10 domande: 8 relative alla parte di teoria (almeno 24 punti) e 2 relative alla parte di laboratorio (al massimo 8 punti). Le domande di teoria possono essere a risposta aperta oppure a risposta chiusa. Le domande a risposta aperta possono richiedere l’enunciazione e la descrizione di proprietà o di tecniche viste a lezione. Possono altresì richiedere l’applicazione della teoria a particolari esempi. Con queste domande si intende valutare la comprensione dei principi visti a lezione, la capacità di enunciarli con l’uso appropriato di un linguaggio scientifico e la capacità di applicarli a casi pratici. Le domande di laboratorio vertono sulla microprogrammazione e sulla programmazione in livello assemblativo.

5. Modalità d'insegnamento:

L’insegnamento è diviso in una parte di teoria e una di laboratorio. Per la parte di teoria sono previste 60 ore lezioni frontali, integrate da esempi e da esercitazioni. Per la parte di laboratorio sono previste 30 ore di attività in laboratorio in cui, con l'ausilio di appositi simulatori e applicativi, si svolgono esercizi e approfondimenti sugli argomenti del corso. La frequenza è facoltativa.

6. Attività di supporto:

Il materiale didattico di supporto sia per il corso A che per il corso B (lucidi, link, esempi di testo di esami ed altro) è disponibile presso il supporto on-line ai corsi I-learn

http://i-learn.educ.di.unito.it/.

Testi e soluzioni di alcuni esercizi svolti in aula possono essere trovati nelle pagine del corso di Architettura degli Elaboratori per gli anni accademici precedenti.

7. Programma:

1. Introduzione all'organizzazione strutturata dei calcolatori 2. Tappe fondamentali dell'architettura dei calcolatori 3. Struttura dei calcolatori, macchina di Von Neumann, macchine virtuali 4. Unità metriche 5. Codifica dell'informazione 1. Numeri binari 2. Conversione tra basi 3. Numeri Negativi 4. Operazioni tra numeri binari 5. Numeri floating point e standard IEEE 754 6. Rappresentazione dei caratteri, stardard ASCII e UNICODE 7. Rappresentazione di immagini e suoni 6. Livello logico digitale 1. Porte logiche e algebra di Boole 2. Circuiti logici 3. Latch, flip-flop, registri, chip di memoria, RAM e ROM 7. Memoria e organizzazione della memoria 1. Organizzazione della memoria 2. Memoria cache 3. Memorie permanenti 8. Il bus, chip di CPU e bus 1. Interfacce di I/O e decodifica degli indirizzi 9. Il livello della microarchitettura 1. Organizzazione della CPU ed esecuzione delle istruzioni 2. Esempio di microarchitettura 3. Notazione Polacca Inversa 4. L'ISA IJVM 5. Il microprogramma per l'architettura Mic-1 10. Il livello ISA 1. Sommario al livello ISA, caratteristiche, modelli di memoria, registri e istruzioni 2. Modalita' di indirizzamento 3. Formati delle istruzioni 4. Differenze fra RISC e CISC 5. Principi di progettazione per i calcolatori moderni, parallelismo a livello delle istruzioni e del processore 6. Ottimizzazione dell'architettura Mic-1 e il Mic-2 7. Un'architettura con pipeline 8. Tipi di dati nel livello ISA 9. Tipi di istruzioni nel livello ISA 10. Controllo del flusso 11. I/O, interrupt, trap 11. Assembler, linker, loader

In laboratorio saranno svolte esercitazioni sulle codifiche, sui circuiti logici, sulla microprogrammazione in Mic-1, sulla scrittura di programmi in linguaggio assemblativo IJVM, con l'ausilio di appositi simulatori e applicativi.

8. Testi consigliati e bibliografia:

Andrew S. Tanenbaum, Todd Austin Architettura dei calcolatori: un approccio strutturale. 6a edizione Pearson Education Italia, 2013. ISBN 9788871929620

oppure

Andrew S. Tanenbaum. Architettura dei calcolatori: un approccio strutturale. 5a Edizione. Pearson - Addison Wesley, 2006. ISBN 8871922719.

Insegnamento **MFN0602 - Basi di Dati**

Insegnamento (inglese):	**Databases**
CFU:	**9**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**2**
Tipologia di Attività Formativa:	**B - caratterizzante**
Docenti:	**Luca ANSELMA (Titolare)** **Sara CAPECCHI (Titolare)** **Giuseppina Barbara DEMO (Titolare)** **Ruggero Gaetano PENSA (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Lo studente deve avere familiarità con i concetti fondamentali della teoria degli insiemi e della logica. Deve aver acquisito capacità di progettare algoritmi fondamentali e programmare in linguaggi ad alto livello.

Eventuali corsi propedeutici

Insegnamenti di Logica, Matematica Discreta e Programmazione 1 e 2

2. Obiettivi formativi:

L'insegnamento è un'introduzione alle basi di dati e ai sistemi di gestione delle medesime (SGBD). Si propone perciò di fornire allo studente le prime competenze teoriche e pratiche sul modello relazionale, sulla progettazione di una base di dati e sull'architettura dei SGBD introducendo le componenti fondamentali. In aula saranno introdotti i concetti teorici di base. Gli studenti acquisiranno conoscenze (teoriche e pratiche) su: progetto di una base dati, modello Entità Associazioni (EA) (o Entity Relationship), modello relazionale, algebra relazionale e SQL, dipendenze funzionali e normalizzazione, elementi di architettura dei SGBD relazionali (concorrenza, ripristino e buffer management, dizionario dei dati, memorizzazione efficiente dei dati). In laboratorio gli studenti lavoreranno al progetto di una base dati e ad esercitazioni su casi di studio in SQL.

3. Risultati dell'apprendimento attesi:

Gli studenti devono aver assimilato i concetti:

- i fondamenti matematici del modello relazionale,
- le interrogazioni in algebra ed in calcolo relazionale e in SQL,
- i fondamenti della teoria della normalizzazione,
- le strutture dati per la memorizzazione efficiente dell'informazione.

Inoltre lo studente deve:

- saper analizzare i requisiti per la progettazione di una base di dati per rappresentarli in forma di diagrammi Entity Relationship o Entità Associazioni (EA) essere in grado di trasformare schemi EA in schemi relazionali,
- essere in grado di trasformare interrogazioni in forma testuale in interrogazioni SQL, essere in grado di eseguire semplici ristrutturazioni di schemi EA sulla base di dati quantitativi (ad esempio: numero stimato di record e criticità delle interrogazioni,
- aver compreso i concetti e l'uso di transazioni.

4. Modalità di verifica dell'apprendimento:

E' obbligatoria l'iscrizione all'ambiente di e-learning Moodle dove gli studenti troveranno forum di discussione, materiale didattico, compiti assegnati e quiz sugli argomenti delle lezioni preparati dai docenti. La modalità d'esame comprende una prova scritta e una di laboratorio. Prova scritta: la prova prevede da 6 a 8 domande in tutto, al fine di verificare l'apprendimento dei concetti di base. Di queste, due domande riguardano argomenti di laboratorio (progettazione ed SQL), le restanti domande riguardano argomenti ed esercizi della parte di teoria. La valutazione è in trentesimi.

Prova di laboratorio: viene assegnato un progetto (su argomenti di progettazione ed SQL) che ciascuno studente decide se svolgere in gruppo o individualmente. In ogni caso la discussione e la valutazione dell'elaborato è individuale e verte sull'intero progetto. La valutazione è in trentesimi.

Il voto finale viene calcolato integrando i voti di scritto e laboratorio. La prova scritta e la discussione del laboratorio non hanno regole di precedenza se non che devono essere sostenute nella stessa sessione di esami.

5. Modalità d'insegnamento:

Lezioni frontali: 60 ore. Attività di laboratorio: 30 ore. La frequenza costante è caldamente consigliata.

6. Attività di supporto:

E' obbligatoria l'iscrizione all'ambiente di e-learning Moodle dove gli studenti troveranno forum di dicussione, materiale didattico e compiti assegnati. E' altresì obbligatoria la consultazione di tali materiali.

7. Programma:

Funzionalità e componenti dei sistemi di gestione di basi di dati. Fondamenti teorici delle basi di dati relazionali:

- il modello relazionale delle basi di dati (definizioni, proprietà principali, vincoli di integrità),
- algebra relazionale,
- introduzione al calcolo relazionale,
- introduzione a SQL (DDL e DML), Dizionario dei dati in un DBMS
- dipendenze funzionali e teoria della normalizzazione,
- memorizzazione efficiente dei dati (B+ alberi),
- cenni alle tecniche di ottimizzazione,
- Introduzione alle transazioni: problemi di concorrenza e di affidabilità, livelli di isolamento.

Progettazione e programmazione delle basi dati:

- Specifica d'interrogazioni e realizzazione in SQL (con esercitazioni in laboratorio su ORACLE/Postgresql).
- Dizionario dei dati (con esercitazioni in laboratorio su ORACLE/Postgresql).
- Definizione e ruolo di "database administrator".
- Cenni ai meccanismi d'autorizzazione offerti da SQL (possibilmente con esercitazioni in laboratorio su ORACLE/Postgresql).
- Introduzione alle metodologie di progettazione del software e loro relazione con la progettazione della basi di dati, argomenti non trattati e relazione con altri insegnamenti (es. Ingegneria del Software).
- Progettazione concettuale/logica, usando il modello ER (Entity Relationship) con eventuali esercizi di reverse modelling.
- Considerazioni sui parametri quantitativi dello schema logico.
- Considerazioni su meccanismi d'indicizzazione.

8. Testi consigliati e bibliografia:

Testo di riferimento: Atzeni, Ceri, Fraternali, Paraboschi, Torlone, "Basi di dati - Modelli e linguaggi di interrogazione", McGraw-Hill, Quarta edizione, 2013.

Materiali aggiuntivi sono forniti dai docenti.

Insegnamento **MFN0612 - Calcolabilità e Complessità A**

Insegnamento (inglese):	**Computability and Complexity - A**
CFU:	**6**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**B - caratterizzante** **D - libera**
Docenti:	**Simonetta RONCHI DELLA ROCCA (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Sono richieste buone conoscenze di logica, programmazione e di algoritmi; inoltre si assume che lo studente possegga le nozioni di linguaggio formale, grammatica e di automa.

Eventuali corsi propedeutici

Matematica Discreta e Logica, Programmazione 1 e 2, Algoritmi e Strutture Dati, Linguaggi Formali e Traduttori.

2. Obiettivi formativi:

Che cos'e' un algoritmo? Quali problemi si possono risolvere con un algoritmo? E in quali casi un algoritmo richiede risorse inaccessibili nella pratica? Il corso affronta questi problemi, trattando anzitutto la teoria della computabilita' sia dal punto di vista classico - macchine di Turing, funzioni ricorsive - che da prospettive piu' attuali, come quella dei programmi while, legate ai linguaggi di programmazione. Si discutono poi i vari possibili criteri di misura delle risorse disponibili (tempo, memoria, cpu) e le classi di complessita', con particolare attenzione alla classica questione P = NP .

3. Risultati dell'apprendimento attesi:

Conoscenza delle basi teoriche dell'informatica, una maggiore consapevolezza delle limitazioni intrinseche all'uso delle macchine ed un'idea delle strategie per sopperire a tali limitazioni.

4. Modalità di verifica dell'apprendimento:

Esame orale. Verra' anche tenuto conto della partecipazione degli studenti agli esercizi in classe.

5. Modalità d'insegnamento:

Le lezioni si svolgono in aula, ed utilizzano la piattaforma Moodle sia per la distribuzione di ulteriore materiale didattico (come note e lucidi a cura del docente) che per le verifiche dell'apprendimento durante il corso.

6. Attività di supporto:

Vedi il sito moodle del corso

7. Programma:

Teoria della computabilita'

- Le Macchine di Turing

- Problemi non risolubili

- Funzioni ricorsive
- Calcolabilita' e Linguaggi di Programmazione

Teoria della complessita'

- Misure e classi di Complessita'
- Classi di Complessita' Temporale
- Classi di Complessita' Spaziale
- Le classi P ed NP
- Problemi NP completi

8. Testi consigliati e bibliografia:

- C. Toffalori et alii, Teoria della calcolabilita' e della complessita', McGraw-Hill 2005

- M. Sipser:"Introduction to the theory of Computation", Course Technology Ptr., 3^edition, 2012.

Insegnamento	**MFN0588 - Calcolo Matriciale e Ricerca Operativa**

Insegnamento (inglese):	**Matrix Calculus and Operational Research**
CFU:	**6**
Settore:	**MAT/09 - RICERCA OPERATIVA**
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**A - di base**
Docenti:	**Roberto ARINGHIERI (Titolare)** **Andrea Cesare GROSSO (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Nozioni di base di algebra e insiemistica comuni nei programmi di matematica della scuola superiore.

Eventuali corsi propedeutici

Nessuno.

2. Obiettivi formativi:

Il corso si propone di fornire agli studenti nozioni generali di calcolo matriciale, algebra e geometria, e nozioni più specifiche di ricerca operativa.

Il calcolo matriciale è uno strumento fondamentale per il calcolo scientifico. La ricerca operativa studia modelli e metodi, basati sulle tecniche introdotte, per l'utilizzo ottimale di risorse scarse (in ambiti produttivi, finanziari, ecc.).

3. Risultati dell'apprendimento attesi:

Lo studente deve apprendere nozioni di geometria di base e le tecniche essenziali per manipolare vettori e matrici in spazi a n dimensioni.

Lo studente deve acquisire la capacità costruire modelli di programmazione lineare sia a variabili continue che a variabili intere partendo dall'enunciato di un problema reale, e la conoscenza dei meccanismi di base e la teoria relativa agli algoritmi che operano su tali modelli.

4. Modalità di verifica dell'apprendimento:

L’esame è costituito da una prova scritta di durata di almeno 2 ore seguita da una prova orale facoltativa.

Regole da seguire durante la partecipazione ad una prova scritta:

- E' vietata la comunicazione, sotto ogni forma, sia tra persone in aula che fuori dall’aula. La violazione di questa regola comporta l’annullamento della prova.
- E’ permesso portare come materiale il formulario del corso.
- E' vietato portare ed utilizzare gli appunti.
- E’ permesso portare la calcolatrice.
- Autovalutazione in 3 passi:
    1. Le soluzioni dell'esame saranno rese disponibili on-line subito dopo la prova scritta.
    2. Lo studente che ha consegnato l'esame ha 24 ore di tempo per decidere di ritirare la propria prova.
    3. Gli studenti che non ritirano lo scritto consegnato, in caso di prova non superata

(ovvero voto<15), consumano 1 delle 3 consegne a disposizione sui 5 appelli disponibili.

Validità dei risultati ottenuti durante le prove che formano l’esame:

- Il voto ottenuto durante una prova rimane valido durante tutto l’Anno Accademico in cui la prova è stata sostenuta.
- La ripetizione di una prova, ovvero presenza effettiva all'appello anche in caso di ritiro, comporta l’annullamento dell’esito della prova precedente.

Regole per il calcolo del voto di un esame:

- La prova scritta viene valutata da 0 a 33 e si considera superata con voto uguali o superiori a 15
- La prova orale viene valutata da -4 a 4.
- Il voto dell’esame si ottiene come somma dei voti ottenuti nelle prove che lo compongono.

5. Modalità d'insegnamento:

Lucidi proiettati in aula e tradizionali spiegazioni alla lavagna.

6. Attività di supporto:

Verranno forniti appunti ed eserciziari, disponibili sulla pagina I-learn del corso.

7. Programma:

1. Vettori e matrici. Operazioni fondamentali.
2. Cenni di geometria in spazi vettoriali.
3. Combinazioni lineari, indipendenza lineare.
4. Insiemi convessi, poliedri.
5. Programmazione lineare.
    - Modellazione.
    - Struttura della regione ammissibile. Soluzioni di base.
    - Algoritmo del simplesso.
    - Dualità.
6. Cenni di programmazione a variabili intere.

8. Testi consigliati e bibliografia:

R.J. Vanderbei, "Linear Programming", Kluwer Academic Publishers (solo per eventuali approfondimenti - reperibile in biblioteca)

Insegnamento	**INF0041 - Computer Forensics: Aspetti Legali e Informatici**

Insegnamento (inglese):	**Computer Forensics: Legal and Computer Aspects**
CFU:	**6**
Settore:	**INF/01 - INFORMATICA** **IUS/02 - DIRITTO PRIVATO COMPARATO**
Periodo didattico:	**0**
Tipologia di Attività Formativa:	**D - libera**
Docenti:	**Cosimo ANGLANO (Titolare)** **Fabio MONTALCINI (Titolare)** **Camillo SACCHETTO (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Per la parte informatica, agli studenti è richiesta una conoscenza di base dei concetti studiati nei corsi di Architettura degli Elaboratori e di Sistemi Operativi. Sono inoltre utili le nozioni trattate nei corsi di Reti e di Sicurezza.

Per la parte giuridica, agli studenti è richiesta una conoscenza di base dei concetti studiati nel corso di Economia e Gestione delle Imprese e Diritto (parte di Diritto).

2. Obiettivi formativi:

La pervasività dell'informatica nella società moderna ha determinato l'insorgenza di fenomeni che in precedenza non erano presenti o lo erano in forma relativamente trascurabile. Da un lato, infatti, i sistemi informatici sono diventati bersaglio di azioni illecite, quali ad esempio le varie forme di cybercrime, il cyber-spionaggio, la sottrazione di know-how aziendale e il sabotaggio di sistemi che sovrintendono a infrastrutture critiche. Dall'altro, i sistemi informatici sono diventati un facilitatore per la commissione di comportamenti illeciti tradizionali, quali il riciclaggio di denaro, il traffico di stupefacenti, le frodi, le molestie, la diffamazione e molto altro. Il sistema giuridico ha dovuto di conseguenza adeguarsi a tali mutamenti, introducendo la cosiddetta "evidenza digitale" (ovvero informazioni aventi valore probatorio memorizzate o trasmesse in formato digitale) nei suoi vari ambiti (civile, penale, tributario e giuslavoristico) quale elemento essenziale per dimostrare la sussistenza di eventuali comportamenti illeciti attuati mediante un dispositivo digitale o aventi tale dispositivo come bersaglio. Ottenere evidenze digitali partendo dai dati grezzi memorizzati su un dispositivo digitale presenta diverse problematiche (quali, ad esempio, l'integrità e l'autenticità dei dati) che richiedono l'adozione di concetti, metodologie e strumenti (sia di tipo informatico, sia di tipo deduttivo) adeguati e specifici per risolverle. Inoltre, l'uso di evidenze digitali nei procedimenti giudiziari richiede l'adozione di prassi legali adeguate allo specifico contesto processuale (civilistico, penalistico, giuslavoristico, tributario). Il corso di "Computer Forensics: aspetti legali e informatici" si pone l'obiettivo di trattare in maniera sistematica le problematiche inerenti l'acquisizione, l'analisi e l'utilizzo di evidenze digitali nei procedimenti giudiziari, nonché di illustrare le diverse soluzioni (sia di natura informatica, sia di natura giuridica) a tali problematiche.

3. Risultati dell'apprendimento attesi:

Per ciò che concerne gli aspetti informatici, lo studente acquisirà la conoscenza dello stato dell'arte nella Computer Forensics relativamente alle metodologie, alle prassi e agli strumenti necessari per l'identificazione, l'acquisizione, la conservazione, l'interpretazione e l'analisi di evidenze digitali memorizzate su dispositivi informatici di vario genere (computer, tablet, smartphone, network appliances, ecc.).

Per ciò che concerne gli aspetti giuridici, lo studente apprenderà il know-how normativo, concettuale ed operativo (sostanziale e processuale) per l'analisi e la verifica degli illeciti (in particolare civili, penali, industriali ed internazionali) attraverso lo studio delle sentenze, dei casi pratici e delle regole concrete fondamentali relative ai principali processi – nazionali ed internazionali – nei quali sono coinvolti l'Informatica ed il Diritto.

Le competenze maturate nel corso, in virtù anche dell'integrazione degli aspetti informatici e giuridici, possono rappresentare una solida base su cui costruire un percorso formativo al termine del quale lo studente potrà intraprendere la carriera del libero professionista oppure operare all'interno di contesti aziendali (pubblici e privati) ovvero in qualità di consulente tecnico del Giudice o delle parti nei processi penali, civili e tributari.

4. Modalità di verifica dell'apprendimento:

L'apprendimento degli argomenti trattati nel corso sarà verificato mediante un esame orale in cui saranno accertate le conoscenze sia degli aspetti informatici che di quelli giuridici trattati nel corso.

5. Modalità d'insegnamento:

Il corso consta di 60 ore di lezioni frontali, suddivise in due parti distinte (da 30 ore cadauna) relative rispettivamente agli aspetti informatici ed a quelli giuridici. Le lezioni delle due parti del corso sono organizzate in maniera sinergica, in conseguenza della stretta interdipendenza degli aspetti informatici e di quelli giuridici: le soluzioni tecnico-informatiche devono infatti rispondere alle esigenze specifiche imposte dal sistema giuridico, mentre le soluzioni giuridiche devono tenere conto sia delle potenzialità offerte dalle metodologie tipiche della Computer Forensics, sia delle loro limitazioni. Pertanto, l'insegnamento delle due parti procederà simultaneamente, distribuendo in modo opportuno le ore di lezione previste in ciascuna settimana sulla base delle specifiche esigenze didattiche richieste dagli argomenti trattati in tale settimana. All'interno delle 60 ore di lezione sono previsti alcuni interventi di carattere seminariale, che saranno definiti durante lo svolgimento del corso, tenuti da esperti operanti nei vari ambiti in cui trova applicazione la Computer Forensics (Polizia Giudiziaria, Magistrati, Avvocati …). Inoltre, nella parte relativa agli aspetti informatici, alle lezioni di carattere metodologico saranno affiancate lezioni in cui saranno effettuate esercitazioni pratiche, nelle quali saranno utilizzati strumenti professionali, hardware e software, per la Computer Forensics.

6. Attività di supporto:

Il corso di avvarrá della piattaforma didattica Moodle, la quale sará utilizzata per ospitare il materiale didattico utilizzato dai docenti durante il corso (slides o altro), nonchè come meccanismo per l'invio di comunicazioni inerenti il corso. Inoltre, tale piattaforma sará utilizzata per la gestione di un forum interattivo, moderato dai docenti, su cui gli studenti potranno avviare discussioni inerenti gli argomenti trattati a lezione o richiedere chiarimenti e approfondimenti ai docenti.

7. Programma:

Parte informatica:

1. Evidenza digitale e computer forensics 2. Metodologie e strumenti per l'acquisizione forense di evidenza digitale: acquisizione di dati memorizzati su computer; acquisizione di dati memorizzati su dispositivi mobili (tablet, smartphone, ecc.); acquisizione di dati trasmessi via rete (intercettazione telematica) 3. Analisi forense dei file system: recupero di file cancellati; analisi dei metadati; identificazione del tipo corretto di file; analisi delle informazioni temporali associate ai file; casi di studio (analisi forense dei file system FAT 32 e NTFS) 4. Analisi forense dei sistemi operativi fissi e mobili: metodologia generale e casi di studio (Windows, Android, iOS) 5. Analisi forense dei dati prodotti da applicazioni utente: metodologia generale e casi di studio (metadati applicativi, cronologia browser Internet, social network, messaggistica istantanea, ecc.) 6. Analisi forense del traffico di rete (cenni) 7. Tecniche di correlazione delle evidenze digitali (cenni) 8. Tecniche di anti-forensic (cenni): cancellazione sicura di file, crittografia, data hiding, reti di anonimizzazione

Parte giuridica:

1. Il concetto di diritto, regola, norma, processo, procedimento, prova 2. La nozione giuridica e tecnico-scientifica di prova digitale, oggetto di prova digitale e fonte di prova digitale (nei principali ambiti giuridici e processuali) 3. Analisi degli aspetti peculiari dei reati informatici (quali ad esempio il danneggiamento informatico, l'accesso abusivo a sistema informatico, violazione del diritto d'autore e cracking, il furto di identità e sostituzione di persona) e dell'"alibi" in ambiente virtuale. 4. Analisi di casi giudiziari penali (italiani e stranieri) finalizzati allo studio delle principali regole processuali con particolare approfondimento dei compiti e delle caratteristiche del Consulente Tecnico 5. Interpretazione della valenza delle prove informatiche nel diritto civile (con particolare disamina degli illeciti compiuti mediante i Social Network e del valore delle investigazioni private effettuate sul web) 6. Diritto dei lavoratori e attività di impresa alla luce delle nuove tecnologie (controlli a distanza e garanzie del lavoratore nell'ambiente digitale; Videosorveglianza, biometria e Privacy) 7. Controlli, accessi, ispezioni e verifiche fiscali e relativa valenza probatoria dei documenti informatici nel Diritto Tributario nazionale ed internazionale (in particolare saranno approfonditi lo scambio automatico di informazioni e dati tra gli Stati e amministrazioni, le nuove forme di residenza e territorialità delle imprese e dei soggetti, le frodi fiscali internazionali attraverso il web, le nuove forme di ricchezza quali criptovalute e Bit Coin)

8. Testi consigliati e bibliografia:

Durante il corso saranno fornite dai docenti dispense, contributi e slide.

Per la parte di informatica, sono inoltre consigliati i seguenti testi di riferimento: 1. J. Sammons. "The Basics of Digital Forensics", seconda edizione, Elsevier/Syngress, 2014, ISBN 9 9780128016350 2. S. Bommisetty, R. Tamma, H. Mahalik. "Practical Mobile Forensics", PACKT Publishing, 2014. ISBN 9781783288311

Per la parte di giuridica, è inoltre consigliato il seguente testo di riferimento: 1. Fabio Montalcini - Camillo Sacchetto, "Diritto Tributario Telematico", Giappichelli, Torino, 2015, ISBN 978-88-3485539-3

Insegnamento	**MFN0604 - Economia e Gestione dell'Impresa e Diritto**

Insegnamento (inglese):	**Management, business administration and computer law**
CFU:	**9**
Settore:	**IUS/02 - DIRITTO PRIVATO COMPARATO** **SECS-P/08 - ECONOMIA E GESTIONE DELLE IMPRESE**
Periodo didattico:	**2**
Tipologia di Attività Formativa:	**C - affine e integrativa**
Docenti:	**Fabio MONTALCINI (Professore a Contratto)** **Marco PIRONTI (Titolare)** **Paola Maria PISANO (Titolare)** **Camillo SACCHETTO (Professore a Contratto)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Il corso non presuppone conoscenze iniziali specifiche

Eventuali corsi propedeutici

NESSUNO

2. Obiettivi formativi:

Modulo di Economia

l'obiettivo del corso è analizzare le caratteristiche basi di una azienda: dal modello di business teorico alla creazione dei processi produttivi, alla comunicazione e vendita dei prodotti.

Verranno studiate le relazioni delle aziende all'interno della catena del valore (Clieti /fornitori) e i loro settori di riferimento. I framework teorici saranno poi applicati per l'analisi di aziende, settori e mercati innovativi al fine di valutare come vengono colte le opportunità che l'innovazione dà alle aziende e ipotizzare sviluppi futuri di modelli. durante il corso gli studenti si confronteranno con casi aziendali reali e con imprenditori. gli studeneti del corso potranno inoltre partecipare al progetto silicon valley study tour e passare una settimana in Silicon Valley dove conoscere le aziende più innovative dell'IT.

Gli studenti hanno inoltre l'opportunità di partecipare: www.ggi-academy.it/

Modulo di Diritto

Agli studenti saranno forniti le nozioni, i metodi, i modelli cognitivi e gli strumenti tecnici necessari alla comprensione dei concetti fondamentali del Diritto dell'Informatica, settore della scienza forense che realizza maggiormente il dialogo teorico e la sintesi operativa tra il diritto (in particolar modo nei suoi profili pratici e processuali) e l'informatica. Ogni lezione, tratterà uno o più argomenti specifici partendo dall’ esame di una vicenda processuale concreta e saranno, pertanto, messe a disposizione degli studenti le conoscenze funzionali alla soluzione del caso operativo e quindi al raggiungimento degli obiettivi didattici di ogni singola tematica. Il Corso si propone l'obbiettivo di analizzare - attraverso l'apprendimento dei principi in materia di privacy e riservatezza, del diritto civile, penale, tributario, del lavoro e industriale - lo studio delle principali metodologie giuridiche e tecnico-informatiche richieste dall’attuale contesto giuridico professionale nonché dai settori, aziende e mercati innovativi.

3. Risultati dell'apprendimento attesi:

Modulo di Economia

CREAZIONE E ANALISI DEI MODELLI DI BUSINESS CREAZIONE DELLE STRATEGIE AZIENDALI ANALISI DEI COMPETITOR, DEL SETTORE, DEL MERCATO.

SOFT SKILL: LAVORO IN TEAM CAPACITA' DI PRESENTARE PROBLEM SOLVING

Modulo di Diritto

Lo studente apprenderà il know-how normativo, concettuale ed operativo (sostanziale e processuale) per l'analisi e la verifica delle principali fattispecie giuridiche inerenti le nuove tecnologie dell'informazione attraverso lo studio delle sentenze, dei casi pratici e delle regole concrete fondamentali relative ai principali procedimenti – nazionali ed internazionali – nei quali sono coinvolti l'Informatica ed il Diritto.

4. Modalità di verifica dell'apprendimento:

Modulo di Economia

70% esame scritto sul materiale delle lezioni 30% lavoro di gruppo i non frequentanti: 70% sul materiale del sito+ libri 30% lavoro individuale.

Modulo di Diritto

L'apprendimento degli argomenti trattati nel corso sarà verificato mediante un esame orale in cui saranno accertate le conoscenze degli aspetti informatico giuridici trattati nel corso.

5. Modalità d'insegnamento:

frontali

6. Attività di supporto:

7. Programma:

Modulo di Economia

business model struttura dell'organizzazione struttura dei processi struttura delle funzioni strategie analisi di settore analisi di mercato

Modulo di Diritto

1) Presentazione tematiche del corso

2) Responsabilità Internet Service Provider

3) Documento informatico e Firme elettroniche

4) Posta Elettronica Certificata (PEC)

5) Diritto d'autore in ambiente web

6) Privacy: Principi generali / Privacy e diritto del lavoro

7) E-commerce e web contracts

8) Identità Digitale e Web Reputation

9) Cookies, marchi, Nomi a dominio

8. Testi consigliati e bibliografia:

Modulo di Economia

Creare modelli di business. Un manuale pratico ed efficace per ispirare chi deve creare o innovare un modello di business Autore Osterwalder Alexander; Pigneur Yves

Modulo di Diritto

Durante il corso saranno fornite dai docenti dispense, contributi e slide. È inoltre consigliato il seguente testo di riferimento: Fabio Montalcini - Camillo Sacchetto (a cura di) "Diritto Tributario Telematico", Giappichelli, Torino, 2016.

Insegnamento	**MFN0617 - Economia e Gestione dell'Innovazione**

Insegnamento (inglese):	**Managing Innovation**
CFU:	**6**
Settore:	**SECS-P/08 - ECONOMIA E GESTIONE DELLE IMPRESE**
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**D - libera**
Docenti:	**Marco PIRONTI (Titolare)** **Paola Maria PISANO (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Conoscenza dei concetti base di Economia e Gestione delle Imprese: analisi di settore analisi di mercato/consumatore modello di business analisi delle organizzazioni strategie delle aziende

Eventuali corsi propedeutici

Corso di Economia e Gestione delle Imprese

2. Obiettivi formativi:

Questo corso ha l'obiettivo di fornire gli skill, tools e approcci per la creazione di un' innovazione sostenibile e di successo nell'era digitale. il corso è multidisciplinare e costruito sulle ultime ricerche legate ai framework di gestione dell'innovazione, sulle metodologie lean per la creazione dei prototipi, sull'additive manufacturing e sull'approccio del design thinking. Insieme, esploreremo come identificare le opportunità dell'economia digitale e creare un'innovazione che va ben oltre il semplice ricopiare o reinventare un prodotto o servizio risolvendo problemi reali della nostra società quali creazione di lavoro, diminuzione di side effect eliminazione di colli di bottiglia all'interno di prodotti, modelli e settori. identificheremo inoltre framework informatici per facilitare l'identificazione,lo sviluppo e la creazione di innovazione in modo economico e sostenibile. l'obiettivo del corso non sarà solo acquistare gli strumenti teorici per creare un innovazione ma anche riuscire ad applicarli per arrivare a creare un'innovazione di prodotto o servizio all'interno di un mercato di riferimento Gli studenti del corso parteciperanno alla European innovation Accademy

3. Risultati dell'apprendimento attesi:

capacità di gestire un processo di creazione dell'innovazione capacità di utilizzare diversi approcci quali: 1)job to be done per il riconoscimento dell'innovazione e dei clienti 2)lean methodology: per la creazione del MINIMUM VIABLE PRODUCT e i testing dei prototipy 3)design thinking: per la comunicazione del prodotto e strumenti

4. Modalità di verifica dell'apprendimento:

le modalità d'esame sono differenti a seconda che lo studente sia frequentante o non frequentate. 1) modalità di esame per i frequentanti 70% valutazione attraverso una prova scritta 30% progetto di gruppo e partecipazione attiva in aula 2)non frequentanti: 70% valutazione attraverso una prova scritta 30%progetto individuale

5. Modalità d'insegnamento:

Le elzioni si basano su lezioni teoriche e laboratori pratici di: tecnologia per la costruzione di prototipi lato software/hardware software di statistica per la definizione dei segmenti di mercato (cluster analysis)

6. Attività di supporto:

le slide del corso verranno inserite sul sito del corso

7. Programma:

---il consumatore ---l'individuazione dell'opportunità ---la value proposition ---creazione del prototipo --Test Lean ---sperimentazione ---design/comunicazione e visualizzazione

8. Testi consigliati e bibliografia:

"what customers want" Tony Ulwich

Insegnamento **MFN0600 - Elementi di Probabilità e Statistica**

Insegnamento (inglese):	**Foundations of Probability and Statistics**
CFU:	**6**
Settore:	**MAT/06 - PROBABILITA\' E STATISTICA MATEMATICA**
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**C - affine e integrativa**
Docenti:	**Marco CAPPIELLO (Titolare)** **Gianluca GARELLO (Titolare)** **Joerg SEILER (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Elementi di teoria intuitiva degli insiemi (relazioni ed operazioni con gli insiemi, insieme delle parti, partizione di un insieme).

Elementi di calcolo combinatorio.

Elementi di analisi matematica (principalmente: limiti, integrali serie numeriche)

Eventuali corsi propedeutici

Matematica discreta e logica, Analisi matematica.

2. Obiettivi formativi:

Comprendere e saper utilizzare i concetti di evento aleatorio, probabilitÃ e variabile aleatoria. Saper riconoscere ed utilizzare le principali distribuzioni di probabilitÃ utilizzate ai fini statistici: Distribuzione binomiale, di Poisson, Normale, t-di Student, chi- Quadro

Apprendere e saper manipolare le principali tecniche di Statistica Inferenziale: stime puntuali, intervalli di confidenza.

Saper utilizzare i principali test di ipotesi.

3. Risultati dell'apprendimento attesi:

Saper leggere dati statistici e individuarne i principali indicatori quantitativi.

Saper effettuare le principali stime di parametri statistici ed inmpostare semplici test di ipotesi.

4. Modalità di verifica dell'apprendimento:

L'esame e' costituito da una prova scritta, costituita da domande a risposta chiusa ed esercizi da svolgere nei dettagli.

Gli studenti che intendono presentarsi ad una prova devono obbligatoriamente registrarsi in rete secondo la procedura di Ateneo.

la prova d'esame ha la durata di 90 minuti.

5. Modalità d'insegnamento:

Lezioni ed esercitazioni frontali in aula

6. Attività di supporto:

- File delle esercitazioni e testi di esame saranno disponibili su I-learn.

7. Programma:

- INTRODUZIONE ALLA STATISTICA

Popolazioni e campioni,

- STATISTICA DESCRITTIVA

Campionamenti. Le grandezze che sintetizzano i dati: media, mediana e moda campionarie; varianza e deviazione standard campionarie; percentili campionari e box plot. La disuguaglianza di Chebychev. Insiemi di dati bivariati e coefficiente di correlazione campionaria.

- ELEMENTI DI PROBABILITA'

Spazi degli esiti discreti e continui, eventi . Algebra degli eventi. Assiomi della probabilita'. Spazi di probabilità finiti. Richiami di calcolo combinatorio Probabilita' condizionata. Eventi indipendenti. Fattorizzazione di un evento e formula di Bayes.

- VARIABILI ALEATORIE

Variabili aleatorie discrete e continue. variabili aleatorie indipendenti. Valore atteso e sue proprieta'. Varianza. Covarianza

- MODELLI DI VARIABILI ALEATORIE

Variabili aleatorie di Bernoulli e binomiali. Variabili aleatorie di Poisson. Variabili aleatorie uniformi. Variabili aleatorie normali o gaussiane. Variabili aleatorie esponenziali. Distribuzioni che derivano da quella normale: le distribuzioni chi-quadro; le distribuzioni t-student; le distribuzioni F-Fisher.

- LA DISTRIBUZIONE DELLE STATISTICHE CAMPIONARIE

La media campionaria. Legge dei Grandi numeri, Il Teorema del limite centrale. La varianza capionaria. Le distribuzioni delle statistiche di popolazioni normali.

- STIMA PARAMETRICA

Intervalli di confidenza per la media di una distribuzione normale con varianza nota e con varianza incognita; intervalli di confidenza per la varianza di una distribuzione normale. Intervalli di confidenza approssimati per la media di una distribuzione di Bernoulli

- VERIFICA DELLE IPOTESI

Livelli di significativita'. La verifica di ipotesi sulla media di una popolazione normale quando la varianza e' nota e quando la varianza e' incognita (il test t). Vericare se due popolazioni normali hanno la stessa media. La verifica di ipotesi sulla varianza di una popolazione normale. La verifica dii ipotesi su una popolazione di Bernoulli.

8. Testi consigliati e bibliografia:

Testi consigliati:

Sheldon M. Ross, PROBABILITA' E STATISTICA per l'ingegneria e le scienze" 2a edizione, APOGEO, 2008.

D. Bertacchi, M.Bramanti, G. Guerra, Esercizi di Calolo delle Probabilità e Statistica, Ed progetto Leonardo, Bologna.

Franco Crivellari, Analisi statistica dei dati con R, Apogeo, 2006.

Insegnamento	**MFN0598 - Fisica**

Insegnamento (inglese):	**Physics**
CFU:	**6**
Settore:	**FIS/01 - FISICA SPERIMENTALE**
Periodo didattico:	**2**
Tipologia di Attività Formativa:	**C - affine e integrativa**
Docenti:	**Martino GAGLIARDI (Titolare)** **Ernesto MIGLIORE (Titolare)** **Igor PESANDO (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Competenze di base di trigonometria, di calcolo vettoriale e di analisi matematica.

Eventuali corsi propedeutici

Corsi di Matematica del I anno.

2. Obiettivi formativi:

Il corso si propone di:

1. introdurre alla conoscenza della basi indispensabili di meccanica, delle principali proprietà del campo elettrico e del campo magnetico, con cenni al comportamento della materia soggetta a tali campi;
2. introdurre alla conoscenza del comportamento degli elementi di un circuito in corrente continua ed in corrente alternata;
3. introdurre alla conoscenza delle principali proprietà delle onde elettromagnetiche.

3. Risultati dell'apprendimento attesi:

Lo studente deve avere acquisito la capacità di esprimere i concetti con un formalismo matematico sufficientemente rigoroso e di collegare fra di loro in modo armonico i principi fondamentali della fisica. Deve inoltre essere in grado di fornire una risposta quantitativa ai problemi assegnati.

4. Modalità di verifica dell'apprendimento:

L'esame di Fisica è costituito dalla sola prova scritta. Il formato dell'esame (2.5h a disposizione) è il seguente:

1. 1 esercizio sull'algebra dei vettori
2. 1 esercizio su campo elettrico oppure su campo magnetico + 1 esercizio su circuiti
3. 1 domanda aperta da svolgere (max 1 facciata) tra 2 estratte, al momento dello scritto, da questa lista

Per superare l'esame occorre:

- avere fatto tutto giusto l'esercizio sui vettori
- avere svolto >50% dei due esercizi su campi E, B e circuiti
- avere risposto in modo esaustivo alla domanda aperta

La parte di esercizi e la parte di teoria contribuiscono in egual misura alla determinazione del voto

finale.

All'esame e` possibile utilizzare la calcolatrice.
Non e` consentito l'uso di libri o appunti ma solo di un formulario distribuito dai docenti in sede di esame. Gli scritti assegnati negli ultimi anni e le relative soluzioni sono reperibili al seguente link

5. Modalità d'insegnamento:

Il corso si svolge attraverso lezioni frontali ed esercitazioni svolte dagli stessi docenti.

6. Attività di supporto:

Supporto tramite web
Il corso e` presente sulla piattaforma di e-learning. Il forum del corso e` usato abitualmente per le comunicazioni con gli studenti e per la distribuzione del materiale didattico.
Il materiale relativo alle esercitazioni e` reperibile al link: http://personalpages.to.infn.it/~gagliard/

Tutoraggio in aula
Vengono fornite ore di tutoraggio (2 ore/settimana) in cui si ha la possibilità di svolgere gli esercizi sotto la guida di un tutore. Gli esercizi proposti sono scelti in stretto rapporto del docente del corso con il tutore.

TUTORATO a.a. 2016-2017
Tutorato, II semestre (Aula da definirsi).

7. Programma:

Meccanica del punto materiale. Richiami di cinematica. Leggi di Newton. Lavoro ed energia. Forze conservative.

Il campo elettrostatico. Teorema di Gauss. Conservatività del campo elettrostatico. Superfici equipotenziali. Conduttori e dielettrici. Capacità elettrica di un conduttore. Condensatori. Densità di energia del campo elettrico. Correnti elettriche. Leggi di Ohm e di Kirchhoff. Circuiti RC.

Il campo magnetico indipendente dal tempo. Magneti. Moto di una carica in campo magnetico; esempi ed applicazioni. Filo percorso da corrente in campo magnetico. Campo magnetico generato da un filo percorso da corrente. Teorema di Ampere. Teorema di Gauss per il campo magnetico.

Campi elettrici e magnetici variabili nel tempo. Induzione elettromagnetica. Legge di Faraday-Henry. Correnti alternate. Legge di Ampere-Maxwell. Autoinduzione. Induttanza del solenoide ideale. Densità di energia del campo magnetico. Circuiti RL. Elementi circuitali in corrente alternata.

Equazioni di Maxwell e onde elettromagnetiche.

Il programma dettagliato relativo al corso sara` pubblicato dai docenti a fine corso.

E` possibile consultare anche il registro delle lezioni dell'A.A. 2015-2016:

- programma dettagliato corso A (A.A. 2015/2016) pdf
- programma dettagliato corso B (A.A. 2012/2013) pdf

8. Testi consigliati e bibliografia:

Testi consigliati (tra cui scegliere):

1. WOLFSON, Fisica, vol.2 - Elettromagnetismo, ottica e fisica moderna, Pearson
2. RESNICK/HALLIDAY/KRANE, Fisica, vol. 2, Casa Editrice Ambrosiana
3. SERWAY/JEWETT Fisica. Per scienze ed ingegneria, vol. 2, EdiSes

Insegnamento **MFN1353 - Interazione Uomo Macchina**

Insegnamento (inglese):	**Human-computer Interaction**
CFU:	**6**
Settore:	
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**D - libera**
Docenti:	**Giovanni SACCO (Titolare)** **Marino SEGNAN (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Si richiede una buona conoscenza delle basi di dati (fornita dal corso di Basi di Dati), dell'analisi e della progettazione object-oriented (fornita dai corsi di Programmazione II e di Algoritmi e strutture dati) e dei fondamenti della programmazione distribuita (fornita dal corso di Programmazione III). Lo studente deve inoltre avere la capacità di scrivere, compilare e verificare la correttezza di programmi in Java.

Eventuali corsi propedeutici

Basi di dati e sperimentazioni, Algoritmi e sperimentazioni, Programmazione II and III.

2. Obiettivi formativi:

Nella progettazione e sviluppo di un’applicazione software si devono tenere in conto due aspetti fondamentali: (i) l’interazione con l’utente che, indipendentemente dall’efficacia delle funzionalità offerte da un’applicazione, influenza sensibilmente il suo successo in termini di utilizzo. L’applicazione deve essere usabile ed accessibile per permettere un uso agevole del software e prevenirne l’abbandono da parte degli utenti. (ii) l’implementazione dell’applicativo, che richiede di guardare “dietro all’interfaccia utente” per andare a fondo su aspetti architetturali e tecnologici che possono influenzare non solo le prestazioni dell’applicazione, la sua scalabilità e robustezza, ma anche le tipologie di servizio che possono effettivamente essere offerte.

Partendo da queste considerazioni, il corso si pone un duplice obiettivo: da una parte, fornire la conoscenza di base necessaria per il disegno corretto di interazioni uomo-macchina, che sta alla base della progettazione di applicazioni di ogni genere (web e non, mobili, etc.). Dall’altra, concentrandosi sulle applicazioni mobili, che sono oggetto specifico del corso, fornire la conoscenza di base necessaria per la progettazione e lo sviluppo di applicazioni interattive, accessibili da terminali mobili (come per esempio gli smart phone), e caratterizzate da una logica applicativa mediamente complessa.

Più precisamente, per quanto riguarda l'interazione uomo-macchina, verranno acquisite competenze sia teoriche sia pratiche nel disegno corretto di interazioni, con specifico riferimento alla progettazione user-centered. Per quanto riguarda gli aspetti architetturali e tecnologici, il corso tratterà dal punto di vista sia teorico che pratico la programmazione lato client per device mobili su piattaforma Android e lo sviluppo di interfacce mobili. Per permettere agli studenti di sperimentare le nozioni apprese durante le ore di teoria in aula il corso include una sostanziale parte di laboratorio. I temi introdotti durante il laboratorio corredano e integrano le conoscenze derivanti dalla parte teorica (knowledge and understanding) e permettono agli studenti di familiarizzare con le metodologie e tecnologie introdotte, anche investigando soluzioni alternative (applying knowledge and understanding).

Inoltre durante le ore di laboratorio è previsto lo sviluppo di un’applicazione realistica con interfaccia mobile ed una introduzione al linguaggio Python. La preparazione e la discussione del progetto di laboratorio sono inoltre volte a stimolare le capacità di organizzare il lavoro in piccoli

gruppi (max 4 studenti), e poi di illustrare verbalmente le soluzioni adottate (communication skills).

3. Risultati dell'apprendimento attesi:

Lo studente dovrà avere assimilato: la progettazione user-centered, la prototipazione, la valutazione dell'usabilità e il disegno inclusivo per categorie speciali di utenti. Inoltre dovrà avere assimilato i concetti rilevanti per analizzare l'interazione: affordance, modelli concettuali, metafore, feedback, etc.

Lo studente acquisirà anche la conoscenza delle varie architetture di riferimento per lo sviluppo di applicazioni web e mobile, e dei loro pro e contro; inoltre lo studente acquisirà la conoscenza dei modelli più comunemente adottati per gestire dialoghi mediamente complessi e articolati tra utente e applicazione, e delle tecnologie attualmente utilizzate per l'implementazione delle applicazioni e delle loro interfacce (web e mobile). Lo studente dovrà essere in grado di sviluppare applicazioni lato client in ambiente Android e sviluppare un programma scritto in linguaggio Python.

4. Modalità di verifica dell'apprendimento:

L'esame è composto da un TEST SCRITTO e da una VERIFICA DI LABORATORIO. Le due prove possono essere sostenute in qualsiasi ordine (cioè, non è necessario aver dato lo scritto per fare la prova di laboratorio, né il vice versa).

TEST SCRITTO: Prova scritta che include esercizi e domande teoriche sul programma del corso. Viene valutata da un minimo di 0 ad un massimo di 15 e si considerano sufficienti i voti >=9. Durante la prova è proibito comunicare con altre persone, presenti in aula o fuori. Inoltre, non si può portare alcun tipo di materiale didattico (appunti, libri, dispense, etc.) ed è vietato usare computer, telefonini o simili. Come da regolamento di Ateneo, ogni studente può sostenere un numero massimo di tre prove scritte durante l'Anno Accademico (cioè, consegnare il proprio elaborato tre volte). Il voto ottenuto durante un test scritto decade se lo studente partecipa ad un altro test scritto e consegna il suo elaborato.

VERIFICA DI LABORATORIO: prevede la discussione del progetto di laboratorio svolto durante il corso. La discussione deve essere effettuata preferibilmente in unica soluzione, con tutti i membri del gruppo di laboratorio presenti. Il voto di laboratorio è un numero compreso tra 0 e 15, si considerano sufficienti i voti >=9. CALCOLO DEL VOTO FINALE DI ESAME: Sia X il voto del test scritto; sia Y il voto di laboratorio. Il voto Fin finale dell'esame si ottiene come segue: Fin = (X+Y)

Note: i voti acquisiti durante la prova di laboratorio, o durante il test scritto, rimangono validi fino al termine della terza sessione d'esame (quella che precede l'inizio del nuovo corso). Quando si superano entrambe le prove, è necessario registrare il voto finale entro i limiti imposti dal Regolamento di Ateneo.

5. Modalità d'insegnamento:

L'insegnamento è diviso in una parte di teoria e una di laboratorio. Per la parte di teoria sono previste 30 ore di lezione frontali che seguono il programma riportato più avanti, integrate da casi di studio e da esercitazioni volte ad illustrare l'applicazione pratica dei concetti appena studiati.

La parte di laboratorio consiste di 30 ore ed è focalizzata sulla programmazione client-side in ambiente Android utilizzando le API Java del sistema operativo. Verra' inoltre sviluppata un' applicazione scritta in linguaggio Python . Le lezioni si svolgono in maniera interattiva e sono corredate da vari esercizi miranti a fornire esempi pratici.

Le sperimentazioni che vengono effettuate durante le ore di laboratorio, strutturate come sequenze di esercizi specifici, sono fondamentali per aiutare gli studenti a comprendere e assimilare i contenuti teorici spiegati a lezione in quanto permettono di mettere in pratica i concetti e le metodologie illustrate su esempi concreti. Inoltre lo sviluppo del progetto di laboratorio permette di consolidare le conoscenze teoriche in un caso realistico di media complessità.

Si consiglia caldamente la frequenza costante alle lezioni di teoria e di laboratorio. E' inoltre

fondamentale che gli studenti si iscrivano al corso online su I-Learn, all’interno del quale i docenti mettono a disposizione materiale didattico di supporto.

6. Attività di supporto:

Vengono forniti on-line i lucidi delle lezioni, link a documentazione disponibile sul Web ed alcuni testi degli esami scritti degli anni precedenti.

Il materiale didattico di supporto (lucidi delle lezioni, link a documentazione, esempi di testo di esami ed altro) è disponibile presso il supporto on-line ai corsi I-learn .

Si noti che i lucidi non sostituiscono il libro di testo, ne' il materiale integrativo ad esso affiancato.

7. Programma:

- Parte Ia - Human-computer interaction (HCI)
    - Human-computer interaction (HCI): Definizioni e contesto, evoluzione di HCI, nuove direzioni.
    - Il fattore umano: percezione (gestalt e affordance), attenzione e memoria, modelli mentali, metafore, il modello di Shneiderman e il modello di Norman.
    - Disegno di interazioni: user-centered design, requisiti funzionali e di usabilità (raccolta, analisi, presentazione), prototipazione, linee guida (con gestione degli errori ed assistenza agli utenti), elementi di tipografia elettronica, di layout e gestione del colore.
    - Tecniche di valutazione: valutazione senza utenti (quantitativa e qualitativa), valutazione con utenti, problemi, presentazione dei risultati.
    - Disegno inclusivo: accessibilità, disegno per utenti di differenti gruppi di età (bambini, anziani), internazionalizzazione.
- Parte Ib - Programmazione di device mobili.
    - Introduzione alla programmazione per mobile.
    - La piattaforma Android e sua architettura.
    - Processi e applicazioni in Android.
    - Il linguaggio Python:differenze rispetto a Java
    - Progettazione di una interfaccia utente in maniera programmatica e dichiarativa.
    - Sviluppare con Python o Java? Confronto tra gli ambienti ed esempi
    - Esempio di sviluppo del lato client di una semplice app per Android.

8. Testi consigliati e bibliografia:

Libri di Testo:

- Polillo, R. - FACILE DA USARE, Edizioni Apogeo, 2010
- Programmazione Web lato Server, di V. Della Mea, L. Di Gaspero, I. Scagnetto, Apogeo, 2007
- Android Training. http://developer.android.com/training/index.html
- Python. http://pythonspot.com/beginner/

Altri testi (per consultazione):

- La caffettiera del masochista, di D. Norman, Apogeo

Insegnamento	**MFN0608 - Interazione Uomo Macchina e Tecnologie Web**

Insegnamento (inglese):	**Human-computer Interaction and Web Technologies**
CFU:	**12**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**B - caratterizzante**
Docenti:	**Liliana ARDISSONO (Titolare)** **Giovanni SACCO (Titolare)** **Marino SEGNAN (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Si richiede una buona conoscenza delle basi di dati (fornita dal corso di Basi di Dati), dei sistemi operativi (fornita dal corso di Sistemi Operativi), dell'analisi e della progettazione object-oriented (fornita dai corsi di Programmazione II e di Algoritmi e strutture dati) e dei fondamenti della programmazione distribuita (fornita dal corso di Programmazione III). Lo studente deve inoltre avere la capacità di scrivere, compilare e verificare la correttezza di programmi in Java.

Eventuali corsi propedeutici

Basi di Dati, Sistemi Operativi, Programmazione II, Algoritmi e strutture dati, Programmazione III

2. Obiettivi formativi:

Nella progettazione e sviluppo di un'applicazione software si devono tenere in conto due aspetti fondamentali: (i) l'interazione con l'utente che, indipendentemente dall'efficacia delle funzionalità offerte da un'applicazione, influenza sensibilmente il suo successo in termini di utilizzo. L'applicazione deve essere usabile ed accessibile per permettere un uso agevole del software e prevenirne l'abbandono da parte degli utenti. (ii) l'implementazione dell'applicativo, che richiede di guardare "dietro all'interfaccia utente" per andare a fondo su aspetti architetturali e tecnologici che possono influenzare non solo le prestazioni dell'applicazione, la sua scalabilità e robustezza, ma anche le tipologie di servizio che possono effettivamente essere offerte.

Partendo da queste premesse, il corso si pone un duplice obiettivo: da una parte, fornire la conoscenza di base necessaria per il disegno corretto di interazioni uomo-macchina, che sta alla base della progettazione di applicazioni di ogni genere (web e non, mobili, etc.). Dall'altra, concentrandosi sulle applicazioni web e mobili, che sono oggetto specifico del corso, fornire la conoscenza di base necessaria per la progettazione e lo sviluppo di applicazioni Web interattive, accessibili da terminali desktop e mobili (come per esempio gli smart phone), e caratterizzate da una logica applicativa mediamente complessa.

Più precisamente, per quanto riguarda l'interazione uomo-macchina, verranno acquisite competenze sia teoriche sia pratiche nel disegno corretto di interazioni, con specifico riferimento alla progettazione user-centered. Per quanto riguarda gli aspetti architetturali e tecnologici, il corso tratterà dal punto di vista sia teorico che pratico: (i) lo sviluppo di pagine web statiche e dinamiche con HTML5-JavaScript, CSS, e cenni a PHP; (ii) la programmazione lato server, utilizzando tecnologie Web in ambiente Java; (iii) la programmazione lato client per device mobili su piattaforma Android e lo sviluppo di interfacce mobili per applicazioni web. Inoltre, il corso tratterà la tecnologia XML, data la sua importanza nella condivisione di informazioni in Internet e nella gestione di interoperabilità tra applicazioni.

Per permettere agli studenti di sperimentare le nozioni apprese durante le ore di teoria in aula il

corso include una sostanziale parte di laboratorio. I temi introdotti durante il laboratorio corredano e integrano le conoscenze derivanti dalla parte teorica (knowledge and understanding) e permettono agli studenti di familiarizzare con le metodologie e tecnologie introdotte, anche investigando soluzioni alternative (applying knowledge and understanding). Inoltre durante le ore di laboratorio è previsto lo sviluppo di un'applicazione realistica con interfaccia sia web che mobile. La preparazione e la discussione del progetto di laboratorio sono inoltre volte a stimolare le capacità di organizzare il lavoro in piccoli gruppi (max 3 studenti), e poi di illustrare verbalmente le soluzioni adottate (communication skills).

3. Risultati dell'apprendimento attesi:

Lo studente acquisira` conoscenza su:

- progettazione user-centered, la prototipazione, la valutazione dell'usabilità e il disegno inclusivo per categorie speciali di utenti.
- concetti rilevanti per analizzare l'interazione: affordance, modelli concettuali, metafore, feedback, etc.
- architetture di riferimento per lo sviluppo di applicazioni web e mobile
- modelli più usati per gestire applicazioni interattive di media complessita`
- tecnologie attualmente utilizzate per l'implementazione delle applicazioni (back-end e interfacce utente web/mobili).

Lo studente acquisira` la capacita` di sviluppare

- pagine dinamiche PHP
- applicazioni Web a 3 livelli in ambiente Java, utilizzando HTML5-JavaScript, CSS, JSP e Java Servlet
- applicazioni mobili.

Per quanto riguarda questo ultimo punto, lo studente sara' in grado di realizzare una applicazione all'interno del sistema Android che interagisca correttamente coi servizi del sistema operativo e le altre applicazioni residenti. Sarà inoltre in grado di saper scrivere un programma usando i costrutti piu' espressivi del linguaggio Python.

4. Modalità di verifica dell'apprendimento:

L'esame è composto da un TEST SCRITTO e da una VERIFICA DI LABORATORIO. Le due prove possono essere sostenute in qualsiasi ordine.

TEST SCRITTO: Prova scritta che include esercizi e domande teoriche sul programma del corso. Viene valutata da un minimo di 0 ad un massimo di 30 e si considerano sufficienti i voti >=18. Durante la prova è proibito comunicare con altre persone, presenti in aula o fuori. Inoltre, non si può portare alcun tipo di materiale didattico (appunti, libri, dispense, etc.) ed è vietato usare computer, telefonini o simili. Come da regolamento di Ateneo, ogni studente può sostenere un numero massimo di tre prove scritte durante l'Anno Accademico (cioè, consegnare il proprio elaborato tre volte). Il voto ottenuto durante un test scritto decade se lo studente partecipa ad un altro test scritto e consegna il suo elaborato.

VERIFICA DI LABORATORIO: consiste di (1) la discussione di un progetto Web svolto durante il corso, che include una applicazione java con interfaccia Web e mobile Android, e la realizzazione di un programma scritto in linguaggio Python. La discussione deve essere effettuata preferibilmente in unica soluzione, con tutti i membri del gruppo di laboratorio presenti. Il voto di laboratorio è un numero intero compreso tra 0 e 30, si considerano sufficienti i voti >=18.

CALCOLO DEL VOTO FINALE DI ESAME: Sia X il voto del test scritto; sia Y il voto di laboratorio. Il voto Fin finale dell'esame si ottiene come segue: Fin = (X+Y)/2

Note: i voti acquisiti durante la prova di laboratorio, o durante il test scritto, rimangono validi fino al termine della terza sessione d'esame (quella che precede l'inizio del nuovo corso). Quando si superano entrambe le prove, è necessario registrare il voto finale entro i limiti imposti dal Regolamento di Ateneo.

5. Modalità d'insegnamento:

L’insegnamento è diviso in una parte di teoria e una di laboratorio. Per la parte di teoria sono previste 60 ore di lezione frontali che seguono il programma riportato più avanti, integrate da casi di studio e da esercitazioni volte ad illustrare l’applicazione pratica dei concetti appena studiati.

La parte di laboratorio è articolata in due moduli da 30 ore ciascuno. Il primo modulo ha per oggetto l’insegnamento delle tecnologie per lo sviluppo di pagine web statiche e dinamiche (HTML5, CSS e JavaScript) e di applicazioni server-side in ambiente java (JSP e Servlet, con accesso a basi di dati remote attraverso JDBC). Il secondo modulo è suddiviso in due parti: programmazione client-side in ambiente Android utilizzando il linguaggio Java; scrittura di un programma in Python. Le lezioni si svolgono in maniera interattiva e sono corredate da vari esercizi miranti a fornire esempi pratici.

Le sperimentazioni che vengono effettuate durante le ore di laboratorio, strutturate come sequenze di esercizi specifici, sono fondamentali per aiutare gli studenti a comprendere e assimilare i contenuti teorici spiegati a lezione in quanto permettono di mettere in pratica i concetti e le metodologie illustrate su esempi concreti. Inoltre lo sviluppo del progetto di laboratorio permette di consolidare le conoscenze teoriche in un caso realistico di media complessità.

Si consiglia caldamente la frequenza costante alle lezioni di teoria e di laboratorio. È inoltre fondamentale che gli studenti si iscrivano al corso online su I-Learn, all’interno del quale i docenti mettono a disposizione materiale didattico di supporto.

6. Attività di supporto:

Vengono forniti on-line i lucidi delle lezioni, link a documentazione disponibile sul Web ed alcuni testi degli esami scritti degli anni precedenti.

Il materiale didattico di supporto (lucidi delle lezioni, link a documentazione, esempi di testo di esami ed altro) è disponibile presso il supporto on-line ai corsi I-learn (vd. le pagine dei corsi (i) Interazione Uomo Macchina e (ii) Interazione Uomo Macchina e Tecnologie Web - parte tecnologica).

Si noti che i lucidi non sostituiscono il libro di testo, ne' il materiale integrativo ad esso affiancato.

7. Programma:

PARTE 1A: HUMAN-COMPUTER INTERACTION (HCI)

- Human-computer interaction (HCI): Definizioni e contesto, evoluzione di HCI, nuove direzioni.
- Il fattore umano: percezione (gestalt e affordance), attenzione e memoria, modelli mentali, metafore, il modello di Shneiderman e il modello di Norman.
- Disegno di interazioni: user-centered design, requisiti funzionali e di usabilità (raccolta, analisi, presentazione), prototipazione, linee guida (con gestione degli errori ed assistenza agli utenti), elementi di tipografia elettronica, di layout e gestione del colore.
- Tecniche di valutazione: valutazione senza utenti (quantitativa e qualitativa), valutazione con utenti, problemi, presentazione dei risultati.
- Disegno inclusivo: accessibilità, disegno per utenti di differenti gruppi di età (bambini, anziani), internazionalizzazione.

PARTE 1B - PROGRAMMAZIONE DI DEVICE MOBILI

- Il sistema operativo Android e le sue peculiarita'. Gli elementi costitutivi di una applicazione utente ed il suo ciclio di vita. Le pratiche consolidate e l'ambiente di sviluppo di una applicazione Android.
- Il linguaggio avanzato Python: principali differenze rispetto al linguaggio Java e le sue caratteristiche piu' interessanti.

PARTE 2 - APPLICAZIONI WEB E TECNOLOGIE DI SUPPORTO

- Architetture delle applicazioni Web: Web browser e Web server.

- Il Pattern Model View Controller per le applicazioni Web - Progettazione e sviluppo di applicazioni Web a 3 livelli basate su MVC:
    - Il primo livello (client dell'applicazione): HTML5, CSS, scripting lato client (JavaScript e AJAX), Java Server Pages (JSP). Raccolta dati (via HTML form).
    - Il secondo livello (logica applicativa): Servlet Java.
    - Il terzo livello (livello dei dati): JavaBeans. Accesso a database relazionali: Java Database Connectivity (JDBC), rappresentazione e gestione di informazioni con XML (XML Schema e DTD, XPath, XML Parsers).
- Introduzione al linguaggio PHP per lo sviluppo di pagine web dinamiche.

8. Testi consigliati e bibliografia:

PARTE 1A:

- Polillo, R. - FACILE DA USARE, Edizioni Apogeo, 2010

PARTE 1B:

- Android: http://www.html.it/guide/guida-android/
- Python: http://www.python.it/doc/newbie/

PARTE 2:

- Programmazione Web lato Server, di V. Della Mea, L. Di Gaspero, I. Scagnetto, Apogeo, 2007

ALTRI RIFERIMENTI (per consultazione):

- La caffettiera del masochista, di D. Norman, Apogeo
- Programmazione Java - tecniche avanzate, di Deitel e Deitel. Ed. Pearson - Prentice Hall
- Principi di Web design, di Joe Sklar, Apogeo

Insegnamento	**MFN0590 - Lingua Inglese I**

Insegnamento (inglese):	**English I**
CFU:	**3**
Settore:	**L-LIN/12 - LINGUA E TRADUZIONE - LINGUA INGLESE**
Periodo didattico:	**1** **2**
Tipologia di Attività Formativa:	**E - prova finale e lingua straniera**
Docenti:	**Francesca CORDERO (Titolare)** **Jeanne Marie GRIFFIN (Esercitatore)** **Daniele Paolo RADICIONI (Titolare)** **Maddalena ZACCHI (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Nessuna.

2. Obiettivi formativi:

Corso di base di Inglese orientato alla comprensione dei testi a alla grammatica di base.

3. Risultati dell'apprendimento attesi:

Al termine dell'insegnamento lo studente deve essere in grado di comprendere e produrre brevi frasi in lingua inglese e leggere semplici testi di argomento scientifico-informatico.

4. Modalità di verifica dell'apprendimento:

Il test finale e` svolto nei laboratori informatici mediante il sistema SET. La prova di Inglese I per informatica e` articolata in due parti, A e B. La parte A (durata 60 min.) deve essere necessariamente superata per poter accedere alla parte B. Il superamento della parte B (durata 30 min.) completa la prova. La valutazione e` automatica e comunicata immediatamente dal sistema allo studente. In ogni appello si svolgono entrambe le parti, in successione; gli studenti che hanno gia` superato la prima parte in un appello precedente devono dare solo la seconda.

Ulteriori informazioni.

Attenzione: Il CCS in Informatica riconosce le certificazioni PET e/o FIRST per *Lingua Inglese I*. Chi non avesse compilato e consegnato l'apposito modulo APU in Segreteria Studenti puo` richiedere il riconoscimento direttamente ai responsabili del corso (a partire da gennaio 2009), presentandosi con il diploma ottenuto e lo statino di *Lingua Inglese I*.

5. Modalità d'insegnamento:

Le lezioni sono svolte in aula dall'esperto linguistico (Dott.ssa Jeanne Griffin).

6. Attività di supporto:

Sono disponibili dispense redatte dall'esperto linguistico.

Le dispense aggiornate sono disponibili anche su I-learn.

7. Programma:

- Grammatica di base

- Lessico
- Pratica

Informazioni dettagliate su I-learn.

8. Testi consigliati e bibliografia:

- New English File Elementary, Oxford University Press
- Essential English Grammar in Use (B1), by Raymond Murphy and Lelio Pallini
- English Grammar in Use (B2), by Raymond Murphy
- Discussions A-Z, Cambridge University Press

Insegnamento	**MFN0610 - Linguaggi e Paradigmi di Programmazione**

Insegnamento (inglese):	**Programming languages and paradigms**
CFU:	**9**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**B - caratterizzante**
Docenti:	**Mariangiola DEZANI (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Conoscenza delle basi della matematica discreta, della programmazione e dei linguaggi formali.

Eventuali corsi propedeutici

Matematica discreta e logica; Programmazione 1.

2. Obiettivi formativi:

L’insegnamento si propone di fornire allo studente una introduzione alla programmazione nell’ambito del paradigma funzionale, con riferimento al linguaggio Haskell; ai sistemi di tipi per i linguaggi funzionali con i relativi algoritmi di inferenza; e ad alcune caratteristiche linguistiche avanzate di tali linguaggi, in particolare le monadi.

3. Risultati dell'apprendimento attesi:

Al termine dell’insegnamento lo studente dovrà dimostrare di avere compreso i principi alla base della programmazione funzionale. Dovrà quindi dimostrare di conoscere le principali tecniche di programmazione utilizzate in tale paradigma e le strutture di dati impiegate (in particolare, le liste), utilizzandole per impostare e risolvere problemi di programmazione. Inoltre, lo studente dovrà dimostrare di sapere fornire, seppure a grandi linee, argomentazioni relative alla correttezza dei programmi realizzati.

4. Modalità di verifica dell'apprendimento:

Esame orale con eventuale seminario ed esercizi su parte teorica e laboratorio. Rispetto alla versione del corso da 6 cfu, la prova di esame comprende esercizi relativi agli argomenti aggiuntivi (monadi).

5. Modalità d'insegnamento:

Lezioni ed esercitazioni in aula ed esercitazioni in laboratorio, in entrambi i casi con sessioni interattive sia su carta che al computer.

6. Attività di supporto:

Non previste.

7. Programma:

Nel seguente elenco di argomenti non viene fatta distinzione tra argomenti svolti in aula ed argomenti svolti in laboratorio, che sono peraltro strettamente connessi. • Introduzione ai paradigmi di programmazione: programmazione imperativa, orientata agli oggetti, logica e funzionale; • Calcolo come riscrittura: le basi dell'esecuzione dei programmi funzionali; •

Espressioni e loro tipi. Tipi di base; • Progettazione di programmi funzionali. Tecniche di ricorsione; • Liste e funzioni del prim'ordine su liste; • Ragionare su programmi funzionali: tecniche di induzione; • Testing di programmi Haskell con QuickCheck; • L'idea di astrazione funzionale. Funzioni di ordine superiore e pattern di calcolo; • Alberi e tipi algebrici generali.

Per la versione da 9 cfu, i seguenti argomenti addizionali: • La valutazione lazy. Strutture di dati potenzialmente infinite. Coinduzione; • Monadi e programmazione imperativa;

8. Testi consigliati e bibliografia:

• Graham Hutton, "Programming in Haskell", Cambridge University Press, 2007. • Simon Thompson, "Haskell: the Craft of Functional Programming" (terza edizione), Addison-Wesley, 2011. • Richard Bird, "Introduction to Functional Programming using Haskell" (seconda edizione), Prentice Hall, 1998. • Bryan O'Sullivan, Don Stewart, and John Goerzen, "Real World Haskell", O'Reilly, 2007. • Miran Lipovača " Learn You a Haskell for Great Good!", 2011.

Insegnamento	**MFN1354 - Linguaggi e Paradigmi di Programmazione**

Insegnamento (inglese):	**Programming languages and paradigms**
CFU:	**6**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**D - libera**
Docenti:	**Mariangiola DEZANI (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Conoscenza delle basi della matematica discreta, della programmazione e dei linguaggi formali.

Eventuali corsi propedeutici

Matematica discreta e logica; Programmazione 1.

2. Obiettivi formativi:

L’insegnamento si propone di fornire allo studente una introduzione alla programmazione nell’ambito del paradigma funzionale, con riferimento al linguaggio Haskell; ai sistemi di tipi per i linguaggi funzionali con i relativi algoritmi di inferenza; e ad alcune caratteristiche linguistiche avanzate di tali linguaggi, in particolare le monadi.

3. Risultati dell'apprendimento attesi:

Al termine dell’insegnamento lo studente dovrà dimostrare di avere compreso i principi alla base della programmazione funzionale. Dovrà quindi dimostrare di conoscere le principali tecniche di programmazione utilizzate in tale paradigma e le strutture di dati impiegate (in particolare, le liste), utilizzandole per impostare e risolvere problemi di programmazione. Inoltre, lo studente dovrà dimostrare di sapere fornire, seppure a grandi linee, argomentazioni relative alla correttezza dei programmi realizzati.

4. Modalità di verifica dell'apprendimento:

Esame orale con eventuale seminario ed esercizi su parte teorica e laboratorio.

5. Modalità d'insegnamento:

Lezioni ed esercitazioni in aula ed esercitazioni in laboratorio, in entrambi i casi con sessioni interattive sia su carta che al computer.

6. Attività di supporto:

Non previste.

7. Programma:

Nel seguente elenco di argomenti non viene fatta distinzione tra argomenti svolti in aula ed argomenti svolti in laboratorio, che sono peraltro strettamente connessi.

• Introduzione ai paradigmi di programmazione: programmazione imperativa, orientata agli oggetti, logica e funzionale; • Calcolo come riscrittura: le basi dell'esecuzione dei programmi funzionali; • Espressioni e loro tipi. Tipi di base; • Progettazione di programmi funzionali. Tecniche

di ricorsione; • Liste e funzioni del prim'ordine su liste; • Ragionare su programmi funzionali: tecniche di induzione; • Testing di programmi Haskell con QuickCheck; • L'idea di astrazione funzionale. Funzioni di ordine superiore e pattern di calcolo; • Alberi e tipi algebrici generali.

Per la versione da 9 cfu, i seguenti argomenti addizionali: • La valutazione lazy. Strutture di dati potenzialmente infinite. Coinduzione; • Monadi e programmazione imperativa;

8. Testi consigliati e bibliografia:

• Graham Hutton, "Programming in Haskell", Cambridge University Press, 2007. • Simon Thompson, "Haskell: the Craft of Functional Programming" (terza edizione), Addison-Wesley, 2011. • Richard Bird, "Introduction to Functional Programming using Haskell" (seconda edizione), Prentice Hall, 1998. • Bryan O'Sullivan, Don Stewart, and John Goerzen, "Real World Haskell", O'Reilly, 2007. • Miran Lipovača " Learn You a Haskell for Great Good!", 2011.

Insegnamento **MFN0603 - Linguaggi Formali e Traduttori**

Insegnamento (inglese):	**Formal Languages and Compilers**
CFU:	**9**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**B - caratterizzante**
Docenti:	**Mario COPPO (Titolare)** **Viviana PATTI (Titolare)** **Jeremy James SPROSTON (Titolare)** **Maddalena ZACCHI (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Lo studente deve avere familiarità con i concetti fondamentali della teoria degli insiemi e della progettazione di algoritmi iterativi e ricorsivi. Deve inoltre aver acquisito capacità di programmare in linguaggi ad alto livello.

Eventuali corsi propedeutici

Le competenze richieste per una proficua frequenza del corso sono fornite dagli insegnamenti: Programmazione I e laboratorio, Programmazione II e laboratorio, Architettura degli elaboratori, Matematica Discreta e Logica.

2. Obiettivi formativi:

Conoscenze nel campo della descrizione formale dei linguaggi e della traduzione (in particolare della compilazione) sono sempre state considerate fondamentali nel bagaglio culturale di un informatico e non possono essere ignorate dagli addetti al settore. Competenze di questo tipo si trovano nei curricula di orientamento informatico-matematico delle Università di tutto il mondo. I linguaggi di programmazione si sono evoluti presentando nuovi problemi di compilazione che hanno portato allo sviluppo di metodi generali per affrontarli. Buona parte della tecnologia di “front-end” dei compilatori, come grammatiche, espressioni regolari, parsificatori e traduttori guidati dalla sintassi, trovano anche applicazione in tutti i programmi in cui sia richiesta l’analisi strutturale di un testo o, in generale, di dati in cui si debba individuare una struttura.

L’insegnamento si propone pertanto di fornire allo studente una visione introduttiva dei problemi connessi alla definizione e alla traduzione dei linguaggi di programmazione, con particolare riferimento al progetto e alla costruzione di compilatori. Le metodologie e le tecniche presentate sono utili in generale come formalismi per definire il comportamento di un sistema o per realizzare componenti che richiedono una traduzione tra rappresentazione diverse di dati. Scopo del laboratorio e' quello di completare la preparazione fornendo competenze pratiche che riguardano l’applicazione dei concetti presentati nelle lezioni di teoria. In particolare l’obiettivo del laboratorio è quello di realizzare un semplice compilatore applicando le tecnologie corrette.

3. Risultati dell'apprendimento attesi:

Comprensione e acquisizione della terminologia tecnica del settore. Conoscenza delle metodologie fondamentali per la descrizione della sintassi di linguaggi formali (ad esempio i linguaggi di programmazione) e delle principali tecniche di parsificazione e traduzione; capacità di utilizzare tali conoscenze per lo sviluppo di sistemi software. Conoscenza e padronanza degli strumenti di base per la progettazione di traduttori.

4. Modalità di verifica dell'apprendimento:

Ai fini della determinazione del voto finale l'esame dell’insegnamento di Linguaggi Formali e Traduttori è diviso in due parti. 1. Scritto. Lo scritto è formato da: a) domande di teoria su tutto il programma svolto a lezione; b) esercizi che intendono verificare che lo studente sia in grado di applicare quanto appreso.

2. Discussione sull'esercitazione finale di laboratorio consistente nella realizzazione di un interprete/compilatore per un semplice linguaggio definito ad hoc.

Per superare l’esame lo studente deve raggiungere la sufficienza in entrambe le parti. Per sostenere la prova di laboratorio è necessario aver superato la prova scritta. Entrambe le prove devono essere superate nella stessa sessione d'esame.

5. Modalità d'insegnamento:

L’insegnamento è suddiviso in una parte di teoria e una di laboratorio. Per la parte di teoria sono previste 60 ore di lezioni frontali. Nel corso delle lezioni vengono svolti in aula esercizi per ognuna delle parti fondamentali in cui si divide l’insegnamento: definizione di automi e di grammatiche, progetto di parsificatori ascendenti o discendenti per semplici linguaggi, definizione di regole di traduzione e algoritmi di traduzione diretta dalla sintassi. Le lezioni finali dell’insegnamento vengono dedicate alla preparazione dell’esame con svolgimento di esercizi tratti da testi d’esame di appelli degli anni precedenti. Agli studenti sono resi disponibili i lucidi eventualmente usati dal docente come supporto alle lezioni.

La parte di laboratorio consiste in un modulo di 30 ore. Agli studenti vengono presentati una serie di esercizi relativi all'applicazione pratica dei concetti trattati nella parte di teoria. Gli esercizi vengono svolti in laboratorio con l'assistenza del docente. I primi esercizi avranno lo scopo di fornire agli studenti l'esperienza necessaria per affrontare l'esercitazione finale, che richiede la realizzazione di un interprete/compilatore.

6. Attività di supporto:

Il materiale didattico di supporto (programma dettagliato, lucidi, esempi di testi d'esame ed altro) è disponibile sul supporto on-line ai corsi I-learn.

7. Programma:

Automi a stati finiti ed espressioni regolari, analisi lessicale. Grammatiche e famiglie di linguaggi. Analisi sintattica: - Parsificazione top-down; - Parsificazione bottom-up. Traduzione diretta dalla sintassi. Generazione del codice intermedio.

Laboratorio: esercizi di programmazione che riguardano, in particolare, l'analisi lessicale (implementazione di automi), l'analisi sintattica (realizzazione di analizzatori a discesa ricorsiva), e la generazione di codice intermedio (sviluppo di un compilatore per un semplice linguaggio).

Il programma dettagliato del corso sarà pubblicato sul supporto on-line ai corsi I-learn

8. Testi consigliati e bibliografia:

A. V. Aho, M. S. Lam, R. Sethi, J.D. Ullman, "Compilatori: Principi, tecniche e strumenti", Pearson Paravia Bruno Mondadori S.p.A., 2009, ISBN 978-88-7192-559-2.

John E. Hopcroft, R. Motwani, J. D. Ullman, “Automi, Linguaggi e Calcolabilità”, Pearson-Addison Wesley, 2009, ISBN 978-88-7192-552-3.

Per la parte di laboratorio, agli studenti viene fornito, oltre ai testi degli esercizi, materiale utile per il loro svolgimento.

Insegnamento **INF0003 - Logica per l'Informatica**

Insegnamento (inglese):	**Logic for Computer Science**
CFU:	**6**
Settore:	**MAT/01 - LOGICA MATEMATICA**
Periodo didattico:	**1** **2**
Tipologia di Attività Formativa:	**D - libera**
Docenti:	**Stefano BERARDI (Titolare)** **Simonetta RONCHI DELLA ROCCA (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

- Conoscenza di base di Logica - Elementi di linguaggi funzionali.

Eventuali corsi propedeutici

- Logica e Matematica Discreta - Linguaggi e Paradigmi di Programmazione

2. Obiettivi formativi:

Si vogliono fornire agli studenti strumenti formali per il ragionamento rigoroso sia in linguaggio naturale che in linguaggi formali. Si vuole utilizzare la logica come strumento per una descrizione rigorosa della specifica di un programma e per porre le basi della verifica automatica di proprietà di programmi.

3. Risultati dell'apprendimento attesi:

Lo studente dovrà dimostrare di aver appreso i principi fondamentali della logica formale Dovrà inoltre dimostrare di di conoscere la relazione della logica formale con la programmazione, sia funzionale che logica.

4. Modalità di verifica dell'apprendimento:

Esame orale.

5. Modalità d'insegnamento:

Lezioni alla lavagna.

6. Attività di supporto:

Non previste

7. Programma:

- Logica proposizionale - Sistemi deduttivi: deduzione naturale, correttezza e completezza - Logica proposizionale e linguaggi di programmazione: isomorfismo di Curry-Howard - Logica dei predicati - Calcolo del primo ordine: deduzione naturale, correttezza e completezza - Metodo di risoluzione.

8. Testi consigliati e bibliografia:

Asperti A., Ciabattoni A., "Logica ad Informatica", Mc Graw-Hill Education, 1996. (parti scelte)

Insegnamento	**MFN0578 - Matematica Discreta e logica**

Insegnamento (inglese):	**Discrete Mathematics and Logic**
CFU:	**12**
Settore:	**MAT/01 - LOGICA MATEMATICA** **MAT/02 - ALGEBRA**
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**A - di base**
Docenti:	**Andrea MORI (Titolare)** **Luca MOTTO ROS (Titolare)** **Margherita ROGGERO (Titolare)** **Matteo VIALE (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Conoscenza delle basi della matematica della scuola superiore: in particolare le operazioni aritmetiche di base, le proprietà delle potenze, le equazioni di primo e secondo grado. Conoscenza della terminologia di base relativa alle parti del discorso: nomi, verbi, proposizioni, aggettivi.

Eventuali corsi propedeutici

Nessuno

2. Obiettivi formativi:

L’insegnamento si propone di fornire allo studente una introduzione alla matematica discreta e alla logica matematica, con particolare riguardo per i loro aspetti piu` rilevanti per la formazione di base di un informatico, in particolare una adeguata familiarita` con le strutture algebriche, il calcolo combinatorio e le principali tecniche di dimostrazione.

3. Risultati dell'apprendimento attesi:

Al termine dell’insegnamento lo studente dovra` dimostrare di avere compreso le strutture basilari dell’algebra dei gruppi e degli anelli, con particolare riguardo alla struttura degli interi, con applicazione all’aritmetica modulare. Dovra` quindi dimostrare di conoscere le principali proprietà di alcuni esempi specifici di gruppi ed anelli e di loro morfismi. Essere in grado di risolvere equazioni diofantee lineari in due variabili e congruenze lineari oltre ad equazioni non commutative tra permutazioni. Dovra` peraltro dimostrare di impostare e risolvere problemi di carattere enumerativo. Lo studente dovra` essere in grado di esprimere i concetti appresi e di impostare la risoluzione di esercizi e problemi utilizzando in modo appropriato il linguaggio e il formalismo degli insiemi, in particolare per quel che riguarda le operazioni insiemistiche di unione, intersezione e prodotto cartesiano, le relazioni, le funzioni. Inoltre, lo studente dovra` dimostrare di sapere riconoscere ed utilizzare le principali tecniche di dimostrazione (diretta, per assurdo, per contrapposizione, per casi) in semplici dimostrazioni di proposizioni relative a strutture algebriche e relazionali. Dovra` essere in grado di utilizzare le principali forme del principio di induzione (ordinaria, forte, principio del minimo) in semplici dimostrazioni aritmetiche o relative alla sintassi formale dei linguaggi proposizionali e del prim’ordine. Dovra` dimostrare di essere in grado di formalizzare mediante formule logiche semplici asserzioni formulate in italiano. Dovra` inoltre dimostrare di conoscere la terminologia di base della teoria dei reticoli e delle algebre di Boole, con particolare attenzione ai concetti di atomo, insieme inferiore (o di ideale d’ordine), distributivita` assieme al loro utilizzo nella rappresentazione dei reticoli distributivi e delle algebre di Boole finite. Infine, lo studente dovra` dimostrare di riconoscere l’esistenza o meno di biiezioni tra insiemi infiniti costruiti mediante le operazioni di unione disgiunta, prodotto cartesiano, insieme potenza, e insieme delle sequenze finite.

4. Modalità di verifica dell'apprendimento:

L’esame consiste di due parti, una di Matematica Discreta e una di Logica, che possono essere affrontate anche in appelli differenti, tuttavia entro lo stesso anno accademico (dall’appello di Gennaio a quello di Settembre). Le prove di esame sono scritte, a meno di motivata richiesta di esame orale da parte dei docenti. Le domande possono riguardare sia la teoria che lo svolgimento di esercizi. Dopo il superamento di una delle due parti, non saranno consentiti piu` di due tentativi per l’altra parte, pena l’annullamento degli esiti positivi gia` conseguiti. L’esito dell’esame e` espresso in trentesimi come media dei punteggi riportati nelle parti di Matematica Discreta e di Logica.

5. Modalità d'insegnamento:

Lezioni frontali (60 ore Matematica Discreta e 60 ore Logica) comprese le esercitazioni in aula.

6. Attività di supporto:

Disponibilita` di un tutor.

7. Programma:

Linguaggio degli insiemi (16 ore circa) • Insiemi: insieme vuoto; sottoinsiemi; unione; intersezione; complementare; insieme delle parti (con particolare attenzione al caso finito). • Corrispondenze, relazioni e funzioni: relazioni d’ordine. • Relazioni di equivalenza e partizioni. • Composizione e inversione di corrispondenze. • Iniettivita`, suriettivita`, composizione e invertibilita` di funzioni.

Calcolo combinatorio (14 ore circa) • Cardinalita` di insiemi finiti • Principi della somma e del prodotto • Disposizioni semplici e con ripetizioni • Combinazioni semplici e con ripetizioni. • Il Teorema del binomio e il triangolo di Pascal-Tartaglia • Il principio di inclusione-esclusione

Strutture algebriche (12 ore circa) • Semigruppi e loro morfismi. • Monoidi e loro morfismi: monoide delle parole • Gruppi e loro morfismi. • Alcuni esempi di strutture algebriche: numeri naturali e interi, gruppo delle biiezioni di un insieme. • Gruppi e sottogruppi ciclici. • Sottogruppi e Teorema di Lagrange. • Anelli: anello delle Matrici. • Corpi e campi: campo dei numeri razionali.

Aritmetica modulare (10 ore circa) • Anelli degli interi e delle classi di resto. • Teorema della divisione • L'algoritmo di Euclide • Identità di Bezout • Equazioni diofantee • Il teorema di Eulero-Fermat

Gruppo delle permutazioni (8 ore circa) • Composizione, potenze e inverse di permutazioni. • Decomposizione in cicli disgiunti e decomposizione in trasposizioni. • Parita` di una permutazione • Sottogruppi del gruppo delle permutazioni.

Logica

Tecniche di dimostrazione (12 ore) • Dimostrazione diretta, per assurdo, per contrapposizione; • Connettivi logici e loro significato in termini di condizioni di verita`; • Tavole di verita` e conseguenza logica tra proposizioni.

Formalizzazione (6 ore) • Linguaggi proposizionali e del prim’ordine: termini, quantificatori, alfabeto non logico, formule; • Schemi di traduzione da linguaggio naturale in linguaggi del prim’ordine (condizione sufficiente, necessaria, per tutti gli n abbastanza grandi, ci sono n arbitrariamente grandi,...)

Il principio di induzione (12 ore) • Forma ordinaria e forte del principio di induzione; • Principio del minimo; • Equivalenza tra forme del principio di induzione; • Induzione strutturale; • Ricorsione.

Reticoli (12 ore) • Ordinamenti parziali, con esempi; • Minimo confine superiore, massimo confine inferiore: reticoli; • Distributivita`; • Applicazioni: reticoli nell’architettura dell’informazione, analisi formale dei concetti.

Algebre di Boole (12 ore) • Algebra degli insiemi; • Algebre di Boole: definizioni; • Algebre di Boole

come reticoli; • Cenni alla rappresentazione delle algebre di Boole finite: atomi; • Calcolo delle funzioni booleane e algebra della commutazione; • Algebra della logica: algebre di Boole e calcolo proposizionale.

Insiemi infiniti (6 ore) • Insiemi numerabili e piu` che numerabili: esempi; • Operazioni infinitarie: unioni e intersezioni, prodotti e somme, con le principali proprieta`.

8. Testi consigliati e bibliografia:

Alberto Facchini, “Algebra e matematica discreta, per studenti di informatica, ingegneria, fisica e matematica con numerosi esempi ed esercizi svolti”. Zanichelli/Decibel, 2000.

Bertone-Roggero: “Appunti di Matematica Discreta”, 2015, https://dl.dropboxusercontent.com/u... (versione aggiornata 2015 https://dl.dropboxusercontent.com/u...)

F. Preparata, R.T. Yeh, “Introduzione alle strutture discrete”, Boringhieri, 1976. Andretta, Cardone: “Dispense di logica matematica”, 2013.

Insegnamento **MFN0633 - Metodi Formali dell'Informatica**

Insegnamento (inglese):	**Formal Methods in Computer Science**
CFU:	**9**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**1** **2**
Tipologia di Attività Formativa:	**B - caratterizzante**
Docenti:	**Simonetta RONCHI DELLA ROCCA (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Sono richieste buone conoscenze di logica, programmazione e algoritmi; inoltre si assume che lo studente possegga le nozioni di linguaggio formale, grammatica e di automa.

Eventuali corsi propedeutici

Matematica Discreta e Logica, Programmazione 1 e 2, Algoritmi e Strutture Dati, Linguaggi Formali e Traduttori.

2. Obiettivi formativi:

Il corso si propone di offrire agli studenti le conoscenze di base per comprendere le nozioni relative alla descrizione formale dei linguaggi di programmazione, al significato dei programmi, alla nozione di funzione calcolabile (e relative proprieta' elementari) e alle nozioni di base della complessità computazionale.

3. Risultati dell'apprendimento attesi:

Conoscenza della semantica operazionale e teoria dei tipi di un semplice linguaggio di programmazione e in generale di tecniche per la descrizione di concetti di linguaggi di programmazione. Nozione di funzione calcolabile e consapevolezza dell'esistenza di funzioni non calcolabili. Nozioni di complessita' in tempo e spazio.

4. Modalità di verifica dell'apprendimento:

L'esame consiste in un esame orale. Verra' anche tenuto conto della partecipazione degli studenti agli esercizi in classe.

5. Modalità d'insegnamento:

Le lezioni si svolgono in aula con l'ausilio di slides, gesso e lavagna. Nel corso delle lezioni vengono proposti esercizi a volte svolti dal docente con la partecipazione degli studenti.

6. Attività di supporto:

Il materiale didattico di supporto (a cura del docente) si puo' reperire nella pagina Moodle del corso stesso.

7. Programma:

Teoria della computabilita'

- Le Macchine di Turing

- Problemi non risolubili

- Funzioni ricorsive
- Calcolabilita' e Linguaggi di Programmazione

Teoria della complessita'

- Misure e classi di Complessita'
- Classi di Complessita' Temporale
- Classi di Complessita' Spaziale
- Le classi P ed NP
- Problemi NP completi

Semantica dei linguaggi di programmazione

- Definizione di un semplice linguaggio imperativo, IMP
- Semantica operazionale di IMP. Semantica input-output e step-by-step
- Semantica denotazionale di IMP
- Equivalenza delle due semantiche
- Accenno alla teoria dei domini
- Semantica assiomatica di IMP

8. Testi consigliati e bibliografia:

- C. Toffalori et alii, Teoria della calcolabilita' e della complessita', McGraw-Hill 2005

- G. Winskel:"La semantica formale dei linguaggi di programmazione", The MIT Press, edizione italiana UTET, 1999 (capitoli scelti).

- M. Sipser:"Introduction to the theory of Computation", Course Technology Ptr., 3^edition, 2012.

Insegnamento **MFN0582 - Programmazione I**

Insegnamento (inglese):	**Programming I**
CFU:	**9**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**A - di base**
Docenti:	**Felice CARDONE (Titolare)** **Massimiliano DE PIERRO (Titolare)** **Luca ROVERSI (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Il corso non presuppone alcuna conoscenza di programmazione o piu' in generale di informatica, al di la' della semplice capacita' di usare un calcolatore con sistema a finestre (gestione di files, elaborazione di testi). Per quanto riguarda la matematica, il modulo presuppone solo conoscenze fondamentali, quali i concetti di numero (naturale, intero, razionale, reale), di funzione, le quattro operazioni, elevamento a potenza, radice, esponenziale, logaritmo, il piano cartesiano, il calcolo letterale elementare. Si presuppongono invece capacita' di ragionamento logico, di astrazione e di risoluzione non meccanica di problemi, accompagnate da una buona padronanza della lingua madre.

Il corso e' propedeutico ad, essenzialmente, tutti i corsi INF/01, pur non essendo obbligatorio aver sostenuto l'esame per seguire corsi successivi e sostenere altri esami INF/01.

La frequenza e' vivamente consigliata.

Eventuali corsi propedeutici

Nessuno.

2. Obiettivi formativi:

Fornire i concetti di base della programmazione strutturata ed imperativa di alto livello, appoggiandosi ad un linguaggio di programmazione di riferimento.

3. Risultati dell'apprendimento attesi:

Capacita' di: (i) formalizzare la soluzione a problemi computazionali di base per mezzo di costrutti linguistici iterativi e ricorsivi, (ii) tradurre la formalizzazione al punto (i) in un linguaggio di programmazione di riferimento, (iii) valutare, anche per mezzo di semplici dimostrazioni formali correttezza parziale e terminazione della soluzione proposta, (iv) valutare l'efficienza della soluzione proposta, (v) rappresentare istantanee dello stato in cui si trovi la memoria di un calcolatore che interpreti programmi ragionevolmente complessi nel linguaggio di programmazione di riferimento.

4. Modalità di verifica dell'apprendimento:

Sono impostate per verificare conoscenza e comprensione degli argomenti sviluppati nel programma didattico, in accordo con i "Descrittori di Dublino", relativi ai learning outcomes.

Per mezzo di un numero congruo di esercizi saranno verificate conoscenze e capacita' di:

- progettazione di algoritmi iterativi e ricorsivi,
- codifica dei programmi che implementino algoritmi iterativi e ricorsivi,

- risolvere esercizi sia su aspetti fondamentali della programmazione di alto livello (correttezza, terminazione e costo degli algoritmi) sia sulla simulazione della gestione della memoria.

Gli esercizi saranno scritti o al calcolatore. La somma dei punteggi degli esercizi proposti permettera` di ottenere voti sino al "30 e Lode".

5. Modalità d'insegnamento:

Le lezioni in aula sono svolte alla lavagna e, se opportuno, con l'ausilio di calcolatore o proiettore per lucidi in formato elettronico.

Il laboratorio e' organizzato a turni. Il numero di turni puo' variare di anno in anno a seconda del numero di iscritti. E' usuale avere due turni T1, T2 per il corso A e due turni T1, T2 per il corso B.

Istruzioni su come accedere al laboratorio, saranno comunicate in aula, durante le prime lezioni, e rese disponibili sul sito di supporto alla didattica.

6. Attività di supporto:

On-line saranno disponibili:

- dispense su argomenti fondamentali del corso, non esaurientemente coperti dal testo di riferimento,
- documentazione disponibile su Internet,
- software.

7. Programma:

- Struttura di base di un calcolatore.
- Informazioni di base su linguaggi di programmazione, differenze tra compilatori e interpreti.
- Algoritmi iterativi e ricorsivi.
- Linguaggio di riferimento: variabili, tipi di dato fondamentali e array, assegnazione e controllo del flusso, procedure e funzioni con parametri.
- Correttezza parziale, terminazione e nozioni di costo degli algoritmi.

8. Testi consigliati e bibliografia:

- Walter Savitch "Programmazione di base e avanzata con Java", Pearson --- ISBN 9788865181904.
- Dispense e lucidi distribuiti dai docenti.

Insegnamento **MFN0585 - Programmazione II**

Insegnamento (inglese):	**Programming II**
CFU:	**9**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**2**
Tipologia di Attività Formativa:	**A - di base**
Docenti:	**Viviana BONO (Titolare)** **Luca PADOVANI (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Agli studenti è richiesta la conoscenza delle nozioni fondamentali di programmazione imperativa nel linguaggio Java (comandi di assegnamento, condizionali e iterativi, tipi di dato semplici e array, nozioni di astrazione procedurale e ricorsione).

Eventuali corsi propedeutici

Programmazione I e Laboratorio

2. Obiettivi formativi:

Il corso si propone di raffinare le capacità di programmare nel linguaggio Java apprese nel corso di Programmazione I e di introdurre le nozioni fondamentali della programmazione orientata agli oggetti. In particolare, il corso illustrerà le *astrazioni fondamentali* per la progettazione del software (classi e oggetti), la definizione di *semplici strutture dati* e operazioni corrispondenti (liste, alberi, pile, code), i meccanismi di base per favorire *riuso e modularità* del software (ereditarietà, polimorfismo, tipi generici), la specifica degli *invarianti di classe* e gestione delle loro violazioni (asserzioni ed eccezioni), così come alcune *classi fondamentali* della libreria Java. Si darà particolare enfasi agli aspetti di buona progettazione del software, utilizzando concetti presi a prestito dall'ingegneria del software e formalismi grafici quali UML.

3. Risultati dell'apprendimento attesi:

Lo studente acquisirà la capacità di progettare e realizzare piccole/medie applicazioni in Java sfruttando gli strumenti di base della programmazione a oggetti, imparerà a utilizzare ed a realizzare alcune strutture dati fondamentali ed approfondirà la conoscenza della libreria standard di Java.

4. Modalità di verifica dell'apprendimento:

L'esame si compone di una verifica di laboratorio e di una prova scritta.

La *verifica di laboratorio* consiste in una prova pratica di programmazione al calcolatore della durata di 45 minuti, viene valutata con un punteggio da 0 a 2 e considerata sufficiente se tale punteggio è 1 o 2. Una volta superata la verifica di laboratorio, il punteggio ottenuto rimarrà valido fino all'appello precedente l'avvio del corso dell'anno successivo. Uno studente può consegnare al massimo 3 elaborati di laboratorio durante l'anno accademico. Può però presentarsi ad ogni appello (previa registrazione) e decidere se consegnare o meno il proprio elaborato.

La *prova scritta*, che può essere sostenuta solo previo superamento della verifica di laboratorio, consiste in esercizi di programmazione e sulle nozioni fondamentali del corso e viene valutata con un punteggio da 0 a 30. È considerata sufficiente se tale punteggio è 18 o superiore.

Il voto finale dell'esame di Programmazione II e Laboratorio è pari al punteggio della prova scritta.

La lode viene assegnata se tale punteggio è 30 ed il punteggio della prova di laboratorio è 2.

5. Modalità d'insegnamento:

L’insegnamento consta di una parte teorica (60 ore) e una pratica (30 ore).

La parte teorica si svolge in aula con l'ausilio di lavagna e gesso ed eventualmente di un calcolatore usato dal docente per la proiezione di lucidi e/o lo sviluppo di semplici programmi.

La parte pratica si svolge nei laboratori didattici in cui il docente illustra esercizi di programmazione che gli studenti risolvono al calcolatore. Il docente (ed eventualmente studenti tutor) supportano gli studenti nella risoluzione degli esercizi assegnati.

6. Attività di supporto:

Il materiale usato durante le lezioni in aula e in laboratorio, i testi degli esercizi e delle prove d’esame sono documentati e/o resi disponibili sulla pagina moodle del corso http://informatica.i-learn.unito.it/.

7. Programma:

Il programma del corso è basato sul testo di riferimento, eventualmente integrato da dispense e/o altro materiale fornito dal docente e reso disponibile sulla pagina moodle del corso. Segue un sommario degli argomenti trattati e dei capitoli relativi sul testo di riferimento. Tale sommario non riflette necessariamente l’ordine di esposizione degli argomenti.

- ripasso dei concetti di base della programmazione imperativa e di Java (Cap. 1-7)
- incapsulamento, definizione di una classe, creazione e uso di oggetti (Cap. 8 e 9)
- specializzazione di una classe, ereditarietà (Cap. 10)
- relazioni tra classi, polimorfismo, classi parzialmente definite, interfacce (Cap. 11)
- generalizzazione di una classe, classi parametriche, tipi generici (Cap. 12)
- invarianti di classe, asserzioni, eccezioni e loro gestione (Cap. 13)
- progettazione e implementazione di strutture dati (liste, alberi, pile, code) (Cap. 15)
- cenni su progettazione a oggetti e UML

8. Testi consigliati e bibliografia:

Libro di testo:

- W. Savitch: Programmazione di base e avanzata con Java, Prima edizione, Pearson, 2014.

Altri testi di riferimento:

- M. Naftalin, P. Wadler: Java Generics and Collections, O'Reilly, 2006.
- G. Pighizzini, M. Ferrari: Dai fondamenti agli oggetti, Terza edizione, Pearson Addison Wesley, 2008.
- Allen B. Downey. How to Think Like a Computer Scientist: Java Version. Disponibile on-line qui.

Insegnamento **MFN0605 - Programmazione III**

Insegnamento (inglese):	**Programming III**
CFU:	**6**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**2**
Tipologia di Attività Formativa:	**B - caratterizzante**
Docenti:	**Liliana ARDISSONO (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Si richiede una buona conoscenza dell'analisi e della progettazione object-oriented (fornita dai corsi di Programmazione II e di Algoritmi e strutture dati) e dei meccanismi di base e delle problematiche della programmazione concorrente (fornita dal corso di Sistemi Operativi). Lo studente deve inoltre avere la capacità di scrivere, compilare e verificare la correttezza di programmi in Java.

Eventuali corsi propedeutici

Programmazione II, Algoritmi e strutture dati, Sistemi Operativi.

2. Obiettivi formativi:

Lo sviluppo di software efficiente e scalabile presuppone la capacità di programmare applicazioni distribuite e concorrenti. In particolare, la programmazione distribuita in ambiente object oriented arricchisce le nozioni di base di programmazione concorrente sfruttando il paradigma ad oggetti per una più chiara scomposizione delle attività da eseguire in parallelo e loro attribuzione alle entità software di competenza, che possono essere modellate come oggetti distribuiti che offrono i relativi servizi. E' quindi fondamentale per il curriculum di un laureato in informatica acquisire competenze approfondite su tali temi.

Il corso si pone l'obiettivo di fornire la conoscenza di base necessaria per la programmazione di applicazioni object-oriented distribuite e concorrenti, usando linguaggi ad alto livello, attraverso (i) l'invocazione remota di metodi degli oggetti, e (ii) la programmazione di thread paralleli, cioè di processi "leggeri" che possono operare su uno o più processori all'interno della stessa applicazione principale. Altro obiettivo fondamentale del corso è la tecnica di programmazione ad eventi per la realizzazione di interfacce grafiche, che stanno alla base di tutte le applicazioni desktop e web basate su finestre. Tutte le conoscenze verranno fornite utilizzando il linguaggio Java come base per le spiegazioni e la sperimentazione.

Per permettere agli studenti di sperimentare le nozioni apprese durante le ore di teoria in aula il corso include una sostanziale parte di laboratorio. I temi introdotti durante il laboratorio corredano e integrano le conoscenze derivanti dalla parte teorica (knowledge and understanding) e permettono agli studenti di familiarizzare con le metodologie e tecnologie introdotte, anche investigando soluzioni alternative (applying knowledge and understanding). Inoltre durante le ore di laboratorio è previsto lo sviluppo di un'applicazione distribuita realistica con interfaccia grafica. La preparazione e la discussione del progetto di laboratorio sono inoltre volte a stimolare le capacità di organizzare il lavoro in piccoli gruppi (max 3 studenti), e poi di illustrare verbalmente le soluzioni adottate (communication skills).

3. Risultati dell'apprendimento attesi:

Lo studente acquisirà la conoscenza dei concetti che stanno alla base della programmazione di interfacce utente grafiche e di applicazioni distribuite e concorrenti in linguaggi ad alto livello. Lo studente dovrà essere in grado di sviluppare applicazioni distribuite in ambiente Java, utilizzando

RMI per l’invocazione remota di metodi, la programmazione a thread per la gestione del parallelismo e SWING per la realizzazione delle interfacce utente grafiche.

4. Modalità di verifica dell'apprendimento:

L'esame è composto da un TEST SCRITTO e da una VERIFICA DI LABORATORIO. Le due prove possono essere sostenute in qualsiasi ordine (cioè, non è necessario aver dato lo scritto per fare la prova di laboratorio, né il vice versa).

- TEST SCRITTO: Prova scritta che include esercizi e domande teoriche sul programma del corso. Viene valutata da un minimo di 0 ad un massimo di 30 e si considerano sufficienti i voti >=18. Durante la prova è proibito comunicare con altre persone, presenti in aula o fuori. Inoltre, non si può portare alcun tipo di materiale didattico (appunti, libri, dispense, etc.) ed è vietato usare computer, telefonini o simili. Come da regolamento di Ateneo, ogni studente può sostenere un numero massimo di tre prove scritte durante l'Anno Accademico (cioè, consegnare il proprio elaborato tre volte). Il voto ottenuto durante un test scritto decade se lo studente partecipa ad un altro test scritto e consegna il suo elaborato.
- VERIFICA DI LABORATORIO: prevede la discussione del progetto di laboratorio svolto durante il corso. La discussione deve essere effettuata preferibilmente in unica soluzione, con tutti i membri del gruppo di laboratorio presenti, e si può tenere sia durante gli appelli appositamente dedicati che su appuntamento (previa email al docente). Il voto di laboratorio è un numero compreso tra 0 e 30, si considerano sufficienti i voti >=18.
- CALCOLO DEL VOTO FINALE DI ESAME: sia X il voto del test scritto; sia Y il voto di laboratorio. Il voto Fin finale dell'esame si ottiene come segue: Fin = (X+Y)/2

Note: i voti acquisiti durante la prova di laboratorio, o durante il test scritto, rimangono validi fino al termine della terza sessione d’esame (quella che precede l'inizio del nuovo corso). Quando si superano entrambe le prove, è necessario registrare il voto finale entro i limiti imposti dal Regolamento di Ateneo.

5. Modalità d'insegnamento:

L’insegnamento è diviso in una parte di teoria e una di laboratorio. Per la parte di teoria sono previste 40 ore di lezione frontali che seguono il programma riportato più avanti, integrate da casi di studio e da esercitazioni volte ad illustrare l’applicazione pratica dei concetti appena studiati. La parte di laboratorio si tiene in altre 20 ore e ha per oggetto l’insegnamento delle tecnologie per lo sviluppo di applicazioni distribuite e concorrenti in Java (RMI, thread), con interfacce grafiche a finestre (SWING). Le lezioni si svolgono in maniera interattiva e sono corredate da vari esercizi miranti a fornire esempi pratici.

Le sperimentazioni che vengono effettuate durante le ore di laboratorio, strutturate come sequenze di esercizi specifici, sono fondamentali per aiutare gli studenti a comprendere e assimilare i contenuti teorici spiegati a lezione in quanto permettono di mettere in pratica i concetti e le metodologie illustrate su esempi concreti. Inoltre lo sviluppo del progetto di laboratorio permette di consolidare le conoscenze teoriche in un caso realistico di media complessità.

Si consiglia caldamente la frequenza costante alle lezioni di teoria e di laboratorio. E’ inoltre fondamentale che gli studenti si iscrivano al corso online su I-Learn, all’interno del quale il docente mette a disposizione materiale didattico di supporto.

6. Attività di supporto:

Il materiale didattico di supporto (lucidi delle lezioni, link a documentazione, esempi di testo di esami ed altro) è disponibile presso il supporto on-line ai corsi: qui (PER ORA PAGINA DEL 2015/16)

Si noti che i lucidi non sostituiscono il libro di testo, né il materiale integrativo ad esso affiancato.

7. Programma:

Programmazione ad eventi in Java: programmare interfacce grafiche.
- Sorgenti di eventi, gestori di eventi, event-driven programming.

- Organizzazione e uso delle interfacce grafiche di Java.
- L'architettura Model-View-Controller (MVC).

Programmazione Multithread:
- Esecuzione concorrente di istruzioni.
- I thread in Java: ciclo di vita dei thread.
- Creazione e sincronizzazione di thread.
- Estensione del modello della memoria in presenza di thread.
- Problemi di sincronizzazione e loro risoluzione mediante il linguaggio Java.

Programmazione in rete in Java:
- L'architettura client-server.
- Uso di socket.
- Polimorfismo e trasferimento di oggetti mediante Java.
- Invocazione remota di metodi (RMI).
- Il modello di esecuzione distribuita di oggetti.

8. Testi consigliati e bibliografia:

Libro di Testo: Silvia Crafa. Oggetti, concorrenza, distribuzione. Società editrice Esculapio, 2014.

Altri testi (approfondimenti):

- K. Arnold, J. Gosling, D. Holmes. Il linguaggio Java, Manuale Ufficiale. Pearson Education Italia, 2006.
- Programmazione Java - tecniche avanzate, di Deitel e Deitel. Ed. Pearson - Prentice Hall

Insegnamento **MFN0635 - Reti di Elaboratori**

Insegnamento (inglese):	**Computer Networks**
CFU:	**12**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**1** **2**
Tipologia di Attività Formativa:	**B - caratterizzante**
Docenti:	**Matteo SERENO (Titolare)** **Franco SIROVICH (Professore a Contratto)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

- Competenza nella programmazione sequenziale C e/o Java (per recepire gli elementi di programmazione concorrente e distribuita presentati nel corso);
- Competenza nelle architetture degli elaboratori e dei sistemi operativi (per apprezzare gli aspetti sistemistici dei sistemi distribuiti oggetto di studio).

Eventuali corsi propedeutici

Programmazione I e II, Architetture, Sistemi Operativi.

2. Obiettivi formativi:

Il corso studia gli elementi fondamentali delle tecnologie di trasmissione del livello data link, dei protocolli di accesso a mezzi condivisi e dei protocolli di trasmissione wireless, la suite di protocolli TCP/IP, e i principi che guidano la strutturazione e la progettazione di applicazioni distribuite.

3. Risultati dell'apprendimento attesi:

- Comprensione dei fenomeni e dei protocolli che sono alla base della trasmissione delle informazioni via cavo e via etere;
- Conoscenza approfondita dei protocolli che compongono la suite TCP/IP e delle loro relazioni;
- Principi che guidano la strutturazione e la progettazione di applicazioni distribuite.

4. Modalità di verifica dell'apprendimento:

La verifica si articolerà in due prove scritte che potranno richiedere complessivamente tra le tre e le quattro ore. La verifica sarà orientata ad accertare la capacità maturata dallo studente nella comprensione dei fenomeni e dei protocolli di livello data link, di accesso a mezzo condiviso, di trasmissione wireless, nella conoscenza approfondita dei protocolli che compongono la suite TCP/IP e delle loro relazioni, e nella capacità di progettare e realizzare semplici applicazioni distribuite.

5. Modalità d'insegnamento:

Le lezioni e le esercitazioni sono svolte con l'ausilio di slide che verranno messe a disposizione sul sito di e-learning del corso di laurea.

6. Attività di supporto:

Il materiale didattico di supporto è disponibile sul siti e-learning del Corso di Laurea.

7. Programma:

- Introduzione ad Internet ed alle reti di calcolatori
- Il livello di collegamento e le reti locali: collegamenti, reti di accesso e reti locali
- Reti Wireless e Mobili
- Il livello di Rete
- Il livello di Trasporto
- Il livello Applicazioni
- Come creare un’applicazione di rete: I socket
- Gestione della rete

8. Testi consigliati e bibliografia:

Libri di testo:

- Reti di Calcolatori e Internet - Un approccio top-down, J. Kurose, K. Ross, Pearson Education, Sesta Edizione.
- Introduzione alla programmazione client-server, D. Maggiorini, Pearson, Addison Wesley

Libri consigliati:

- Internetworking con TCP/IP. Vol. 1: Principi, protocolli e architetture, D. Comer, Pearson Prentice Hall

Insegnamento **MFN1362 - Reti I**

Insegnamento (inglese):	**Computer Networks I**
CFU:	**6**
Settore:	
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**B - caratterizzante**
Docenti:	**Marco BOTTA (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Lo studente deve conoscere prima di seguire il corso i fondamenti della programmazione e dei sistemi operativi.

Eventuali corsi propedeutici

Sistemi Operativi, Programmazione I e II

2. Obiettivi formativi:

Il corso si propone di fornire agli studenti nozioni base sulle reti di calcolatori e una comprensione approfondita della suite di protocolli TCP/IP. Inoltre, attraverso l'uso di software ed esempi pratici, il corso fornisce agli studenti una comprensione concreta dei meccanismi di comunicazione tra dispositivi di rete e calcolatori.

3. Risultati dell'apprendimento attesi:

Il corso si propone di preparare gli studenti allo sviluppo di un progetto di tipo generico che coinvolga una tecnologia di rete relativa alla famiglia di protocolli TCP/IP. Al tempo stesso, gli studenti che superano con successo l'esame potranno inserirsi anche in gruppi di lavoro tecnici che debbano affrontare problematiche di sviluppo di applicazioni distribuite di progettazione, implementazione e/o adeguamento di protocolli di rete.

4. Modalità di verifica dell'apprendimento:

La verifica viene fatta sulle conoscenze acquisite durante il corso. La durata della prova scritta è di 2 ore e prevede domande relative agli argomenti trattati nel Corso. L'esame può avere un seguito in forma orale, che è facoltativo per lo studente.

5. Modalità d'insegnamento:

Le lezioni sono di tipo frontale in aula con esempi dal PC del docente collegato in rete e al proiettore durante la lezione

6. Attività di supporto:

Diverso materiale reperibile tramite: Reti e Sistemi Distribuiti

7. Programma:

Fondamenti su reti di calcolatori. Si seguirà l'approccio top-down.

Il livello applicativo: suite di applicazioni TCP/IP (Web, Posta, DNS, File transfer, sistemi P2P) Il livello transport: controllo della congestione e del flusso end-to-end Il livello rete: instradamento e reti IP Il livello link: condivisione del mezzo e controllo di flusso. Il livello di comunicazione fisico: mezzi di comunicazione, modulazione, multiplexing. Reti wireless e mobilità.

8. Testi consigliati e bibliografia:

J. Kurose - K. Ross. Computer Networking: A Top-Down approach featuring the Internet. 5/e Pearson - Addison Wesley

Edizione italiana: J. Kurose - K. Ross. Reti di calcolatori e internet - Un approccio top-down 4a Edizione Pearson - Addison Wesley

Testo supplementari: D. Comer. Internetworking with TCP/IP Volume Unico, Prentice Hall

Barabási Albert-László. Link. La scienza delle reti Einaudi. 2004

Insegnamento **INF0002 - Servizi Web**

Insegnamento (inglese):	**Web Services**
CFU:	**6**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**D - libera**
Docenti:	**Liliana ARDISSONO (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Si richiede una buona conoscenza delle basi di dati (fornita dal corso di Basi di Dati), dei sistemi operativi (fornita dal corso di Sistemi Operativi), dell'analisi e della progettazione object-oriented (fornita dai corsi di Programmazione II e di Algoritmi e strutture dati) e dei fondamenti della programmazione distribuita (fornita dal corso di Programmazione III). Lo studente deve inoltre avere la capacità di scrivere, compilare e verificare la correttezza di programmi in Java.

Eventuali corsi propedeutici

Basi di Dati, Sistemi Operativi, Interazione Uomo-Macchina, Programmazione II, Algoritmi e Strutture Dati, Programmazione III.

2. Obiettivi formativi:

Con la enorme diffusione delle applicazioni web diventa fondamentale nel curriculum di un laureato in informatica il conoscere a fondo le architetture delle stesse e le tecnologie più diffuse per il loro sviluppo. In particolare, l'implementazione dell'applicativo, che richiede di guardare "dietro all'interfaccia utente" per andare a fondo su aspetti architetturali e tecnologici che possono influenzare non solo le prestazioni dell'applicazione, la sua scalabilità e robustezza, ma anche le tipologie di servizio che possono essere effettivamente offerte.

Il corso si pone come obiettivo di fornire la conoscenza di base necessaria per la progettazione e lo sviluppo di applicazioni Web interattive, accessibili da terminali desktop e mobili (grazie all'uso di linguaggi di interfaccia utente cross-platform), e caratterizzate da una logica applicativa mediamente complessa. In ultimo ci si propone di formare programmatori capaci di sviluppare applicazioni web di qualità e basate su architetture standard, largamente utilizzate nel mondo aziendale.

Le tecnologie presentate sono note come Server-side Programming e riguardano la progettazione e lo sviluppo di applicazioni basate su architetture modulari che possono accedere a sorgenti dati eterogenee (come basi dati relazionali, file, etc.) allo scopo di fornire all'utente servizi complessi. Più precisamente, il corso tratterà dal punto di vista sia teorico che pratico: (i) lo sviluppo di pagine web statiche e dinamiche con HTML5-JavaScript, CSS, e PHP, e (ii) la programmazione lato server, utilizzando tecnologie Web in ambiente Java. Inoltre, il corso tratterà la tecnologia XML, data la sua importanza nella condivisione di informazioni in Internet e nella gestione di interoperabilità tra applicazioni.

Per permettere agli studenti di sperimentare le nozioni apprese durante le ore di teoria in aula il corso include una sostanziale parte di laboratorio. I temi introdotti durante il laboratorio corredano e integrano le conoscenze derivanti dalla parte teorica (knowledge and understanding) e permettono agli studenti di familiarizzare con le metodologie e tecnologie introdotte, anche investigando soluzioni alternative (applying knowledge and understanding). Inoltre durante le ore di laboratorio è previsto lo sviluppo di un'applicazione realistica con interfaccia web. La preparazione e la discussione del progetto di laboratorio sono inoltre volte a stimolare le capacità di organizzare il lavoro in piccoli gruppi (max 3 studenti), e poi di illustrare verbalmente le

soluzioni adottate (communication skills).

3. Risultati dell'apprendimento attesi:

Lo studente acquisirà la conoscenza delle architetture di riferimento per lo sviluppo di applicazioni web modulari. Inoltre lo studente acquisirà la conoscenza dei modelli più usati per gestire applicazioni interattive di media complessita`, e delle tecnologie attualmente utilizzate per l'implementazione delle applicazioni e della loro interfaccia utente. Lo studente acquisira` la capacita` di sviluppare pagine dinamiche PHP, e applicazioni Web a 3 livelli in ambiente Java, utilizzando HTML5-JavaScript, CSS, JSP e Java Servlet.

4. Modalità di verifica dell'apprendimento:

L'esame è composto da un TEST SCRITTO e da una VERIFICA DI LABORATORIO. Le due prove possono essere sostenute in qualsiasi ordine.

TEST SCRITTO

Prova scritta che include esercizi e domande teoriche sul programma del corso. Viene valutata da un minimo di 0 ad un massimo di 30 e si considerano sufficienti i voti >=18. Durante la prova è proibito comunicare con altre persone, presenti in aula o fuori. Inoltre, non si può portare alcun tipo di materiale didattico (appunti, libri, dispense, etc.) ed è vietato usare computer, telefonini o simili. Come da regolamento di Ateneo, ogni studente può sostenere un numero massimo di 3 prove scritte durante l'Anno Accademico (cioè, consegnare il proprio elaborato 3 volte). Il voto ottenuto durante un test scritto decade se lo studente partecipa ad un altro test scritto e consegna il suo elaborato.

VERIFICA DI LABORATORIO

Prevede la discussione del progetto di laboratorio svolto durante il corso. La discussione deve essere effettuata preferibilmente in unica soluzione, con tutti i membri del gruppo di laboratorio presenti. Il voto di laboratorio è un numero compreso tra 0 e 30, si considerano sufficienti i voti >=18.

CALCOLO DEL VOTO FINALE DI ESAME

Sia X il voto del test scritto; sia Y il voto di laboratorio. Il voto Fin finale dell'esame si ottiene come segue: Fin = (X+Y)/2

NB: i voti acquisiti durante la prova di laboratorio, o durante il test scritto, rimangono validi fino al termine della terza sessione d'esame (quella che precede l'inizio del nuovo corso). Quando si superano entrambe le prove, è necessario registrare il voto finale entro i limiti imposti dal Regolamento di Ateneo.

5. Modalità d'insegnamento:

L'insegnamento è diviso in una parte di teoria e una di laboratorio. Per la parte di teoria sono previste 30 ore di lezione frontali, che seguono il programma del corso, integrate da casi di studio e da esercitazioni volte ad illustrare l'applicazione pratica dei concetti appena studiati.

La parte di laboratorio si tiene in altre 30 ore e ha per oggetto l'insegnamento delle tecnologie per lo sviluppo di pagine web statiche e dinamiche (PHP, HTML5 e JavaScript) e di applicazioni server-side in ambiente java (JSP e Java Servlet, con accesso a basi di dati remote attraverso JDBC). Le lezioni si svolgono al calcolatore, in maniera interattiva, e sono corredate da vari esercizi miranti a fornire esempi pratici.

Le lezioni in aula sono svolte principalmente con l'ausilio del calcolatore (proiezione di lucidi). Le esercitazioni in laboratorio sono svolte a calcolatore, utilizzando ambienti di sviluppo professionali, come Eclipse e/o NetBeans.

Le sperimentazioni che vengono effettuate durante le ore di laboratorio, strutturate come

sequenze di esercizi specifici, sono fondamentali per aiutare gli studenti a comprendere e assimilare i contenuti teorici spiegati a lezione in quanto permettono di mettere in pratica i concetti e le metodologie illustrate su esempi concreti. Inoltre lo sviluppo del progetto di laboratorio permette di consolidare le conoscenze teoriche in un caso realistico di media complessità.

Si consiglia caldamente la frequenza costante alle lezioni di teoria e di laboratorio. E' inoltre fondamentale che gli studenti si iscrivano al corso online su I-Learn, all'interno del quale il docente mette a disposizione materiale didattico di supporto.

6. Attività di supporto:

Il materiale didattico di supporto (lucidi delle lezioni, link a documentazione, esempi di testo di esami ed altro) è disponibile presso il supporto on-line ai corsi: qui (PER ORA PAGINA DEL 2015/16)

Si noti che i lucidi non sostituiscono il libro di testo, né il materiale integrativo ad esso affiancato.

7. Programma:

Architetture delle applicazioni Web: Web browser e Web server.

Il Pattern Model View Controller per le applicazioni Web - Progettazione e sviluppo di applicazioni Web a 3 livelli basate su MVC:

- Il primo livello (client dell'applicazione): HTML5, CSS, scripting lato client(JavaScript e AJAX), Java Server Pages (JSP). Raccolta dati (via HTML form).
- Il secondo livello (logica applicativa): Servlet Java.
- Il terzo livello (livello dei dati): JavaBeans. Accesso a database relazionali: Java Database Connectivity (JDBC), rappresentazione e gestione di informazioni con XML (XML Schema e DTD, XPath, XML Parsers).

Introduzione al linguaggio PHP per lo sviluppo di pagine web dinamiche.

8. Testi consigliati e bibliografia:

Libro di testo: Programmazione Web lato Server, di V. Della Mea, L. Di Gaspero, I. Scagnetto, Apogeo, 2007

Altri testi (per gli interessati):

- Programmazione Java - tecniche avanzate, di Deitel e Deitel. Ed. Pearson - Prentice Hall
- Principi di Web design, di Joel Sklar. Ed. Apogeo

Insegnamento **MFN0636 - Sicurezza**

Insegnamento (inglese):	**Computer and Network Security**
CFU:	**6**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**2**
Tipologia di Attività Formativa:	**B - caratterizzante** **D - libera**
Docenti:	**Francesco BERGADANO (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Si presuppone la conoscenza dei sistemi operativi e delle reti di calcolatori basate sui protocolli della suite TCP/IP. La conoscenza di specifiche piattaforme di sviluppo e programmazione, quali Java e C++, considerata di aiuto alla comprensione degli argomenti svolti nel corso.

Eventuali corsi propedeutici

Sistemi Operativi, Reti di Calcolatori

2. Obiettivi formativi:

Il corso si propone di fornire agli studenti gli strumenti crittografici e tecnici utilizzati per garantire la sicurezza di reti e calcolatori. Inoltre, attraverso l'uso di esempi pratici, il corso fornisce agli studenti una comprensione concreta dei maggiori rischi di sicurezza e delle soluzioni disponibili

3. Risultati dell'apprendimento attesi:

Il corso si propone di preparare gli studenti a lavorare in azienda per la gestione dei sistemi informatici, cooperando con i livelli organizzativi per garantire la sicurezza del sistema informativo nel suo insieme.

4. Modalità di verifica dell'apprendimento:

La verifica viene fatta sulle conoscenze acquisite durante il corso. La durata della prova scritta e' di un'ora e prevede domande aperte e domande a risposta multipla, relative agli argomenti trattati nel corso

5. Modalità d'insegnamento:

Le lezioni sono di tipo frontale in aula con esempi dal PC del docente collegato in rete e al proiettore durante la lezione.

6. Attività di supporto:

slide online per alcune parti del corso, per il resto si fa riferimento al libro di testo.

7. Programma:

Strumenti crittografici: cifrari simmetrici e asimmetrici, funzioni di hash, firma elettronica
Sicurezza della rete privata: analisi dei rischi di sicurezza informativa, controllo di accesso, protezione da virus, sistemi firewall, reti private virtuali

8. Testi consigliati e bibliografia:

Libro di testo: William Stallings: Cryptography and Network Security Prentice Hall, Seconda

Edizione, 1998

Altri Testi per consultazione: Bruce Schneier: Applied Cryptography, John Wiley and sons, 1994.
David Curry: Unix System Security, Addison-Wesley, 1992.

Insegnamento	**MFN0618 - Sistemi Informativi**

Insegnamento (inglese):	**Information Systems**
CFU:	**6**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**D - libera**
Docenti:	**Roberto MICALIZIO (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Si richiede allo studente di padroneggiare i concetti relativi alle basi di dati, alla programmazione e allo studio degli algoritmi.

Eventuali corsi propedeutici

Basi Dati e Sperimentazioni, i corsi di Programmazione I e Programmazione II, e il corso di Algoritmi e Strutture Dati.

2. Obiettivi formativi:

Il corso di Sistemi Informativi ha lo scopo di dare una panoramica delle maggiori e più diffuse applicazioni delle basi di dati nel mondo del lavoro e dell'impresa in cui buona parte dei processi aziendali sono ormai automatizzati. A tal fine, il corso introdurrà nozioni basilari Business Process Management (BPM) con la possibilità di sperimentare la modellazione di processi di business mediante lo standard BPMN.
Inoltre, il corse vuole anche dare una panoramica sui sistemi di pianificazione delle risorse aziendali (Enterprise Resource Planning - ERP), i sistemi integrati di gestione e profilazione del cliente (CRM), i sistemi di supporto alla decisione (DSS) e alle piattaforme di Business Intelligence: le Data Warehouse e le primitive per l'analisi on-line dei dati (OLAP).

3. Risultati dell'apprendimento attesi:

Conoscenza delle principali ambiti di applicazione dei sistemi informativi. Progettazione dei dati e dei processi di business anche interaziendali.
Conoscenza di una suite di Business Intelligence (analisi dei dati per grossi volumi).

4. Modalità di verifica dell'apprendimento:

L'esame è orale in cui si verificano l'apprendimento dei concetti visti a lezione, l'autonomia e la capacità di ragionamento acquisite dallo studente. Durante l'orale lo studente potrà esporre i risultati ottenuti durante la propria sperimentazione in laboratorio.

5. Modalità d'insegnamento:

Le lezioni si svolgono in aula con l'ausilio di slides e di materiale distribuito via Moodle. Le esercitazioni si volgeranno in laboratorio tramite una suite di strumenti software open source (come Bonita BPM, Pentaho, RapidMiner, KNIME o SpagoBI).

6. Attività di supporto:

Si veda il sito Moodle del corso a cura della docente.

7. Programma:

Gli argomenti e le relative tempistiche sono indicativi e verranno calibrati in base alle conoscenze

dell'uditorio.

1. Introduzione ai sistemi informativi aziendali (4 ore) (Modello organizzativo, modello funzionale, modello informatico)

2. Sistemi ERP (8 ore) (Le suite ERP; Paradigma ERP; Piattaforme software; Offerta ERP; Trasformazione dell’impresa)

3. Integrazione con il cliente: i sistemi CRM (6 ore) (Ruolo dei sistemi CRM nelle aziende; Schema architetturale; Il paradigma; Esempi; Suite di package software; Evoluzione dei CRM)

4. Piattaforme di knowledge management (6 ore) (Modello della conoscenza; Sistemi informatici; Progettazione dei sistemi di KM; Modello di successo)

5. Piattaforme di Business Intelligence e DSS (12 ore) (Livello delle fonti; Data warehouse; ETL; Progettazione del sistema di warehousing; Livello di elaborazione: reporting e DSS; Motori di analisi e data mining; Suite software per i sistemi direzionali; Sistemi CRM analitici)

6. Gestione dell'azienda orientata ai processi (24 ore) (Studio e sperimentazione di diversi linguaggi di modellazione quali: UML,BMM, BPMN. Studio di casi reali).

8. Testi consigliati e bibliografia:

- G. Bracchi, C. Francalanci, G. Motta, Sistemi informativi d'impresa, McGraw-Hill, Milano, 2010, ISBN: 978-88-386-6328-4

Libro di consultazione per la parte di Data Warehouse:

- Ralph Kimball, Margy Ross, The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling, John Wiley & Sons, 2nd Edition, 2002

Libro di consultazione per la parte di Business Process Management:

- Antonio Di Leva, La Gestione dell’Azienda Basata sui Processi e i Sistemi Informativi Aziendali, Celid, 1st Edition, 2014, ISBN 978-88-6789-017-0

Insegnamento **MFN0607 - Sistemi Intelligenti**

Insegnamento (inglese):	**Intelligent Systems**
CFU:	**6**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**2**
Tipologia di Attività Formativa:	**B - caratterizzante** **D - libera**
Docenti:	**Cristina BAROGLIO (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Poiché Sistemi Intelligenti è il primo corso che tratta argomenti di Intelligenza Artificiale, le competenze attese in ingresso riguardano competenze nel settore informatico. In particolare: - conoscenza di algoritmi su alberi e grafi con relative nozioni di complessità - esperienza di programmazione con particolare riferimento a programmazione ad oggetti (organizzazione in classi e sottoclassi, ereditarietà) - nozioni di logica (calcolo proposizionale e calcolo dei predicati del primo ordine) - nozioni di modelli semantici dei dati nelle basi dati.

Eventuali corsi propedeutici

Gli studenti che sono iscritti al corso di laurea di Informatica di Torino acquisiscono le competenze in ingresso sopra elencate seguendo gli insegnamenti di: - “Algoritmi e strutture dati”, - “Programmazione I e II”, - “Basi di dati”, - “Matematica Discreta e Logica” (e sostenendo i relativi esami)

2. Obiettivi formativi:

Il corso si propone di fornire una introduzione generale alle problematiche nel settore dell'Intelligenza Artificiale, con particolare attenzione a come sia possibile costruire un sistema dotato di capacità autonome di risoluzione di problemi, di ragionamento e di apprendimento quando abbia a disposizione una rappresentazione simbolica del mondo.

Il corso si articola in tre parti principali: - Risoluzione automatica di problemi - Rappresentazione della conoscenza e ragionamento - Nozione di agente intelligente che agisce, ragiona ed apprende

Data la natura introduttiva del corso e la durata del corso, molte problematiche avanzate di Intelligenza Artificiale trovano collocazione nei corsi offerti per l'indirizzo "Sistemi per il Trattamento dell'Informazione" della laurea magistrale in Informatica

3. Risultati dell'apprendimento attesi:

Dal punto di vista della acquisizione di contenuti metodologici, gli studenti conosceranno le principali strategie di ricerca per la risoluzione automatica di problemi, i fondamenti della rappresentazione della conoscenza mediante formalismi logici e approcci strutturati (ontologie) e relativi meccanismi inferenziali.

Avranno inoltre acquisito le nozioni di base su architettura di un agente intelligente e su apprendimento automatico di conoscenza a partire da esempi.

Gli studenti acquisteranno una prima forma di consapevolezza sulla necessità di fare dei trade-off (in termini di qualità della soluzione e costo computazionale). tra strategie e metodologie alternative che risolvono la stessa classe di problemi.

L’esame orale richiede inoltre allo studente di acquisire una buona (discreta) capacità espositiva

unita all’uso corretto della terminologia del settore (e più in generale dell’informatica).

4. Modalità di verifica dell'apprendimento:

L'esame consiste in una interrogazione individuale sugli argomenti trattati nel corso. Tipicamente una delle domande richiede la capacità di applicare una metodologia/strategia per risolvere un semplice problema simile a quelli illustrati a lezione. Pertanto nell’esame orale, oltre a verificare la conoscenza dei concetti e delle metodologie, si intende verificare la capacità di applicazione dei concetti e metodologie a problemi specifici. Essendo l’esame un orale, è possibile inoltre verificare la chiarezza di esposizione e l’uso corretto della terminologia.

5. Modalità d'insegnamento:

Le lezioni si svolgono in aula. Oltre alle lezione tipiche frontali in cui il docente illustra gli aspetti metodologici, nell’insegnamento numerose ore sono dedicate ad esempi ed esercitazioni (svolte dal docente). Vengono anche illustrati alcuni esempi significativi di strumenti software e di applicazioni di Intelligenza artificiale a problemi complessi del mondo reale.

6. Attività di supporto:

Verra' reso disponibile sulla apposita pagina del corso all'interno del servizio I LEARN. Tale materiale prevede ulteriori esempi di utilizzo dei meccanismi di ragionamento e di strategie di ricerca oltre a quelli contenuti sul libro di testo

7. Programma:

Come già detto l'insegnamemnto è una introduzione ai concetti basilari di Intelligenza articiaicile e si articola in tre parti strettamente connesse.

Parte 1) RISOLUZIONE AUTOMATICA DI PROBLEMI In questa parte si affronta la problematica di come definire il concetto di problema e di soluzione, di distinguere tra soluzione e soluzione ottima. Sono studiati tre approcci alla risoluzione di problemi: ricerca nello spazio degli stati, ricerca in spazi con avversario (giochi ad informazione completa), risoluzione di problemi mediante soddisfacimento di vincoli. Per ciascun approccio si discutono le principali strategie di ricerca: ampiezza, profondità, iterative deepening (per le ricerche cieche nello spazio degli stati), A* e Recursive Best First Strategy (per le ricerche euristiche), Min-Max e Alfa-beta (per i giochi con avversario), backtracking, forward propagation e arc consistency per meccanismi basati su soddisfacimento di vincoli. Particolare attenzione viene data alla garanzie offerte dalle diverse strategie in termini di qualità della soluzione e di complessità computazionale.

Parte 2) RAPPRESENTAZIONE DELLA CONOSCENZA E RAGIONAMENTO Il problema della rappresentazione della conoscenza e dei relativi meccanismi inferenziali viene affrontato studiando due principali famiglie di approcci alla rappresentazione della conoscenza: formalismi logici e rappresentazioni strutturate. Per quanto riguarda i formalismi logici si vede come sia il calcolo proposizionale che il calcolo dei predicati del primo ordine possano essere utilizzati per rappresentare conoscenza sul mondo e si vede come i meccanismi inferenziali (modus ponens, resolution, etc.) possano essere adoperati per fornire servizi utili (es. risposta a domande, verifica consistenza, ecc.). Si analizza anche come una rappresentazione a regole permetta meccanismi di ragionamento più efficienti (forward e backward chaining). Notevole attenzione viene data alla rappresentazione della conoscenza strutturata introducendo tassonomie, classi, individui, ereditarietà singola e multipla, inferenze specializzate. Queste nozioni vengono analizzate ed esemplificate mediante uso del linguaggio ontologico OWL2 (proposto e supportato da W3C).

Parte 3) AGENTI E APPRENDIMENTO AUTOMATICO In questa parte conclusiva si introduce la nozione di agente intelligente che opera in un ambiente e si fa vedere come l’agente possa avere sia comportamenti reattivi che deliberativi a seconda del compito assegnato. Si illustra come agente debba avere capacità di risoluzione automatica di problemi e di ragionamento sullo stato del mondo e sul suo stato. Si descrive brevemente come l’apprendimento automatico sia una delle caratteristiche essenziali per ottenere un agente intelligente. Vengono introdotte solo nozioni elementari con particolare riguardo all'apprendimento da esempi (in particolare apprendimento di alberi di decisione). Viene infine fatta una introduzione alle reti neuronali come strumento per passare dal livello sub simbolico a quello simbolico.

8. Testi consigliati e bibliografia:

S. J. Russell e P. Norvig, Artificial Intelligence: a modern approach , Third edition, 2010, Pearson
Nell’ambito del corso verranno affrontate le problematiche descritte nei capitoli 1,2,3,5,6,7,8,9, 18 del testo. Si ricorda che esiste anche la traduzione italiana della prima parte della terza edizione (volume 1) e la traduzione integrale della seconda edizione (volume 1 e 2), entrambe edite da Pearson/Prentice Hall.

OWL 2 Web Ontology Language Primer (Second Edition) http://www.w3.org/TR/owl2-primer/

Insegnamento **MFN0601 - Sistemi Operativi**

Insegnamento (inglese):	**Operating Systems**
CFU:	**12**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**B - caratterizzante**
Docenti:	**Luca ANSELMA (Titolare)** **Cristina BAROGLIO (Titolare)** **Luigi DI CARO (Titolare)** **Rossano GAETA (Titolare)** **Daniele GUNETTI (Titolare)** **Roberto MICALIZIO (Titolare)** **Daniele Paolo RADICIONI (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Agli studenti è richiesta una conoscenza di base dell'architettura di un computer (secondo quanto studiato nel corso di Architetture degli Elaboratori I) e dei concetti di base di programmazione (secondo quanto studiato nel corso di Programmazione I). Gli studenti dovranno inoltre essere in grado di padroneggiare i sistemi di enumerazione binario (base due) ed esadecimale (base sedici).

Eventuali corsi propedeutici

Costituiscono prerequisiti i contenuti dei corsi di:

* Architettura degli Elaboratori I * Programmazione I

2. Obiettivi formativi:

Il sistema operativo costituisce l'interfaccia fondamentale tra l'utilizzatore di un computer e il computer stesso. Parte essenziale del curriculum di base di un laureato in informatica è la conoscenza di come il sistema operativo sia in grado di amministrare le varie componenti hardware di cui è composto un computer. Queste modalità di amministrazione devono essere il più possibile trasparenti al generico utilizzatore del computer, ma devono essere conosciute a fondo da ogni specialista del settore. L'insegnamento fornisce dunque una conoscenza di base dell'architettura interna e del funzionamento dei moderni sistemi operativi, e di come, ai fini di garantire un ragionevole compromesso tra efficienza, sicurezza e facilità d'uso, vengono amministrate le risorse fondamentali della macchina su cui il sistema operativo è installato: il processore, la memoria principale e la memoria secondaria.

Per la parte di laboratorio gli obiettivi formativi sono l'apprendimento del linguaggio C, utilizzato per la programmazione nell'ambiente del sistema operativo Unix. La parte di laboratorio mira a fornire allo studente una conoscenza (teorica e pratica) di base sui comandi della shell, sulla gestione dei processi, sugli strumenti di inter-process communication e sulla gestione dei segnali forniti dal sistema, oltre che alcuni rudimenti di programmazione bash.

I temi introdotti durante il laboratorio corredano e integrano le conoscenze derivanti dalla parte teorica (knowledge and understanding), al tempo stesso presentando esempi di problemi realistici di comunicazione e sincronizzazione su cui gli studenti sono sollecitati a cimentarsi, anche investigando soluzioni alternative (applying knowledge and understanding). La preparazione e la discussione del progetto sono inoltre volte a stimolare le capacità di organizzare il lavoro in piccoli gruppi (2-3 studenti), e poi di illustrare verbalmente le soluzioni adottate (communication skills).

- Corso A: TUTTE LE INFORMAZIONI SUL CORSO E IL MATERIALE DIDATTICO SI TROVERANNO ALL'URL:

Sistemi Operativi

- Corso B: TUTTE LE INFORMAZIONI SUL CORSO E IL MATERIALE DIDATTICO SI TROVERANNO IN MOODLE

3. Risultati dell'apprendimento attesi:

Lo studente acquisirà la conoscenza dell'architettura e del funzionamento dei moderni sistemi operativi, e dovrà essere in grado di ragionare sulle prestazioni fornite dal sistema e su eventuali inefficienze. Avrà inoltre appreso i fondamenti della programmazione C e concorrente e la capacità di sviluppare programmi concorrenti in grado di interagire fra loro senza causare anomalie o blocchi del sistema. Infine, avrà una conoscenza di base di come le risorse della macchina (in particolar modo il tempo di CPU, lo spazio di memoria primaria e lo spazio di memoria secondaria) possano essere sfruttate al meglio.

Lo studente dovrà essere in grado di sviluppare programmi scritti nel linguaggio C, e programmi di sistema e concorrenti codificati con i comandi propri dell'ambiente del sistema operativo Unix; dovrà inoltre essere in grado di ragionare su alcuni problemi di comunicazione fra processi e sincronizzazione utilizzando gli strumenti introdotti durante il laboratorio.

4. Modalità di verifica dell'apprendimento:

Ai fini della determinazione del voto finale l'esame dell'insegnamento di Sistemi Operativi è diviso in 3 parti, 2 obbligatorie ed una facoltativa.

1. Scritto obbligatorio. Permette di prendere un voto massimo di 27/30. Lo scritto è diviso a sua volta in due parti: a) domande di teoria su tutto il programma dell'insegnamento della parte di teoria, e in particolare sulla gestione dei processi, della memoria primaria e della memoria secondaria. Lo studente dovrà dimostrare di essere in grado di ragionare su situazioni specifiche, su casi di studio e su esempi numerici che possono rispecchiare problematiche reali dell'implementazione dei sistemi operativi e del loro funzionamento. b) domande sul laboratorio Unix e sul linguaggio C. 2. Discussione sull'esercitazione finale di laboratorio obbligatoria. Permette di incrementare di al massimo 3 punti il voto conseguito allo scritto (questa parte viene sostenuta oralmente con il docente del corso di laboratorio Unix, e deve essere sostenuta prima dell'esame scritto). 3. Orale facoltativo. Integra il voto ottenuto sostenendo le parti di esame 1) e 2), e permette di prendere un voto massimo di 30/30 e lode.

Per la parte di teoria, nel corso delle lezioni è previsto lo svolgimento di esercizi in aula per ognuna delle parti fondamentali in cui si divide l'insegnamento (gestione dei processi, gestione della memoria primaria, gestione della memoria secondaria). Alla fine dell'insegnamento vengono tenute alcune lezioni di preparazione all'esame in cui vengono svolti i testi d'esame di appelli degli anni passati.

Per la parte di laboratorio, durante ciascuna lezione sono assegnati specifici esercizi; al termine di ogni macro-argomento sono proposte semplici esercitazioni riepilogative da svolgere in preparazione all'esercitazione finale, in modo da favorire l'assimilazione graduale degli argomenti introdotti durante il corso. L'esercitazione finale generalmente consiste in un progetto di maggiore impegno, e richiede di sviluppare un'applicazione in cui lo studente deve mostrare di padroneggiare i concetti e gli strumenti presentati durante il laboratorio, producendo inoltre una breve relazione in cui sono illustrati i problemi e le relative soluzioni.

La prova scritta comprende alcune domande sia sul linguaggio C sia sulla parte UNIX.

5. Modalità d'insegnamento:

L'insegnamento è diviso in una parte di teoria e una di laboratorio.

Per la parte di teoria sono previste 60 ore di lezione frontali che seguono il programma riportato più avanti, integrate da casi di studio e da esercitazioni volte ad illustrare l'applicazione pratica dei concetti appena studiati.

La parte di laboratorio è articolata in due moduli da 30 ore ciascuno, il primo avente per oggetto l'insegnamento del linguaggio C, e il secondo rivolto alla programmazione in ambiente UNIX. Le lezioni si svolgono in maniera interattiva e sono corredate da vari esercizi miranti a fornire esempi pratici.

6. Attività di supporto:

Dispense e lucidi usati a lezione, programmi di esempio.

- Per la parte di teoria del corso A il materiale è disponibile

qui

- Per la parte di teoria del corso B il materiale sarà disponibile in Moodle

7. Programma:

NOTA: Par la parte di teoria, il programma è basato sul TESTO DI RIFERIMENTO.

PARTE DI TEORIA:

* Introduzione al Corso di Sistemi Operativi

* PARTE I: GENERALITA'

o Introduzione (cap. 1)

o Strutture dei Sistemi Operativi (cap. 2)

* PARTE II: GESTIONE DEI PROCESSI

o Processi (cap. 3)

o Thread (cap. 4)

o Scheduling della CPU (cap. 5)

o Sincronizzazione dei Processi (cap. 6)

o Deadlock (Stallo di Processi) (cap. 7)

* PARTE III: GESTIONE DELLA MEMORIA (PRIMARIA)

o Memoria Centrale (cap. 8)

o Memoria Virtuale (cap. 9)

* PARTE IV: GESTIONE DELLA MEMORIA SECONDARIA

o Interfaccia del File System (cap. 10)

o Realizzazione del File System (cap. 11)

o Memoria Secondaria e Terziaria (Gestione dell'Hard disk)

==========

PARTE DI LABORATORIO:

- Linguaggio C
- Introduzione a Unix (comandi, shell, file system, diritti d'accesso, ridirezione, pipe)

- Make e makefile
- System call per la creazione e la sincronizzazione di processi
- System call per L'InterProcess Communication e per la gestione di segnali
- Esercitazioni pratiche, in particolare: esercitazioni finalizzate ad imparare il linguaggio C, ad utilizzare Unix e a sviluppare programmi concorrenti

8. Testi consigliati e bibliografia:

Teoria (TESTO DI RIFERIMENTO):

Silberschatz-Galvin-Gagne. Sistemi Operativi - ottava o nona edizione – Agli studenti sono anche resi disponibili fin dall’inizio del corso i lucidi usati dal docente nelle lezioni di teoria.

Laboratorio: • A. Kelley e I. Pohl: "C didattica e programmazione" (presente in Biblioteca) • R. Stevens, S. Rago: "Advanced Programming in the UNIX Environment", 2a ed., Addison Wesley • M.J. Bach: "The Design of the UNIX Operating System", Prentice-Hall International Editions (L3167 escluso dal prestito), per la parte relativa all'implementazione del kernel Unix. • S.R. Bourne: "The Unix System" (L1856 escluso dal prestito, L1803, L2140) • K. Christian: "The Unix Operating System", John Wiley & Sons (L3147 escluso dal prestito)

Insegnamento **INF0004 - Storia dell'Informatica**

Insegnamento (inglese):	**History of Computer Science**
CFU:	**6**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**1** **2**
Tipologia di Attività Formativa:	**D - libera**
Docenti:	**Felice CARDONE (Titolare)** **Daniele GUNETTI (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Conoscenza di base dell'architettura di un computer e di un sistema operativo. Dimestichezza di base con i principali paradigmi di programmazione.

Eventuali corsi propedeutici

Architetture degli elaboratori I. Programmazione I e II. Sistemi Operativi.

2. Obiettivi formativi:

Dare una visione di insieme e una prospettiva storica dei momenti salienti dell'evoluzione delle idee teoriche e pratiche dell'informatica.

3. Risultati dell'apprendimento attesi:

Acquisire una visione di insieme e una prospettiva storica dei momenti salienti dell'eveoluzione delle idee teoriche e pratiche dell'informatica.

4. Modalità di verifica dell'apprendimento:

Esame scritto

5. Modalità d'insegnamento:

Orale

6. Attività di supporto:

7. Programma:

Il corso fornisce una introduzione alla storia dell’informatica, dedicando particolare attenzione ad alcune tappe che hanno fornito un contributo essenziale alla definizione della forma attuale dell’informatica (per esempio, lo sviluppo di Internet con il suo retroterra culturale e tecnologico). L’interesse centrale del corso è per le idee e la loro evoluzione, più che per le singole innovazioni tecnologiche: non si tratta di una rassegna di modelli di macchina calcolatrice, ma di una introduzione ai modi di pensiero, alle metafore ed alle visioni che hanno caratterizzato la scienza dell’informazione e del calcolo attraverso la storia.

La prima parte del corso sarà dedicata ad una storia dell’evoluzione delle architetture, dei sistemi operativi e dei linguaggi di programmazione, insieme ad alcuni cenni di storia dell’informatica commerciale e di Internet.

La seconda parte comprende, accanto ad un’introduzione istituzionale alla storia del calcolo basata su uno dei libri di testo elencati, anche una serie di lezioni monografiche dedicate a temi e lavori classici in varie aree dell’informatica.

L’esame del corso prevede un compito finale il cui scopo sarà quello di verificare la comprensione della portata tecnologica e culturale degli eventi salienti della storia dell’informatica.

8. Testi consigliati e bibliografia:

M. Campbell-Kelly and W. Aspray. Computer: A History of the Information Machine. Basic Books, 1996.

P.E. Ceruzzi. A History of Modern Computing (2nd edition ed.). MIT Press, 2003.

M. Davis. Il calcolatore universale. Da Leibniz a Turing. Adelphi, 2012.

M. Priestley. A Science of Operations: Machines, Logic and the Invention of Programming. Springer, 2011.

B. Randell (Ed.), The Origins of Digital Computers: Selected Papers (3rd ed.). Springer-Verlag, 1982.

Insegnamento **MFN0606 - Sviluppo delle Applicazioni Software**

Insegnamento (inglese):	**Development of Software Applications**
CFU:	**9**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**2**
Tipologia di Attività Formativa:	**B - caratterizzante**
Docenti:	**Viviana BONO (Titolare)** **Claudia PICARDI (Titolare)** **Gianluca TORTA (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Conoscenza dei concetti studiati negli insegnamenti di Programmazione I e II.

Eventuali corsi propedeutici

Programmazione I e II.

2. Obiettivi formativi:

Il corso si propone di introdurre gli studenti ai concetti di base dell’ingegneria del software e allo sviluppo di applicazioni software, utilizzando la metodologia Agile Unified Process (UP), che sfrutta il linguaggio di modellazione UML. Lo studente dovrà saper sviluppare un'applicazione significativa individuando con chiarezza la logica applicativa, l'interazione con le basi di dati e le interfacce richieste dai requisiti. Inoltre dovrà imparare a pianificare il lavoro secondo i canoni dello sviluppo dei progetti: lavoro di gruppo, definizione degli obiettivi e delle fasi di sviluppo.

Il corso ha una forte caratterizzazione sperimentale. Per questa ragione è suddiviso in un modulo di 4 CFU di lezioni frontali e una modulo di laboratorio di 5 CFU.

3. Risultati dell'apprendimento attesi:

Lo studente acquisirà:

- conoscenze di base sull’ingegneria del software;
- conoscenza teorica e pratica della metodologia UP e dell'uso di UML in tale ambito;
- conoscenza di un tool a supporto della notazione UML;
- conoscenza della progettazione attraverso pattern di progettazione e pattern architetturali;
- capacità di presentare i risultati mediante una adeguata relazione.

4. Modalità di verifica dell'apprendimento:

L’esame consiste di tre parti. La prima parte è rappresentata da un progetto di gruppo sviluppato in laboratorio, i cui risultati sono illustrati da una breve relazione. La valutazione del progetto è fatta mediante una discussione di gruppo, seguita da una discussione individuale. La valutazione di tale parte è espressa in trentesimi. La seconda e la terza parte consistono ciascuna di una prova orale: (i) sugli argomenti di ingegneria del software; (ii) su UML, UP e Design Pattern. La valutazione per ciascuna delle prove orali viene espressa in trentesimi. La votazione finale è data da una media pesata: la parte di laboratorio ha un peso di 0.5, le due prove orali hanno ciascuna un peso di 0.25, la lode viene aggiunta se la media pesata è strettamente maggiore di 30. Le tre parti si possono sostenere insieme (nello stesso appello) oppure separatamente, in qualsiasi ordine.

5. Modalità d'insegnamento:

L’insegnamento è diviso in due parti di teoria e una di laboratorio.

Per la parte di teoria sono previste 40 ore di lezione frontali che seguono il programma riportato più avanti, integrate da casi di studio e da esercitazioni volte ad illustrare l’applicazione pratica dei concetti studiati.

La parte di laboratorio è composta da 50 ore e consiste nello sviluppo guidato di un progetto di medie dimensioni, realizzato in gruppi utilizzando la metodologia UP.

6. Attività di supporto:

Vengono effettuate consegne periodiche degli artefatti UP per il progetto di laboratorio, che man mano vengono corretti e commentati durante le ore di laboratorio.

7. Programma:

PRIMA PARTE DI TEORIA Elementi di ingegneria del software: modelli Waterfall, Spirale, V-shaped, Componend-based Development, metodologie Agili tra cui: SCRUM e extreme programming, testing: unit testing, acceptance test, white e black box testing, controllo delle versioni.

SECONDA PARTE DI TEORIA Introduzione all'UML: use case diagram, class diagram, object diagram, sequence diagram, communication diagram, state chart, activity diagram. Una metodologia Agile: Unified Process (UP). Tale metodologia verrà descritta in dettaglio a lezione durante le ore di teoria e applicata a uno studio di caso nel laboratorio. I passi previsti, in breve, sono:

- Pianificazione delle fasi di sviluppo: esse sono suddivise in ideazione, elaborazione e costruzione, possiamo pensare ad esempio a una iterazione di ideazione, due iterazioni di elaborazione e una iterazione di costruzione. Di tutte le iterazioni si prevede una durata e i documenti prodotti (tabella delle attività).
- Prima iterazione (di ideazione). A partire da una descrizione informale del progetto da sviluppare, si comincia l’analisi dei requisiti con: scelta degli attori e descrizione dei loro obiettivi, individuazione dei casi d’uso (documento prodotto: use case diagram). Si fa poi una suddivisione fra i casi d’uso a seconda della loro priorità (ad esempio: alta, media, bassa), secondo dei criteri da fissare a priori. Si produce il class diagram di dominio. Si dettagliano i casi d’uso ad alta priorità, si fanno i sequence diagram di sistema per individuare le operazioni che corrisponderanno a eventi legati all’interfaccia utente e si scrivono i contratti di quelle operazioni che si ritengono più complesse. Si producono anche il documento di visione, il documento delle specifiche supplementari, la prima versione del glossario.
- Seconda e terza iterazione (di elaborazione). Durante queste iterazioni si eseguono sia attività di design e implementazione che di analisi. Per quanto riguarda la progettazione, si sceglie la classe Controller (la classe nel sistema che “parla” con l’interfaccia utente). Si dettagliano i corpi delle operazioni dei casi d’uso ad alta priorità tramite sequence o communication diagram. Si produce un secondo class diagram, in cui si dettagliano campi e metodi e si applicano i design pattern per organizzare il software. Si cominciano a progettare la base di dati e l’interfaccia utente. Si cominciano a sviluppare gli unit test e l’implementazione dei casi d’uso ad alta priorità.

Per quanto riguarda l’analisi, si rivedono i casi d’uso sviluppati nel passo precedente. Si dettagliano i casi d’uso a priorità media e il class diagram di dominio. Si fanno i sequence diagram di sistema e i contratti per tali casi d’uso. Si aggiorna il glossario.

- Quarta iterazione (di costruzione). Si progettano e si implementano i casi d'uso a media priorità. Si rivedono eventualmente quelli ad alta priorità. Si procede con gli unit test (insieme eventualmente ad altri test). Si fa il deployment.

PARTE DI LABORATORIO Si veda la seconda parte di teoria.

8. Testi consigliati e bibliografia:

Craig Larman. Applicare UML e i pattern. Terza edizione (seconda edizione italiana), Pearson (Capitoli 1-21; Capitoli 23-26; cenni ai Capitoli 28, 29, 30, 31, 33, 34; Capitolo 36; cenni al Capitolo 38).

Roger S. Pressman, Bruce R. Maxim. Software Engineering: A Practitioner's Approach, 8/e. McGraw-Hill.

Ian Sommerville. Software Engineering, 10/e. Pearson.

Altri testi consigliati:

Fowler. UML Distilled. Terza edizione italiana. Pearson-Addison Wesley.

Gamma, Helm, Johnson, Vlissides. Design pattern. Prima edizione italiana. Pearson Education Italia-Addison Wesley.

NOTE Software a disposizione nei laboratori: Modelio e Visual Paradigm.

Il materiale del corso consiste in slide, testi e soluzioni di esercizi, esempi, riferimenti bibliografici e articoli di approfondimento per la parte di teoria; in piu', per la parte di laboratorio, sono forniti artefatti completi e semi-lavorati appartenenti al progetto proposto. Tutto il materiale viene pubblicato sulla piattaforma I-learn (http://informatica.i-learn.unito.it/).

Insegnamento	**MFN0634 - Tecnologie Web**

Insegnamento (inglese):	**Web Technologies**
CFU:	**6**
Settore:	**INF/01 - INFORMATICA**
Periodo didattico:	**1**
Tipologia di Attività Formativa:	**B - caratterizzante** **D - libera**
Docenti:	**Giancarlo Francesco RUFFO (Titolare)**

1. Prerequisiti e Propedeuticità:

Competenze attese in ingresso

Lo studente deve mostrare di possedere una buona familiarità con i principi della progammazione (imperativa, ad oggetti e basata su eventi). Inoltre, deve conoscere le basi operative per gestire una base di dati basata su SQL e per configurare/installare pacchetti software nel proprio sistema operativo.

Eventuali corsi propedeutici

- MFN0582 - Programmazione I
- MFN0585 - Programmazione II
- MFN0601 - Sistemi Operativi
- MFN0602 - Basi di Dati

2. Obiettivi formativi:

Gli obiettivi di questo corso sono i seguenti:

- Imparare a produrre siti Web dinamici, animati, interattivi e collegati ad un database in back end;
- Imparare diversi linguaggi e tecnologie per lo sviluppo Web client-side, quali HTML5, CSS, JavaScript, JQuery
- Imparare principi della programmazione server side tramite principalmente PHP e MySQL, sfruttando strumenti opensource come i comuni browser web e il server web Apache.

3. Risultati dell'apprendimento attesi:

HTML5, PhP, JavaScript, CSS, Web services, mySQL, version control, cloud deployment

4. Modalità di verifica dell'apprendimento:

Esercizi di laboratorio (almeno 4) (20%): saranno corretti e valutati i vari esercizi di laboratorio consegnati.

Relazione (30%): deve contenere la descrizione del progetto che si intende realizzare e deve essere consegnata almeno un mese prima della data di appello scelta (2000-3000 parole). Conterrà una descrizione funzionale, il wireframe del sito ed altre specifiche che saranno definite dal docente durante il corso.

Progetto finale (50%): progetto individuale che implementa quanto dichiarato dallo studente nella

relazione consegnata precedentemente. Il sito sarà sottoposto alle opportune fasi di test e valutato di conseguenza.

Per superare l'esame gli studenti devono raggiungere e superare un totale del 60% quando tutte le singoli parti sono state terminate e sommate insieme. Gli studenti non frequentanti che non potranno consegnare gli esercizi di laboratorio, potranno sostenere un esame orale sostitutivo basato sul progetto consegnato.

5. Modalità d'insegnamento:

Le lezioni teoriche saranno svolte in modalità tradizionale/frontale, con l'ausilio di diapositive che saranno proiettate in aula. La parte applicativa del corso sarà svolta in laboratorio informatico e gli studenti saranno incentivati a svolgere gli esercizi in aula e consegnarli per la correzione. Le diapositive saranno messe a disposizione degli studenti come materiale integrativo.

6. Attività di supporto:

Il materiale didattico di supporto sarà reso tramite un'apposita pagina web accessibile tramite il sistema di supporto alla didattica Moodle gestito dal nostro Corso di Laurea.

7. Programma:

- Progettazione base ed implementazione di siti Web
- Presentazione delle diverse strategie di navigazione e di organizzazione dei siti
- Tecnologie client-side, tra cui HTML5, CSS, Javascript, JSON e JQuery
- Tecnologie server side, facendo particolare attenzione alle implementazioni in PhP
- Gestione dei dati in back end
- Tecnologie emergenti (MVC, bootstrap, angular, versioning con github, etc.)

8. Testi consigliati e bibliografia:

J. Miller, V. Kirst, Marty Stepp. [Web Programming Step by Step http://www.webstepbook.com/index.sh... 2nd edition (2012)