27 luglio 2021

mercoledì 21 agosto 2024 15:24

Esame ASD 27 luglio 2021 - 3 domande (gli studenti/studentesse con certificazione DSA o che sostengono l'esame in modalità integrata devono fare solo le prime due e DEVONO SCRIVERE SUL FOGLIO, ACCANTO AL NOME, "DSA" oppure "INTEGRATA"); tempo totale 45 minuti; punteggio massimo 6.9; sufficienza 4 punti

Domanda #1 (per tutti)

Si consideri una tabella di hash con liste di collisione che implementa un dizionario D in cui le chiavi sono stringhe di 4 cifre decimali che indichiamo con c1, c2, c3, c4 e i valori sono stringhe.
La tabella di hash ha 7 bucket indicizzati da 0 a 6 e la funzione di hash è **h: (c1+c2+c3+c4) mod 7**.

Passo 1 (senza voto): scrivete sul foglio della risposta le cifre 0 4 0 1 4 0, poi la vostra matricola universitaria (7 cifre), poi la vostra matricola universitaria al contrario (7 cifre), poi la vostra data di nascita nel formato ANNO (4 cifre) MESE (2 cifre) GIORNO (2 cifre), e infine le cifre 1 3 0 3

Ad esempio, se la vostra matricola è 1234567 e siete nati il primo gennaio 2000, dovrete scrivere sul foglio

0 4 0 1 4 0	1 2 3 4 5 6 7	7 6 5 4 3 2 1	2 0 0 0 0 1 0 1	1 3 0 3
0 4 0 1 4 0	*matricola*	*matricola al contrario*	*data nascita*	1 3 0 3

Passo 2 (senza voto): disegnate sul foglio della risposta uno schema come quello mostrato sotto e immettete i dati indicati:

1. nella prima riga, il numero che avete generato al Passo 1, diviso in gruppi di quattro cifre: questi gruppi rappresenteranno le chiavi
2. nella seconda riga, i valori associati alle chiavi, ovvero le stringhe "amo", "barca", "cesto", "daino", "edera", "foglia", "gatto", "hotel"
3. nella terza riga, il valore di hash associato a ogni chiave, che **dovrete calcolare** usando la formula h: (c1+c2+c3+c4) mod 7

0401	4012	3456	7765	4321	2000	0101	1303
amo	barca	cesto	daino	edera	foglia	gatto	hotel
val. di h(0401)	val. di h(4012)	val. di h(3456)	...	...	...	...	...

Passo 3 (1/3 del punteggio): Il dizionario D implementato dalla tabella di hash è inizialmente vuoto. **Disegnate** la tabella di hash che si ottiene dagli inserimenti delle stringhe "amo", "barca", "cesto", "daino", "edera", "foglia", "gatto", "hotel" associate alle chiavi su quattro cifre così come le avete calcolate al Passo 2, nell'ordine in cui le avete scritte da sinistra verso destra. Ricordate che in un dizionario non si possono ammettere chiavi duplicate: nel caso ce ne fossero ignorate l'inserimento del duplicato e scrivete esplicitamente sul foglio "questa coppia chiave-valore è stata ignorata perché la chiave è duplicata". Continuando l'esempio, si dovrebbe disegnare la tabella di hash risultante da:

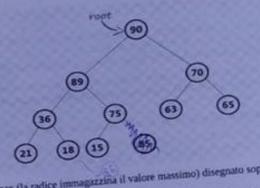
```
insert(D, "0401", "amo").
insert(D, "3012", "barca").
insert(D, "3456", "cesto").
.....
```

Passo 4 (1/3 del punteggio): quali sono le due principali "buone proprietà" che una tabella di hash deve avere? Rispetto alle chiavi che avete ottenuto al Passo 2.1, la funzione di hash **h** ha queste due "buone proprietà"? Motivate esaurientemente la risposta. **Risposte generiche e non relative alla funzione h applicata alle chiavi così come ottenute al Passo 2.1 comportano l'assegnazione di 0 punti.**

Passo 5 (1/3 del punteggio): sotto l'ipotesi che la funzione di hash abbia tutte le "buone proprietà" che dovrebbe avere, qual è la complessità dell'operazione di **cancellazione** di un elemento data la sua chiave nel caso peggiore? Non basta copiare la risposta dalle slide del corso o da altro materiale didattico: è necessario spiegare cosa rappresentano i parametri che caratterizzano la risposta e motivarla esaurientemente.

La mancanza di motivazione chiara ed esauriente comporta l'assegnazione di 0 punti a questa domanda.

Domanda #2 (per tutti)



A

Si consideri lo Heap Binario di tipo max (la radice immagazzina il valore massimo) disegnato sopra, che indicheremo con A.

[1/2 del punteggio] Si spieghino dettagliatamente, mediante disegni chiari e autoesplicativi, i passaggi principali della chiamata
`insert(86, "elem", A);`
effettuata sullo heap A.
La chiamata `insert(86, "elem", A)` rappresenta il caso peggiore della operazione insert sullo heap A, rispetto alla complessità temporale? Motivare la risposta.

[1/2 del punteggio] Si spieghino dettagliatamente, mediante disegni chiari e autoesplicativi, i passaggi principali della chiamata
`deleteMax(A);`
effettuata sullo heap A modificato a seguito dell'inserimento dell'elemento con chiave 86.

Risposte generiche e che non fanno riferimento allo heap A verranno considerate insufficienti.

Domanda #3 (se siete in possesso di una certificazione DSA oppure se state svolgendo lo scritto in modalità integrata non dovete rispondere a questa domanda)

[1/2 punteggio] Si scriva quanto vale la complessità di quicksort nel caso peggiore e si illustri dettagliatamente **quando** si ricade nel caso peggiore e **come** si calcola la complessità di quicksort nel caso peggiore.

[1/2 punteggio] Si simuli una esecuzione della chiamata di quicksort sulla sequenza riportata sotto, con il pivot scelto sempre, "per magia", come elemento mediano della sottosequenza da riordinare. Questa situazione coincide con il caso migliore di quicksort, con il caso peggiore, o con il caso medio? Qual è la complessità di quicksort in questa specifica situazione?

1 100 2 90 3 80 70 60 50

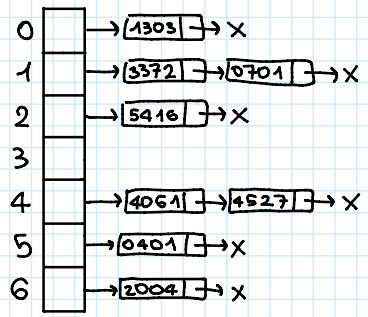
Domanda 1.1

040140 6145273 3725416 20040701 1303

3. nella terza riga, il valore di hash associato a ogni chiave, ...
(c1+c2+c3+c4) mod 7

0401	4061	4527	3372	5416	2004	0701	1303
amo	barca	cesto	daino	edera	foglia	gatto	hotel
5	4	4	1	2	6	1	0

5 mod 7 11 mod 7 18 mod 7 15 mod 7 16 mod 7 6 mod 7 8 mod 7 7 mod 7



Domanda 1.2

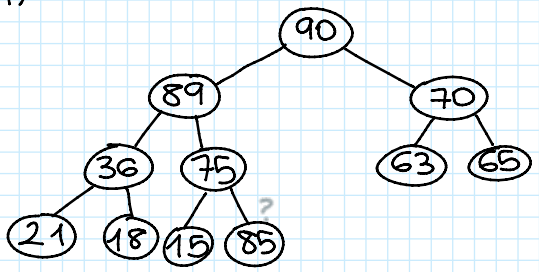
Le due principali buone proprietà sono l'essere calcolabile in tempo costante e l'uniformità. La prima proprietà è soddisfatta in quanto la funzione di hash viene calcolata sempre su chiavi formate da 4 cifre; inoltre, le operazioni aritmetiche del calcolo sono semplici, quindi calcolabili in tempo costante (somma e modulo). La seconda proprietà invece non è soddisfatta: il bucket 3 è vuoto e i bucket 1 e 4 son più pieni degli altri: questo significa che alcune chiavi generano più frequentemente lo stesso valore, quindi l'uniformità non è possibile.

Domanda 1.3

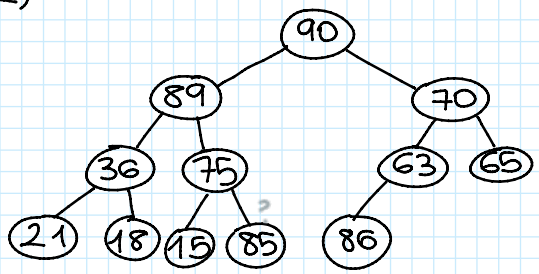
Nel caso peggiore, la complessità è $O(n/m)$. Questo perchè se la funzione di hash ha tutte le "buone proprietà", allora ogni bucket ha n/m elementi. Quindi se la ricerca avviene in $O(n/m)$ e la cancellazione avviene in $O(1)$, la complessità della delete è $O(n/m \cdot 1) = O(n/m)$.

Domanda 2.1

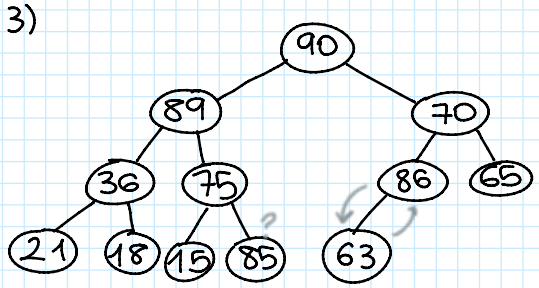
1)



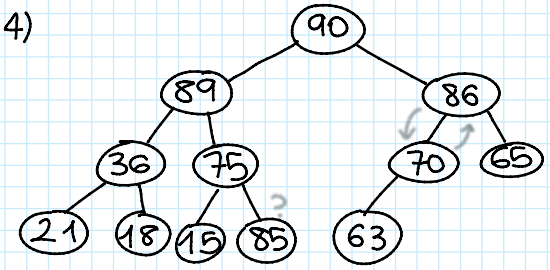
2)



3)



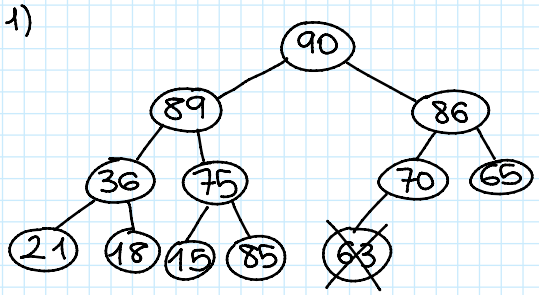
4)



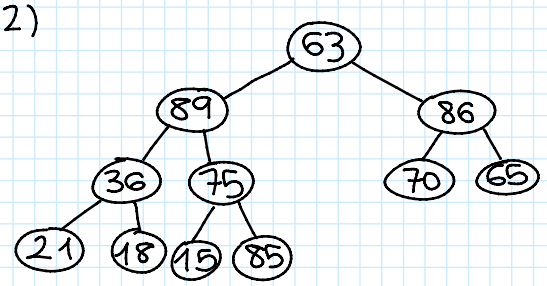
Non rappresenta il caso peggiore in quanto 86 non arriva fino alla radice.

Domanda 2.2

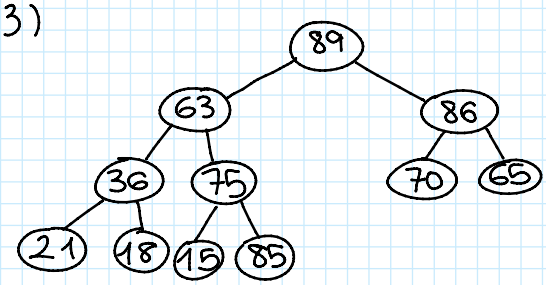
1)



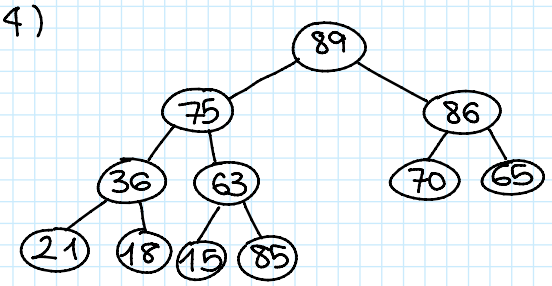
2)



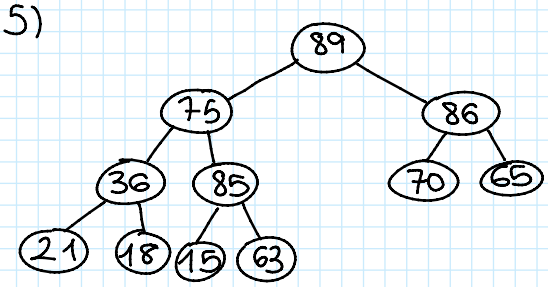
3)



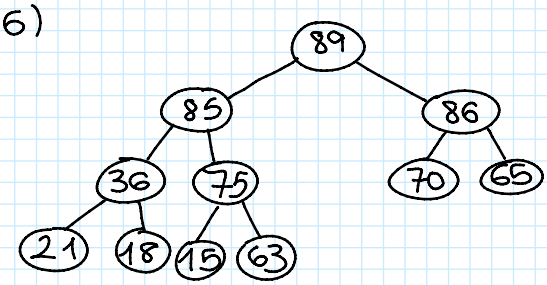
4)



5)



6)



Domanda 3.1

Nel caso peggiore, quicksort ha complessità $O(n^2)$ e si ricade in questo caso quando si sceglie come pivot l'elemento minore o maggiore della sequenza. Per calcolare la complessità si sommano le operazioni fatte ad ogni chiamata ricorsiva. La "prima volta" si eseguono n operazioni, la seconda volta n-1 operazioni e così via fino all'ultimo livello in cui si esegue 1 operazione. Questo calcolo si sviluppa nella sommatoria per "i" che va da 1 a n.

Quindi $n+(n-1)+(n-2)+\ldots+1=n^2$.

Domanda 3.2

1 100 2 90 3 80 70 60 50 Ci ritroviamo nel caso migliore e la complessità è $O(n \log n)$

Domanda 3.2

1 100 2 90 3 80 70 60 50

1 2 3 50 60 100 90 80 70

1 2 3 50 60 70 80 100 90

1 2 3 50 60 70 80 90 100

Ci ritroviamo nel caso migliore e la complessità è $O(n \log n)$