

# Algoritmica – Esame di Laboratorio

8/11/2013

## Istruzioni

Risolvete il seguente esercizio prestando particolare attenzione alla formattazione dell'input e dell'output. La correzione avverrà in maniera automatica eseguendo dei test e confrontando l'output prodotto dalla vostra soluzione con l'output atteso. Si ricorda che è possibile verificare la correttezza del vostro programma su un sottoinsieme dei input/output utilizzati. 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 redirectione 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 corrisponda a quanto contenuto nel file `output0.txt`. Per effettuare un controllo automatico sul primo file input `input0.txt` potete eseguire i comandi:

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

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

Una volta consegnata, la vostra soluzione verrà valutata nel server di consegna utilizzando altri file di test non accessibili. Si ricorda di avvisare i docenti una volta che il server ha accettato una soluzione come corretta.

## Esercizio

Scrivere un programma che prenda in input un intero  $K$  e una stringa  $S$  costituita solo da caratteri alfanumerici (a-Z e 0-9, senza spazi) e avente una lunghezza compresa tra i 3 e i 500 caratteri.

L'esercizio consiste nell'inserire tutti i 3-grammi della stringa  $S$  (con ripetizioni) in un albero binario di ricerca **senza ribilanciamento**. Un 3-gramma di  $S$  è una sequenza di tre caratteri consecutivi che occorre in  $S$ . Ad esempio, la stringa  $S = \text{mississippi}$  di lunghezza  $N = 11$  contiene 9 ( $= N - 2$ ) 3-grammi non distinti:  $S[0, 2] = \text{mis}$ ,  $S[1, 3] = \text{iss}$ ,  $S[2, 4] = \text{ssi}$ ,  $S[3, 5] = \text{sis}$ ,  $S[4, 6] = \text{iss}$ ,  $S[5, 7] = \text{ssi}$ ,  $S[6, 8] = \text{sip}$ ,  $S[7, 9] = \text{ipp}$ ,  $S[8, 10] = \text{ppi}$ .

Il programma dovrà scandire  $S$  da sinistra verso destra e inserire nell'albero binario ogni 3-gramma incontrato. L'inserimento **deve** essere tale per cui, per un qualsiasi nodo, il sottoalbero sinistro contenga i 3-grammi **lessicograficamente minori o uguali** al 3-gramma nel nodo, mentre il sottoalbero destro contenga i 3-grammi **lessicograficamente maggiori**. La **dimensione** di un nodo  $u$  dell'albero è data dal numero di nodi nel sottoalbero radicato in  $u$ ,  $u$  compreso.

Dopo aver costruito l'albero binario, il programma dovrà stampare tutti i 3-grammi contenuti in nodi aventi dimensione **minore o uguale** a  $K$ . I 3-grammi devono essere stampati, uno per riga, ordinati lessicograficamente.

L'input è formattato nel seguente modo. La prima riga contiene l'intero  $K$ , la seconda riga contiene la stringa  $S$ .

L'output invece è costituito dai 3-grammi memorizzati in nodi aventi dimensione minore o uguale a  $K$ . I 3-grammi devono essere stampati uno per riga e ordinati lessicograficamente.

## Esempio

### Input

```
3
mississippi
```

### Output

```
ipp
iss
iss
ppi
sip
ssi
```

