3MNO Algoritmi e Programmazione

Appello del 22/02/2017 - Prova di teoria (12 punti)

1. (2.5 punti)

Sia data la generica equazione alle ricorrenze

$$T(n) = aT(n/b) + cn \qquad n>1$$
$$T(1) = 1 \qquad n=1$$

dove a, b e c sono coefficienti interi, mediante il metodo dello sviluppo (unfolding), se ne determinino le 3 soluzioni per i casi a=b, a<b e a>b.

2. (2 punti)

Sia data la sequenza di interi, supposta memorizzata in un vettore:

10 1 27 14 17 0 8 55 19 41 23 91 31 37 7 3 13

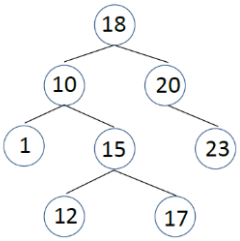
Si eseguano i primi 2 passi dell'algoritmo di quicksort per ottenere un ordinamento ascendente**.** NB: I passi sono da intendersi, impropriamente, come in ampiezza sull'albero della ricorsione, non in profondità. Si chiede, pertanto, che siano ritornate le 2 partizioni del vettore originale e le due partizioni delle partizioni trovate al punto precedente.

3. (1 punto)

Si converta la seguente espressione da forma infissa a forma postfissa: A * ((B * C) + (D * (E + F)))

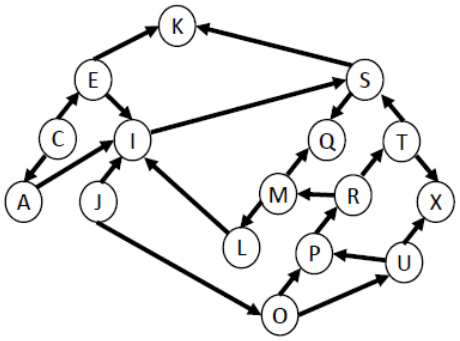
4. (2 punti)

Si inseriscano in radice nel BST di figura in sequenza le chiavi: 16, 19 e 22 e poi si cancelli la chiave 19. Si disegni l'albero ai passi significativi.



5. (2.5 punti)

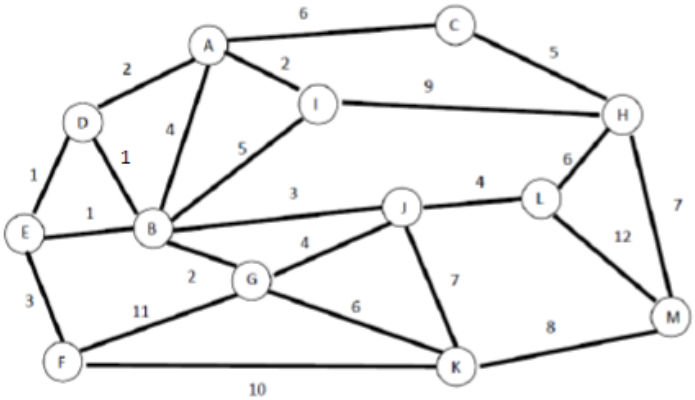
Dato il seguente grafo orientato:



se ne determinino mediante l'algoritmo di Kosaraju le componenti fortemente connesse. Si consideri **M** come vertice di partenza e, qualora necessario, si trattino i vertici secondo l'ordine alfabetico. Si riportino i passaggi rilevanti:

6. (2 punti)

Si determini mediante l'algoritmo di Kruskal l'albero ricoprente minimo per il grafo non orientato, pesato e connesso in figura illustrando i passaggi intermedi del procedimento adottato.



03MNO Algoritmi e Programmazione

Appello del 22/02/2017 - Prova di programmazione (18 punti)

Un supermercato offre prodotti acquistabili singolarmente a prezzo pieno e offerte speciali a prezzo ridotto. In ogni offerta sono raggruppati almeno due prodotti. Tutti i prodotti di un'offerta sono venduti congiuntamente. Il cliente può sfruttare ogni offerta al massimo una volta.

I prodotti e le offerte del supermercato sono riportati in un primo file (`catalogo.txt`), organizzato come segue:

- sulla prima riga sono presenti due interi `N e O`, rappresentanti il numero di prodotti disponibili e il numero di offerte
- sulle `N` righe successive sono riportati i prodotti, ognuno mediante un identificativo alfanumerico e il prezzo unitario (intero)
- seguono `O` blocchi di righe, in cui sono riportati i dettagli di ogni offerta, una per blocco, organizzata come segue:
  - una terna con l'identificativo alfanumerico per l'offerta, il prezzo (intero) e il numero `X` di oggetti distinti che l'offerta comprende
  - seguono `X` coppie, una per riga, a rappresentare l'identificativo di ogni prodotto incluso nell'offerta e la sua quantità

La lista della spesa è contenuta in un secondo file (`spesa.txt`), organizzato come segue:

- sulla prima riga presente un intero `M`, rappresentante il numero di prodotti distinti da acquistare
- sulle `M` righe successive sono riportati l'identificativo di ogni prodotto voluto e la quantità da acquistare (al massimo 9 unità per pezzo).

Esempio di `catalogo.txt` (indentato per leggibilità):

```
6 4
      Latte 10
      Pane 6
      Biscotti 16
      Nutella 12
      Zucchero 3
      Caffe 3
OFF1 19 2
      Latte 2
      Caffe 1
OFF2 28 2
      Nutella 2
      Pane 1
OFF3 10 2
      Pane 2
      Zucchero 1
OFF4 18 2
      Latte 2
      Zucchero 1
```

Esempio di `spesa.txt`:

```
3
Latte 4
Nutella 1
Pane 3
```

Si scriva un programma in C che, a partire dai due file sopra descritti:

- generi una struttura dati, in cui acquisire i dati del primo file, realizzata mediante due vettori dinamici, uno per i prodotti e uno per le offerte. Ogni offerta contiene una collezione di ***riferimenti*** (puntatori o indici) ai prodotti compresi: la si realizzi a scelta come vettore o lista. Occorre inoltre realizzare, per ogni prodotto, la lista delle offerte che includono tale prodotto (si consiglia una lista di ***riferimenti*** mediante indici, nel caso di vettore di offerte, o puntatori a offerte)
- fornisca una funzione che, dato l'identificativo di un prodotto, elenchi tutte le offerte in cui il prodotto compare
- fornisca una funzione che, dati gli identificativi di due offerte, elenchi gli eventuali prodotti in comune
- una volta letto il secondo file, determini l'insieme ottimo di acquisti di prodotti singoli e/o di offerte, che permetta di soddisfare la lista della spesa. Si noti che è possibile acquistare più oggetti del necessario, purché il costo sia complessivamente minimo. Si ricordi che ogni offerta può essere sfruttata al massimo una volta.

Con i dati degli esempi precedenti, la soluzione ottima include OFF1, OFF3 e OFF4 + 1 Pane e 1 Nutella presi singolarmente, per un totale di 65.

03MNO Algoritmi e Programmazione

Appello del 22/02/2017 - Prova di programmazione (12 punti)

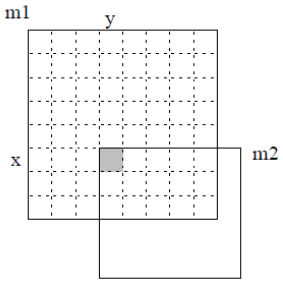
1. (**2 punti**)

Si scriva una funzione caratterizzata dal seguente prototipo

```
void f(int **m1, int r1, int c1, int **m2, int r2, int c2, int x, int y);
```

che riceve due matrici `m1` e `m2` di dimensioni `r1` x `c1` e `r2` x `c2`, rispettivamente, e una coppia di coordinate `x` e `y`.

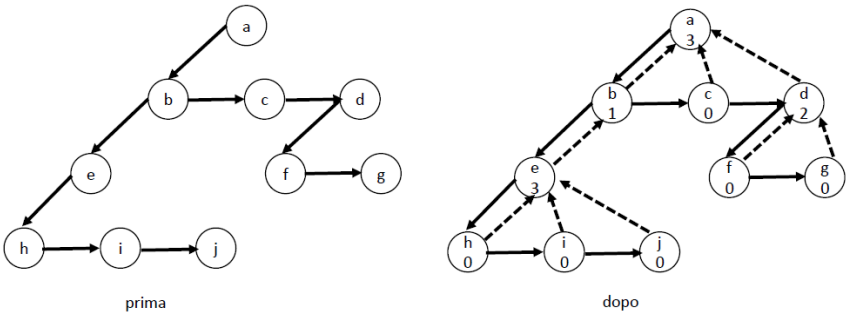
La funzione deve sovrascrivere la porzione di matrice `m1`, a partire dall'angolo in alto a sinistra identificato da `x` e `y`, con i contenuti di `m2`. Poiché non c'è garanzia circa le relative dimensioni delle matrici, la funzione deve effettuare tutti i controlli del caso per non sforare le dimensioni di `m1`, come illustrato nella seguente figura:



2. (**4 punti**)

Dato un albero in formato left-child right-sibling:

- si definisca con una struct il nodo, in cui, oltre alle altre informazioni legate alla rappresentazione left-child right-sibling, vi siano un campo intero per memorizzare il numero di figli e un campo di puntatore al padr
- si scriva una funzione ricorsiva `void processTree(link root);`
- in grado di memorizzare in ogni nodo dell'albero il numero dei suoi figli e il puntatore al padre:



3. (**6 punti**)

Si consideri un insieme di `n` elementi distinti identificati univocamente con un intero da `0` a `n-1`. Per ogni coppia di elementi è noto se questi possono essere separati o no. Tale informazione è riportata in una matrice quadrata di dimensione `n` x `n`. Ogni cella `m[i][j]` della matrice riporta il valore `1` (`0`) se due elementi `i` e `j` sono (non sono) separabili.

Si scriva una funzione dal prototipo:

```
int *f(int n, int k, int **m)
```

il cui obiettivo sia di trovare una partizione dell'insieme di partenza in due sottoinsiemi, il secondo dei quali contenga almeno `k` oggetti, tali per cui nessuna coppia di elementi indivisibili venga separata.

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

- ***indicare nell'elaborato e nella relazione (oltre a nome, cognome e numero di matricola) anche il nome del corso per cui si sta sostenendo l'esame (AP, APA I+II).***
- ***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 sabato 25/02/2017, 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 discussione o esame orale, <u>sulla base dell'elaborato svolto in aula</u>. Non verranno corretti i compiti di cui non sarà stata inviata la relazione nei tempi stabiliti.***