

## Algoritmi e Strutture Dati - 14/07/15

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

### Esercizio 1 – Punti $\geq 6$

Trovare il limite **inferiore** per la seguente equazione di ricorrenza, utilizzando il metodo di sostituzione (detto anche per tentativi)

$$T(n) = \begin{cases} 2T(\lfloor n/8 \rfloor) + \sqrt[3]{n} & n > 1 \\ 1 & n \leq 1 \end{cases}$$

### Esercizio 2 – Punti $\geq 6$

Si supponga che Alice, Bob e Carl siano posizionati su tre nodi in un grafo *orientato*  $G = (V, E)$ , associato ad una funzione di peso  $w$  con pesi interi non negativi. Alice e Bob vogliono incontrarsi in un altro nodo del grafo, senza che Carl lo venga a sapere.

Scrivere un algoritmo efficiente che sia in grado di individuare il nodo di incontro ottimale tra Alice e Bob, in modo tale che la somma dei cammini percorsi da entrambi (dalla loro posizione iniziale al nodo di incontro) sia minimizzata. L'algoritmo deve restituire tale somma minima. I due percorsi non devono passare per il nodo di Carl. Il punto di incontro non deve essere il nodo in cui risiede Carl. Discutere correttezza e complessità computazionale.

### Esercizio 3 – Punti $\geq 9$

In un grafo non orientato  $G = (V, E)$  un sottoinsieme  $A \subseteq E$  degli archi è detto univoco se non contiene due o più archi uscenti dallo stesso nodo.

- Dimostrare che l'algoritmo greedy che seleziona gli archi in ordine decrescente di peso (e quindi inserisce sicuramente l'arco di peso maggiore) non funziona correttamente.
- Scrivere un algoritmo che preso in input un grafo  $G$  non orientato con pesi positivi sugli archi, trovi un insieme univoco  $A$  di archi di  $G$  la cui somma dei pesi sia massima. Discutere la correttezza dell'algoritmo proposto e valutarne la complessità computazionale.

### Esercizio 4 – Punti $\geq 12$

Si supponga di dover organizzare una partita di calcetto e di dover dividere i  $2n$  partecipanti in due squadre composte da  $n$  giocatori che siano il più possibile bilanciate. Per bilanciare le squadre, si consideri un vettore  $b[1 \dots 2n]$  che associa al giocatore  $i$ -esimo la sua bravura  $b[i]$ . Scrivere un algoritmo che restituisca **true** se è possibile suddividere i giocatori in due squadre di  $n$  giocatori la cui somma delle bravura sia uguale. Discutere la complessità computazionale.

Esempio:  $b[1] = 3$ ,  $b[2] = 5$ ,  $b[3] = 4$ ,  $b[4] = 2$ , la risposta è **true** perché è possibile fare due squadre  $\{1, 3\}$  e  $\{2, 4\}$  che hanno la stessa bravura  $3 + 4 = 5 + 2 = 7$ .