03AAX Algoritmi e Strutture Dati

Appello del 11/05/2023 - Prova di Teoria (12 punti)

1. (**2.5 punti**)
Si risolva la seguente equazione alle ricorrenze mediante il metodo dello sviluppo (unfolding):

$$T(n) = 3T(n/4) + 3n \qquad n > 1$$
$$T(1) = 1 \qquad n = 1$$

2. (**2.0 punti**)
Sia data la sequenza di interi, supposta memorizzata in un vettore:

7 2 6 31 3 67 11 18 8 41 0 5 9 10

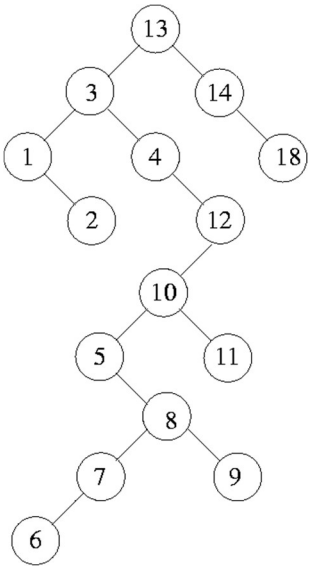
la si trasformi in un heap, ipotizzando di usare un vettore come struttura dati. Si riportino graficamente i passi significativi della costruzione dell'heap ed il risultato finale. Si ipotizzi che, alla fine, nella radice dell'heap sia memorizzato il valore minimo si eseguano su tale heap i primi 2 passi dell'algoritmo di heapsort.
NB: la sequenza è già memorizzata nel vettore e rappresenta una configurazione intermedia per cui la proprietà di heap non è ancora soddisfatta.

3. **(2.5 punti**)
Data la catena di matrici (A_1, A_2, A_3, A_4) di dimensioni (3x4), (4x2), (2x5) e (5x3) rispettivamente, si determini mediante un algoritmo di programmazione dinamica la parentesizzazione ottima del prodotto di matrici che minimizza il numero di moltiplicazioni.

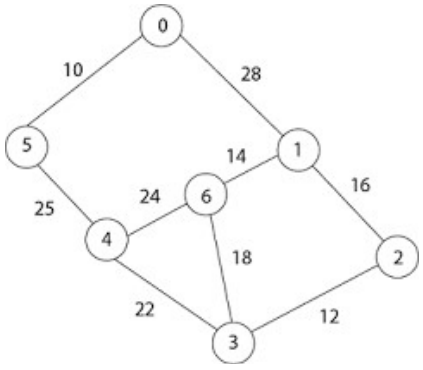
4. **(1 punto)**
Si converta la seguente espressione da forma prefissa a forma postfissa:

/ * - A B / C D / E * - F G + H I

5. **(2.0 punti)**
Si partizioni il seguente BST attorno alla sesta chiave più piccola. In ogni nodo compare la chiave intera, altre eventuali informazioni non sono riportate.



6. **(2.0 punti**)
Dato il seguente grafo non orientato, connesso e pesato, se ne determini un minimum spanning tree applicando l'algoritmo di Prim a partire dal vertice **0,** disegnando l'albero e ritornando come risultato il valore del peso minimo. Si esplicitino i passi intermedi.



03AAX Algoritmi e Strutture Dati

Appello del 11/05/2023 - Prova di programmazione (18 punti)

1. (**18 punti**)

Dato un grafo non orientato non pesato $G = (V, E)$ si definisce *triangle packing* una collezione $V_1, V_2, \dots, V_k$ di sottoinsiemi disgiunti di vertici, ognuno contenente esattamente tre vertici, tali per cui per ogni $\mathrm{V_i} = \{\mathrm{u_i}, \mathrm{v_i}, \mathrm{w_i}\}$ con $1 \leq i \leq k$ tutti e tre gli archi (u_i, v_i), (u_i, w_i) e (v_i, w_i) esistono e appartengono a E. Un triangle packing può non coprire tutti i vertici del grafo. Per gli scopi del problema si è interessati all'individuazione di un triangle packing a cardinalità massima.

Strutture dati e acquisizione

Fornire la definizione e implementazione delle strutture dati reputate necessarie a modellare le informazioni del problema, quali il grafo e un triangle packing, e le funzioni di acquisizione dei dati stessi. In caso di organizzazione delle strutture dati su più file, indicare esplicitamente il modulo di riferimento.
Si assuma che il grafo in input sia riportato in un file di nome `grafo.txt`, organizzato come segue:

- Sulla prima riga appare il numero V di vertici
- Seguno un numero indefinito di righe riportanti coppie (u, v) di interi, con $0 \leq u, v < V$ a rappresentare gli archi del grafo

Problema di verifica

Data una soluzione proposta, verificare che questa rappresenti effettivamente un triangle packing, tenendo in considerazione la definizione teorica proposta in precedenza. Si trascuri la richiesta di cardinalità massima in questo contesto. La soluzione proposta deve essere letta da file il cui nome e formato è a discrezione del candidato, che è tenuto a fornire anche una breve spiegazione dei contenuti del file stesso.

Problema di ricerca e ottimizzazione

Identificare, se possibile, un triangle packing a cardinalità massima per il grafo dato in input.

03AAX Algoritmi e Programmazione

Appello del 11/05/2023 - Prova di programmazione (12 punti)

1. (**2 punti**)

Sia data una matrice `M` di dimensione `r x c` contenente elementi interi. Scrivere una funzione che generi una matrice `M'` di dimensione `r x c` derivata da `M` in cui ogni elemento `[i][j]` assume il valore della somma cumulata di tutti gli elementi di indice inferiore, lungo la riga `i` e la colonna `j`, incluso l'elemento `[i][j]` originale il cui contributo è contato una singola volta. La matrice `M'` sia allocata dentro alla funzione. Completare opportunamente il prototipo in modo che la nuova matrice sia disponibile al chiamante.

```
void f(int **M, int r, int c, ...);
```

Esempio:

$$M = \begin{Bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{Bmatrix} \Rightarrow M' = \begin{Bmatrix} 1 & 3 & 6 \\ 5 & 11 & 18 \\ 12 & 22 & 33 \end{Bmatrix}$$

2. (**4 punti**)

Si fornisca la definizione delle strutture dati `LIST` e `NODE`, come ADT di I categoria e quasi ADT rispettivamente, per rappresentare una lista singolo linkata di interi, senza sentinelle. Suddividere il codice in modo opportuno tra file `.h` e `.c`

Si scriva una funzione `void f(LIST l)` che ricevuta in input una lista di interi (rappresentata facendo riferimento ai tipi definiti in precedenza), già ordinata in ordine crescente, compatti i contenuti della lista cancellando tutti i nodi uguali consecutivi, ad eccezione del primo di ogni gruppo.

Non è ammesso l'utilizzo di funzioni di libreria.

Esempio:

$$1 \rightarrow 1 \rightarrow 2 \rightarrow 2 \rightarrow 2 \rightarrow 5 \rightarrow 7 \rightarrow 7$$

diventa

$$1 \rightarrow 2 \rightarrow 5 \rightarrow 7$$

3. **(6 punti**)

Una matrice binaria `M` di dimensione `O x S` rappresenta l'appartenenza di una serie di oggetti a una serie di insiemi. Un oggetto `o_i` appartiene all'insieme `s_j` se nella matrice appare un 1 nella cella `[i][j]`. L'unico altro valore ammesso nella matrice è il valore 0, per indicare non appartenenza. Ogni colonna della matrice rappresenta quindi un sottoinsieme dell'insieme universo di oggetti. Scrivere una funzione ricorsiva in grado di identificare, se possibile, l'insieme a cardinalità minima di sottoinsiemi disgiunti in grado di coprire l'insieme universo. Si giustifichi la scelta del modello combinatorio adottato. Si descrivano i criteri di pruning adottati o il motivo della loro assenza.

PER ENTRAMBE LE PROVE DI PROGRAMMAZIONE (18 o 12 punti):

- ***indicare nell'elaborato e nella relazione nome, cognome e numero di matricola.***
- ***se non indicato diversamente, è consentito utilizzare chiamate a funzioni <u>standard</u>, quali ordinamento per vettori, funzioni su FIFO, LIFO, liste, BST, tabelle di hash, grafi e altre strutture dati, considerate come librerie esterne.***
- ***gli header file devono essere allegati all'elaborato (il loro contenuto riportato nell'elaborato stesso). Le funzioni richiamate, inoltre, dovranno essere incluse nella versione del programma allegata alla relazione. I modelli delle funzioni ricorsive non sono considerati funzioni standard.***
- ***consegna delle relazioni (per entrambe le tipologie di prova di programmazione): entro il 14/05/2023, alle ore 23:59, mediante caricamento su Portale. Le istruzioni per il caricamento sono pubblicate sul Portale nella sezione Materiale). QUALORA IL CODICE CARICATO CON LA RELAZIONE NON COMPILI CORRETTAMENTE, VERRÀ APPLICATA UNA PENALIZZAZIONE. Si ricorda che la valutazione del compito viene fatta, senza la presenza del candidato, sulla base dell'elaborato svolto in aula. Non verranno corretti i compiti di cui non sarà stata inviata la relazione nei tempi stabiliti.***