Laboratorio di Algoritmi e Strutture Dati

Docenti: M. Goldwurm, S. Aguzzoli

Appello del 4 luglio 2002

Progetto "Filotto"
Consegna entro il 22 luglio 2002

Il problema

Obiettivo del progetto è lo studio delle configurazioni di simboli che possono verificarsi durante una partita di *Filotto Generalizzato*.

Il terreno di gioco è costituito da una griglia bidimensionale di $n \times n$ caselle. A turno, ognuno dei k giocatori occupa una delle caselle libere con il suo simbolo. Vince il giocatore che per primo realizza una *fila* (orizzontale, verticale o diagonale) di h dei suoi simboli. Ad ogni nuova partita occorre fissare la dimensione n del piano, il numero k di giocatori, la lunghezza vincente $h \geq 3$.

Formalmente, il *piano di gioco* di dimensione n è l'insieme dei punti

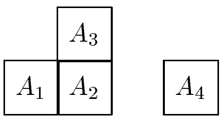
$$\text{Piano}(n) \; = \; \{ \, (x, y) \in \mathbf{N} \times \mathbf{N} \quad | \quad 0 \leq x \leq n, \; 0 \leq y \leq n \}.$$

Una *casella* è un quadrato nel piano il cui lato ha dimensione unitaria; più precisamente, data una coppia di naturali (a, b), una casella in *posizione* (a, b) è data dall'insieme dei punti (vertici del quadrato):

$$\text{Casella}(a, b) \; = \; \{ \, (x, y) \quad | \quad a \leq x \leq a+1, \; b \leq y \leq b+1 \}.$$

Ad esempio, $\text{Casella}(3, 5) \; = \; \{(3, 5), (3, 6), (4, 6), (4, 5)\}$ A ogni casella è associato un valore intero in $\{0, 1, \ldots, k\}$. Il valore 0 indica che la casella è vuota, mentre un valore $0 < m \leq k$ indica che la casella è occupata dal giocatore m.

Diciamo che due caselle sono *adiacenti* se hanno in comune almeno un punto (quindi, ogni casella è adiacente a se stessa); nella figura qui sotto le caselle A_1 e A_2 sono adiacenti, A_2 e A_3 sono adiacenti, e anche A_1 e A_3 sono adiacenti. Al contrario A_4 non è adiacente ad alcuna delle altre tre caselle.



Un *cammino* da A_1 a A_h è una sequenza di caselle occupate (ossia, contenenti un valore diverso da 0) $A_1, \ldots, A_h$ tali che A_i è adiacente a A_{i+1} per $1 \leq i \leq h-1$. Diciamo che un insieme $\mathcal{P}$ di caselle occupate è *connesso* se, per ogni $A, B \in \mathcal{P}$, esiste un cammino da A a B contenuto in $\mathcal{P}$. Un *blocco* è un insieme di caselle occupate $\mathcal{A}$ tale che $\mathcal{A}$ è connesso e, per ogni casella occupata $P \notin \mathcal{A}$, $\mathcal{A} \cup \{P\}$ non è connesso.

Dato un intero $0 < m \leq k$, una *m-fila* è un cammino $\mathbf{F}$ contenente almeno *tre* caselle tale che:

1. Tutte le caselle in $\mathbf{F}$ hanno valore m (vale a dire, sono tutte occupate dal giocatore m).

2. Le caselle in $\mathbf{F}$ sono disposte in uno dei seguenti modi:

(a) *Orizzontalmente*: esistono due interi x e y e un intero i tali che l'insieme delle coordinate delle caselle nella fila è descritto da $\{(x+j, y) \mid 0 \leq j < i\}$.

(b) *Verticalmente*: esistono due interi x e y e un intero i tali che l'insieme delle coordinate delle caselle nella fila è descritto da $\{(x, y+j) \mid 0 \leq j < i\}$.

(c) *Diagonalmente*: esistono due interi x e y e un intero i tali che l'insieme delle coordinate delle caselle nella fila è descritto da $\{(x+j, y+j) \mid 0 \leq j < i\}$ oppure da $\{(x+j, y-j) \mid 0 \leq j < i\}$.

Una m-fila è *vincente* se la sua lunghezza (ossia, il numero delle caselle che la compongono) è $\geq h$.

Una casella C è *buona* per il giocatore m rispetto a un blocco $\mathbf{B}$ se C è libera (il suo valore è 0) ed è tale che assegnando a C il valore m, si viene a creare almeno una nuova m-fila $\mathbf{F}$ (eventualmente allungando una m-fila già presente o unendo due m-file preesistenti) contenente la casella C e almeno una delle caselle appartenenti al blocco $\mathbf{B}$.

Una casella è buona per il giocatore m se è buona per m rispetto ad almeno un blocco.

Una casella C è *cattiva* per il giocatore m se essa è buona per qualche giocatore p diverso da m.

Si consideri l'esempio seguente dove $n = 12$, $k = 3$, $h = 4$ (si osservi che la situazione descritta non può verificarsi in una partita reale, a causa del diverso numero di mosse effettuate dai 3 giocatori. Questo aspetto del gioco è trascurato nel nostro progetto).

									$\boxed{1}$		
									1		
									1		
						$\boxed{2}$	$\boxed{3}$		$\boxed{*}$		$\boxed{3}$
					2	$\boxed{2}$	**2**	**3**	**3**	**3**	$\boxed{3}$
		$\boxed{1}$	**3**	**2**			**3**	**1**	**3**		
		1	**2**			$\boxed{3}$		$\boxed{3}$	$\boxed{3}$	$\boxed{3}$	
		1									
		1									
		$\boxed{1}$									

Le caselle prive di bordo e contenenti un numero rappresentano le caselle occupate dal giocatore corrispondente. Tutte le altre caselle (comprese quelle bordate) sono libere. Osserviamo che sono presenti tre blocchi, contenenti tutte e sole le caselle le cui coordinate sono descritte nei tre insiemi seguenti

$$\mathbf{B_1} = \{(2,2), (2,3), (3,3), (3,4), (4,4), (5,5)\}$$
$$\mathbf{B_2} = \{(7,4), (7,5), (8,4), (8,5), (9,4), (9,5), (10,5)\}$$
$$\mathbf{B_3} = \{(9,7), (9,8)\}$$

Si noti che, ad esempio, $\{(3,3), (4,4), (5,5)\}$ è una 2-fila diagonale di lunghezza 3, $\{(2,1), (2,2), (2,3)\}$ è una 1-fila verticale di lunghezza 3, mentre non vi sono file vincenti. Ogni casella bordata contenente m è buona per il giocatore m. Ad esempio, la casella $(2,0)$ è buona per 1 rispetto al blocco $\mathbf{B_1}$, in quanto allunga la 1-fila verticale $\{(2,1), (2,2), (2,3)\}$ producendo la nuova 1-fila $\{(2,0), (2,1), (2,2), (2,3)\}$, che è vincente; la casella contrassegnata da $*$ è buona per diversi giocatori: infatti la sua occupazione da parte del giocatore 3 produce la 3-fila $\{(7,4), (8,5)\,(9,6)\}$ e la 3-fila $\{(9,4), (9,5), (9,6)\}$ mentre la sua occupazione da parte del giocatore 1 produce la 1-fila $\{(9,6), (9,7), (9,8)\}$. Si noti che l'occupazione da parte del giocatore 2 della casella $(6,6)$, che allunga la 2-fila $\{(3,3), (4,4), (5,5)\}$, riduce il numero di blocchi (essendo tale casella adiacente sia a $\mathbf{B_1}$ che a $\mathbf{B_2}$), inoltre produce la 2-fila vincente $\{(3,3), (4,4), (5,5), (6,6)\}$. Per concludere, si osservi che la casella di coordinate $(6,5)$ è buona per il giocatore 2 sia rispetto al blocco $\mathbf{B_1}$ che al blocco $\mathbf{B_2}$ pur non allungando alcuna 2-fila preesistente, ma creandone una nuova di lunghezza 3.

Si richiede di implementare una struttura dati efficiente che permette di eseguire le operazioni seguenti (si tenga presente che la dimensione del piano n può essere grande rispetto al numero di caselle occupate, quindi *non è sicuramente efficiente rappresentare il piano mediante una bit-map di tipo* $n \times n$).

- **crea (n, k, h)**

  Crea un piano di gioco vuoto di dimensione **n** (se un piano è già definito viene eliminato). Stabilisce che vi sono **k** giocatori e che la lunghezza vincente è **h**.

- **inserisci (m, x, y)**

  Se Casella(**x, y**) ha valore 0 (è libera) assegna a Casella(**x, y**) il valore **m** ($0 < m \leq k$), altrimenti non compie alcuna operazione.

- **elimina (x, y)**

  Assegna a Casella(**x, y**) il valore 0.

- **listacaselle (m)**

  Fornisce la lista di tutte le caselle occupate dal giocatore m, secondo le specifiche sotto riportate.

- **suggerisci (m)**

  Indica, secondo le specifiche sotto riportate, la lista delle caselle *buone* per il giocatore m.

- **suggerisci (x, y)**

  Detto **m** il giocatore che occupa Casella(**x, y**), indica, secondo le specifiche sotto riportate, la lista delle caselle *buone* per m rispetto al blocco cui appartiene Casella(**x, y**).
  Se invece Casella(**x, y**) ha valore 0 (è libera), allora non compie alcuna operazione.

- **blocca(m)**

  Indica, secondo le specifiche sotto riportate, la lista delle caselle *cattive* per il giocatore **m**.

- **vince(m)**

  Restituisce 1 se esiste almeno una m-fila vincente, 0 altrimenti.

Specifiche di implementazione

Il programma deve leggere dallo standard input (`stdin`) una sequenza di linee (separate da `\n`), ciascuna delle quali corrisponde a una linea della prima colonna della Tabella 1, dove x, y, z sono numeri naturali e i vari elementi sulla linea sono separati da uno o più spazi. Quando una linea 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 linea.

Si noti che non devono essere presenti vincoli sulla dimensione del piano e sul numero di celle (se non quelli determinati dal tipo di dato intero). 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()`.

Formato per la visualizzazione delle liste di caselle

Nei casi in cui l'output richiede un elenco di caselle, il formato a cui attenersi è il seguente.

LINEA DI INPUT	OPERAZIONE
`c` x y z	**crea (x, y, z)**
`i` x y z	**inserisci (x, y, z)**
`e` x y	**elimina (x, y)**
`l` x	Stampa l'output di **listacaselle (x)**
`s` x	Stampa l'output di **suggerisci (x)**
`t` x y	Stampa l'output di **suggerisci (x, y)**
`b` x	Stampa l'output di **blocca (x)**
`v` x	Stampa il valore di **vince (x)**
`f`	Stampa `f` e termina l'esecuzione del programma

Tabella 1: Specifiche del programma

- La prima riga è costituita da una copia della riga di input corrispondente.
- Le righe successive descrivono la lista delle caselle secondo la specifica seguente. Assumiamo che l'insieme delle caselle da visualizzare sia

$$\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$$

dove, per ogni $i \in \{1, \ldots, n-1\}$, valga $(x_i, y_i) < (x_{i+1}, y_{i+1})$ rispetto all' ordine stretto $<$ definito da:

$$(x_i, y_i) < (x_{i+1}, y_{i+1}) \qquad \text{se e solo se} \qquad x_i < x_{i+1} \quad \text{oppure} \quad x_i = x_{i+1} \text{ e } y_i < y_{i+1}.$$

L'output deve essere:

```
x1 y1
x2 y2
..
xn yn
```

Esempio. Si supponga che le linee di input siano:

```
c 20 3 4
i 1 3 3
i 1 4 4
i 2 5 5
i 1 3 5
s 1
i 3 4 7
```

```
i 3 5 7
i 3 6 7
b 2
e 4 4
i 1 3 2
t 3 5
s 1
v 3
i 3 3 7
v 3
f
```

L'output prodotto dal programma deve essere:

```
s 1
2 2
2 6
3 4
5 3
b 2
2 2
2 6
3 4
3 7
5 3
7 7
t 3 5
3 4
s 1
3 1
3 4
0
1
f
```

Presentazione del progetto

Il progetto deve essere inviato per posta elettronica all'indirizzo `aguzzoli@dsi.unimi.it` entro il 22 luglio 2002. La discussione del progetto e l'esame orale si svolgeranno il 25 luglio 2002 in luogo e orario da stabilirsi.

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 analizza il costo delle diverse operazioni.

I due file devono essere contenuti in un unico file `.zip` il cui nome dovrà essere `cognome.zip`. La relazione e il codice devono riportare il vostro nome, cognome e matricola.

Una copia cartacea della relazione e del codice deve inoltre essere consegnata al dr. Aguzzoli sempre entro il 22 luglio 2002 (lasciandola eventualmente nella sua casella postale presso il dipartimento in via Comelico).

Si ricorda infine di presentarsi alla prova orale con una copia stampata della relazione e del codice.

Per ogni ulteriore chiarimento:
E-mail: `aguzzoli@dsi.unimi.it`
Ricevimento: il mercoledì, ore 15-16, stanza S211.

Avvisi

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

Lo svolgimento del progetto è una prova d'esame da svolgere *individualmente*.