Algoritmica – Esame di Laboratorio

Corso A e B

08/02/2013

Istruzioni

Risolvete il seguente esercizio, prestando particolare attenzione alla formattazione dell'input e dell'output, in quanto la correzione avverrà in maniera automatica.
Per consegnare un elaborato dovete fornire il codice sorgente attraverso il comando `./consegna` che avete nella vostra home directory e che provvede ad inviare il vostro esercizio al server di valutazione. Il comando deve essere utilizzato dal vostro terminale nel seguente modo:

```
./consegna sorgente.c numEx
```

dove:

- `sorgente.c` è il nome del file che contiene la soluzione che avete elaborato, ricordando che il percorso deve essere specificato a partire dalla vostra home directory
- `numEx` è l'identificativo numerico dell'esercizio a cui fa riferimento la soluzione (in questa prova l'esercizio è solo il numero 1, per cui al posto di `numEx` dovete mettere sempre `1`).

Il comando `consegna` può essere utilizzato molteplici volte, per cui è possibile sovrascrivere la propria soluzione per un dato esercizio. Di tutte le consegne per un dato esercizio, viene corretta soltanto l'ultima. Il file da consegnare deve contenere nelle prime righe un commento `C` che specifica il vostro Nome, Cognome e Numero di Matricola. Per esempio:

```
/*
    Nome: Alan
    Cognome: Turing
    Matricola: 193700
*/
```

File non contenenti tali informazioni NON saranno ritenuti validi. Lo script di consegna, prima di inviare la vostra soluzione al server, proverà ad eseguire il vostro codice utilizzando alcuni file di input predefiniti e controllerà

la correttezza dell'output prodotto. Tali file di input e output per la verifica del codice sono stati collocati nella cartella `dati` della vostra home directory. Dentro la cartella `dati` a loro volta i file di input e output per i test sono suddivisi in tante cartelle quanti sono gli esercizi. In questa prova abbiamo solamente l'esercizio 1, per cui dentro la cartella `dati` trovate una sola altra cartella nominata `1` dove sono presenti i file di input e output per effettuare il test dell'esercizio 1. Questi file sono nominati secondo lo schema: `input0.txt output0.txt input1.txt output1.txt ...` Questi file possono essere usati anche da voi per testare il vostro codice e verificare poi manualmente la correttezza dell'output prodotto. Per effettuare le vostre prove potete infatti utilizzare il comando del terminale per la redirezione dell'input. Ad esempio:

```
./compilato < dati/1/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 corrisponda a quanto contenuto nel file `dati/1/output0.txt`. Per effettuare un controllo in automatico sul primo file input `input0.txt` potete eseguire i comandi:

```
./compilato < dati/1/input0.txt > res0
diff res0 dati/1/ouput0.txt
```

Il primo comando infatti esegue la vostra soluzione e stampa l'output prodotto nel file `res0`, il secondo infine controlla le differenze fra l'output prodotto da voi e quello corretto.

La consegna andrà a buon fine solo se il vostro codice riesce a superare tutti i test contenuti nella cartella `dati`. Eventualmente lo script di consegna vi informa per quali di questi test il vostro codice non risponde correttamente. Nel caso invece la vostra soluzione passi i test, vi verrà chiesto di specificare il vostro numero di matricola e la soluzione sarà consegnata. Una volta effettuata la prima consegna, il numero di matricola verrà associato alla macchina e tramite quella macchina non sarà possibile consegnare altri elaborati se non per quello specifico numero di matricola.
Una volta consegnata, la vostra soluzione verrà valutata nel server di consegna utilizzando altri file di test, da voi non accessibili. Il consiglio è quindi quello di non provare ciecamente e ripetutamente la vostra soluzione con i file di test che vi sono forniti ma cercare di ragionare sul codice che avete scritto perché il fatto che il vostro codice passi i test di consegna non significa necessariamente che questo sia corretto in assoluto.

Esercizio 1

Scrivere un programma che legga da tastiera due interi N e D e una sequenza A di N coppie [**intero,stringa**]. Ogni coppia rappresenta il nodo di un albero binario di ricerca. Il programma deve:

- Inserire uno alla volta, nell'ordine dato, le coppie [**intero,stringa**] di A in un albero binario di ricerca **senza ribilanciamento**. L'inserimento deve essere tale per cui, per un qualsiasi nodo, il sottalbero sinistro contenga le coppie la cui stringa sia lessicograficamente **minore o uguale** della stringa del nodo, mentre il sottoalbero destro contiene coppie la cui stringa è lessicograficamente **maggiore**.
- Stampare **in ordine non decrescente** tutte gli interi che si trovano nell'albero in nodi a profondità D. Si ricorda che la profondità di un nodo è uguale alla sua **distanza dalla radice e che la profondità della radice è 0**

Le stringhe devono essere costituite da caratteri alfanumerici (a-Z e 0-9, senza spazi), lunghe **al massimo** 1000 caratteri e **minimo** un carattere.

L'input è formattato nel seguente modo: la prima riga contiene i due interi N e D separati da uno spazio, seguita da $2N$ righe. Ogni coppia è rappresentata da due righe consecutive: la prima riga contiene il valore intero, la seconda la stringa associata. Si assuma che N sia sempre maggiore di zero.

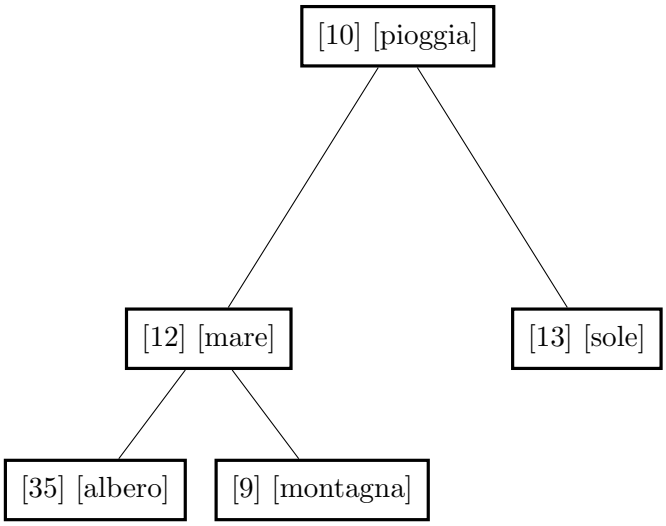
L'output, se esistono nodi a profondità D, deve stampare nella prima riga il numero di nodi trovati e successivamente una riga per ogni intero trovato. **Si ricorda che gli interi devono essere stampati in ordine non decrescente** Se non esistono nodi a profondità D, in programma stampa 0. In entrambi i casi la risposta deve essere seguita dal carattere di ritorno a capo **\n**

È consentito l'utilizzo di librerie standard, come **strcmp**,**strlen**,**qsort**...

Esempi

Input

```
5 2
10
pioggia
12
mare
35
albero
13
sole
9
montagna
```



Output

```
2
9
35
```

Input

```
5 3
10
pioggia
12
mare
35
albero
13
sole
9
montagna
```

Output

```
0
```