Programma
Introduzione agli algoritmi a.a. 2016/2017

Parte I

INTRODUZIONE: Definizione di algoritmo. Studio di algoritmi: progetto, correttezza e analisi. Analisi del tempo di esecuzione 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;
Algoritmi per l'inserimento e la cancellazione di un elemento in un MaxHeap.