Analisi e progettazione di algoritmi

(III anno Laurea Triennale - a.a. 2018/19)

Prova scritta 27 giugno 2019

NB: I punteggi sono indicativi.

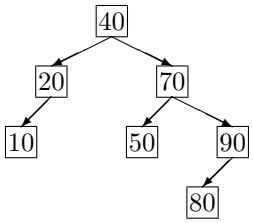
Esercizio 1 - Ordinamenti (punti 5) Dato lo schema di quicksort per liste collegate (linked list):

- Sia l la lista da ordinare.
- Se $|l| > 1$:
  1. Prendi il primo elemento p di l, eliminandolo da l;
  2. Forma due liste l_1 ed l_2, dove l_1 contiene gli elementi di l con chiave $\leq p$ e l_2 quelli con chiave $> p$;
  3. Ordina ricorsivamente l_1;
  4. Ordina ricorsivamente l_2;
  5. Poni $l = l_1 \cdot p \cdot l_2$, dove il simbolo $\cdot$ indica la concatenazione.

Assumere che le liste collegate siano implementate come *code*, e quindi supportino in tempo costante le operazioni:
– *make_empty* che crea una nuova coda vuota e la ritorna;
– *enqueue* che aggiunge un elemento in fondo;
– *dequeue* che rimuove l'elemento in cima e lo ritorna;
– *size* che ritorna il numero di elementi.

1. Scrivere in pseudocodice i passi 1,2 dello schema. Nota bene: assumere le primitive sulle code come date, non bisogna darne lo pseudocodice.
2. Valutare la complessità temporale e spaziale del frammento di codice scritto.
3. Fornire un esempio di input con 4 elementi in cui si veda che l'algoritmo di quicksort così ottenuto non è stabile. Per indicare due occorrenze diverse della chiave x usare la notazione per indicare x_A, x_B. Far vedere come l'algoritmo modifica tale input.

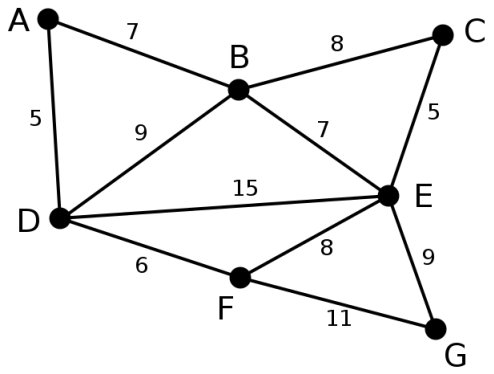
Esercizio 2 - Strutture dati (punti 6) Dato il seguente albero AVL:



1. indicare i fattori di bilanciamento dei nodi.
2. dare una sequenza di inserimento che potrebbe avere prodotto tale albero; la sequenza è unica?
3. sull'albero dato inserire in successione queste tre chiavi, facendo vedere che cosa succede e spiegando perché: 30, 75, 100.

Esercizio 3 – Grafi (Punti 7)

Si consideri il seguente grafo pesato.



Applicando l'algoritmo di Dijkstra, si determinino i pesi dei cammini minimi che collegano il nodo A con tutti gli altri nodi. Più precisamente, si completi la seguente tabella:

	A	B	C	D	E	F	G
0	0	∞	∞	∞	∞	∞	∞
1							
2							
...							

dove ogni riga corrisponde a un'iterazione, e ogni casella contiene: per i nodi per i quali è già stata trovata la distanza definitiva, un simbolo speciale (per esempio -), per gli altri la distanza provvisoria corrente. Si considerino gli adiacenti a un nodo in ordine alfabetico.

Esercizio 4 – Tecniche algoritmiche (Punti 8) Siano $X[1..n]$ e $Y[1..n]$ due sequenze. Indichiamo con $\exists CS[i,j,k]$, con $0 \leq i,j,k \leq n$, il problema di decidere se esiste una sottosequenza comune di $X[1..i]$ e $Y[1..j]$ di lunghezza (almeno) k.

1. Si definisca induttivamente $\exists CS[i,j,k]$ giustificando la correttezza della definizione.
2. Si descriva un corrispondente algoritmo di programmazione dinamica ($\exists CS$ sarà quindi una matrice a valori booleani), indicandone la complessità.
3. Si descriva come ottenere anche una delle sottosequenze di di $X[1..i]$ e $Y[1..j]$ di lunghezza (almeno) k, se ne esistono.

Esercizio 5 - Analisi di complessità (Punti 7) Per la soluzione di un problema $\mathcal{P}$, abbiamo a disposizione un algoritmo iterativo di complessità $O(n^2)$. Inoltre abbiamo un algoritmo ricorsivo con la seguente relazione di ricorrenza:

$$T(n) = a\,T(n/4) + n^2$$

1. Cosa possiamo dire sulla delimitazione superiore e inferiore del problema $\mathcal{P}$?
2. Per quali valori di a è preferibile (asintoticamente) l'algoritmo iterativo?