Lookup database

Creare un programma **lookup-database.c** in linguaggio C che accetti invocazioni sulla riga di comando del tipo:

`lookup-database <db-file1> <db-file2> <...> <db-filen>`

Il programma dovrà fondamentalmente leggere n file *database* con coppie del tipo *(nome, valore)*, permettendo all'utente di effettuare ricerche interattive sugli stessi attraverso una chiave. Ciascun file database è un file testuale in cui ogni riga ha la struttura *"nome:valore"* dove *nome* può contenere degli spazi e/o altri simboli e *valore* è un numero intero. Per una maggiore comprensione, vedere i file d'esempio allegati.

Il programma, una volta avviato creerà *n+2* thread che condivideranno una struttura dati a cui accederanno utilizzando lo strumento *rwlock* (quantità e modalità di impiego da determinare da parte dello studente).

La struttura dati condivisa conterrà: - una lista dinamica singolarmente linkata (implementata dallo studente) in cui verranno caricati i dati secondo le regole definite di seguito; - altri elementi ritenuti utili/necessari.

I ruoli dei thread saranno i seguenti: - ciascun thread $i \in \{1, \ldots, n\}$ leggerà l'i-esimo file passato da riga di comando e ne manterrà il contenuto in memoria sulla lista dinamica presente nella struttura dati condivisa; il file potrebbe avere un numero arbitrario di righe non noto a priori. Inoltre, ciascun thread, dopo aver inserito un nuovo elemento all'interno della lista, dovrà bloccarsi per *8* secondi (utilizzando la funzione *sleep*).

- un thread Q si occuperà di prendere una chiave dallo standard input ed effettuerà una ricerca in tempo reale sulla lista singolamente linkata, stampando in output il valore associato (se la chiave esiste)
- un thread C si occuperà, ogni *8* secondi, di conteggiare il numero di entry (nodi) presenti all'interno della lista singolarmente linkata e stamperà tale statistica a video

Al termine degli inserimenti da parte dei primi n thread, il thread Q dovrà continuare a leggere eventuali query dallo standard input e restituire in output il risultato.

Il programma deve terminare se l'utente invia al thread Q, la keyword **quit**.