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

Algoritmi e Strutture Dati - 31/05/13

Esercizio 1 – Punti ≥ 6 (Parte B)

Si consideri un grafo non orientato connesso $G = (V, E)$ i cui archi hanno tutti lo stesso peso $w > 1$, tranne uno che ha peso $w - 1$. Scrivere un algoritmo efficiente per calcolare un minimum spanning tree di G.

Discutere informalmente la correttezza della soluzione proposta e calcolare la complessità computazionale.

Esercizio 2 – Punti ≥ 6 (Parte A)

Dati un grafo non orientato $G = (V, E)$, un nodo $s \in V$ e un intero $k > 0$, scrivere un algoritmo di complessità ottima che ritorna **true** se e solo se la componente connessa cui appartiene il nodo s ha un numero di nodi $\leq k$ (il nodo s deve essere incluso nel conteggio, quindi se s è un nodo isolato, la componente connessa cui appartiene ha un nodo).

Discutere informalmente la correttezza della soluzione proposta e calcolare la complessità computazionale.

Esercizio 3 – Punti ≥ 9 (Parte B)

Nella sagra del vostro paese, è stato deciso di realizzare lo sfilatino alla Nutella più grande del mondo, lungo L centimetri. Ora si tratta di realizzare un altro record: il maggior numero di persone servite con lo stesso sfilatino. Alla sagra sono presenti n persone, dove la persona i-esima chiede un *segmento* di sfilatino lungo $V[i]$ centimetri; secondo il regolamento, ogni richiesta va servita esattamente, ovvero se la persona i verrà servita, riceverà il segmento richiesto. Tutte le lunghezze sono intere. Scrivere un algoritmo che restituisca il numero massimo di persone che possono essere servite con lo sfilatino. Non è necessario utilizzare tutto lo sfilatino.
Oltre a calcolare la complessità dell'algoritmo proposto, discutere anche la correttezza, menzionando la tecnica scelta e specificando bene perché tale tecnica può essere applicata in questo caso.

Esercizio 4 – Punti ≥ 12 (Parte B)

Dopo tante promesse, l'IMU sulla prima casa non è stata abolita. Avete a disposizione tanti spiccioli, più che sufficienti a pagare l'IMU, e decidete per protesta di pagare la vostra rata di T centesimi di euro con il **maggior** numero di monete possibili. Avete a disposizione n monete, dove la moneta i-esima ha valore $V[i]$ centesimi. Scrivere un algoritmo che restituisca il massimo numero di monete che possono essere utilizzate per pagare **esattamente** L centesimi, o $-\infty$ se le monete che abbiamo non permettono di pagare esattamente L centesimi. Ad esempio, potreste avere a dispozione 6 monete così distribuite:

$$V = \{2, 1, 5, 2, 2, 5\}$$

il che significa che avete una moneta da 1 centesimo, tre da 2 centesimi e due da 5 centesimi. Per pagare 7 centesimi potreste usare una moneta da 5 e una da 2, ma in realtà utilizzare tre monete da 2 centesimi e una da 1 è la soluzione ottimale, perchè utilizza il maggior numero di monete.
Oltre a calcolare la complessità dell'algoritmo proposto, discutere anche la correttezza, menzionando la tecnica scelta ed specificando bene perché tale tecnica può essere applicata in questo caso.