Algoritmi e Strutture Dati – Prova di Laboratorio

11/02/2025

Istruzioni

Risolvere il seguente esercizio implementando un programma in un singolo file .cpp, completo di funzione *main*. Si presti particolare attenzione alla formattazione dell'input e dell'output, e alla complessità target indicata per ciascuna funzionalità. Nel caso la complessità target non sia specificata, si richiede che sia la migliore possibile. La lettura dell'input e la scrittura dell'output **DEVONO** essere effettuate tramite gli stream **cin** e **cout** rispettivamente. La correzione avverrà prima in maniera automatica inviando il file .cpp al server indicato in aula. Quest'ultimo esegue dei test confrontando l'output prodotto dalla vostra soluzione con l'output atteso. In caso la verifica abbia esito positivo sarà possibile consegnare il compito, il quale verrà valuato dai docenti in termini di complessità. Si ricorda che è possibile testare la correttezza del vostro programma in locale su un sottoinsieme dei input/output utilizzati nella seguente maniera. I file di input e output per i test sono nominati secondo lo schema: `input0.txt output0.txt input1.txt output1.txt ...` Per effettuare le vostre prove potete utilizzare il comando del terminale per la redirezione dell'input. Ad esempio

```
./compilato < input0.txt
```

effettua il test del vostro codice sui dati contenuti nel primo file di input, assumendo che `compilato` contenga la compilazione della vostra soluzione e che si trovi nella vostra home directory. Dovete aspettarvi che l'output coincida con quello contenuto nel file `output0.txt`. Per effettuare un controllo automatico sul primo file input `input0.txt` potete eseguire la sequenza di comandi

```
./compilato < input0.txt | diff - output0.txt
```

Questa esegue la vostra soluzione e controlla le differenze fra l'output prodotto e quello corretto.

Esercizio

Si consideri un sistema che memorizza informazioni relative ad un insieme di hotel. Ogni hotel è caratterizzato da un intero *ID*, una stringa *nome* e un float *prezzo medio PM*, che indica il prezzo medio delle stanze. Il sistema memorizza gli hotel in una tabella hash con liste di trabocco, utilizzando l'*ID* come chiave. Una volta inseriti tutti gli hotel, il sistema permette di inserire delle coppie $\{H, P\}$, ciascuna rappresentante una prenotazione presso l'hotel con ID H per una stanza ad un prezzo P. Ad ogni inserimento, PM viene aggiornato con la media dei prezzi P (per camere presso l'hotel H) inseriti precedentemente. La media PM di k prezzi è calcolata come

$$PM = \frac{P_1 + P_2 + ...P_k}{k}$$

Si scriva un programma che:

- legga da tastiera N coppie $\{intero,\ stringa\}$, ciascuna rappresentante rispettivamente l'*ID* e il *nome* dell'hotel, e le inserisca all'interno della tabella hash all'indirizzo ottenuto seguendo la seguente funzione

  $$h(x) = \{[(\ a \times x\ ) +\ b\ ]\ \%\ p\ \}\ \%\ (N * 2)$$

  dove p=999149 , a=1000, b=2000 e x è l'*ID* dell'hotel.

- legga da tastiera M prenotazioni specificate da $\{H,\ P\}$, ciascuna rappresentante l'*ID* dell'hotel, e il costo P della prenotazione.

- stampi il *nome* dei primi al più K hotel, ordinati in maniera non decrescente per prezzo medio. A parità di prezzo, gli hotel devono essere ordinati lessicograficamente per *nome*.

L'**input** è formattato nel seguente modo: la prima riga contiene gli interi N, M e K. Seguono N righe contenenti una coppia $\{intero, stringa\}$ ciascuna. Seguono infine M righe contenenti una coppia $\{intero, float\}$ ciascuna.

L'**output** contiene gli elementi della soluzione, uno per riga.

Esempio

Input

```
3 5 2
1 Boomerang
2 Waikiki
3 Safari
1 10
1 20
2 30
3 15
3 15
```

Output

```
Boomerang
Safari
```