Algoritmi e Strutture Dati

Star Wars: Il Risveglio dell'Algoritmo - La gara degli sgusci (`swrace`)

Testo del problema

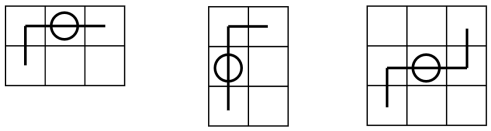
Slides originali su: `https://judge.science.unitn.it/slides/asd19/prog2.pdf`
In una galassia lontana lontana il Maestro Jedi Obi-Al Montresor e i suoi due padawan stanno esplorando l'universo in cerca di nuovi cavalieri Jedi. Anni di studi da parte dei Jedi hanno portato alla scoperta che una componente fondamentale della Forza è **l'Algoritmo:** per potere diventare dei Jedi è necessario conoscere le vie della Forza e per farlo è *necessario conoscere gli algoritmi*. La mancata conoscenza degli algoritmi porta al **Lato Oscuro della Forza**, come dice il Maestro Montresor: «Gli algoritmi inefficienti portano al Lato Oscuro. Gli algoritmi inefficienti conducono all'ira, l'ira all'odio; l'odio conduce alla sofferenza.» Per mostrare il vostro valore come Jedi, sareste sottoposti a una prova. Come *Anakin Skywalker*, dovrete partecipare a una **gara di sgusci.** Grazie all'aiuto di una vecchia conoscenza, il pilota di X-Wing Cristian Solo, vi viene fornita una mappa dell'area in cui si svolgerà la gara. In questa specialità valgono le seguenti regole:

R1) La gara si svolge su una mappa prestabilita, suddivisa in quadrati, alcuni dei quali possono contenere degli anelli.

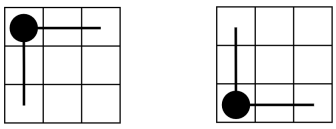
R2) Ogni anello può essere di colore bianco oppure nero.

R3) L'obiettivo è quello di creare un percorso sulla mappa che connetta più anelli possibili tra loro, formando un **percorso continuo**, **non auto-intersecante** e **preferibilmente chiuso**.

R4) Se durante il percorso si raggiunge un **anello bianco**, questo deve essere attraversato in maniera rettilinea, curvando nel quadrato immediatamente precedente **e/o** successivo. Ad esempio:



R5) Se durante il percorso si raggiunge un **anello nero**, al contrario, attraversandolo si deve curvare di novanta gradi, procedendo in modo rettilineo nei quadrati precedente **e** successivo. Ad esempio:



La gara ha inizio

Per iniziare vi verrà fornita la mappa $N \times M$ dell'area dove si svolgerà la gara, costituita da B anelli neri e W anelli bianchi, di cui vi saranno comunicate le posizioni (r, c) (riga e colonna).

Obiettivo

Il vostro compito è vincere la gara.

- Per fare ciò dovrete descrivere un percorso che rispetti le regole stabilite e attraversi il **maggior numero di anelli possibile**. Il vostro punteggiò sarà in corrispondenza con la percentuale di anelli attraversati.
- Il percorso deve passare in ogni quadrato *al massimo* una volta e deve essere continuo.
- Il percorso può essere sia **chiuso** (se alla fine della gara si torna al punto di partenza), sia **aperto** (se la gara si interrompe in un punto diverso da quello di partenza). I percorsi aperti sono concessi, ma con una penalità: il punteggio sarà dimezzato.

Note:

- Vi è concesso che i percorsi aperti possano iniziare o terminare in un anello. Quindi in tal caso, per questi anelli, è concesso non completare la richiesta delle regole.
- Vi garantiamo che esiste sempre una soluzione ottima per ogni mappa, ovvero un percorso chiuso valido che attraversa tutti gli anelli.

Esempi

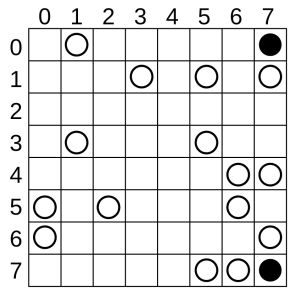


Figura 1: La mappa della gara, con i rispettivi anelli bianchi e neri.

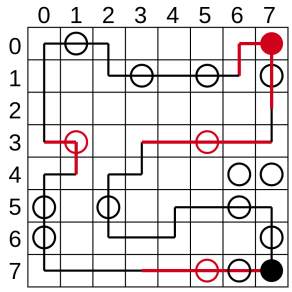


Figura 2: Output sbagliato: gli anelli evidenziati in rosso sono stati attraversati in modo invalido.

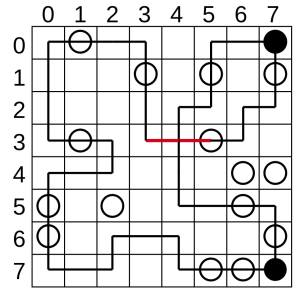


Figura 3: Output sbagliato: una casella è stata percorsa due volte.

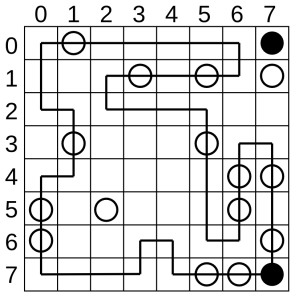


Figura 4: Soluzione parziale chiusa, 14 anelli su 17 sono stati attraversati. **Punteggio:** `4.11/5`.

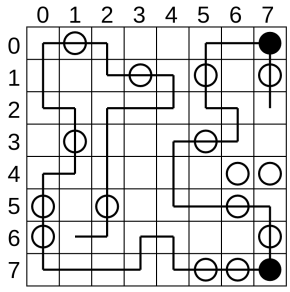


Figura 5: Soluzione parziale aperta, 15 anelli su 17 sono stati attraversati. **Punteggio:** `2.21/5`.

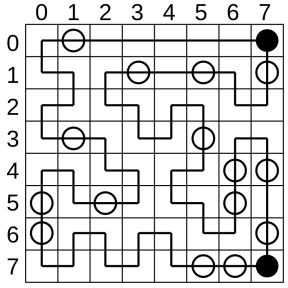


Figura 6: Soluzione completa chiusa, 17 anelli su 17 sono stati attraversati. **Punteggio:** `5/5`.

Input/Output

Input: un file con la mappa del luogo in cui si svolgerà la gara. Il file di input è costituito da $1 + B + W$ righe.

- La prima riga riporta 4 numeri, N, M, B e W, rispettivamente il numero di righe e di colonne della mappa, il numero di anelli neri ed il numero di anelli bianchi presenti.
- Le successive B righe sono costituite da 2 interi ciascuna r c, che indicano riga e colonna delle posizioni degli anelli neri.
- Le successive W righe sono costituite da 2 interi ciascuna r c, che indicano riga e colonna delle posizioni degli anelli bianchi.

Output: un file con le vostre proposte di percorsi sulla mappa, ogni proposta è composta da un'unica riga

$$A \;\; L \;\; r \;\; c \;\; d_1 d_2 d_3 \cdots d_L \#$$

dove:

- A è numero totale di anelli attraversati;
- L è la lunghezza totale del percorso effettuato;
- (r, c) è la posizione di partenza del percorso (riga e colonna);
- $d_1 \cdots d_L$ è una stringa di lunghezza L che identifica il percorso. L'i-esimo carattere della stringa fornisce la direzione in cui muoversi all'i-esimo step. Il carattere deve essere uno tra U, D, L, R (up, down, left, right);
- Ogni soluzione deve essere terminata da un cancelletto #.

Punteggio

- Ci sono 20 casi di test: ogni test assegna un punteggio di massimo 5 punti.
- Un percorso proposto è valido se rispetta tutte le regole richieste. Percorsi non validi fanno **zero punti**!
- I percorsi validi **chiusi** ottengono questo punteggio:

$$5 \cdot \left(\frac{A}{B + W} \right),$$

dove A è il numero di anelli attraversati.

- I percorsi validi **non chiusi** ottengono questo punteggio:

$$5 \cdot \left(\frac{A}{(B+W)} \right) \cdot \frac{1}{2},$$

  dove A è il numero di anelli attraversati.
- Per ogni file di output viene calcolato il punteggio dato dall'**ultimo** percorso presente nel file che termina con un cancelletto #.

Esempi (punteggio)

Con riferimento all'input dato in Figura 1, ecco due esempi di calcolo del punteggio:

1. Nell'esempio in Figura 4 è presentata una soluzione parziale chiusa: sono stati attraversati $A = 14$ anelli su $B + W = 17$, quindi il punteggio sarà $5 \cdot \frac{14}{17} = 4.11$ su un massimo di 5.
2. Nell'esempio in Figura 5 è presentata una soluzione parziale aperta: sono stati attraversati $A = 15$ anelli su $B + W = 17$, quindi il punteggio sarà $5 \cdot \frac{15}{17} \cdot \frac{1}{2} = 2.21$ su un massimo di 5.
3. Nell'esempio in Figura 6 è presentata una soluzione ottima chiusa: sono stati attraversati $A = 17$ anelli su $B + W = 17$, quindi il punteggio sarà $5 \cdot \frac{17}{17} = 5$ su un massimo di 5.

Valutazione

Per valutazione del progetto:

- Conta il punteggio dell'**ultimo sorgente** inviato al sistema;
- Il progetto è superato con un punteggio non inferiore a 30 punti;
- C'è un limite di 80 sottoposizioni per gruppo;

Limiti e assunzioni

Limiti generali

- $1 < N, M \leq 256$
- $1 < B, W \leq 8000$
- $0 \leq r < N, 0 \leq c < M$, per ogni posizione (r, c) di un anello.

Casi di test

- Ci sono 20 casi di test in totale.
- In almeno 6 casi su 20 esiste un percorso rettangolare come soluzione ottima.
- In almeno 10 casi su 20 $1 < B, W \leq 20$.

Limiti delle risorse

- Tempo di esecuzione: 3 secondi
- Memoria: 16 MB

Dataset di esempio

Per gli input forniti nel dataset di esempio non è stata calcolata una soluzione ottima. Per questo motivo il dataset non contiene anche i relativi output, solitamente messi a disposizione.

Istruzioni di compilazione

Di seguito riportiamo le istruzioni per testare i vostri progammi su vari sistemi. Si suppone che il sorgente con il vostro codice si chiami file `swrace.cpp`. I file `swrace.cpp`, `grader.cpp` e `swrace.h` devo stare nella stessa cartella.

Sistemi GNU/Linux

```
/usr/bin/g++ -DEVAL -std=c++11 -O2 -pipe -static -s -o swrace swrace.cpp grader.cpp
```

Sistemi Mac OS X

Su sistemi Mac OS X usate il seguente comando di compilazione:

```
/usr/bin/g++ -DEVAL -std=c++11 -O2 -pipe -o swrace swrace.cpp grader.cpp
```

Se ottente un errore del tipo: `use of undeclared identifier quick_exit`, sostituite in `grader.cpp` l'istruzione `quick_exit(EXIT_SUCCESS);` con `exit(EXIT_SUCCESS);`.

Sistemi Windows

Per il sistema Windows 10 potete installare il "Windows Subsystem for Linux"[1]. Successivamente potete installare i tool necessari per usare Visual Studio Code[2] o Visual Studio 2017[3] seguendo le relative guide riportate nelle note. Usando questo sistema fate attenzione a dove salvate i file e a quale nome gli date in quanto potreste avere delle difficoltà con percorsi che contengano spazi e caratteri speciali.
In alternativa, o per sistemi precedenti a Windows 10 potete installare *Cygwin*[4], un ambiente completamente POSIX-compatibile per Windows. Anche in questo caso esistono guide per configurare i comuni editor disponibili su Windows di modo che utilizzino l'ambiente Cygwin, come per esempio Visual Studio[5].
Una volta installato Cygwin è possibile simulare quanto avviane su arena compilando il proprio sorgente senza includere l'header `swrace.h` e il grader `grader.cpp`:

```
/usr/bin/g++ -DEVAL -std=c++11 -O2 -pipe -static -s -o swrace swrace.cpp
```

e lanciare il comando come:

```
timeout.exe 2 ./swrace
```

`timeout.exe` arresterà il programma dopo 3 secondi.

[1] https://docs.microsoft.com/en-us/windows/wsl/install-win10
[2] https://code.visualstudio.com/docs/cpp/config-wsl
[3] https://devblogs.microsoft.com/cppblog/targeting-windows-subsystem-for-linux-from-visual-studio/
[4] https://www.cygwin.com/
[5] https://devblogs.microsoft.com/cppblog/using-mingw-and-cygwin-with-visual-cpp-and-open-folder/

Esempi di input/output

File `input.txt`	File `output.txt`
8 8 2 15 0 7 7 7 7 5 1 7 1 3 3 5 6 0 5 0 6 7 0 1 5 2 4 7 3 1 5 6 7 6 4 6 1 5	17 64 0 0 RRRRRRRDDLULLLLDRDRURDDLDRDRU UURDDDDLLLULDLULDLUUURDRRULULLURULU#

File `input.txt`	File `output.txt`
16 16 10 21 9 14 3 11 0 11 9 11 14 0 12 11 12 8 14 8 3 14 4 0 10 11 11 11 0 10 13 8 1 11 12 9 8 14 14 1 9 12 5 0 12 10 4 1 3 13 4 14 2 3 0 5 9 13 3 12 2 11 14 7 13 0	31 56 0 4 RRRRRRRDDDRRRDDDDDDLLLDDDLLLD DLLLLLLLLUUUUUUUUUURRURUURU#

File `input.txt`	File `output.txt`

`5 5 4 3`
`0 1`
`4 4`
`4 1`
`0 4`
`0 3`
`3 4`
`1 4`
	`7 14 0 1 DDDDRRRUUUULLL#`