Esercitazione di laboratorio n. 8

Esercizio n.1: **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 problem-solving: 3.5)

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.2: 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.

Esercizio n. 3: Gioco di ruolo
Competenze: Vettori di `struct`, strutture dati dinamiche, vettori dinamici, riallocazione dinamica (Puntatori e strutture dati dinamiche 2.5.5, 3.3.3, 3.2.3)

Sia dato un file di testo (`pg.txt`), contenente i dettagli di alcuni personaggi di un gioco di ruolo, organizzato come segue:

- il numero di personaggi presenti nel file non è noto a priori
- i dettagli di ogni personaggio sono riportati in ragione di uno per riga. In ogni riga sono presenti tre stringhe quali un codice identificativo univoco, il nome del personaggio e la sua classe. Segue sulla stessa riga una sestupla a rappresentare le statistiche di base del personaggio, nella forma `<hp> <mp> <atk> <def> <mag> <spr>`. Ai fini dell'esercizio non è rilevante conoscere il significato dei campi della sestupla,
- il codice è nella forma `PGXXXX`, dove `X` rappresenta una cifra nell'intervallo 0-9
- il nome e la classe di ogni personaggio sono rappresentati da una stringa, priva di spazi, di massimo 50 caratteri alfabetici (maiuscoli o minuscoli)
- le statistiche sono dei numeri interi positivi o nulli
- tutti i campi sono separati da uno o più spazi.

In un secondo file di testo (`inventario.txt`), sono memorizzati i dettagli di una serie di oggetti a cui i personaggi del gioco hanno accesso. Il file è organizzato come segue:

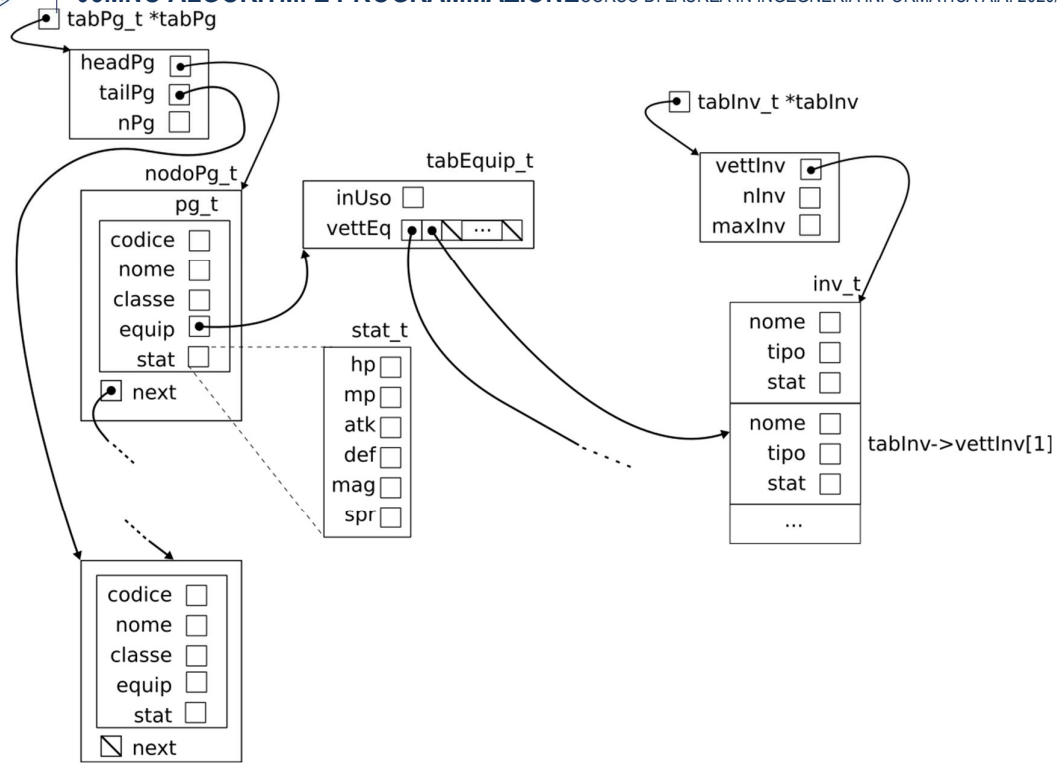
- sulla prima è presente il numero `O` di oggetti
- sulle `O` righe successive appaiono i dettagli di ogni oggetto disponibile
- ogni oggetto è caratterizzato da un nome, una tipologia e da una sestupla a rappresentare i modificatori alle statistiche base di un personaggio
- il nome e la tipologia sono rappresentati da una stringa, priva di spazi, di massimo 50 caratteri alfabetici (maiuscoli o minuscoli)
- i modificatori alle statistiche sono dei numeri interi (potenzialmente anche negativi), quindi possono essere visti come *bonus* (se positivi) o *malus* (se negativi).

Ogni personaggio può fare uso degli oggetti disponibili nell'inventario e comporre liberamente il proprio equipaggiamento, fino ad un massimo di otto elementi.

Ai fini dell'esercizio, si imposti la struttura dati in modo che sia coerente con la rappresentazione grafica proposta in figura (i nomi dei tipi e dei campi sono solo a titolo di esempio). Se necessario, si aggiungano tutti i campi addizionali ritenuti opportuni. Si presti particolare attenzione all'uso di strutture *wrapper* per la memorizzazione delle collezioni (per personaggi, oggetti e equipaggiamenti).

Si scriva un programma in C che permetta di:

- caricare in una lista l'elenco di personaggi
- caricare in un vettore di strutture, allocato dinamicamente, l'elenco di oggetti
- aggiungere un nuovo personaggio
- eliminare un personaggio
- aggiungere/rimuovere un oggetto dall'equipaggiamento di un personaggio
- calcolare le statistiche di un personaggio tenendo in considerazione i suoi parametri base e l'equipaggiamento corrente, applicando quindi i bonus e i malus dell'equipaggiamento stesso. Se una certa statistica risultasse inferiore a 0 in seguito all'applicazione dei malus, in fase di visualizzazione si stampi 0, per convenzione, anziché il valore negativo assunto dalla statistica stessa.



Valutazione: gli esercizi 2 e 3 saranno oggetto di valutazione
Scadenza: caricamento di quanto valutato: entro le 23:59 del 18/12/2020.