Algoritmi e Strutture Dati – Prova di Laboratorio

03/02/2023

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 per la gestione di alberi binari di ricerca (ABR) a etichette intere positive. Ogni nodo è caratterizzato da un intero ID, che ne rappresenta l'etichetta, e da un *nome* in formato stringa.

Sia definita la seguente proprietà **P**: dato un nodo x; sia d la distanza tra x e la radice dell'albero; sia infine f il numero di nodi discendenti di x. Si dice che il nodo x soddisfa la proprietà **P** se d=f. **NOTA:** La distanza tra il nodo radice e se stesso è **0**.

Scrivere un programma che:

- legga da tastiera una sequenza di N coppie $[intero, stringa]$ e le inserisca in un albero binario di ricerca; utilizzando il valore intero come etichetta. Le coppie devono essere inserite nello stesso ordine con cui vengono lette. (per convenzione, i valori $\leq$ vanno inseriti a sinistra);
- stampi, in ordine lessicografico, il *nome* dei nodi dell'albero che soddisfano la proprietà **P**.

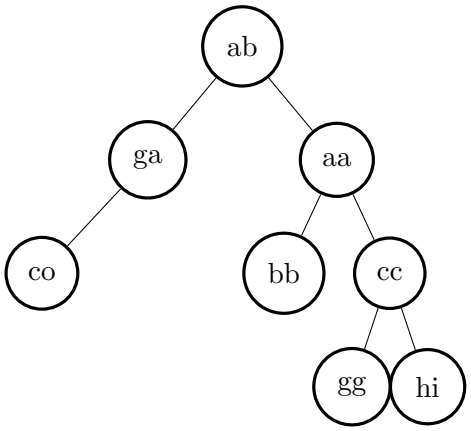
L'**input** è formattato nel seguente modo: la prima riga contiene l'intero N. Seguono N righe contenenti una coppia $[intero, stringa]$ ciascuna, con gli elementi della coppia separati da uno spazio.

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

Esempio 1

Input

```
8
100 ab
120 aa
90 ga
115 bb
140 cc
130 gg
200 hi
1 co
```



Output

```
cc
ga
```