

**Algoritmi e Strutture Dati****18/06/12**

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

**Esercizio 1 – Punti  $\geq 6$** 

Scrivere un algoritmo che preso in input un albero binario  $T$  i cui nodi sono associati ad un valore intero  $T.key$ , restituisca il numero di nodi dell'albero il cui valore è pari al livello del nodo. Vi ricordo che il livello del nodo è pari al numero di archi che devono essere attraversati per raggiungere il nodo dalla radice. Per cui la radice ha livello 0, i suoi figli hanno livello 1, etc.

**Esercizio 2 – Punti  $\geq 8$** 

Si consideri una versione di MergeSort in cui il vettore viene suddiviso in  $\sqrt{n}$  sottovettori di  $\sqrt{n}$  elementi, ad ognuno dei quali viene applicato MergeSort in modo ricorsivo, come mostrato nello pseudocodice seguente.

---

**MergeSort(integer[]  $A$ , integer  $i$ , integer  $j$ )**


---

**if**  $j - i + 1 < 4$  **then**

    InsertionSort( $A, i, j$ )

    **return**
**integer**  $step \leftarrow \lfloor \sqrt{j - i + 1} \rfloor$ 
**integer**  $start \leftarrow i$ 
**while**  $start \leq j$  **do**

    MergeSort( $A, start, \min(start + step - 1, j)$ )

     $start \leftarrow start + step$ 

Merge( $A, i, j, step$ )

---

La procedura Merge(), non mostrata qui per brevità, effettua un'operazione di unione fra i  $\sqrt{n}$  sottovettori, selezionando ad ogni passo il valore minore e avanzando l'indice sul corrispondente sottovettore.

1. Discutete della complessità della Merge().
2. Scrivete l'equazione di ricorrenza associata a questa versione di MergeSort.
3. Risolvete l'equazione di ricorrenza ottenuta al punto precedente.

**Esercizio 3 – Punti  $\geq 6$** 

Sia  $G = (V, E)$  un grafo non orientato,  $q \in V$  un nodo di  $G$  e  $w : E \rightarrow \mathbb{R}^+$  una funzione peso a valori positivi. Un  $q$ -cammino in  $G$  da  $u$  a  $v$  è un cammino in  $G$  da  $u$  a  $v$  passante per  $q$ . Un  $q$ -cammino in  $G$  da  $u$  a  $v$  si dice minimo se il suo peso è minimo rispetto a tutti i  $q$ -cammini in  $G$  da  $u$  a  $v$ . Si descriva un algoritmo efficiente che calcoli per ciascuna coppia di nodi  $(u, v) \in V \times V$  il costo del  $q$ -cammino minimo da  $u$  a  $v$ ; se ne valuti la complessità computazionale.

**Esercizio 4 – Punti  $\geq 10$** 

Considerate una scacchiera composta da **tre** colonne e  $n$  righe. Ad ogni casella è associato un intero positivo, memorizzato nella matrice  $A[1 \dots n, 1 \dots 3]$ . Su ogni casella è possibile piazzare una pedina; un piazzamento di pedine è regolare se non esistono due pedine adiacenti in orizzontale o verticale. Il valore di un piazzamento regolare è dato dalla somma degli interi associati alle caselle in cui ci sono pedine. Progettare un algoritmo che data la matrice  $A$ , determina il valore del piazzamento regolare di valore massimo. Utilizzando la programmazione dinamica, è possibile risolvere questo problema in tempo  $O(n)$ .