Fondamenti di Informatica ◇ 2021-22

Laboratorio #5 del 30-11-2021

Il poeta

Avete deciso di avviarvi alla professione di poeta. Siete risoluti ma, forse a causa della passione collaterale per la scienza, volete qualche garanzia nel caso la fantasia e creatività occasionalmente facciano cilecca. Decidete quindi che la soluzione giusta sia dotarsi di un supporto tecnologico software, ma dopo una rapida analisi in Internet decidete che nessuna software house ha il prodotto che fa per voi, e dovete quindi fare voi. Vi organizzate con un gruppo di amici per realizzare l'applicazione che vi aiuti a creare ... partendo dalle funzionalità che vi sembrano essenziali, che lavorano sui versi (stringhe) che comporranno il poema.

```
#define |
a_poem\
that\
compiles

#include \
<a\source\
which\makes\you\
smile>

So(){
if (you.dare())
go(read, behold);
new SourceCodeChallenge()
->hits->the(road);
}

submit(your code){
AND if(you(re)) proud();
you_ll(trumph); for(sure;;);
AND claim(first, prize);
}

// (C) 2015 www.sourceCodePoetry.com
// "oh come all ye coders the challenge is here"
```

Lunghezza

Una funzionalità irrinunciabile è il controllo della metrica. Decidete per una soluzione approssimativa ma veloce da realizzare: dati due versi, si vuole sapere se questi hanno una lunghezza "compatibile", ossia se le loro lunghezze differiscono al più di $\pm$ 20%.

Rima

Decidete di catalogare tutti i versi di particolare effetto in una base di dati: lo scopo è salvare le perle di eventuali periodi di intensa capacità creativa per riusarle in lavori successivi. Nel momento in cui vi incastrate nel mezzo del lavoro, una visita alla base di dati potrebbe offrire qualche spunto interessante. Naturalmente, però, non è possibile pescare frammenti dalla base di dati a casaccio.
Vi proponete allora di realizzare un filtro che consiste nel trovare solo i versi in rima con una frase data. Realizzate quindi una funzionalità per cui dati due versi, sapete se il secondo è in rima con il primo oppure no. Una condizione che ritenete accettabilmente buona per la rima è che le ultime tre lettere dei versi coincidano (attenzione ai versi corti!).

La cesura

Gli attacchi dei versi sono da sempre il vostro problema, Dato un verso, dovete *troncarlo* in presenza di un punto, o punto e virgola, e restituire la parte iniziale. Se però il verso originale non contiene tali simboli, non vi verrà restituito nulla ... e rimarrete con un pugno di mosche.

Le figure retoriche

Verificare le forme retoriche è piuttosto noioso, e desiderate automatizzare l'operazione. In particolare il vostro stile punta molto su assonanze e allitterazioni.
Per l'assonanza verificate che coincidano le ultime due vocali dei due versi.
Per l'allitterazione trovate una approssimazione sufficientemente efficace: un verso presenta un'allitterazione se contiene almeno 3 parole consecutive che iniziano con la stessa lettera (senza distinguere tra maiuscole e minuscole).
Volete quindi disporre di una funzionalità che vi dica, dati due versi, se contengono allitterazioni e se producono assonanza.

Una nuova *vision*

Il supporto tecnologico realizzato vi dà molte soddisfazioni e decidete di venderlo ... Se la *vision* di Bill Gates era "un computer in ogni casa", la vostra è, forti della vostra esperienza personale, che il vostro software possa realizzare "un poeta in ogni casa".
Dopo una revisione di tutto il codice concludete che prima della vendita sarebbe meglio rendere più robusto il controllo delle allitterazioni, unico tallone d'Achille del vostro applicativo. La nuova specifica che vi proponete, per rispondere alla domanda "è una allitterazione?" è che esista una lettera all'interno del verso che compaia con una frequenza maggiore del 20% del totale delle lettere del verso. Con questa soluzione la funzionalità vi dice anche qual è questa lettera.

L'applicazione

Avendo capito e deciso quali funzionalità vi possono servire durante la creazione di un nuovo poema, cercate di mettere tutto insieme per avere a portata di un comando quello che vi serve ...
Vi ingegnate per avere un menù iniziale che vi chieda cosa volete fare, considerate le possibili opzioni:

0. per oggi ho lavorato abbastanza ... esco
1. dati due versi, controlla la metrica (la lunghezza)
2. dati due versi, verificare se sono in rima
3. dato un verso, cercarne uno tra quanti non ancora usati, che sia in rima
4. dato un verso, visualizzarne, se esiste, la cesura
5. cercare un verso tra quanti non ancora usati, cui sia possibile applicare una cesura
6. dati due versi, verificare se c'è assonanza
7. dato un verso, cercarne uno tra quanti non ancora usati, che abbia assonanza
8. dato un verso, sapere se c'è allitterazione[1]
9. data una lettera, cercare tra i versi non ancora usati uno che presenti un'allitterazione con quella lettera

Dettagli tecnici

- ◇ definire LENMAXVERSO a piacimento ma ragionevole ... un verso sta su una singola riga.
- ◇ in nessun caso maiuscole/minuscole devono causare problemi ... "Chi crede che possa importare al poeta che C e c in C sono diverse?"
- ◇ per memorizzare tutti i versi pensati ma non ancora usati, vi avvalete di un file `futuribili.txt`, in cui, salvate tutto il vostro prezioso sapere

Si utilizzino i sottoprogrammi delle librerie `string.h`, `stdlib.h`, `ctype.h` e simili quando utili, senza sviluppare ex-novo cose che già esistono.

Esempi

Odi et amo. Quare id faciam ...	cesura	Odi et amo
Sempre caro mi fu quest'ermo colle ...	cesura	`NULL`
di **m**e **m**edesmo **m**eco **m**i vergogno e ...	allitterazione	**m**
... m**ar**e – ... s**al**e	assonanza	si

[1] Realizzate la specifica che vi piace di più