Laboratorio di Algoritmi e Strutture Dati

Docenti: M. Torelli, S. Aguzzoli

Progetto "Incastri"

valido per l'appello di gennaio 2011

Premessa

La realizzazione del progetto è una prova d'esame da svolgersi **individualmente**. I progetti giudicati frutto di **copiatura** saranno **estromessi** d'ufficio dalla valutazione.

Si richiede allo studente di effettuare un **adeguato collaudo** del proprio progetto su numerosi esempi diversi per verificarne la correttezza.

La versione aggiornata del progetto è pubblicata in `.pdf` sul sito:
`http://homes.dsi.unimi.it/~aguzzoli/algo.htm`.
Si consiglia di consultare periodicamente questo sito per eventuali correzioni e/o precisazioni relative al testo del progetto.

1 Organizzazione degli appelli e modalità di consegna

Il presente progetto è valido per l'appello di gennaio 2011 (ultimo appello dell'anno accademico 2009/2010) e deve essere consegnato entro il 14 gennaio 2011.

Le discussioni dei progetti per i due appelli si svolgeranno in date e luoghi da specificarsi. Il calendario dei colloqui sarà disponibile sulla pagina del corso `http://homes.dsi.unimi.it/~aguzzoli/algo.htm` qualche giorno dopo il termine di consegna del progetto.

Il progetto va inviato per posta elettronica all'indirizzo `aguzzoli@dsi.unimi.it` entro le date sopra indicate. Occorre presentare:

1. il codice sorgente (rigorosamente ANSI C, compilabile con **gcc**);
2. una sintetica relazione (formato pdf o rtf) che illustra le strutture dati utilizzate e le scelte implementative, analizzando il costo delle diverse operazioni richieste dalla specifica.

I due o più file (file sorgenti C + relazione) devono essere contenuti in un unico file `.zip` il cui nome dovrà essere della forma `cognome_matricola.zip`. La relazione e il codice devono riportare nome, cognome e matricola. Una copia cartacea della relazione e del codice deve inoltre essere consegnata al docente entro le scadenze fissate (lasciandola eventualmente nella sua casella postale presso il dipartimento in via Comelico). Si ricorda infine di presentarsi al colloquio con una copia stampata della relazione e del codice.

Il problema

Il progetto si ispira ad un gioco da tavola, composto da tessere, che devono essere incastrate l'un l'altra a formare delle catene (o sequenze). Su ogni tessera è scritta una parola; inoltre ciascuno dei i bordi laterali di una tessera può essere incastrato con il bordo di un'altra tessera: i bordi sono identificati da

un simbolo peculiare e due tessere si possono incastrare tra loro soltanto se sono identificate dallo stesso simbolo.

Formalmente, sia $\mathcal{A}$ l'insieme delle lettere minuscole dell'alfabeto. Una *tessera* t è definita da una tripla (α, β, σ), dove α, β e σ sono parole di lunghezza *arbitraria* sull'alfabeto $\mathcal{A}$; α e β rappresentano i simboli che identificano i bordi di t, mentre σ è il nome di t.

Per ottenere una catena, possiamo immaginare di disporre le tessere in una sequenza. Ogni tessera può essere disposta in due modi diversi. Denotiamo una *disposizione* di una tessera t con $+t$ o $-t$: se t è la tessera (α, β, σ), allora $+t$ indica la tessera disposta con il bordo di tipo α a sinistra e il bordo di tipo β a destra; viceversa $-t$ indica la tessera disposta con il bordo di tipo β a sinistra e il bordo di tipo α a destra. Formalmente, il *simbolo sinistro* $Simbolo_S$ e il *simbolo destro* $Simbolo_D$ della tessera t sono definiti nel modo seguente:

- $Simbolo_S(+t) = \alpha$ e $Simbolo_D(+t) = \beta$;
- $Simbolo_S(-t) = \beta$ e $Simbolo_D(-t) = \alpha$.

Una *catena* τ è data da una sequenza di disposizioni di tessere

$$T_1, T_2, \ldots, T_n$$

tali che:

- per ogni $1 \leq i \leq n$, T_i è una disposizione di una tessera t_i che non appartiene ad alcun altra catena precedentemente posizionata sulla tavola da gioco;
- per ogni $1 \leq i \leq n$ e $1 \leq j \leq n$, $i \neq j$ implica $t_i \neq t_j$;
- per ogni $1 \leq i \leq n-1$, $Simbolo_D(T_i) = Simbolo_S(T_i + 1)$.

La *lunghezza* di τ è n (numero di tessere usate in τ). Se $Simbolo_S(T_1) = \alpha$ e $Simbolo_D(T_n) = \beta$, diciamo che τ è una catena dal simbolo α al simbolo β. Una catena *minima* da α a β è una catena da α a β avente lunghezza minima fra tutte le possibili catene da α a β che si possono formare usando tessere non ancora disposte sul tavolo da gioco.

Esempio

Siano date le seguenti tessere:

$t_1 = (quadrato, croce, pane)$, $t_2 = (quadrato, fiore, cane)$, $t_3 = (luna, fiore, casa)$, $t_4 = (croce, luna, cosa)$

Allora le catene da *quadrato* a *luna* sono i seguenti:

$$+t_1, +t_4 \quad \text{e} \quad +t_2, +t_3$$

entrambe di lunghezza minima 2. Le catene da *quadrato* a *croce* sono:

$$+t_1 \quad \text{e} \quad +t_2, -t_3, -t_4$$

e la catena $+t_1$, che ha lunghezza 1, è l'unica catena da *quadrato* a *croce* di lunghezza minima.

Si consideri una catena $T_1, T_2, \ldots, T_n$ e, per ogni $i = 1, \ldots, n$, si denoti con σ_i il nome della tessera t_i (di cui T_i è la disposizione nella catena). Il *nome* della catena $T_1, T_2, \ldots, T_n$ è la stringa $\sigma_1\sigma_2\cdots\sigma_n$ ottenuta concatenando i nomi delle singole tessere. La lettura del nome $\sigma_1\sigma_2\cdots\sigma_n$ è tanto più *cacofonica* quanto più i nomi delle tessere adiacenti sono simili tra loro.

Formalmente, date due stringhe $\sigma_1 = a_1a_2\cdots a_u$ e $\sigma_2 = b_1b_2\cdots b_v$, diciamo che $\sigma = c_1c_2\cdots c_z$ è una *sottostringa* di σ_1 e σ_2 se σ verifica le seguenti proprietà:

1. Per ogni $h = 1, \ldots, z$ esistono indici $i(h)$ e $j(h)$ con $1 \leq i(h) \leq u$ e $1 \leq j(h) \leq v$ tali che $c_h = a_{i(h)} = b_{j(h)}$.
2. Per ogni coppia di indici h_1, h_2 con $1 \leq h_1 < h_2 \leq z$ si ha che $i(h_1) < i(h_2)$ e $j(h_1) < j(h_2)$.

Diciamo che σ è una *sottostringa massima* di σ_1 e σ_2 se σ è una sottostringa di σ_1 e σ_2 avente lunghezza massima fra tutte le sottostringhe di σ_1 e σ_2. Si noti che due stringhe possono ammettere più di una sottostringa massima, tutte della stessa lunghezza. Ad esempio, le sottostringhe massime di *abc* e *bac* sono *ac* e *bc*, di lunghezza 2. Con $\delta(\sigma_1, \sigma_2)$ denotiamo la lunghezza di una sottostringa massima di σ_1 e σ_2. Sia τ una catena e sia $\sigma_1\sigma_2\cdots\sigma_n$ il suo nome; l'*indice di cacofonia* di τ è data da

$$\sum_{i=1}^{n-1} \delta(\sigma_i, \sigma_{i+1})\,.$$

Esempio

Il nome della catena $+t_2, -t_3, -t_4$, costruita nell'esempio precedente, è *canecasacosa*. Si noti che $\delta(cane, casa) = 2$ poiché l'unica sottostringa massima è *ca*, mentre $\delta(casa, cosa) = 3$ poiché l'unica sottostringa massima è *csa*, dunque l'indice di cacofonia della catena risulta essere $2 + 3 = 5$.

Si richiede di implementare una struttura dati efficiente che permetta di eseguire le operazioni seguenti:

- **tessera** (α, β, σ)

  Se esiste già una tessera di nome σ oppure se α è uguale a β, non compie alcuna operazione. Altrimenti, crea la tessera (α, β, σ).

- **catena**(α, β)

  Crea e posiziona sul tavolo da gioco una catena di lunghezza minima dal simbolo α al simbolo β. Se non è possibile crearlo, stampa il messaggio:

  `non esiste catena da` α `a` β

- **stampaCatena**(σ)

  Se non esiste alcuna tessera di nome σ oppure se la tessera di nome σ non appartiene ad alcuna catena sul tavolo da gioco, non compie alcuna operazione.

  Altrimenti, stampa la catena cui appartiene la tessera σ secondo il formato specificato nell'apposita sezione.

- **cancellaCatena**(σ)

  Se non esiste alcuna tessera di nome σ oppure se la tessera di nome σ non appartiene ad alcuna catena, non compie alcuna operazione.

  Altrimenti, sia τ la catena cui appartiene la tessera di nome σ. Allora, τ è cancellato e tutte le tessere che compongono τ possono essere riutilizzate nella definizione di nuove catene.

- **max**(σ_1, σ_2)

  Se almeno una tra σ_1 e σ_2 non è un nome di tessera, non compie alcuna operazione. Altrimenti, stampa su una nuova linea una sottostringa massima σ di σ_1 e σ_2 (se σ è la stringa nulla, la linea rimane vuota).

- **indiceCacofonia**(σ)

  Se non esiste alcuna tessera di nome σ oppure se la tessera di nome σ non appartiene ad alcuna catena, non compie alcuna operazione.

  Altrimenti stampa l'indice di cacofonia della catena cui appartiene la tessera di nome σ.

Si noti che le operazioni richieste sono liberamente implementabili; in particolare, non vanno necessariamente intese come prototipi di funzioni.

Specifiche di implementazione

Il programma deve leggere dallo standard input (`stdin`) una sequenza di righe (separate da `\n`), ciascuna delle quali corrisponde a una riga della prima colonna della Tabella 1, dove α, β e σ sono stringhe finite sull'alfabeto $\mathcal{A}$ di lunghezza *arbitraria*. I vari elementi sulla riga sono separati da uno o più spazi. Quando una riga è letta, viene eseguita l'operazione associata; le operazioni di stampa sono effettuate sullo standard output (`stdout`), e ogni operazione deve iniziare su una nuova riga.

Riga di input	Operazione
`t` $\alpha\ \beta\ \sigma$	**tessera** (α, β, σ)
`C` $\alpha\ \beta$	**catena** (α, β)
`s` σ	**stampaCatena** (σ)
`c` σ	**cancellaCatena** (σ)
`m` $\alpha\ \beta$	**max** (α, β)
`i` σ	**indiceCacofonia** (σ)
`f`	Termina l'esecuzione del programma

Tabella 1: Specifiche del programma

Note

1. Non devono essere presenti vincoli sul numero di tessere e catene esistenti; potete invece assumere che tutte le stringhe corrispondenti ai simboli e ai nomi delle tessera siano lunghe al più 15 caratteri. Non si richiede – anzi si sconsiglia – l'uso di grafica, se non per test personali: in modo particolare, non si usi `conio.h` e neppure `clrscr()`.

2. Per semplicità si suppone che l'input sia sempre conforme alle specifiche di Tabella 1, per cui non è necessario controllare la correttezza dell'input. Per leggere l'input si usino le funzioni standard ANSI C `getchar()` e/o `scanf()`.

3. **Formato per la stampa di una catena**

Si supponga che la catena da stampare sia

$$T_1, T_2, \ldots, T_n$$

e che le tessere che lo costituiscono siano $t_i = (\alpha_i, \beta_i, \sigma_i)$ per ogni $1 \leq i \leq n$. Si ponga

$$\gamma_i = \sigma_i : \alpha_i, \beta_i \qquad \text{se} \qquad T_i = +t_i \,,$$

$$\gamma_i = \sigma_i : \beta_i, \alpha_i \qquad \text{se} \qquad T_i = -t_i \,.$$

L'output da produrre è il seguente:

(
γ_1
γ_2
$\vdots$
γ_n
)

Esempio
Si supponga che le righe di input siano:

```
t cerchio quadrato cammello
t sole croce topolino
t stella fiore elefante
t sole stella cane
t luna stella cavallo
t cerchio luna cappello
t fiore quadrato mela
t croce fiore topo
t cuore croce tela
t croce sole tavolo
m mela tela
m cappello cammello
m cane topo
m topo topolino
m gatto  topo
m tela elefante
C stella cerchio
s cavallo
i cappello
C stella cerchio
s elefante
i mela
C fiore cerchio
c cammello
C fiore cerchio
s mela
i cammello
c mela
```

```
c cappello
t cuore cerchio pesca
C croce croce
s tavolo
i topolino
C croce croce
s cammello
i pesca
f
```

L'output prodotto dal programma deve essere:

```
ela
caello

topo
to
ela
(
cavallo: stella, luna
cappello: luna, cerchio
)
5
(
elefante: stella, fiore
mela: fiore, quadrato
cammello: quadrato, cerchio
)
6
non esiste catena da fiore a cerchio
(
mela: fiore, quadrato
cammello: quadrato, cerchio
)
3
(
topolino: croce, sole
tavolo: sole, croce
)
4
(
tela: croce, cuore
pesca: cuore, cerchio
cammello: cerchio, quadrato
mela: quadrato, fiore
topo: fiore, croce
)
7
```