# UNIVERSITÀ DEGLI STUDI DI PALERMO

| | |
|---|---|
| **DIPARTIMENTO** | Ingegneria |
| **ANNO ACCADEMICO OFFERTA** | 2023/2024 |
| **ANNO ACCADEMICO EROGAZIONE** | 2024/2025 |
| **CORSO DILAUREA** | INGEGNERIA ROBOTICA |
| **INSEGNAMENTO** | ALGORITMI E STRUTTURE DATI |
| **TIPO DI ATTIVITA'** | B |
| **AMBITO** | 50289-Ingegneria informatica |
| **CODICE INSEGNAMENTO** | 01175 |
| **SETTORI SCIENTIFICO-DISCIPLINARI** | ING-INF/05 |
| **DOCENTE RESPONSABILE** | LO PRESTI LILIANA Professore Associato Univ. di PALERMO |
| **ALTRI DOCENTI** | |
| **CFU** | 6 |
| **NUMERO DI ORE RISERVATE ALLO STUDIO PERSONALE** | 96 |
| **NUMERO DI ORE RISERVATE ALLA DIDATTICA ASSISTITA** | 54 |
| **PROPEDEUTICITA'** | |
| **MUTUAZIONI** | ALGORITMI E STRUTTURE DATI - Corso: INGEGNERIA INFORMATICA<br>ALGORITMI E STRUTTURE DATI - Corso: COMP'UTER ENGINEERING |
| **ANNO DI CORSO** | 2 |
| **PERIODO DELLE LEZIONI** | 1° semestre |
| **MODALITA' DI FREQUENZA** | Facoltativa |
| **TIPO DI VALUTAZIONE** | Voto in trentesimi |
| **ORARIO DI RICEVIMENTO DEGLI STUDENTI** | **LO PRESTI LILIANA**<br>Martedì 16:00 17:00 |

**DOCENTE:** Prof.ssa LILIANA LO PRESTI

| | |
|---|---|
| **PREREQUISITI** | Buona conoscenza di elementi di programmazione (linguaggio C) |
| **RISULTATI DI APPRENDIMENTO ATTESI** | - Conoscenza e capacita' di comprensione<br>Al termine del corso, lo studente avra' conoscenza delle problematiche inerenti gli algoritmi e le strutture dati. In particolare, lo studente conoscera' nel dettaglio le metodologie di analisi degli algoritmi, le strutture dati elementari, gli algoritmi di ricerca e di ordinamento, conoscenze di base relative ai grafi e ai principali algoritmi sui grafi.<br>Per il raggiungimento di questo obiettivo il corso comprende: lezioni frontali; analisi e discussione di frammenti di programmi.<br>- Capacita' di applicare conoscenza e comprensione<br>Lo studente sara' in grado di valutare le caratteristiche, i vantaggi e le limitazioni dei principali algoritmi e strutture dati. Sara' in grado di progettare, analizzare e valutare le soluzioni software a problemi di media complessita'. Sara' anche in grado di sviluppare nuove soluzioni software, valutandone la qualita' in termini di semplicita', efficacia ed efficienza.<br>Per il raggiungimento di questo obiettivo il corso comprende esercitazioni da svolgere in aula relative all'analisi di frammenti di codice/pseudo-codice e alla progettazione di algoritmi efficienti.<br>Per la verifica di questo obiettivo l'esame comprendera' l'analisi di algoritmi gia' esistenti, l'applicazione di tecniche note e la progettazione di un algoritmo a partire dalla descrizione testuale del problema da risolvere.<br>- Autonomia di giudizio<br>Lo studente sara' in grado sia di effettuare l'analisi di un problema che di progettare, a partire da precise specifiche, una opportuna soluzione software. Sara' in grado di valutare la qualita' di una soluzione software in termini di semplicita', leggibilita', efficienza e possibilita' di riutilizzo.<br>Per il raggiungimento di questo obiettivo il corso comprende: analisi e discussioni di frammenti di codice/pseudo-codice; lezioni relative alle tecniche di analisi degli algoritmi; discussioni su possibili vantaggi e svantaggi derivanti dall'utilizzo di particolari strutture dati. Lo studente verra' incoraggiato inizialmente a trovare e valutare autonomamente soluzioni ai problemi posti, al fine di potere comprendere la qualita' e l'utilita' delle soluzioni proposte durante il corso.<br>Per la verifica di questo obiettivo l'esame comprendera' la progettazione di un algoritmo. Lo studente dovra' effettuare delle scelte progettuali in autonomia quali, per esempio, la scelta della struttura dati idonea all'implementazione di una soluzione efficiente.<br>- Abilita' comunicative<br>Lo studente acquisira' la capacita' di comunicare ed esprimere problematiche inerenti l'oggetto del corso. Sara' in grado di sostenere conversazioni su tematiche relative alla implementazione software di algoritmi e strutture dati efficienti. Sara' 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 complessita'.<br>Per il raggiungimento di questo obiettivo il corso comprendera' esercitazioni in aula durante le quali gli studenti proporranno soluzioni agli esercizi proposti e discuteranno le eventuali difficolta' riscontrate.<br>Per la verifica di questo obiettivo l'esame comprendera' la discussione orale sugli argomenti trattati nel corso.<br>- Capacita' d'apprendimento<br>Lo studente dovra' sviluppare la capacita' di apprendere i processi di analisi e di sintesi relativi alla codifica di algoritmi di media complessita' e alla relativa implementazione di librerie e strumenti software.<br>Per il raggiungimento di questo obiettivo il corso comprende: esercitazioni da svolgere autonomamente; discussione delle eventuali difficolta' riscontrate.<br>Per la verifica di questo obiettivo l'esame comprendera' la discussione di alcuni argomenti introdotti a lezione e il cui approfondimento e' lasciato agli studenti. |
| **VALUTAZIONE DELL'APPRENDIMENTO** | Sono previste due prove: una prova scritta ed una orale.<br>La prova scritta richiede lo svolgimento di esercizi atti ad accertare la capacita' del candidato di analizzare il costo computazione di un algoritmo, comprendere e simulare il funzionamento di un algoritmo, progettare un algoritmo utilizzando le tecniche di programmazione apprese durante il corso, applicare gli algoritmi studiati durante il corso. I candidati che raggiungono la sufficienza (18/30) nella prova scritta sono ammessi a sostenere la prova orale.<br>La prova orale consiste della discussione della prova scritta e di domande atte ad accertare la conoscenza del candidato degli argomenti trattati nel corso.<br>Il voto finale in trentesimi, nell'intervallo 18/30-30/30 con Lode, e' ottenuto come media della valutazione complessiva della prova scritta e di quella orale.<br>In accordo con i descrittori di Dublino, la formulazione delle prove permette una valutazione dei risultati attesi in relazione al voto finale come segue:<br>- da 18/30 a 20/30: mediocre o sufficiente conoscenza e capacita' di comprensione degli argomenti trattati, parziale capacita' di applicazione delle conoscenze acquisite per la risoluzione dei problemi proposti; parziale autonomia di giudizio, abilita' comunicative e capacita' di apprendere. |

| | |
|---|---|
| | - da 21/30 a 23/30: più che sufficiente conoscenza e capacita' di comprensione degli argomenti trattati, sufficiente capacita' di applicazione delle conoscenze acquisite per la risoluzione dei problemi proposti, sufficiente autonomia di giudizio, abilita' comunicative e capacita' di apprendere.<br>- da 24/30 a 26/30: discreta conoscenza e capacita' di comprensione degli argomenti trattati, discreta capacita' di applicazione delle conoscenze acquisite per la risoluzione dei problemi proposti, sufficiente autonomia di giudizio, abilita' comunicative e capacita' di apprendere.<br>- da 27/30 a 30/30 e lode: buona o eccellente conoscenza e capacita' di comprensione degli argomenti trattati, buona o eccellente capacita' di applicazione delle conoscenze acquisite per la risoluzione dei problemi proposti, buona o eccellente autonomia di giudizio, abilita' comunicative e capacita' di apprendere<br>Requisito minimo per il superamento dell'esame e' la dimostrata conoscenza, nelle due prove, delle nozioni di base relative a: analisi e comprensione di algoritmi, calcolo della complessita' di un algoritmo, gestione di strutture dati elementari (array, pile, code, alberi e grafi) e di strutture complesse (tabelle hash e code con priorita'), algoritmi di ordinamento e di ricerca. |
| **OBIETTIVI FORMATIVI** | Il corso trattera' in maniera approfondita lo studio degli algoritmi e delle strutture dati. Lo studente acquisira' una buona conoscenza degli aspetti relativi l'analisi dell'efficienza di un algoritmo, specifici algoritmi di ricerca e di ordinamento dei dati, l'implementazione di strutture dati quali pile, code, alberi, tabelle hash e code con priorita'. Il corso puntera' ad evidenziare i vantaggi e gli svantaggi delle tecniche presentate in base al problema che si intende risolvere. Il corso mira ad introdurre i concetti relativi a particolari tecniche di programmazione quali divide et impera, programmazione dinamica e strategie greedy. Infine, verranno trattate problematiche relative ad algoritmi per i grafi e specifici algoritmi per la robotica. |
| **ORGANIZZAZIONE DELLA DIDATTICA** | Lezioni frontali ed esercitazioni |
| **TESTI CONSIGLIATI** | Per l'analisi degli algoritmi, le strutture dati elementari e complesse, gli algoritmi di ordinamento e ricerca (For the analysis of algorithms, elementary and complex data structures, searching and sorting algorithms):<br>C. Demetrescu, I. Finocchi, G. F. Italiano (2008). Algoritmi e strutture dati - seconda edizione. McGraw-Hill<br><br>Per gli algoritmi relativi al campo della robotica, Capitolo 6 di (For algorithms related to the robotics field):<br>Dudek, G., & Jenkin, M. (2010). Computational Principles of Mobile Robotics Cambridge: Cambridge University Press. doi:10.1017/CBO9780511780929.009<br><br>Per ulteriori approfondimenti (For further studies, and available also in English):<br>T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, L. Colussi (2010). Introduzione agli algoritmi e strutture dati - terza edizione. McGraw-Hill |

## PROGRAMMA

| ORE | Lezioni |
|---|---|
| 6 | Introduzione agli algoritmi. Algoritmi iterativi e ricorsivi. Occupazione di memoria ed efficienza di un algoritmo. Analisi di algoritmi. Le notazioni asintotiche: O-grande, Omega-grande, Theta-grande. Analisi del caso peggiore, migliore e medio. Analisi di algoritmi ricorsivi attraverso metodo dell'iterazione e teorema master. |
| 5 | Definizione di struttura dati; operazioni tipiche per la gestione di strutture dati (inserimento, cancellazione e ricerca); rappresentazioni indicizzate e collegate; array dinamici; strutture collegate: record e puntatori; lista semplice e lista doppiamente concatenata. Pile e code. Alberi: definizioni di base; tecniche di rappresentazione di alberi in memoria (vettore dei padri, vettore posizionale, puntatori ai figli, lista di puntatori ai figli, primo figlio-fratello succesivo); algoritmi di visita di alberi e loro complessita' computazionale. |
| 4 | Il problema della ricerca. Ricerca sequenziale e ricerca binaria. Il problema dell'ordinamento. Lower-bound degli algoritmi di ordinamento basati sul metodo del confronto nel caso peggiore. Algoritmi di ordinamento con upper-bound quadratico: selectionSort, bubbleSort. Algoritmi di ordinamento in tempo n log n: mergeSort e quickSort. Ordinamento in loco. Ordinamento di interi: integerSort, bucketSort. |
| 3 | Alberi binari di ricerca e operazioni di ricerca, inserimento e cancellazione. Alberi AVL e fattore di bilanciamento. Operazioni di rotazione SS, DS, SD, DD. |
| 2 | Tabelle ad accesso diretto e fattore di carico; tabelle hash; definizione di collisione; funzioni hash perfette e non perfette; uniformita' delle funzioni hash; risoluzione delle collisioni: tabelle hash con liste di collisione; analisi dei costi per le tabelle hash. |
| 2 | Code con priorita' implementate attraverso d-heap. Operazioni di inserimento e cancellazione. |
| 5 | Tecniche algoritmiche: divide et impera; programmazione dinamica; strategie greedy. Applicazioni di programmazione dinamica: sotto-vettore di massimo valore; cammini di valore minimo; distanza tra stringhe (edit distance); massima sotto-sequenza comune. Esempi di tecniche greedy. Cenni sui problemi di ottimizzazione numerica. Definizione del problema dello zaino. Soluzione greedy per il problema dello zaino. Introduzione all'algoritmo di discesa lungo il gradiente. |

# PROGRAMMA

| ORE | Lezioni |
|---|---|
| 6 | Grafi e rappresentazione di grafi in memoria: lista di archi, liste di adiacenza, liste di incidenza, matrici di adiacenza, matrici di incidenza. Algoritmi di visita dei grafi; grafi connessi e calcolo delle componenti connesse di un grafo non orientato; connettivita' forte in grafi orientati e calcolo delle componenti fortemente connesse in grafi orientati; definizione di minimo albero ricoprente; algoritmi di Prim e Kruskal; definizione di cammino minimo; distanza tra nodi; algoritmo di Bellman-Ford e algoritmo di Dijkstra. |
| 2 | Teoria della NP-completezza: cenni sulle macchine non deterministiche; classi di complessita': problemi in P, NP, NP-C e NP- H. Esempi. |
| 6 | Algoritmi per la rappresentazione dell'ambiente nella robotica mobile. Rappresentazione dello spazio attraverso griglie. Griglie di occupazione (occupancy map). Decomposizione attraverso QuadTree. Triangolazione di Delaunay. Tassellazione di Voronoi. Pianificazione di cammini. Metodi dei potenziali. Grafi di visibilità. |

| ORE | Esercitazioni |
|---|---|
| 3 | Stima del costo computazionale e dell'occupazione di memoria di algoritmi. Individuazione della classe di complessita' di un algoritmo. |
| 3 | Funzionamento di strutture dati: alberi binari di ricerca; tabelle hash, alberi AVL, code con priorita'. |
| 5 | Progettazione di algoritmi per la risoluzione di problemi di media complessita'. |
| 2 | Simulazione di algoritmi per i grafi e per la robotica. |