

FACOLTÀ	Ingegneria
ANNO ACCADEMICO	2015-2016
CORSO DI LAUREA	Ingegneria Informatica e delle Telecomunicazioni
INSEGNAMENTO	Algoritmi e Strutture Dati
TIPO DI ATTIVITÀ	Caratterizzante
AMBITO DISCIPLINARE	Ingegneria Informatica
CODICE INSEGNAMENTO	01175
ARTICOLAZIONE IN MODULI	NO
NUMERO MODULI	1
SETTORI SCIENTIFICO DISCIPLINARI	ING-INF/05
DOCENTE RESPONSABILE	Giorgio Vassallo Ricercatore Confermato Università degli Studi di Palermo
DOCENTE COINVOLTO	
CFU	6
NUMERO DI ORE RISERVATE ALLO STUDIO PERSONALE	96
NUMERO DI ORE RISERVATE ALLE ATTIVITÀ DIDATTICHE ASSISTITE	54
PROPEDEUTICITÀ	Calcolatori Elettronici, Programmazione
ANNO DI CORSO	Terzo
SEDE DI SVOLGIMENTO DELLE LEZIONI	Consultare il sito www.ingegneria.unipa.it
ORGANIZZAZIONE DELLA DIDATTICA	Lezioni frontali Esercitazioni in aula e nelle aule informatiche
MODALITÀ DI FREQUENZA	Facoltativa
METODI DI VALUTAZIONE	Prova Scritta e Orale
TIPO DI VALUTAZIONE	Voto in trentesimi
PERIODO DELLE LEZIONI	Consultare il sito www.ingegneria.unipa.it
CALENDARIO DELLE ATTIVITÀ DIDATTICHE	Da definire

RISULTATI DI APPRENDIMENTO ATTESI

Conoscenza e capacità di comprensione

Lo studente alla fine del corso acquisirà una buona conoscenza dei principali algoritmi e delle più importanti strutture dati utilizzate nella programmazione avanzata. Sarà in grado di analizzare e comprendere il codice sorgente dei principali algoritmi utilizzati per lo sviluppo del software. La capacità di comprensione dello studente verrà valutata, dopo l'esposizione dei principali concetti, durante le lezioni frontali con un dialogo diretto con gli studenti.

Capacità di applicare conoscenza e comprensione

Lo studente sarà in grado di valutare le caratteristiche, i vantaggi e le limitazioni dei principali algoritmi e strutture dati. Sarà in grado di progettare, analizzare e valutare le soluzioni software a problemi di media complessità. Sarà anche in grado di sviluppare nuove soluzioni software, valutandone la qualità in termini di semplicità, efficacia ed efficienza. Tale capacità verrà valutata principalmente durante le ore di esercitazione.

Autonomia di giudizio

Lo studente sarà in grado sia di effettuare l'analisi di un problema che di progettare, a partire da precise specifiche, una opportuna soluzione software. Sarà in grado di valutarne la qualità di una soluzione software in termini di semplicità, leggibilità, efficienza e possibilità di riutilizzo. L'autonomia di giudizio verrà valutata esaminando le soluzioni proposte dagli studenti a problemi software di media complessità. Lo studente verrà incoraggiato inizialmente a trovare e valutare autonomamente soluzioni ai problemi posti, al fine di potere comprendere la qualità e l'utilità delle soluzioni proposte successivamente dal docente.

Abilità comunicative

Lo studente acquisirà la capacità di comunicare ed esprimere problematiche inerenti l'oggetto del corso. Sarà in grado di sostenere conversazioni su tematiche relative alla implementazioni software di algoritmi e strutture dati efficienti. Sarà in grado di utilizzare un linguaggio semplice e chiaro per la descrizione dei processi di analisi e di sintesi di soluzioni software a problemi di media complessità. Il carattere interattivo delle lezioni dovrà permettere la valutazione e il miglioramento delle abilità comunicative dello studente.

Capacità d'apprendimento

Lo studente dovrà sviluppare la capacità di apprendere i processi di analisi e di sintesi relativi alla codifica di algoritmi di media complessità e alla relativa implementazione di librerie e strumenti software. Il grado di apprendimento sarà valutato non in base alla capacità di memorizzare concetti specifici ma in base alla capacità di ricostruire *ex novo* partendo dal minor numero possibile di idee generali di base soluzioni software ottimali.

ORE FRONTALI	LEZIONI FRONTALI
5	Introduzione al concetto di algoritmo, complessità computazionale degli algoritmi. Semplici esempi di algoritmi per la gestione di vettori e matrici: Prodotto di matrici di numeri reali e complessi. Utilizzo di una matrice di

	interi per la simulazione di macchine a stati finiti. Algoritmi di ordinamento e calcolo della loro complessità computazionale. Implementazione di funzioni per la gestione di insiemi con operatori bitwise.
4	Liste semplici, liste circolari. Inserimento, ricerca e cancellazione di un elemento in una lista. Esempi di strutture dati elementari: pile e code. Code aperte ai due estremi, code con priorità. Implementazione di una coda con priorità con heap binario e con heap binomiale. Utilizzo di una pila per l'implementazione di un semplice interprete di espressioni matematiche in notazione polacca inversa.
4	Algoritmi sulle stringhe. Distanza di Levenshtein. Alberi binari di ricerca. Ordinamento, inserimento e ricerca di dati in un albero binario. Calcolo della complessità computazionale delle operazioni su alberi binari.
4	Funzioni di hashing e implementazione di una tabella hash con un vettore di liste semplici. Utilizzo di tabelle hash e alberi binari per l'implementazione di insiemi e di mappe associative semplici e multiple.
	ESERCITAZIONI
10	Esercitazioni sulla sintesi, l'analisi e le applicazioni degli algoritmi studiati nel modulo. Codifica degli algoritmi studiati in linguaggio C o in java.
TESTI CONSIGLIATI	Luciano M. Baroni et al., Programmazione Scientifica, Pearson Education

ORE FRONTALI	LEZIONI FRONTALI
5	Concetto di grafo, grafi orientati e non orientati, matrice di incidenza. Strutture dati per l'implementazione di un grafo. Algoritmo di Dijkstra e di Bellman-Ford per trovare il percorso di costo minimo in un grafo. Problemi NP: percorso euleriano minimo in un grafo. Tecniche di rilassamento per la soluzione approssimata di problemi NP. Minimo albero di copertura.
4	Algoritmi di ottimizzazione: discesa lungo il gradiente, metodo del gradiente coniugato. Tecniche di compressione dell'informazione, tecniche di entropy encoding e algoritmo di Huffman.
4	Strutture dati per la geometria, algebra geometrica, concetto di multivettore. Prodotto geometrico. Esempi di algoritmi e strutture dati per l'implementazione delle principali operazioni dell'algebra geometrica. Algoritmi e strutture dati per la codifica di oggetti geometrici elementari (punti rette, piani) con l'utilizzo dell'algebra geometrica.
4	Introduzione alla analisi numerica. Integrazione numerica di funzioni. Integrazione con il metodo Monte Carlo. Integrazione di equazioni differenziali.
	ESERCITAZIONI

10	Implementazione in linguaggio C e/o java degli algoritmi e delle soluzioni software studiate nel modulo.
TESTI CONSIGLIATI	Luciano M. Baroni et al., Programmazione Scientifica, Pearson Education