Laboratorio di Algoritmi e Strutture Dati

Docente: V. Lonati

Progetto "Mattoncini"

valido per gli appelli di gennaio e febbraio 2024

Indice

1 Organizzazione degli appelli e modalità di consegna

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 nella pagina del corso `https://myariel.unimi.it/course/view.php?id=1221`. Si consiglia di consultare periodicamente questo sito per eventuali correzioni e/o precisazioni relative al testo del progetto. Per ogni ulteriore chiarimento potete chiedere un appuntamento scrivendo una mail all'indirizzo `lonati@di.unimi.it` .

La presente traccia è valida per gli appelli di gennaio e febbraio 2024.

Il progetto svolto va consegnato tramite il sistema di upload (`https://upload.di.unimi.it`)del Dipartimento entro una delle seguenti scadenze:

1. Prima scadenza: 15 gennaio 2024. Data indicativa della discussione: 22 gennaio 2024
2. Seconda scadenza: 29 gennaio 2024. Data indicativa della discussione: 5 febbraio 2024
3. Terza scadenza: 12 febbraio 2024. Data indicativa della discussione: 19 febbraio 2024

Occorre consegnare:

1. il codice sorgente Go;

2. una relazione in formato `.pdf`, di lunghezza compresa tra 3 e 10 pagine, che illustra le strutture dati utilizzate e le scelte implementative, in relazione sia al codice sia al problema descritto nella traccia; la relazione deve inoltre contenere l'analisi dei costi delle diverse operazioni richieste dalle specifiche;

3. almeno 5 esempi di input e relativi output usati per testare il programma.

I file devono essere contenuti in un unico archivio `.zip`. Tutti i file nell'archivio, compresa la relazione, devono riportare nome, cognome e matricola dell'autore.

È possibile consegnare una sola volta: se un progetto viene valutato insufficiente, non verranno valutate ulteriori consegne.

Le date delle discussioni sono da considerarsi indicative e potranno subire variazioni. Qualche giorno dopo ciascuna scadenza per la consegna, i progetti consegnati verranno corretti e verrà pubblicato un avviso con la data e l'orario della discussione e l'elenco delle persone convocate per la discussione.

È necessario presentarsi alla discussione con un computer portatile con i file consegnati (oppure richiedere con adeguato anticipo di svolgere la discussione in una aula con computer).

2 Criteri di valutazione dei progetti

Nella valutazione del progetto si terrà conto delle seguenti dimensioni: capacità di modellare il problema mediante strutture di dati opportune; capacità di progettare soluzioni algoritmiche efficienti; capacità di implementare in Go le strutture di dati e gli algoritmi scelti, in maniera appropriata rispetto al tipo di elaborazione richiesta; chiarezza espositiva e proprietà di linguaggio nell'illustrare le scelte di modellazione e di implementazione fatte (sia in relazione al problema che in relazione al codice); correttezza formale (sintattica) del codice Go prodotto; funzionamento del programma (correttezza e completezza degli output prodotti dal programma); qualità del codice (codice ripetuto/ridondante/intricato/oscuro, strutturazione del codice, ecc).

Il progetto verrà estromesso dalla valutazione in uno qualunque dei seguenti casi:

1. il programma non compila;

2. il programma non contiene la definizione dei tipi e delle funzioni indicate nelle specifiche di implementazione;

3. il programma contiene le funzioni specificate, ma con segnatura diversa.

3 Il problema

Il progetto si ispira ad un gioco da tavola, composto da mattoncini che devono essere incastrati l'un l'altro a formare delle file di mattoncini.

Mattoncini

Su ogni mattoncino è scritta una parola; inoltre ciascuno dei bordi laterali di un mattoncino ha una forma particolare e può essere incastrato con il bordo di un altro mattoncino: due mattoncini si possono incastrare tra loro soltanto se hanno un bordo con la stessa forma.

Formalmente, un *mattoncino* m è definito da una tripla (α, β, σ), dove α, β e σ sono stringhe di lunghezza *arbitraria*; α e β denotano le forme ai bordi di m, σ il nome di m.

I mattoncini sono tutti diversi tra loro, e sono raccolti in una *scatola*.

Durante il gioco i mattoncini vengono disposti orizzontalmente sul tavolo. Ogni mattoncino può essere disposto in due modi diversi: più precisamente il mattoncino identificato dalla tripla (α, β, σ) può essere disposto con α a sinistra e β a destra, oppure con β a sinistra e α a destra.

Per comodità, dato un mattoncino identificato dalla tripla $t = (\alpha, \beta, \sigma)$, nel seguito scriveremo $+t$ per indicare il mattoncino identificato da t disposto con la forma α a sinistra e $-t$ per indicare il mattoncino identificato da t disposto con la forma α a destra.

File di mattoncini

Una *fila* si ottiene incastrando i mattoncini in orizzontale uno di seguito all'altro:

$$m_1,\, m_2,\, \ldots,\, m_k$$

In una fila i mattoncini devono essere disposti in modo che i bordi si incastrino: dunque il bordo a destra di m_i deve avere la stessa forma del bordo a sinistra di m_{i+1}, per ogni i da 1 a $k-1$ (inclusi).

Naturalmente se un mattoncino è usato per costruire una fila, non sarà più presente nella scatola e non potrà essere più utilizzato per costruire altre file. In particolare questo vuol dire che lo stesso mattoncino non può comparire due volte nella stessa fila.

La lunghezza di una fila è definita dal numero di mattoncini che la formano; ad esempio la lunghezza di $m_1,\, m_2,\, \ldots,\, m_k$ è k. Se α denota la forma del bordo a sinistra del primo mattoncino di una fila F e β denota la forma a destra dell'ultimo mattoncino di F, allora diciamo che F è una fila da α a β. Una fila *minima* da α a β è una fila da α a β avente lunghezza minima fra tutte le possibili file da α a β.

Esempio Consideriamo i mattoncini identificati dalle seguenti triple:

$$t_1 = (\texttt{quadrato}, \texttt{croce}, \texttt{PANE}) \qquad t_2 = (\texttt{quadrato}, \texttt{fiore}, \texttt{CANE})$$
$$t_3 = (\texttt{luna}, \texttt{fiore}, \texttt{CASA}), \qquad t_4 = (\texttt{croce}, \texttt{luna}, \texttt{COSA})$$

Allora le file da `quadrato` a `luna`, entrambe di lunghezza minima 2, sono le seguenti:

$$+t1, +t4 \qquad \text{e} \qquad +t2, -t3.$$

Le file da `quadrato` a `croce` sono:

$$+t1 \qquad \text{e} \qquad +t2, -t3, -t4$$

e la fila $+t1$, che ha lunghezza 1, è l'unica fila da `quadrato` a `croce` di lunghezza minima. □

Nomi cacofonici

Il nome di una fila è ottenuto concatenando, in ordine da sinistra verso destra, i nomi dei singoli mattoncini che la compongono. Più formalmente, se σ_i è il nome del mattoncino m_i per ogni i da 1 a n, allora il nome della fila $m_1,\, m_2,\, \ldots,\, m_n$ è dato dalla concatenazione di $\sigma_1, \sigma_2, \ldots, \sigma_n$.

La lettura del nome $\sigma_1, \sigma_2, \ldots, \sigma_n$ è tanto più *cacofonica* quanto più i nomi dei mattoncini adiacenti sono simili tra loro. Definiamo più precisamente questo concetto.

Date due stringhe σ e τ, diciamo che ρ è una *sottostringa comune a* σ e τ se tutte le lettere di ρ compaiono anche in σ e τ, nello stesso ordine in cui compaiono in ρ (anche non consecutivamente). Diciamo che ρ è una *sottostringa massima* comune a σ e τ se ρ è una sottostringa comune a σ e τ avente lunghezza

massima tra tutte le sottostringhe comuni a σ e τ. Ad esempio le sottostringhe massime comuni a `ABC` e `BAC` sono `AC` e `BC`, entrambe di lunghezza 2. Indichiamo con $\delta(\sigma, \tau)$ la lunghezza della sottostringa massima comune a σ e τ.

L'*indice di cacofonia* di una fila è data dalla somma delle lunghezze delle sottostringhe massime associate a coppie di mattoncini vicini nella fila. Formalmente, se $F = m_1, m_2, \ldots, m_n$ e σ_i è il nome del mattoncino m_i per ogni i, allora l'indice di cacofonia di F è dato da

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

Esempio Il nome della fila $+t_2, -t_3, -t_4$, costruita nell'esempio precedente, è `CANECASACOSA`. L'indice di cacofonia della fila risulta essere $2 + 3 = 5$. Infatti $\delta(\texttt{CANE}, \texttt{CASA}) = 2$ poiché l'unica sottostringa massima è `CA`, mentre $\delta(\texttt{CASA}, \texttt{COSA}) = 3$ poiché l'unica sottostringa massima è `CSA`. □

Trasformazione di file

Data una qualunque fila $F = m_1, m_2, \ldots, m_n$ consideriamo la sequenza delle forme ai bordi di ciascun mattoncino, partendo da quello di sinistra di m_1. Tale sequenza è detta *sequenza di forme definita da* F.

Esempio Nell'esempio precedente, la sequenza di forme definita dalla fila $+t_1, +t_4$ è `quadrato`, `croce`, `luna`. La sequenza di forme definita da $+t_2, -t_3$ è `quadrato`, `fiore`, `luna`. □

Diciamo che *dalla fila F si può ottenere la sequenza di forme $s = f_1, f_2, f_n$* se esiste una fila F' che definisce la sequenza di forme s e che può essere ottenuta da F effettuando una sequenza di operazioni elementari di *inserimento* o *cancellazione* di mattoncini.

Più precisamente, un'operazione di inserimento in una fila si ha quando un mattoncino (preso dalla scatola) si inserisce all'inizio della fila, alla fine della fila, oppure tra due mattoncini già nella fila. Non è necessario che si ottenga di nuovo una fila, ovvero che il nuovo mattoncino si incastri con i mattoncini vicini.

Un'operazione di cancellazione si ha quando si toglie un mattoncino dalla fila. Non è necessario che si ottenga di nuovo una fila (perciò, nel caso si tolga un mattoncino che non è né il primo né l'ultimo della fila, non è necessario che i due mattoncini ai lati di quello tolto si incastrino tra loro. Il mattoncino tolto dalla fila viene rimesso nella scatola).

In ciascuna operazione si può cambiare a piacere la disposizione dei mattoncini (invertendo cioè i due lati destro/sinistro).

Se è possibile ottenere una sequenza di forme $s = f_1, f_2, \ldots, f_n$ da F, il *costo* del passaggio da F a s è dato dal minimo numero di operazioni elementari necessarie per trasformare F in un'altra fila F' che definisce la sequenza di forme s. In caso contrario, il costo del passaggio da F a s è indefinito.

Si noti che, come sottolineato anche sopra, nei passaggi intermedi non è necessario che la sequenza di mattoncini costituisca una fila. Inoltre i cambiamenti di disposizione dei mattoncini non influiscono sul calcolo del costo.

Esempio Consideriamo i mattoncini $m_1 = (\texttt{a}, \texttt{b}, \texttt{uno})$, $m_2 = (\texttt{c}, \texttt{b}, \texttt{due})$, $m_3 = (\texttt{c}, \texttt{d}, \texttt{tre})$. Supponiamo di avere già costruito la fila $F = +m_1, -m_2, +m_3$ che definisce la sequenza di forme $s = \texttt{a}, \texttt{b}, \texttt{c}, \texttt{d}$.

Consideriamo la sequenza di forme $s' = \texttt{d}, \texttt{c}, \texttt{a}, \texttt{b}, \texttt{c}$.

Se non ci sono altri mattoncini nella scatola, il costo di passaggio da F a s' è indefinito (infatti, non è possibile in alcun modo aggiungere alla prima fila altri mattoncini).

Supponiamo ora che la scatola contenga i mattoncini $m_4 = (\texttt{a}, \texttt{b}, \texttt{quattro})$, $m_5 = (\texttt{a}, \texttt{c}, \texttt{cinque})$ e $m_6 = (\texttt{c}, \texttt{a}, \texttt{sei})$. In questo caso il costo di passaggio da F a s' è 3. Un esempio di sequenza di tre operazioni elementari che permette di ottenere s' da F è:

1. Eliminazione di m_3 da F. Dopo questa operazione, si ottiene $+m_1, -m_2$. La scatola contiene ora i mattoncini m_3, m_4, m_5, m_6.
2. Inserimento di $-m_3$ all'inizio. Si ottiene: $-m_3, +m_1, -m_2$. La scatola ora contiene i mattoncini m_4, m_5, m_6. Si noti che questa non è una fila.
3. Inserimento di $-m_5$ (dopo m_3). Si ottiene: $-m_3$, $-m_5$, $+m_1$, $-m_2$. La scatola contiene ora i mattoncini m_4 e m_6.

L'ultima sequenza costituisce una fila F', la quale definisce la sequenza cercata $s' = \texttt{d}, \texttt{c}, \texttt{a}, \texttt{b}, \texttt{c}$. È facile verificare che non è possibile produrre s' da F con meno di 3 operazioni elementari, quindi 3 è il costo di passaggio.

Calcoliamo ora il costo di passaggio dalla fila $F = F = +m_1, -m_2, +m_3$, (che definisce la sequenza $s = \texttt{a}, \texttt{b}, \texttt{c}, \texttt{d}$) alla sequenza $s' = \texttt{b}, \texttt{a}, \texttt{c}, \texttt{d}$, avendo nella scatola i mattoncini m_4, m_5 e m_6. Possiamo togliere il mattoncino m_2 e inserire al suo posto il mattoncino m_5, ottenendo, in ordine, i mattoncini m_1, m_5, m_3. Invertendo la disposizione del mattoncino m_1 si ottiene la fila $F' = -m_1, +m_5, +m_3$, la quale definisce la sequenza di forme $s' = \texttt{b}, \texttt{a}, \texttt{c}, \texttt{d}$. Poiché non è possibile ottenere s' da F con meno di due operazioni elementari, il costo di passaggio da F a s' è 2. Si noti che l'inversione della disposizione di m_1 non incide nel calcolo del costo. □

4 Specifiche di progettazione

Si richiede di modellare il gioco con strutture dati opportune e di progettare algoritmi che permettano di eseguire efficientemente le operazioni elencate sotto.

Le scelte di modellazione e di progettazione fatte devono essere discusse nella relazione, includendo l'analisi dei costi risultanti per le diverse operazioni.

Si richiede inoltre di predisporre una rassegna *esauriente* di esempi che potrebbero essere usati per testare il programma e che mettono in evidenza particolari caratteristiche del suo funzionamento (non solo casi tipici di input, ma anche casi limite e/o situazioni patologiche, oppure input che evidenzino la differenza di prestazioni tra le soluzioni progettuali scelte e altre meno interessanti).

- **inserisciMattoncino** (α, β, σ)

  Se esiste già un mattoncino di nome σ oppure se α è uguale a β, non compie alcuna operazione. Altrimenti, inserisce nella scatola il mattoncino definito dalla tripla (α, β, σ).

- **stampaMattoncino** (σ)

  Se non esiste alcun mattoncino di nome σ non compie alcuna operazione. Altrimenti, stampa il mattoncino con nome σ, secondo il formato specificato nell'apposita sezione.

- **disponiFila** $(\pm\sigma_1, \pm\sigma_2, \ldots, \pm\sigma_n)$

  dove $\pm$ indica uno dei due simboli + o −. Verifica se nella scatola ci sono i mattoncini di nome $\sigma_1, \sigma_2, \ldots, \sigma_n$ e se la sequenza di mattoncini $\pm\sigma_1, \pm\sigma_2, \ldots, \pm\sigma_n$ costituisce una fila; in questo caso, toglie dalla scatola i mattoncini che la compongono e li dispone sul tavolo formando la fila. In caso contrario, non compie alcuna operazione.

- **stampaFila** (σ)

  Se non esiste alcun mattoncino di nome σ, oppure se il mattoncino di nome σ non appartiene ad alcuna fila sul tavolo da gioco, non compie alcuna operazione. Altrimenti, stampa la fila cui appartiene il mattoncino con nome σ, secondo il formato specificato nell'apposita sezione.

- **eliminaFila** (σ)

  Se non esiste alcun mattoncino di nome σ, oppure se il mattoncino di nome σ non appartiene ad alcuna fila sul tavolo da gioco, non compie alcuna operazione. Altrimenti, sia F la fila cui appartiene il mattoncino di nome σ. La fila F è rimossa dal tavolo e tutti i mattoncini che la compongono sono rimessi nella scatola.

- **disponiFilaMinima** (α, β)

  Crea e posiziona sul tavolo da gioco una fila di lunghezza minima da α a β. Tutti i mattoncini della fila devono essere presi dalla scatola. Se non è possibile creare alcuna fila da α a β, stampa il messaggio: `non esiste fila da` α `a` β

- **sottostringaMassima** (σ, τ)

  Stampa su una nuova riga una sottostringa massima ρ di σ e τ (se ρ è la stringa nulla, stampa una riga vuota).

- **indiceCacofonia** (σ)

  Se non esiste alcun mattoncino di nome σ oppure se il mattoncino di nome σ non appartiene ad alcuna fila, non compie alcuna operazione.

  Altrimenti stampa l'indice di cacofonia della fila cui appartiene il mattoncino di nome σ.

- **costo** ($\sigma, \alpha_1, \alpha_2, \ldots, \alpha_n$)

  Se σ non fa parte di alcuna fila, non compie alcuna operazione.

  Altrimenti, detta F la fila cui appartiene il mattoncino di nome σ, stampa il costo del passaggio da F alla sequenza di forme $s = \alpha_1, \alpha_2, \ldots, \alpha_n$ (stampa `indefinito` se il costo di passaggio è indefinito).

5 Specifiche di implementazione

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

2. Il programma deve contenere (almeno) la definizione di questi 3 tipi:

   - `gioco`, che rappresenta tutte le informazioni riguardanti lo stato del gioco (inclusi i mattoncini presenti nella scatola e le file disposte sul tavolo di gioco).
   - `mattoncino`, che rappresenta un mattoncino del gioco
   - `fila`, che rappresenta una fila di mattoncini disposti sul tavolo

3. Il programma deve contenere (almeno) cinque funzioni, corrispondenti alle prime cinque operazioni descritte nelle specifiche di progettazione e ai comandi indicati nella Tabella 1. Le funzioni devono avere queste segnature:

Linea di input	Operazione
`m` $\alpha\ \beta\ \sigma$	**inserisciMattoncino** (α, β, σ)
`s` σ	**stampaMattoncino** (σ)
`d` $\pm\sigma_1\ \pm\sigma_2\ \ldots\ \pm\sigma_n$	**disponiFila** $(\pm\sigma_1, \pm\sigma_2, \ldots, \pm\sigma_n)$
`S` σ	**stampaFila** (σ)
`e` σ	**eliminaFila** (σ)
`f` $\alpha\ \beta$	**disponiFilaMinima** (α, β)
`M` $\sigma\ \tau$	**sottostringaMassima** (σ, τ)
`i` σ	**indiceCacofonia** (σ)
`c` $\sigma\ \alpha_1\ \alpha_2 \ldots \alpha_n$	**costo** $(\sigma, \alpha_1, \alpha_2, \ldots, \alpha_n)$
`q`	Termina l'esecuzione del programma

Tabella 1: Specifiche del programma

- `inserisciMattoncino(g gioco, alpha, beta, sigma string)`
- `stampaMattoncino(g gioco, sigma string)`
- `disponiFila(g gioco, listaNomi string)`
  dove `listaNomi` è un stringa della forma $\pm\sigma_1\ \pm\sigma_2\ \ldots\ \pm\sigma_n$
- `stampaFila(g gioco, sigma string)`
- `eliminaFila(g gioco, sigma string)`

Se queste funzioni non sono presenti nel programma o hanno una segnatura diversa da quella indicata, il progetto verrà estromesso dalla valutazione.

Note

1. Non devono essere presenti vincoli sul numero di mattoncini, sul numero di file disposte sul tavolo da gioco, sulla lunghezza delle stringhe α, β, σ che definiscono i mattoncini.

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.

3. **Formato per la stampa di un mattoncino**

   Il mattoncino identificato dalla tripla (α, β, σ) va stampato in questo formato:

   $$\sigma\colon \alpha,\ \beta$$

4. **Formato per la stampa di una fila**

   Si supponga che la fila di nome σ da stampare sia

   $$\pm m_1, \pm m_2, \ldots, \pm m_n$$

   e che i mattoncini che la costituiscono siano $m_i = (\alpha_i, \beta_i, \sigma_i)$ per ogni i da 1 a n.

   Definiamo γ_i come segue:

   - se m_i è preceduto dal segno +, allora γ_i è la stringa $\sigma_i : \alpha_i,\ \beta_i$
   - se m_i è preceduto dal segno -, allora γ_i è la stringa $\sigma_i : \beta_i,\ \alpha_i$

   L'output da produrre è il seguente:

   (
   γ_1
   γ_2
   $\vdots$
   γ_n
)

6 Qualche esempio di esecuzione

6.1 Inserimento di mattoncini

Si supponga che le linee di input siano:

```
m cerchio quadrato cammello
m stella fiore elefante
m luna stella cavallo
s elefante
q
```

L'output prodotto dal programma deve essere:

```
elefante: stella, fiore
```

6.2 Disponi fila e indice di cacofonia

Si supponga che le linee di input siano:

```
m cerchio quadrato cammello
m stella fiore elefante
m luna stella cavallo
m cerchio luna cappello
m fiore quadrato mela
d +elefante +mela -cammello
S mela
i mela
q
```

L'output prodotto dal programma deve essere:

```
(
elefante: stella, fiore
mela: fiore, quadrato
cammello: quadrato, cerchio
)
6
```

6.3 Fila minima

Si supponga che le linee di input siano:

```
m cerchio quadrato cammello
m stella fiore elefante
m luna stella cavallo
m cerchio luna cappello
m fiore quadrato mela
f stella cerchio
S cavallo
q
```

L'output prodotto dal programma deve essere:

```
(
cavallo: stella, luna
cappello: luna, cerchio
)
```

6.4 Sottostringa massima

Si supponga che le linee di input siano:

```
M cappello cammello
q
```

L'output prodotto dal programma deve essere:

```
caello
```

6.5 Un esempio più lungo

Si supponga che le linee di input siano:

```
m cerchio quadrato cammello
m sole croce topolino
m stella fiore elefante
m sole stella cane
```

```
m luna stella cavallo
m cerchio luna cappello
m fiore quadrato mela
m croce fiore topo
m cuore croce tela
m croce sole tavolo
M mela tela
M cappello cammello
M cane topo
M topo topolino
M gatto topo
M tela elefante
f stella cerchio
S cavallo
i cappello
f stella cerchio
S elefante
i mela
f fiore cerchio
e cammello
f fiore cerchio
S mela
i cammello
e mela
e cappello
m cuore cerchio pesca
f croce croce
S tavolo
i topolino
f croce croce
S cammello
i pesca
q
```

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 fila 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
```