Cognome: **Nome**: **# Matricola**: **Riga**: **Col**:

Algoritmi e Strutture Dati - Prova d'esame
06/06/11

Esercizio 0 Scrivere correttamente nome, cognome, numero di matricola, riga e colonna.

Esercizio 1 - Punti ≥ 6

Si consideri la seguente equazione di ricorrenza:

$$T(n) = \begin{cases} 11/5n + T(\lfloor n/5 \rfloor) + T(\lfloor 7n/10 \rfloor) & n > 1 \\ 1 & n \leq 1 \end{cases}$$

Individuare limiti inferiori e superiori tramite il metodo di sostituzione.

Esercizio 2 - Punti ≥ 8

Descrivere un algoritmo nearest(**integer**[ ] V, **integer** n, **integer** k) che, dato un vettore V di n interi distinti e un intero positivo $k \leq n$, stampi k numeri di V che sono più vicini alla mediana dei valori in V, dove la misura di distanza dalla mediana è data dalla distanza in valore assoluto da essa. Discutere complessità e correttezza.
Ad esempio, sia $n = 11$, $V = \{1, 6, 4, 3, 9, 12, 15, 2, 8, 10, 11\}$ e $k = 2$; la mediana di V è 8 e i due valori più vicini sono $6, 9$ (che hanno distanza $2, 1$) *oppure* $9, 10$ (che hanno distanza $1, 2$). In altre parole, se il k-esimo e il $k + 1$-esimo numero sono equidistanti dalla mediana, se ne sceglie uno. Notate che la mediana stessa non viene contata fra i valori da restituire.
Nota: è facile descrivere un algoritmo $O(n \log n)$; esistono tuttavia algoritmi $O(n)$. La valutazione dipenderà dall'efficienza dell'algoritmo trovato.

Esercizio 3 - Punti ≥ 10

Dato un grafo G e due sottoinsiemi V_1 e V_2 dei suoi vertici si definisce distanza tra V_1 e V_2 la distanza minima per andare da un nodo in V_1 ad un nodo in V_2. Nel caso V_1 e V_2 non siano disgiunti allora il valore è 0. Descrivere un algoritmo mindist(GRAPH G, SET V_1, SET V_2) che restituisce la distanza minima (in numero di archi). Discutere complessità e correttezza, assumendo che l'implementazione degli insiemi sia tale che il costo di verificare l'appartenenza di un elemento all'insieme abbia costo $O(1)$.
Nota: è facile descrivere un algoritmo $O(nm)$; esistono tuttavia algoritmi di complessità $O(n^2)$ (con matrice di adiacenza) e $O(m + n)$. La valutazione dipenderà dall'efficienza dell'algoritmo trovato.

Esercizio 4 - Punti ≥ 12

Si consideri un insieme n di scatole tridimensionali, dove ogni scatola S_i è definita dalle dimensioni $x_i \times y_i \times z_i$. Una scatola S_i è *minore* di una scatola S_j ($S_i \subset S_j$) se, per ogni asse, S_i ha dimensione minore di S_j: $x_i < x_j$, $y_i < y_j$, $z_i < z_j$. Le scatole non possono essere ruotate. Le scatole non sono ordinate per alcun asse, ma vanno considerate nell'ordine dato: una scatola S_i può essere inserita in una scatola S_j solo se $S_i \subset S_j$ e $i < j$. Vogliamo trovare un sottoinsieme massimale $S_{i_1}, S_{i_2}, \ldots, S_{i_k}$ di scatole inseribili l'una nell'altra, tali cioè che $i_1 < i_2 < \cdots < i_k$ e $S_{i_1} \subset S_{i_2} \subset \cdots \subset S_{i_k}$.

1. Si scriva un algoritmo che determini la dimensione massima di questo sottoinsieme
2. Se ne discuta la correttezza e complessità
3. Si scriva un algoritmo aggiuntivo che stampa un insieme massimale di scatole.