

Iniziato giovedì, 6 giugno 2024, 18:02

Stato Completato

Terminato giovedì, 6 giugno 2024, 18:30

Tempo impiegato 28 min. 47 secondi

Valutazione 5,94 su un massimo di 8,10 (73,33%)

Domanda 1

Risposta corretta

Punteggio ottenuto 0,54 su 0,54

Questa domanda vale 0,54 punti

Si consideri il seguente codice:

```
Error funzioneOffuscata(const Key k, TipoDiDatoOffuscato& s)
{
    CosaOffuscata current = s[h(k)];
    CosaOffuscata prev = emptyCosa;
    while (current!=emptyCosa) {
        if ((current->elem).key == k)
        {
            if (prev != emptyCosa)
                prev->next = current->next;
            else
                s[h(k)] = current->next;
            delete current;
            return OK;
        }
        else
        {
            prev = current;
            current = current->next;
        }
    }
    return FAIL;
}
```

Di quale funzione si tratta?

- ☐ a. Della funzione deleteMax in un binary search tree
- ☐ b. Della funzione deleteElem in uno heap binario
- ☒ c. Della funzione deleteElem in una tabella di hash con liste di collisione ✓
- ☐ d. Della funzione deleteElem in un binary search tree
- ☐ e. Della funzione deleteElem in una lista circolare con sentinella
- ☐ f. Della funzione deleteElem in una tabella ad accesso diretto
- ☐ g. Della funzione deleteMax in uno heap binario

La risposta corretta è:

Della funzione deleteElem in una tabella di hash con liste di collisione

Domanda 2

Risposta errata

Punteggio ottenuto 0,00 su 0,27

Un cammino in un grafo è

Scegli un'alternativa:

- ☐ a. una sequenza di vertici $[v_0, v_1, \dots, v_n]$ tali che ogni coppia di vertici consecutivi nella sequenza (v_i, v_{i+1}) è connessa da un arco e tutti gli archi (v_i, v_{i+1}) sono uguali tra loro
- ☐ b. una sequenza di vertici $[v_0, v_1, \dots, v_0]$ tali che ogni coppia di vertici consecutivi nella sequenza (v_i, v_{i+1}) è connessa da un arco e il primo vertice coincide con l'ultimo (si noti che v_0 è il primo e l'ultimo elemento della sequenza)
- ☐ c. una sequenza di vertici $[v_0, v_1, \dots, v_n]$ tali che ogni coppia di vertici consecutivi nella sequenza (v_i, v_{i+1}) è connessa da un arco e tutti gli archi (v_i, v_{i+1}) sono distinti tra loro
- ☒ d. una sequenza di vertici $[v_0, v_1, \dots, v_n]$ tali che ogni coppia di vertici consecutivi nella sequenza (v_i, v_{i+1}) è connessa da un arco e non ci sono vertici ripetuti nella sequenza

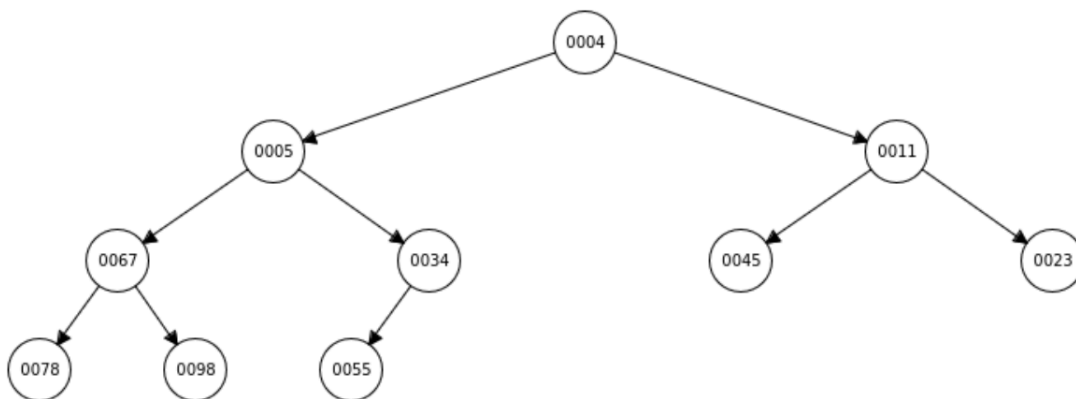
La risposta corretta è: una sequenza di vertici $[v_0, v_1, \dots, v_n]$ tali che ogni coppia di vertici consecutivi nella sequenza (v_i, v_{i+1}) è connessa da un arco e tutti gli archi (v_i, v_{i+1}) sono distinti tra loro

Domanda 3

Risposta errata

Punteggio ottenuto 0,00 su 0,27

Si consideri l'albero binario radicato in 3 rappresentato sotto: qual è l'ordine con cui vengono visitati i nodi con una visita simmetrica?



- ☒ a. 4-5-11-23-34-45-55-67-76-98
- ☐ b. 4-5-11-45-23-55-34-67-76-98
- ☐ c. 78-67-98-5-55-34-4-45-11-23
- ☐ d. 78-98-5-55-34-4-45-11-67-23

La risposta corretta è: 78-67-98-5-55-34-4-45-11-23

Domanda 4

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Qual è la complessità temporale dell'operazione del TDD Set **emptySet**: -> **Set** nel caso migliore, quando l'insieme è implementato con un **bit vector di n elementi**?

Scegli un'alternativa:

- ☐ a. Theta (n^2)
- ☐ b. Theta (1)
- ☒ c. Theta (n) ✓

La risposta corretta è: Theta (n)

Domanda 5

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Durante il corso abbiamo accennato a un algoritmo per cercare i cammini minimi in un grafo con pesi non negativi sugli archi. Come si chiama questo algoritmo?

Scegli un'alternativa:

- ☐ a. Algoritmo degli alberi AVL
- ☐ b. Algoritmo di ricerca in ampiezza (BFS)
- ☐ c. Algoritmo di ricerca in profondità (DFS)
- ☒ d. Algoritmo di Dijkstra ✓

La risposta corretta è: Algoritmo di Dijkstra

Domanda 6

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Le tabelle ad accesso diretto...

Scegli un'alternativa:

- ☒ a. sono molto rigide: le chiavi possono solo essere dei numeri interi, ed il loro range deve essere per forza $[0, m-1]$, con m dimensione della tabella ✓
- ☐ b. sono estremamente vantaggiose in termini di occupazione dello spazio: se nel dizionario sono presenti n elementi, occupo uno spazio proporzionale ad n. In particolare, se il dizionario è vuoto non occupo spazio.
- ☐ c. sono estremamente flessibili: le chiavi possono essere elementi qualunque, ad esempio stringhe, caratteri, etc.

La risposta corretta è: sono molto rigide: le chiavi possono solo essere dei numeri interi, ed il loro range deve essere per forza $[0, m-1]$, con m dimensione della tabella

Domanda 7

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Si consideri la struttura dati "liste di adiacenza con vertici memorizzati in una lista" per rappresentare un grafo non orientato con **n** nodi ed **m** archi. La complessità temporale di **removeVertex** del vertice **v** è, nel caso peggiore

Scegli un'alternativa:

- ☐ a. $\Theta(m)$
- ☒ b. $\Theta(n + m)$ ✓
- ☐ c. $\Theta(\text{grado}(v))$
- ☐ d. $\Theta(n)$

La risposta corretta è: $\Theta(n + m)$

Domanda 8

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Contesto:

Analisi di complessità delle operazioni su liste, Struttura dati: array dinamico con size e maxsize (e ridimensionamento)

Dimensione della lista: n

La posizione di Elem appartiene a \mathbb{N} , insieme dei numeri naturali

Nel contesto indicato sopra, qual è la complessità, nel caso migliore e peggiore, di **get: $\mathbb{N} \times \text{List} \rightarrow \text{Elem}$**

Attenzione: N non è la dimensione della lista ma è l'insieme dei numeri naturali, da cui si prende un valore che indica la posizione di Elem

Scegli un'alternativa:

- ☐ a. $\Theta(1)$ (se la posizione di passata a get == 0) nel caso migliore; $\Theta(n)$ (se la posizione passata a get $\geq n$) nel caso peggiore
- ☒ b. $\Theta(1)$ nel caso migliore e peggiore ✓
- ☐ c. $\Theta(n)$ nel caso migliore e peggiore

La risposta corretta è: $\Theta(1)$ nel caso migliore e peggiore

Domanda 9

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

I Binary Search Tree e gli heap binari

Scegli un'alternativa:

- ☐ a. sono entrambi alberi binari completi
- ☐ b. sono entrambi alberi generici in cui l'altezza può variare da n (con n numero dei nodi) a n^2
- ☐ c. sono entrambi alberi binari quasi completi
- ☒ d. sono entrambi alberi binari ✓
- ☐ e. sono entrambi alberi generici in cui l'altezza è sempre al più $\log n$, ma può essere anche minore

La risposta corretta è: sono entrambi alberi binari

Domanda 10

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Un albero **d-ario** è un albero in cui....

Scegli un'alternativa:

- ☐ a. tutti i nodi **tranne le foglie** hanno **esattamente grado d**
- ☒ b. i nodi hanno **al più grado d** ✓
- ☐ c. tutti i nodi **tranne la radice** hanno **esattamente grado d**

La risposta corretta è: i nodi hanno **al più grado d**

Domanda 11

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Contesto: Caso peggiore di lettura di n dati da uno stream e inserimento in **ultima posizione** in lista collegata semplicemente, **con controllo che nella lista non ci siano elementi duplicati**.

Si consideri il seguente codice:

```
void read_list(istream& input_stream, basic_list& list)
{
    create_empty(list);
    int d;
    while (ReadData(input_stream, d))
        last_insert_no_duplicates(list, d);
}
```

Qual è la complessità temporale della funzione *read_list* nel **caso peggiore**?

Si supponga che gli elementi letti da *input_stream* siano n e che *list* sia una lista di interi semplice (collegata semplicemente, non circolare, senza sentinella).

create_empty inizializza list come lista vuota e ha complessità costante nel caso migliore e peggiore; *ReadData* legge un elemento alla volta da *input_stream* restituendo false se non ci sono più elementi da leggere e true altrimenti; ha complessità costante nel caso migliore e peggiore.

last_insert_no_duplicates inserisce un elemento **nell'ultima posizione di list** solo se l'elemento **non è già presente in list**; se è presente, non lo re-inserisce.

Scegli un'alternativa:

- ☐ a. Theta (n)
- ☐ b. Theta (2^n)
- ☒ c. Theta (n^2) ✓
- ☐ d. Theta (1)
- ☐ e. Theta ($n!$)
- ☐ f. Theta ($\log n$)

La risposta corretta è: Theta (n^2)

Domanda 12

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Sia $f(n) = 12 + 3^n + 5n^2 + 7n! + 9n \log n + \log n$
Selezionare l'unica risposta corretta.

Scegli un'alternativa:

- ☐ a. $f(n)$ appartiene a $O(1)$
- ☐ b. $f(n)$ appartiene a $\Theta(2^n)$
- ☒ c. $f(n)$ appartiene a $O(n!)$ ✓
- ☐ d. $f(n)$ appartiene a $\Theta(n \log n)$

La risposta corretta è: $f(n)$ appartiene a $O(n!)$

Domanda 13

Risposta errata

Punteggio ottenuto 0,00 su 0,27

Contesto: si consideri **merge**, funzione ausiliaria di **mergeSort**.

Sia A un array di lunghezza **n**; siano **inizio** e **fine** gli indici che **delimitano la porzione di A** (con inizio e fine compresi) sulla quale viene chiamata merge per fondere la **prima metà della porzione di A compresa tra inizio e fine** con la **seconda metà della porzione di A compresa tra inizio e fine**. Qual è la complessità temporale di merge nel caso migliore e peggiore?

Scegli un'alternativa:

- ☐ a. $\Theta(n)$ nel caso peggiore e $\Theta(1)$ nel caso migliore, quello in cui $n = 1$
- ☐ b. $\Theta(\text{fine} - \text{inizio})$ nel caso migliore e peggiore
- ☒ c. $\Theta(n)$ nel caso migliore e peggiore ❌

La risposta corretta è: $\Theta(\text{fine} - \text{inizio})$ nel caso migliore e peggiore

Domanda 14

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

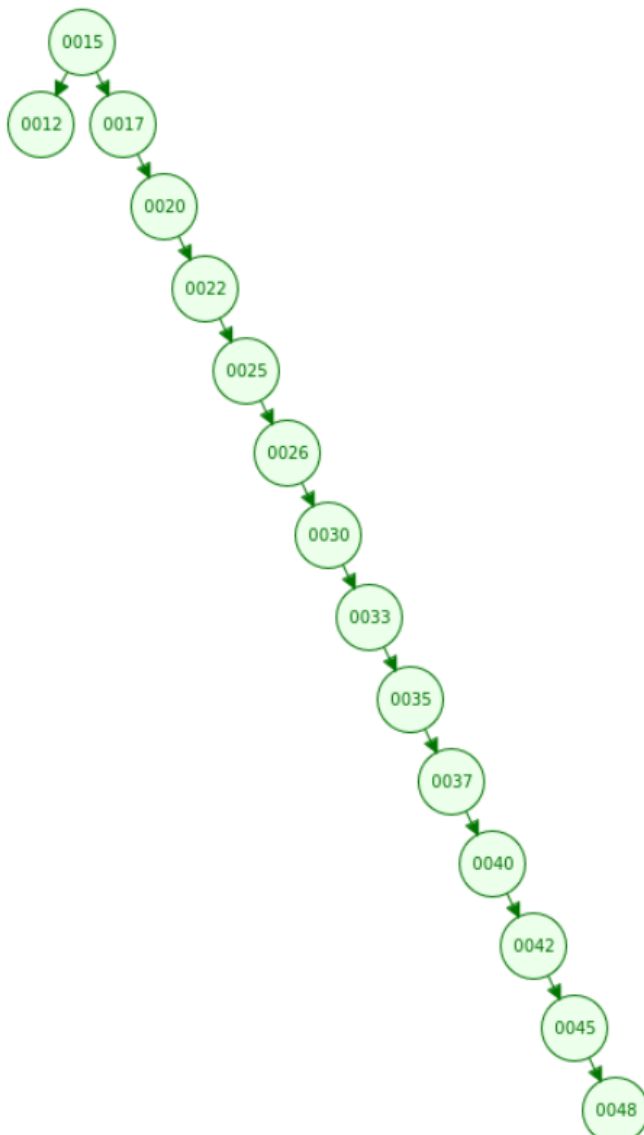
Si consideri un Binary Search Tree inizialmente vuoto, sul quale si eseguono le seguenti operazioni di inserimento (nelle quali come al solito mostriamo solo la chiave), nell'ordine in cui le vedete dall'alto in basso: si inserisce prima l'elemento con chiave 12, poi l'elemento con chiave 15, etc.

```
insert(12, ....);  
insert(15, ....);  
insert(17, ....);  
insert(20, ....);  
insert(22, ....);  
insert(25, ....);  
insert(26, ....);  
insert(30, ....);  
insert(33, ....);  
insert(35, ....);  
insert(37, ....);  
insert(40, ....);  
insert(42, ....);  
insert(45, ....);  
insert(48, ....);
```

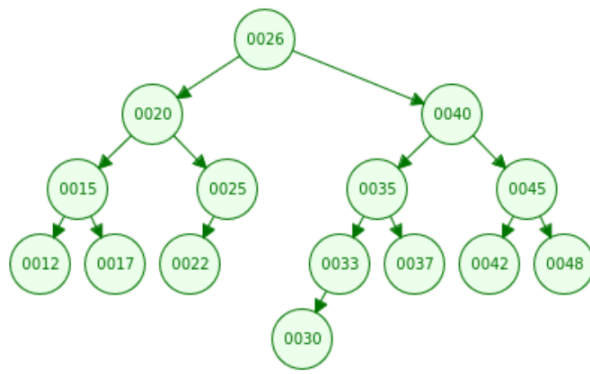
Qual è il BST risultante dagli inserimenti?

Scegli un'alternativa:

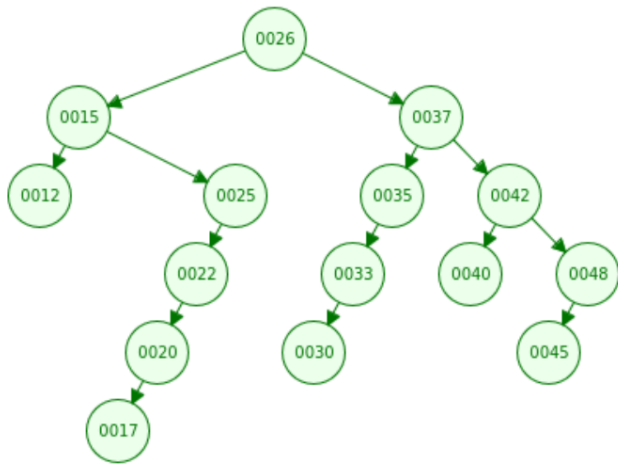
☐ a.



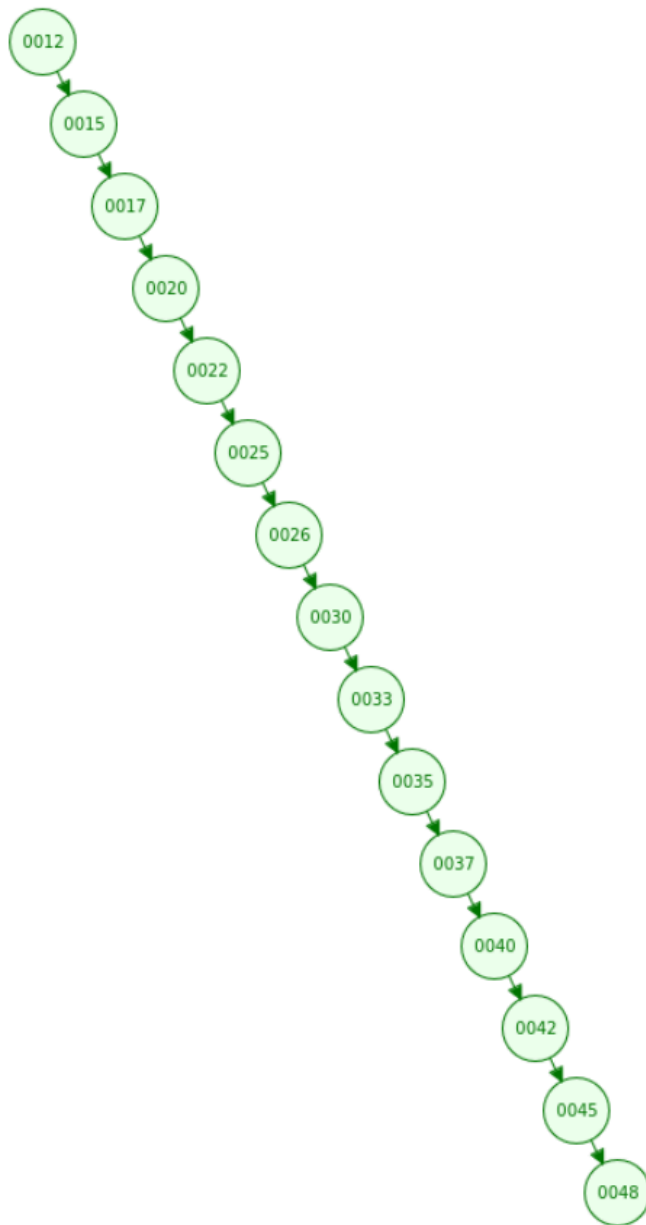
● b.



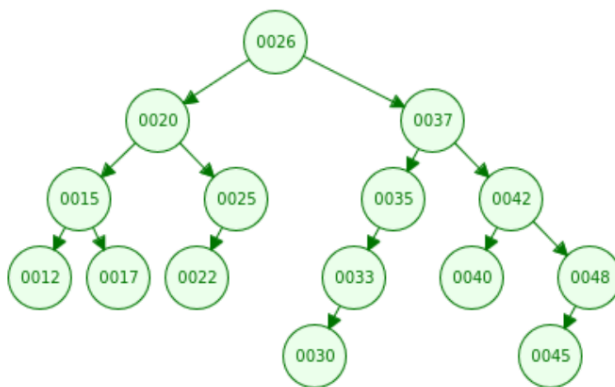
● c.



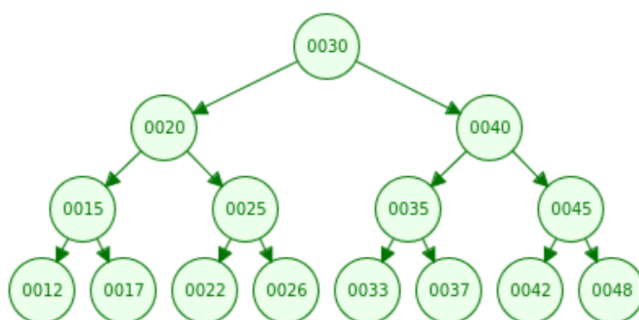
☒ d.

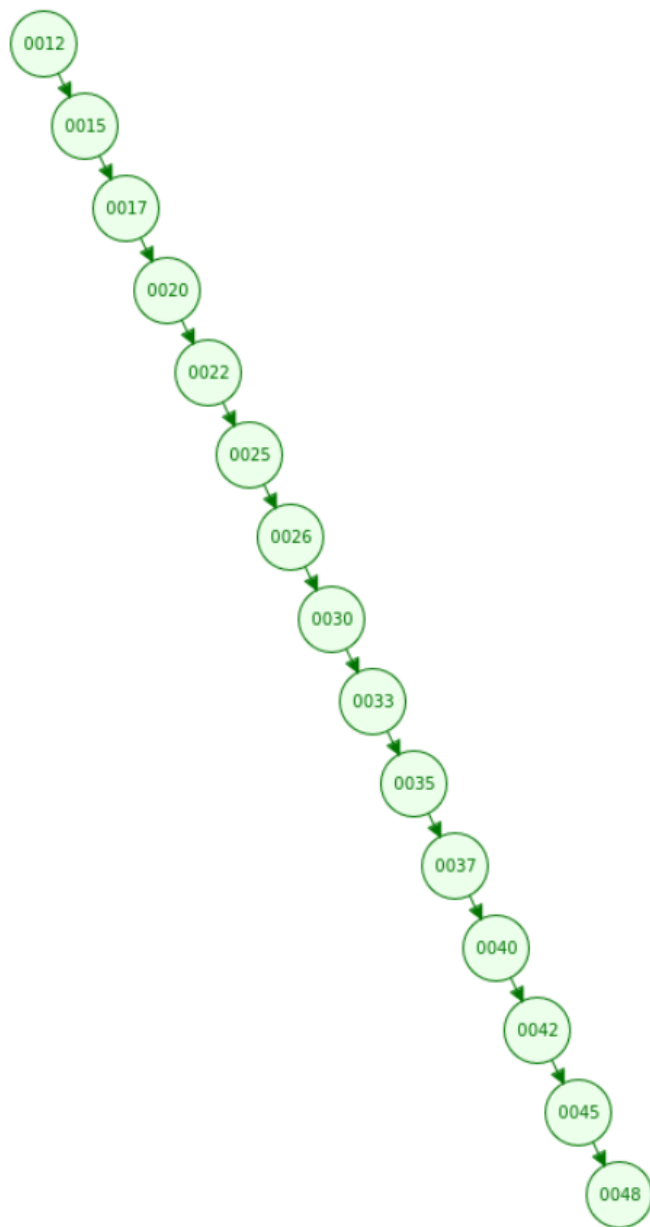


☐ e.



☐ f.





La risposta corretta è:

Domanda 15

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Nella struttura dati "tabella di hash con liste di collisione" una buona funzione di hash dovrebbe essere...

Scegli un'alternativa:

- ☐ a. calcolabile in tempo **costante** e **surgettiva**
- ☐ b. calcolabile in tempo **lineare** nel numero di elementi nella tabella e **iniettiva**
- ☐ c. calcolabile in tempo **lineare** nel numero di elementi nella tabella e **bigettiva**
- ☐ d. calcolabile in tempo **costante** e **bigettiva**
- ☐ e. calcolabile in tempo **costante** e **iniettiva**
- ☒ f. calcolabile in tempo **costante** e in grado di **distribuire in modo uniforme** le chiavi nello spazio degli indici della tabella ✓
- ☐ g. calcolabile in tempo **lineare** nel numero di elementi nella tabella e in grado di **distribuire in modo uniforme** le chiavi nello spazio degli indici della tabella

La risposta corretta è: calcolabile in tempo **costante** e in grado di **distribuire in modo uniforme** le chiavi nello spazio degli indici della tabella

Domanda 16

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Contesto:

Analisi di complessità delle operazioni su liste, Struttura dati: array dinamico con size e maxsize (e ridimensionamento)

Dimensione della lista: n

La posizione di Elem appartiene a N , insieme dei numeri naturali

Nel contesto indicato sopra, qual è la complessità temporale, nel caso migliore e peggiore, di **set: $N \times Elem \times List \rightarrow List$**

Attenzione: N non è la dimensione della lista ma è l'insieme dei numeri naturali, da cui si prende un valore che indica la posizione di Elem

Scegli un'alternativa:

- ☐ a. $\Theta(n)$ nel caso migliore e peggiore
- ☐ b. $\Theta(1)$ nel caso migliore, $\Theta(n)$ nel caso peggiore
- ☒ c. $\Theta(1)$ nel caso migliore e peggiore ✓

La risposta corretta è: $\Theta(1)$ nel caso migliore e peggiore

Domanda 17

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Si consideri il **TDD Dictionary** implementato con un **Binary Search Tree** con **n** nodi.

Qual è la complessità temporale dell'operazione **deleteElem: Key x Dictionary -> Dictionary** nel caso peggiore?

Scegli un'alternativa:

- ☐ a. Theta (1)
- ☒ b. Theta (h), con h altezza del BST ✓
- ☐ c. Theta (log n)

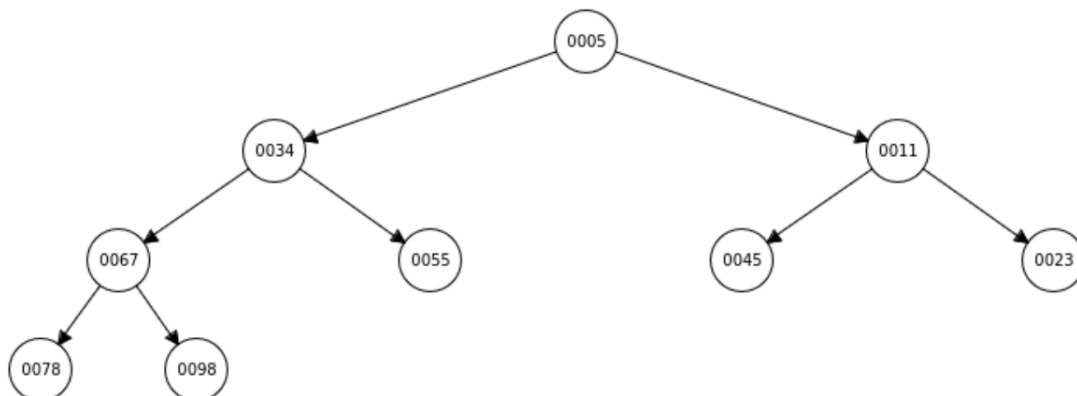
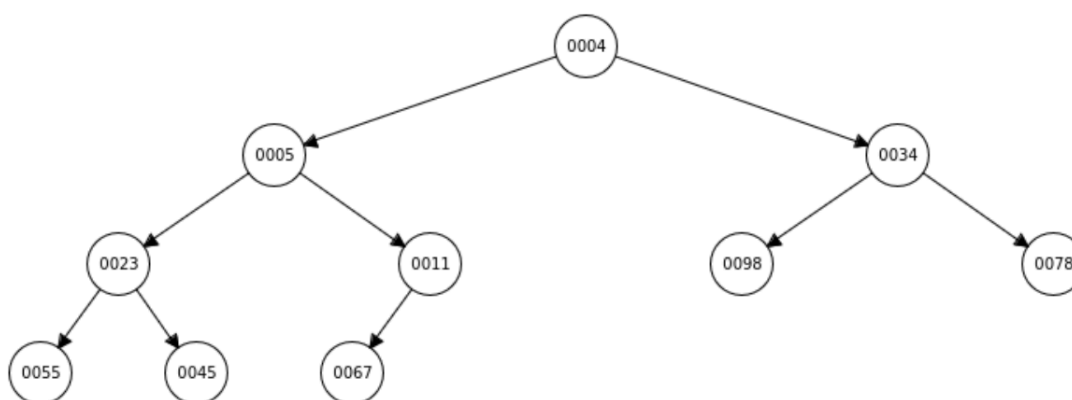
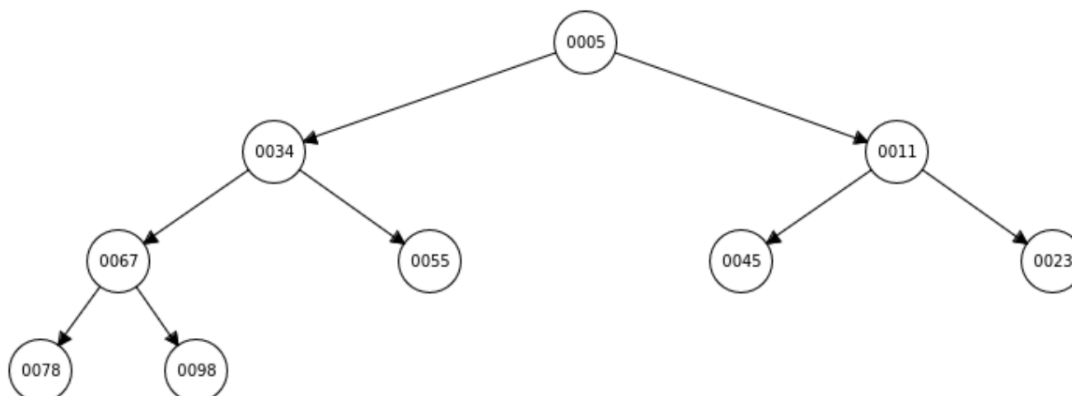
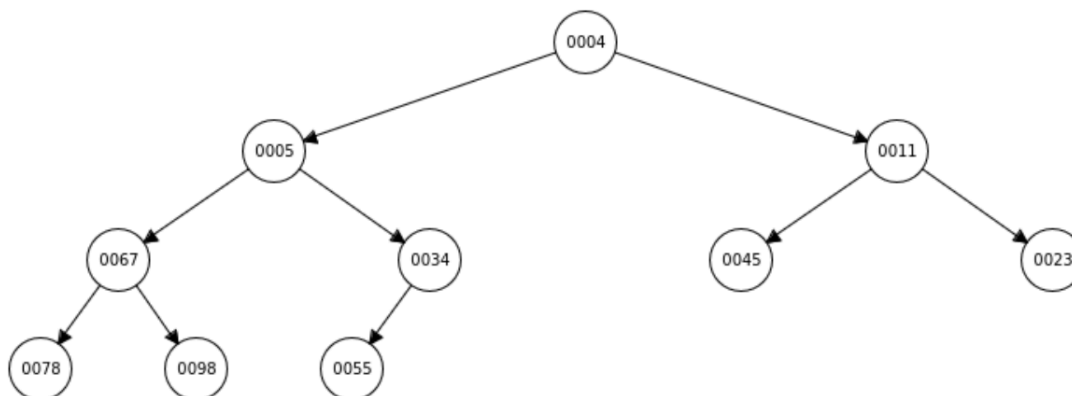
La risposta corretta è: Theta (h), con h altezza del BST

Domanda 18

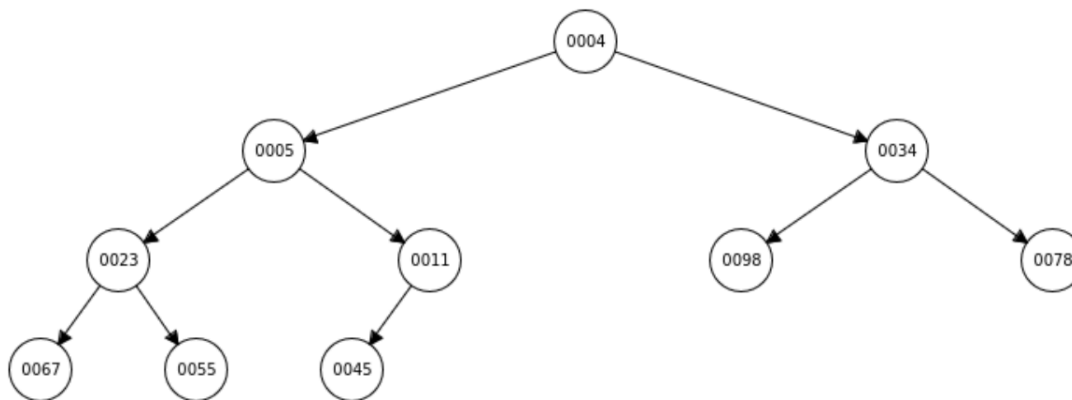
Risposta corretta

Punteggio ottenuto 0,27 su 0,27

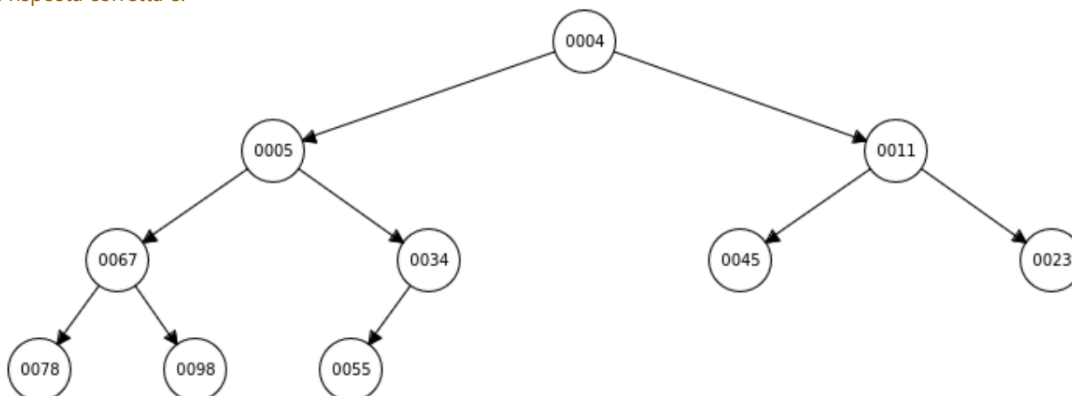
Si consideri questo heap binario di tipo min. Come viene modificato lo heap a seguito dell'inserimento del numero **0004**?

☐ a.☐ b.☒ c.

☐ d.



La risposta corretta è:



Domanda 19

Risposta errata

Punteggio ottenuto 0,00 su 0,27

Si consideri il **TDD Coda con Priorità** implementato con uno **heap binario** con **n** nodi.

Qual è la complessità temporale dell'operazione **deleteMax: PriorityQueue -> PriorityQueue** nel caso peggiore?

Scegli un'alternativa:

- ☒ a. Theta (1) ❌
- ☐ b. Theta (n)
- ☐ c. Theta (log n)

La risposta corretta è: Theta (log n)

Domanda 20

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Una struttura dati indicizzata (cioè un array) supporta l'implementazione efficiente delle operazioni del **TDD Tree** quando

Scegli un'alternativa:

- ☐ a. l'albero da rappresentare è un BST
- ☒ b. l'albero da rappresentare è un albero d-ario quasi completo ✓
- ☐ c. l'albero da rappresentare è generico

La risposta corretta è: l'albero da rappresentare è un albero d-ario quasi completo

Domanda 21

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Si consideri il seguente algoritmo di ordinamento che chiameremo "cosaSort":

```
void cosaSort(vector<int>& v)
{
    int current, prev;
    unsigned int size = v.size();
    for (unsigned int i=1; i<size; ++i)
    { current=i;
      prev=i-1;
      while(prev>=0 && v[current]<v[prev])
      {
          scambia(v, current, prev);
          --current;
          --prev;
      }
    }
}
```

Che algoritmo è, e qual è la sua complessità temporale nel caso migliore e peggiore?

Scegli un'alternativa:

- ☐ a. Si tratta di insertionSort che ha complessità temporale $\Theta(n \log n)$ nel caso migliore e $\Theta(n^2)$ nel caso peggiore
- ☐ b. Si tratta di insertionSort che ha complessità temporale $\Theta(n^2)$ nel caso migliore e peggiore
- ☐ c. Si tratta di selectionSort che ha complessità temporale $\Theta(n \log n)$ nel caso migliore e $\Theta(n^2)$ nel caso peggiore
- ☐ d. Si tratta di selectionSort che ha complessità temporale $\Theta(n)$ nel caso migliore e $\Theta(n^2)$ nel caso peggiore
- ☒ e. Si tratta di insertionSort che ha complessità temporale $\Theta(n)$ nel caso migliore e $\Theta(n^2)$ nel caso peggiore ✓
- ☐ f. Si tratta di selectionSort che ha complessità temporale $\Theta(n^2)$ nel caso migliore e peggiore

La risposta corretta è: Si tratta di insertionSort che ha complessità temporale $\Theta(n)$ nel caso migliore e $\Theta(n^2)$ nel caso peggiore

Domanda 22

Risposta errata

Punteggio ottenuto 0,00 su 0,27

Si consideri la struttura dati "liste di adiacenza con vertici memorizzati in un array" per rappresentare un grafo non orientato con **n** nodi ed **m** archi. La complessità di **incidentEdges** chiamata sul vertice **v** è

Scegli un'alternativa:

- ☐ a. $\Theta(1)$ nel caso migliore e $\Theta(n)$ nel caso peggiore
- ☒ b. $\Theta(1)$ nel caso migliore e $\Theta(\text{grado}(v))$ nel caso peggiore ✗
- ☐ c. $\Theta(\text{grado}(v))$ sia nel caso migliore che nel caso peggiore
- ☐ d. $\Theta(1)$ sia nel caso migliore che nel caso peggiore

La risposta corretta è: $\Theta(\text{grado}(v))$ sia nel caso migliore che nel caso peggiore

Domanda 23

Risposta errata

Punteggio ottenuto 0,00 su 0,27

In quale situazione la complessità temporale di quicksort si calcola seguendo un approccio simile a quello del calcolo della complessità temporale di mergesort?

Scegli un'alternativa:

- ☐ a. Quando come pivot viene sempre selezionato il **mediano** degli elementi su cui quicksort viene chiamata
- ☐ b. Quando come pivot viene sempre selezionato **il minimo o il massimo** degli elementi su cui quicksort viene chiamata
- ☒ c. Quando come pivot viene sempre selezionato il valore che si ottiene come **media aritmetica** degli elementi su cui quicksort viene chiamata ✗

La risposta corretta è: Quando come pivot viene sempre selezionato il **mediano** degli elementi su cui quicksort viene chiamata

Domanda 24

Risposta errata

Punteggio ottenuto 0,00 su 0,27

Formalmente, a cosa corrisponde **l'interfaccia di un tipo di dato**?

Scegli un'alternativa:

- ☐ a. a una struttura dati
- ☐ b. a una segnatura many-sorted, ovvero eterogenea
- ☐ c. a un insieme
- ☒ d. a un'algebra, solitamente eterogenea, su una segnatura ✗
- ☐ e. a un elemento di un insieme

La risposta corretta è: a una segnatura many-sorted, ovvero eterogenea

Domanda 25

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Si consideri la struttura dati "matrice di adiacenza" per rappresentare un grafo non orientato con **n** nodi ed **m** archi. La complessità temporale di **addEdge** dell'arco **(x,y)** è, nel caso peggiore

Scegli un'alternativa:

- ☒ a. $\Theta(1)$ ✓
- ☐ b. $\Theta(n^2)$ perché potresti dover ridimensionare la matrice
- ☐ c. $\Theta(n)$
- ☐ d. $\Theta(\min(\text{grado}(x), \text{grado}(y)))$
- ☐ e. $\Theta(m)$

La risposta corretta è: $\Theta(1)$ **Domanda 26**

Risposta errata

Punteggio ottenuto 0,00 su 0,27

Un grafo **G'** è detto albero di ricoprimento di un grafo **G** se

Scegli un'alternativa:

- ☐ a. **G'** ha gli stessi vertici di **G** ed è un grafo completo
- ☐ b. **G'** è un sottografo di ricoprimento di **G** ed è anche un albero libero
- ☒ c. **G'** ha gli stessi archi di **G**, ha un sottoinsieme proprio dei vertici di **G** ed è privo di cicli ✗

La risposta corretta è: **G'** è un sottografo di ricoprimento di **G** ed è anche un albero libero**Domanda 27**

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Quale delle seguenti affermazioni sui Binary Search Tree (BST) è corretta

Scegli un'alternativa:

- ☐ a. Un BST è un **albero binario quasi completo** con alcune proprietà che le chiavi associate ai nodi devono rispettare
- ☐ b. Un BST è un **albero binario completo** con alcune proprietà che le chiavi associate ai nodi devono rispettare
- ☒ c. Un BST è un **albero binario** con alcune proprietà che le chiavi associate ai nodi devono rispettare ✓

La risposta corretta è: Un BST è un **albero binario** con alcune proprietà che le chiavi associate ai nodi devono rispettare

Domanda 28

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Contesto:

Analisi di complessità delle operazioni su set, struttura dati: array **ORDINATO** con size e maxsize.

Dimensione del primo insieme: **n**

Dimensione del secondo insieme: **m**

Nel contesto indicato sopra, qual è la complessità temporale nel caso peggiore di **setUnion: Set x Set → Set**

Scegli un'alternativa:

- ☐ a. Theta (m)
- ☐ b. Theta (n * m)
- ☒ c. Theta (n + m) ✓
- ☐ d. Theta (n)

La risposta corretta è: Theta (n + m)

Domanda 29

Risposta corretta

Punteggio ottenuto 0,27 su 0,27

Siano **m** = dimensione della tabella di hash ed **n** = numero di elementi presenti nel dizionario; come si definisce il fattore di carico **alpha**?

Scegli un'alternativa:

- ☐ a. $\alpha = v[h(k)]$
- ☒ b. $\alpha = n/m$ ✓
- ☐ c. $\alpha = m/n$

La risposta corretta è: $\alpha = n/m$