Esercizi svolti a lezione

Problema 1

In un corso di laurea sono previsti un certo numero di esami obbligatori. Esistono inoltre dei vincoli di propedeuticità: se un esame A è propedeutico ad un esame B allora B deve essere sostenuto in una sessione d'esami successiva a quella in cui è stato sostenuto A. Se due o più esami non violano alcun vincolo di propedeuticità allora possono essere sostenuti tutti all'interno della stessa sessione. Supponendo di superare ogni esame al primo tentativo fornire un algoritmo lineare che calcoli il minimo numero di sessioni necessarie per sostenere tutti gli esami. Per ogni sessione d'esami, fornire inoltre l'insieme di esami da sostenere.
Suggerimento: Considerare il grafo delle propedeuticità e le sue proprietà strutturali.

Problema 2

Su un grafo non orientato e non pesato $G = (V, E)$ si possono muovere due robot telecomandabili a distanza. All'inizio i due robot sono posizionati su due nodi del grafo, diciamo s_1 e s_2. In ogni istante di tempo è possibile effettuare la seguente *mossa*: ordinare a uno dei due robot di spostarsi dal nodo su cui è in un nodo adiacente. L'obiettivo è portare il robot che si trova su s_1 in un certo nodo t_1 e il robot che si trova su s_2 in un certo nodo t_2. Le antenne dei robot, però, suffrono di problemi di interferenze: se i robot finiscono troppo vicini l'uno con l'altro non riescono più a ricevere il segnale e quindi a muoversi. Per questo motivo si vuole che i robot in ogni istante di tempo sono sempre a distanza reciproca (nel grafo) di almeno k, dove k è un parametro del problema. Progettare un algoritmo che trovi il minimo numero di mosse che porta i robot nelle posizioni desiderate.

Problema 3

È dato un grafo non orientato $G = (V, E)$. Si vogliono posizionare delle monete sui vertici di G. Per posizionare una moneta, questa deve dapprima essere piazzata su di un vertice x di G non ancora occupato e successivamente spostata su un vertice adiacente ad x, anch'esso non occupato. Progettare un algoritmo lineare che trovi il modo di posizionare il maggior numero di

monete possibili sui vertici di G.
Suggerimento: Pensare al caso in cui G è un albero ed estendere la soluzione trovata a grafi generici.

Problema 4

Dato un grafo $G = (V, E)$ con pesi positivi sugli archi ed un insieme di k centri $C = \{c_1, c_2, \ldots, c_k\} \subseteq V$, si richiede di partizionare l'insieme V in k insiemi $V_1, V_2, \ldots, V_k$ in modo che tutti i vertici in un insieme V_i siamo più vicini al vertice c_i che ad ognuno degli altri centri in C. Formalmente, se $d(u, v)$ indica la distanza tra i vertici u e v in G, deve valere:

$$d(u, c_i) \leq d(u, c_j) \; \forall i, j = 1, \ldots, k \; \forall u \in V_i$$

Progettare un algoritmo che risolva tale problema in tempo $O(|E|+|V| \log |V|)$.

Problema 5

Sia $G = (V, E, w)$ un grafo orientato con pesi positivi sugli archi in cui alcuni archi sono speciali e sono chiamati archi *blu*. Progettare un algoritmo che, dato G, due nodi $s, t \in V$ e un intero k, calcoli un cammino di costo minimo da s a t che usa *al più* k archi blu.

Problema 6

Dato un grafo diretto aciclico (DAG) $G = (V, E)$ e due nodi $s, t \in V$ progettare un algoritmo lineare (in $|V|$ e $|E|$) che calcoli il numero di cammini distinti in G da s a t. Suggerimento: si usi la programmazione dinamica.