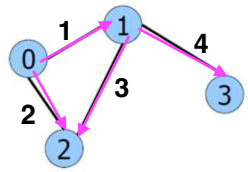
Esercitazione di laboratorio n. 4

Esercizio n.1: Vertex Cover
Competenze: esplorazione dello spazio delle soluzioni con i modelli del Calcolo Combinatorio (Ricorsione e problem-solving: 3.2, 3.3)
Sia dato un grafo non orientato G di N vertici, identificati da interi nell'intervallo 0..N-1, ed E archi, identificati come coppie di vertici. Il grafo è memorizzato su di un file, nella cui prima riga compaiono N ed E, mentre nelle E righe successive compaiono, uno per riga, gli archi nella forma u v.

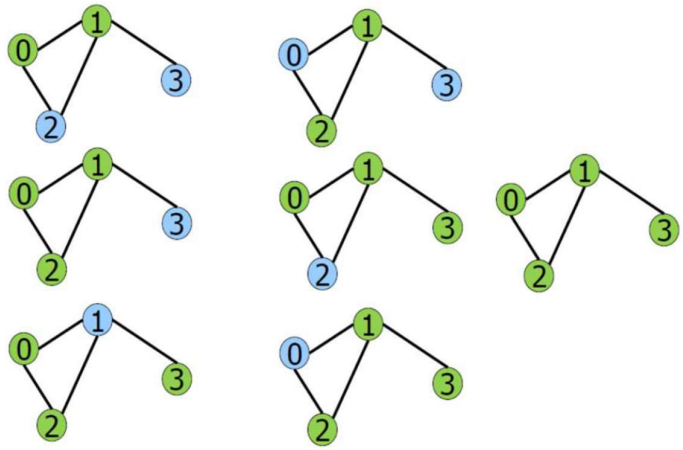
Un vertex cover è un sottoinsieme W dei vertici tale che tutti gli archi in E abbiano almeno uno dei 2 vertici su cui incidono in W: $\forall (u,v) \in E,\ u \in W$ OR $v \in W$.

Dopo aver letto da file il grafo G ed aver memorizzato le informazioni rilevanti in opportune strutture dati, si elenchino tutti i vertex cover.

Esempio: per il grafo seguente



esistono i seguenti vertex cover: (0, 1), (1, 2), (0, 1,2), (0, 1,3), (0, 2,3), (1, 2,3), (0,1,2,3).



Si osservi che questo esercizio non richiede conoscenze di Teoria dei Grafi.

Esercizio n.2: **Anagrafica con liste**
Competenze: creazione e gestione di liste concatenate (Puntatori e strutture dati dinamiche: 4.1)
I dettagli di una anagrafica sono memorizzati in file di testo composti da un numero indefinito di righe nella seguente forma:
<codice> <nome> <cognome> <data_di_nascita> <via> <citta'> <cap>

Il campo <data_di_nascita> è nella forma gg/mm/aaaa, <cap> è un numero intero, mentre tutti i campi rimanenti sono stringhe senza spazi di massimo 50 caratteri. <codice> è nella forma AXXXX, dove X rappresenta una cifra nell'intervallo 0-9, ed è univoco nell'intera anagrafica. I dettagli dell'anagrafica vanno racchiusi in un opportuno tipo di dato Item.

L'anagrafica va memorizzata in una lista ordinata per data di nascita (le persone più giovani **appaiono prima nella lista).**

Si scriva un programma in C che, una volta inizializzata una lista vuota, offra le seguenti funzionalità:

- acquisizione ed inserimento ordinato di un nuovo elemento in lista (da tastiera)
- acquisizione ed inserimento ordinato di nuovi elementi in lista (da file)
- ricerca, per codice, di un elemento
- cancellazione (con estrazione del dato) di un elemento dalla lista, previa ricerca per codice
- cancellazione (con estrazione del dato) di tutti gli elementi con date comprese tra 2 date lette da tastiera. Si consiglia, anziché di realizzare una funzione che cancelli dalla lista questi elementi, restituendoli memorizzati in una lista o in un vettore dinamico, di implementare una funzione che estragga e restituisca al programma chiamante il primo degli elementi appartenenti all'intervallo. Il programma chiamante itererà la chiamata di questa funzione, stampando il risultato, per tutti gli elementi dell'intervallo
- stampa della lista su file.

Per le funzioni di ricerca e cancellazione è richiesto che la funzione che opera sulle liste ritorni **l'elemento trovato o cancellato al programma chiamante, che provvede alla stampa.**

Esercizio n.3: Collane e pietre preziose

Competenze: esplorazione dello spazio delle soluzioni con i modelli del Calcolo Combinatorio (Ricorsione e problem-solving: 3.2.4, 3.3.6), problemi di ottimizzazione (Ricorsione e problemsolving: 3.5)

introdurre condizioni di pruning

Un gioielliere ha a disposizione z zaffiri, r rubini, t topazi e s smeraldi per creare una collana infilando una pietra dopo l'altra. Deve però soddisfare le seguenti regole:

- uno zaffiro deve essere seguito immediatamente o da un altro zaffiro o da un rubino
- uno smeraldo deve essere seguito immediatamente o da un altro smeraldo o da un topazio
- un rubino deve essere seguito immediatamente o da uno smeraldo o da un topazio
- un topazio deve essere seguito immediatamente o da uno zaffiro o da un rubino.

Si scriva una funzione C che calcoli la lunghezza e visualizzi la composizione di una collana a lunghezza massima che rispetti le regole di cui sopra. La lunghezza della collana è il numero di pietre preziose che la compongono.

Osservazione: la lunghezza della soluzione non è nota a priori, ma può variare tra 1 e (z+r+t+s).

Suggerimento: l'esercizio può essere risolto adottando un approccio simile a quello delle disposizioni con ripetizione, visto a lezione, se opportunamente adottato ai requisiti del problema. Una volta impostato il modello ricorsivo, si scriva poi la funzione di filtro (verifica di accettabilità) e di ottimizzazione. Si valuti infine la possibilità di introdurre criteri di pruning.

Nota: si presti attenzione al crescere del tempo di esecuzione richiesto dall'identificazione della soluzione a fronte dell'aumentare dei valori dei parametri di ingresso per il problema (numero di pietre preziose disponibili).

Esercizio n.4: Collane e pietre preziose (versione 2)
Si consideri il contesto introdotto dall'esercizio precedente. Ogni tipologia di pietra è caratterizzata dal suo valore (intero non negativo) (val_z, val_s, val_r, val_t). Il valore della collana è dato dalla somma dei valori delle singole pietre che la compongono. Indicando con n_z, n_s, n_r e n_t il numero di zaffiri, smeraldi, rubini e topazi, il valore della collana è:

val = val_z* n_z + val_s* n_s + val_r* n_r + val_t* n_t

La composizione della collana deve rispettare tutte le regole introdotte nell'esercizio precedente. In aggiunta, devono essere rispettati i seguenti criteri:

- nessuna tipologia di pietra si può ripetere più di max_rip volte consecutive
- nella collana, il numero di zaffiri non può superare il numero di smeraldi

Si scriva una funzione C che calcoli la composizione di una collana a valore massimo che rispetti le regole di cui sopra.

Valutazione: Gli esercizi 1, 2 ed uno a scelta tra l'esercizio 3 e 4 saranno oggetto di valutazione
Scadenza caricamento di quanto valutato: entro le 23:59 del 16/12/2022.