Domande orale teoria APA

Questa è una raccolta di domande prese dagli orali di APA negli anni: domande prese e riordinate da qui.

Le domande sono state private della risposta e inserite senza i doppioni (o almeno ci ho provato).

- Definizione DAG
- Cammino di Hamilton
- Partition di Binary Search Trees
- Struttura dati adatta per rappresentare una Binary Search Tree
- Equazione alle ricorrenze
- Ricerca dicotomica
- Caratteristiche di un albero: altezza, profondità, grado e rango
- Differenza tra grafo e albero
- Differenza tra Binary Search Tree e Heap
- Algoritmi di ricerca stabili
- Differenza tra l'algoritmo di Dijkstra e di Bellman-Ford
- Differenze tra il paradigma greedy e la programmazione dinamica
- Tabelle di hash
- Valore alpha
- Definizione componente connessa in un grafo
- Complessità heapbuild
- Complessità heapify
- Complessità PQInsert
- Algoritmo disposizioni ripetute con equazione alle ricorrenze
- Cammini di una certa lunghezza per grafi orientati e non orientati
- Complessità degli algoritmi di Dijkstra e Bellmann-Ford
- Codici di Huffman: a cosa servono?
- Ordinamento topologico inverso
- Massimo, minimo, predecessore e successore nel Binary Search Tree
- Fattore dimensionamento nelle tabelle di hash
- Nella visita di profondità è possibile rilevare cicli?
- Equazione alle ricorrenze del merge sort
- Tipologia di algoritmi (P, NP, etc)
- Ricerca in profondità e archi T F B C
- Albero completo e bilanciato: differenze
- Scrivere un albero binario che è bilanciato ma non completo
- Complessità delle operazioni negli algoritmi su un Binary Search Tree
- Stack
- Stack Frame
- Ricorsione tail recursive e perchè è utile
- Cos'è un ciclo in un grafo
- Proprietà dell'heap
- Puoi salvare un Binary Search Tree in un array? (Differenze con heap)
- Limite inferiore di complessità defli algoritmi basati sul confronto
- Che struttura dati usa la ricerca in ampiezza?
- Che struttura dati usa l'algoritmo di Dijkstra e come funziona?
- Proprietà di un heap

- Matrice e lista di adiacenze, vantaggi e complessità
- Come implementare la lista di adiacenze quando non si ha l'informazione sul numero di vertici del grafo?
- Cos'è un albero ricoprente minimo?
- Esempio di un problema NP-Completo
- Hashing (come implementarlo, collisioni, delete in caso di Open Addressing)
- Complessità degli algoritmi di ordinamento basati su confronto e dimostrazione
- Punti di articolazione
- Componenti fortemente connesse
- Complessità del grafo trasposto
- Equazione alle ricorrenze del merge sort
- Complessità di una visita in profondità
- Cammino semplice
- Programmazione dinamica
- Differenza tra programmazione dinamica e memoization
- Algoritmi di ordinamento ricorsivi basati sul confronto: quali di questi sono in loco?
- Possibili implementazioni di coda a priorità: è possibile implementarla anche con un vettore non ordinato?
- Delete in una tabella di hash
- Limite inferiore lasco degli algoritmi di ordinamento basati sul confronto
- Tabella di simboli
- Tabella di hash
- Quick Sort: qual è il caso peggiore?
- Costo della ricerca in profondità
- Rappresentazioni dei grafi (matrice o lista delle adiacenze) con vantaggi e svantaggi
- Tabelle di simboli ad accesso diretto
- Cancellazione in una tabella di simboli con open addressing (solamente teorico, no codice)
- Come si trasformano le stringhe in interi per le tabelle di hash
- Il limite inferiore degli algoritmi basati sul confronto
- Come funziona la ricorsione (salvataggio nello stack, indirizzo di ritorno)
- Dov'è lo stack fisicamente e come è strutturato
- Algoritmi greedy
- Backtrack
- Cos'è la partition e a cosa serve?
- Cos'è il rango di un Binary Search Tree?
- Cosa sono gli IBST (e come funzionano)?
- Permutazioni (algoritmo)
- Tabelle di simboli ad accesso diretto, con costi delle operazioni, pro e contro
- Tabelle di hash ed occupazione della memoria
- Metodi per implementare una coda a priorità
- Confronto tra Binary Search Trees e Heap
- Codice che calcola l'altezza di un albero binario con paradigma divide et impera
- Shellsort
- Quando un algoritmo si dice stabile/non stabile e in loco/non in loco
- Cosa significa greedy
- Perché l'algoritmo di Dijkstra è greedy?
- Grado di un vertice

- Punti di articolazione
- Cos'è una tabella di simboli e qual è l'operazione che va ottimizzata al meglio?
- Disegnare un grafo composto da un'unica componente fortemente connessa
- Complessità dell'algoritmo di HeapSort
- Archi sicuri
- Tagli
- Archi leggeri
- Teorema e corollario degli archi sicuri
- Algoritmi greedy
- Inversione di una lista
- Binary Search Tree: funzione che ritorna il numero di nodi che hanno esattamente un figlio
- Dato un cammino minimo tra due vertici A e B, se inserisco un vertice C tra i due vertici: perché i sottocammini che vanno da A a C e da C a B sono considerati dei sottocammini minimi?
- Heap e funzioni di Heapify, BuildHeap e HeapSort
- Costi di tali funzioni
- Approssimazione di Stirling
- Binary Search Tree: vari casi di cancellazione, successore/predecessore, funzione select
- Grafo completamente connesso
- Numero di vertici e a quale algoritmo del calcolo combinatorio è possibile associare il numero di questi vertici
- Powerset
- Quante soluzioni generano le permutazioni ripetute di n elementi
- Funzione che generi tutte le permutazioni di 3 bits in pseudocodice
- Equazione alle ricorrenze generica
- Complessità dell'algoritmo di partition e sua struttura generica
- Definizione ricorsiva di un Binary Search Tree
- Tabelle di simboli Universal Hashing
- Polinomio di Horner
- Quale può essere uno dei problemi del metodo ricorsivo?
- Come può essere risolto?
- All'interno di una funzione ricorsiva, in quale punto faresti il controllo sul fatto che il problema sia già stato calcolato?
- Quale potrebbe essere una struttura utile a questo fine?
- Cosa dovrebbe contenere questa struttura dati?
- Come possiamo definire un albero tramite un grafo, cos'è un albero, albero bilanciato e completo
- Differenza tra cammini minimi e alberi ricoprenti minimi
- QuickSort e perché non consideriamo il caso peggiore
- Alberi non radicati
- Realloc: come funziona, quando la usiamo, come la usiamo, perché raddoppiamo la dimensione e non aumentiamo semplicemente di uno e costo della realloc
- Tabella di hash: valore di M nell'open addressing, fattore di carico, costo di inserzione e double hashing (con codice)
- Metodi per rappresentare un nodo di un Binary Search Tree
- Analisi della complessità della moltiplicazione tra due interi
- Quando l'algoritmo di Dijkstra fornisce una soluzione ottima e quando no
- Come implementare i cammini massimi
- Cammini massimi nei DAG

- Paradigma greedy: appetibilità fisse e variabili
- Complessità di un Binary Search Tree non bilanciato e complessità di un Binary Search Tree bilanciato
- Come funziona il partizionamento di un Binary Search Tree?
- Spiegare il significato di “left child right sibling”
- Esiste un limite inferiore alla complessità degli algoritmi di ordinamento? Dimostra come si ricava
- Qual’è la complessità del QuickSort? Scrivi l’equazione alle ricorrenze nel caso peggiore
- Limite inferiore degli ordinamenti basati sul confronto e dimostrazione (formula di Stirling)
- Gestione degli elementi cancellati in una hash table
- Come si può implementare un ordinamento basato sulla hash table
- Cos’è e come si trova una chiave di un dato rango in un Binary Search Tree?
- Quanto valgono A e B (nella equazione alle ricorrenze) per la ricerca dicotomica?
- Quando si verifica il caso peggiore dell’algoritmo di QuickSort?
- Cosa rappresenta la n nell’equazione alle ricorrenze del QuickSort?
- Modelli di calcolo combinatorio per il problema delle 8 regine
- Problema della colorazione di una mappa (grafo planare) come adattamento del problema di partizione di un grafo
- Heap: cosa sono, operazioni principali e complessità
- Algoritmi derivanti dai minimum spanning trees
- IBST: descrizione ed estensioni da eseguire al Binary Search Tree di base per implementarlo
- Condizione per cui due intervalli si intersecano
- Order Binary Search Tree: domande su numero di nodi, altezza dell’albero
- Rotazione di un Binary Search Tree
- Differenze tra combinazioni, disposizioni, permutazioni
- Dimostrazione complessità
- Omega grande per algoritmi di ordinamento basati sul confronto
- In che senso gli algoritmi di ricerca dei cammini minimi e dei Minimum Spanning Tree sono greedy?
- Notazione sigma grande e theta grande
- Cos’è la relaxation, come viene utilizzata nell’algoritmo di Dijkstra
- Cos’è il rango in un Binary Search Tree, come modificare l’ADT Binary Search Tree per applicare algoritmi di ricerca della chiave con rango k
- Dal punto di vista insiemistico, gli alberi bilanciati sono un sottoinsieme degli alberi completi, o il contrario?
- PQ Change
- Come abbattere la complessità dell’algoritmo PQ Change?
- Albero non radicato
- Partition di un Binary Search Tree: breve spiegazione del funzionamento e utilizzo della funzione (occorre spiegare che viene utilizzata nell’algoritmo di _____ dei Binary Search Tree)