Programma Introduzione agli algoritmi a.a. 2015/2016

Parte I

INTRODUZIONE: Definizione di algoritmo. Studio di algoritmi: progetto, correttezza e analisi. Analisi della complessità computazionale di un algoritmo: approccio sperimentale e teorico.

COMPLESSITA' COMPUTAZIONALE: Definizione di tempo di esecuzione: caso migliore e peggiore. Efficienza asintotica degli algoritmi: limite asintotico superiore e notazione O (o grande), limite asintotico inferiore, notazione Ω (omega grande), e limite asintotico stretto, notazione Θ (theta grande).
Ricerca sequenziale in un array (o lista). Ricerca binaria in un array (o lista).

ALGORITMI DI ORDINAMENTO: Il problema dell'ordinamento. Algoritmi di ordinamento incrementali: InsertionSort, SelectionSort e BubbleSort. Introduzione al concetto di albero: albero come grafo connesso aciclico. alberi radicati, alberi binari. Numero di nodi, numero di foglie e altezza di un albero. Limitazione inferiore e superiore per l'altezza di un albero binario in termini del suo numero di nodi. Alberi di decisione. Teorema sulla limitazione inferiore per il tempo di esecuzione del problema dell'ordinamento, nel modello basato su confronti (dimostrazione).
Algoritmo di fusione di due vettori ordinati.
Algoritmi lineari di partizione: Partition semplice e quello di Hoare.
Un algoritmo lineare per dividere un array in tre parti, Tripartition.
Ordinamenti in tempo lineare: CountingSort. Stabilità degli algoritmi di ordinamento.
L' Heap binario: un array visto come albero binario. MaxHeap e l'operazione di lettura e estrazione del massimo in un MaxHeap. La funzione, MaxHeapify, che ripristina la proprietà del maxheap violata in una unica posizione i, data in input insieme al maxheap. Algoritmo per la trasformazione di un array qualunque in un MaxHeap, BuildMaxHeap, con la dimostrazione della linearità del suo tempo di esecuzione; algoritmo Heap-sort;

Parte II

ALGORITMI RICORSIVI: Esempi di algoritmi ricorsivi.
Algoritmi di ordinamento ricorsivi: MergeSort e QuickSort e il loro tempo di esecuzione nei vari casi.
Analisi di algoritmi ricorsivi. Equazioni di ricorrenza. Soluzione di una equazione di ricorrenza: metodo di sostituzione.

STRUTTURE DATI ELEMENTARI ED ALBERI:
Pile, code: operazioni caratterizzanti e cenni alla loro implementazione con strutture ad accesso diretto, arrays, e ad accesso sequenziale, liste

concatenate. Esempi di algoritmi che utilizzano una pila (verifica buon bilanciamento di una stringa di parentesi, calcolo dello span di un array) o di una coda (visita per livelli di un albero binario).
Code di priorità: operazioni caratterizzanti e implementazione con un heap binario. Algoritmi per incrementare la chiave di un elemento e per aggiungere o cancellare un elemento in un maxheap;
Rappresentazioni in memoria di alberi binari. Visite di alberi binari: preordine, postordine, inordine. Risoluzione dell'equazione di ricorrenza relativa agli algoritmi ricorsivi di visita. Calcolo ricorsivo dell'altezza di un albero binario, del numero di nodi e di quello delle foglie in un albero binario con l' analisi del tempo di esecuzione.

DIZIONARI: La struttura dati Dizionario. Implementazioni banali: vettori ordinati e liste collegate. Alberi binari di ricerca (ABR): algoritmi per la ricerca, l'inserimento e la cancellazione su un ABR e analisi del tempo di esecuzione.
Calcolo del minimo e del massimo in un ABR, calcolo del precedente e del successivo in un ABR. Visita inorder iterativa e l'analisi del suo tempo di esecuzione.
Alberi binari di ricerca bilanciati. Alberi bilanciati in altezza, AVL. Dimostrazione che l'altezza di un albero AVL è logaritmica. Fattore di bilanciamento e rotazioni (Left-Left, Right-Right, Right-Left e Left-Right) e il loro uso nel ribilanciare un AVL sbilanciato a seguito di un inserimento o di una cancellazione.