Disegni di Mondrian (`disegno`)

Al museo d'arte moderna di Bergamo sta per aprire una nuova mostra! Il pittore protagonista sarà *Piet Mondrian*, famoso per i suoi quadri geometrici.

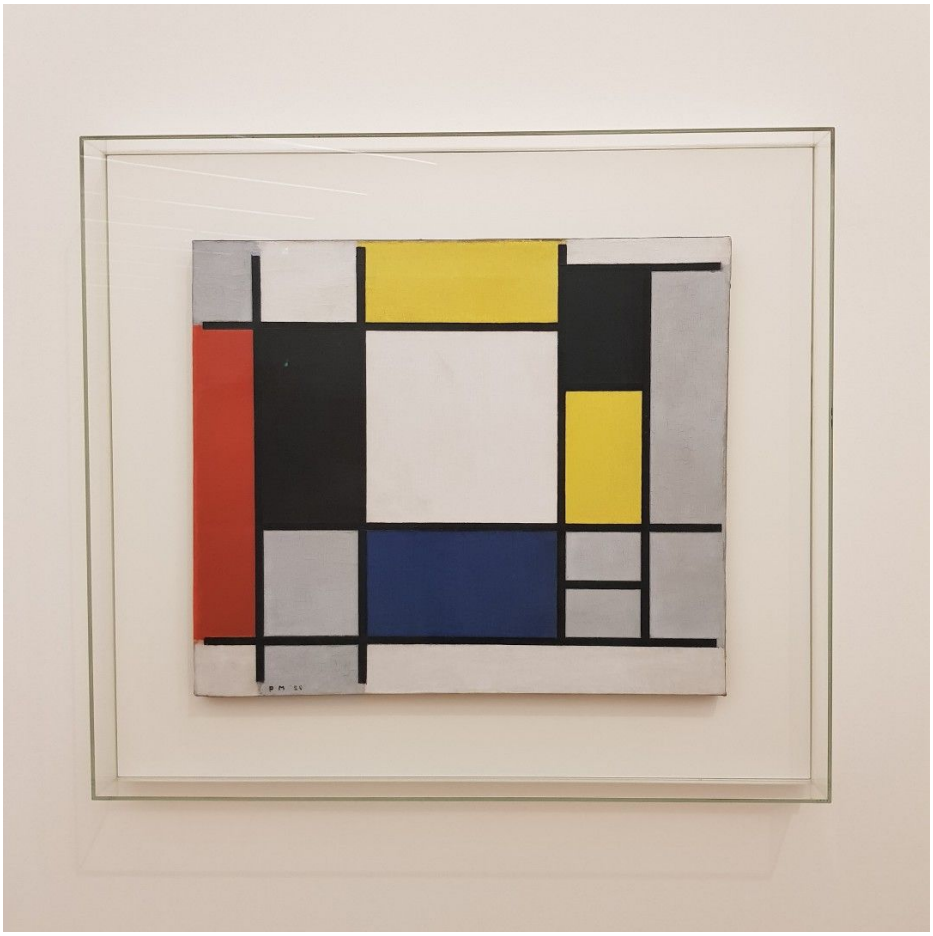


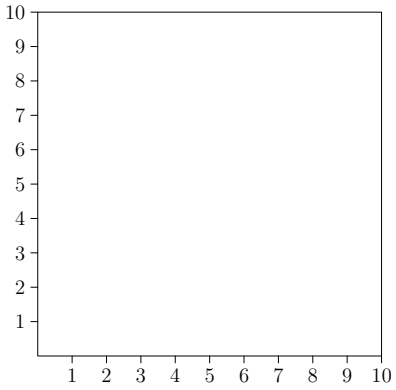
Figura 1: Un vero quadro di Mondrian.

Le opere sono appena arrivate e Filippo Vasarin, noto critico d'arte delle OII[1], ha il compito di assicurarsi che siano tutte autentiche. Infatti, dopo anni di studi, è l'unico esperto che ha capito perfettamente qual era la procedura di lavoro che Mondrian applicava a ogni suo quadro.

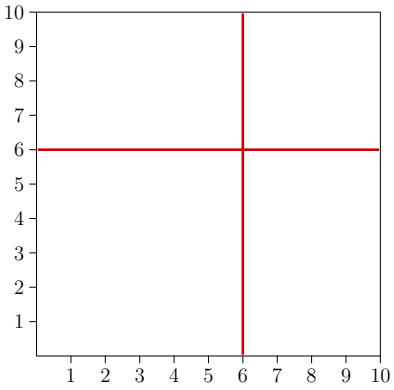
Mondrian partiva sempre con una tela quadrata bianca di un certo lato L, poi eseguiva una serie di *mosse*. In ciascuna mossa, sceglieva un punto interno al quadrato (a distanza intera da tutti i bordi della tela) che non appartenesse a nessun segmento già disegnato. Poi, tracciava il più lungo segmento orizzontale e il più lungo segmento verticale che passassero per quel punto e che non intersecassero altri segmenti se non agli estremi. Il punto fissato diventava quindi il centro di una croce (coi bracci non necessariamente di uguale lunghezza, ma ciascuno della massima lunghezza possibile rispettati i criteri).

Eseguito un certo numero di mosse Mondrian, soddisfatto, iniziava a colorare i rettangoli così creati. Di seguito trovi un esempio di realizzazione di un disegno di Mondrian.

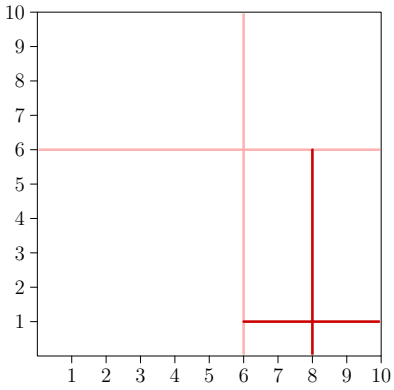
[1] *Oii Italiane di Informatica* (OII[1]).



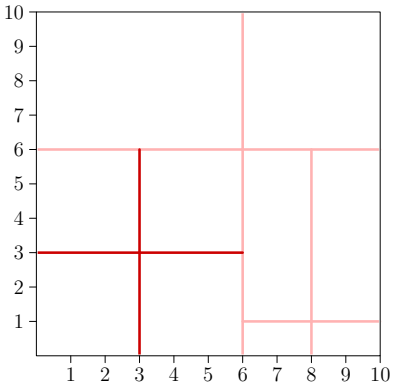
(a) Inizio (quadrato vuoto)



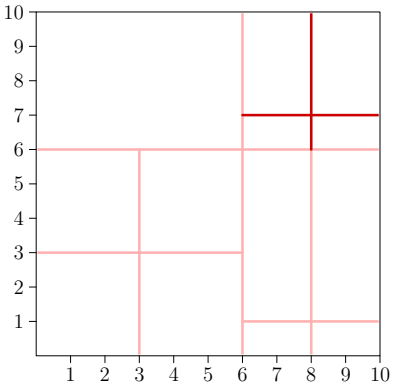
(b) Prima *mossa*



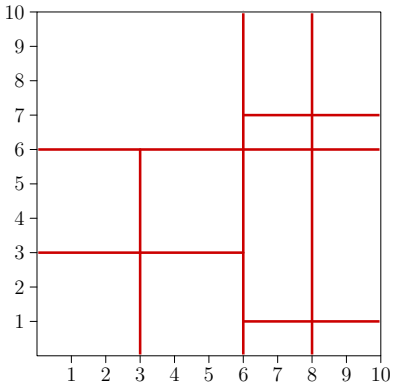
(c) Seconda *mossa*



(d) Terza *mossa*



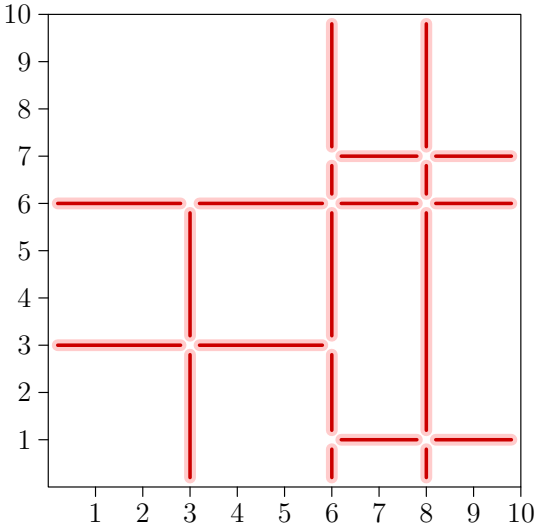
(e) Quarta *mossa*



(f) Situazione finale

Assumendo che nessuno sia mai riuscito a imitare Mondrian, ti è assegnato il compito di determinare se un quadro è un originale o un falso. Per provare che è un quadro è un originale, devi anche trovare la sequenza di mosse eseguite da Mondrian per realizzarlo.

Formalmente, ignoriamo i colori e identifichiamo un disegno con l'insieme di segmenti massimali che lo compongono. Un segmento massimale è un segmento che interseca altri segmenti solo (al più) agli estremi e che sia della lunghezza massima possibile. Nell'esempio di prima, il quadro corrisponde ai segmenti evidenziati in figura.



Dato l'elenco di segmenti massimali, aiuta Filippo a trovare la sequenza di mosse che produce quel quadro!

Implementazione

Dovrai sottoporre un unico file, con estensione `.cpp`.

> ☞ Tra gli allegati a questo task troverai un template `disegno.cpp` con un esempio di implementazione.

Il template e il grader includono un header `disegno.h` in cui è definita la `struct Point`. Questa struttura contiene i campi `x` e `y` e rappresenta un punto di coordinate $(\texttt{x}, \texttt{y})$.

> ✍ Per assicurarti che la tua sottoposizione compili correttamente, ti consigliamo di usare il template allegato.

Dovrai implementare la seguente funzione:

C++
```
vector<Point> draw(int N, int L, vector<Point> A, vector<Point> B);
```

- L'intero N è il numero di segmenti.
- L'intero L è la lunghezza del lato del quadrato.
- I vettori A e B, di dimensione N, descrivono i segmenti.
  Per ogni $i = 0, \ldots, N-1$:
  - se l'i-esimo segmento è orizzontale e collega i punti (x_1, y) e (x_2, y), con $x_1 < x_2$, si ha $A[i].\texttt{x} = x_1$, $A[i].\texttt{y} = y$, $B[i].\texttt{x} = x_2$, e $B[i].\texttt{y} = y$;
  - se l'i-esimo segmento è verticale e collega i punti (x, y_1) e (x, y_2), con $y_1 < y_2$, si ha $A[i].\texttt{x} = x$, $A[i].\texttt{y} = y_1$, $B[i].\texttt{x} = x$, e $B[i].\texttt{y} = y_2$.
- Se non è possibile realizzare il disegno descritto dall'input tramite una sequenza di mosse valide, la funzione deve restituire un **vettore vuoto**. Altrimenti, la funzione deve restituire un vettore di mosse che, se effettuate nell'ordine dato, producono il disegno in input. Ogni mossa è descritta da una `struct Point` che contiene le coordinate del centro della mossa.

Il grader chiamerà la funzione `draw` e stamperà in output le informazioni restituite.

Grader di prova

Nella directory relativa a questo problema è presente una versione semplificata del grader usato durante la correzione, che potete usare per testare le vostre soluzioni in locale. Il grader di esempio legge i dati da `stdin`, chiama la funzione che dovete implementare e scrive su `stdout`, secondo il seguente formato.

Il file di input è composto da $N + 1$ righe, contenenti:

- Riga 1: gli interi N ed L;
- Riga $2 + i$ $(0 \leq i < N)$: gli interi $A[i].\texttt{x}$, $A[i].\texttt{y}$, $B[i].\texttt{x}$, e $B[i].\texttt{y}$.

Se la funzione `draw` restituisce un vettore vuoto, il file di output è composto da una sola riga contenente l'intero 0. Altrimenti, sia $M > 0$ la lunghezza del vettore restituito. Il file di output è composto da $M+1$ righe, contenenti:

- Riga 1: l'intero M;
- Riga $2 + i$ $(0 \leq i < M)$: i campi `x` e `y` dell'i-esimo elemento del vettore restituito.

Assunzioni

- $4 \leq N \leq 1\,000\,000$.
- $2 \leq L \leq 1\,000\,000\,000$.
- $0 \leq A[i].\texttt{x} \leq L$, $0 \leq A[i].\texttt{y} \leq L$ per ogni $i = 0, \ldots, N-1$.
- $0 \leq B[i].\texttt{x} \leq L$, $0 \leq B[i].\texttt{y} \leq L$ per ogni $i = 0, \ldots, N-1$.
- Per ogni $i = 0, \ldots, N-1$, si ha esattamente una delle seguenti:
  - $A[i].\texttt{y} = B[i].\texttt{y}$ e $A[i].\texttt{x} < B[i].\texttt{x}$ (segmento orizzontale).
    In questo caso, vale ulteriormente $0 < A[i].\texttt{y} < L$.
  - $A[i].\texttt{x} = B[i].\texttt{x}$ e $A[i].\texttt{y} < B[i].\texttt{y}$ (segmento verticale).
    In questo caso, vale ulteriormente $0 < A[i].\texttt{x} < L$.
- Ogni estremo di un segmento orizzontale è anche l'estremo di un segmento verticale, oppure appartiene a un lato del quadrato. Ogni estremo di un segmento verticale è anche l'estremo di un segmento orizzontale, oppure appartiene a un lato del quadrato.
- Due segmenti distinti possono intersecarsi solo agli estremi.

Assegnazione del punteggio

Il tuo programma verrà testato su diversi test case raggruppati in subtask.

Il punteggio di un subtask è il minimo punteggio ottenuto su uno dei test case che lo compongono, moltiplicato per il punteggio del subtask.

Diciamo che una sequenza di M mosse è *variegata* se le coordinate x dei loro centri sono M interi distinti, e le coordinate y dei loro centri sono M interi distinti.

Diciamo che un disegno è una *griglia completa* se per ogni $0 < x < L$ e $0 \leq y < L$ esiste un segmento i tale che $A[i] = (x, y)$ e $B[i] = (x, y+1)$, e per ogni $0 \leq x < L$ e $0 < y < L$ esiste un segmento j tale che $A[j] = (x, y)$ e $B[j] = (x+1, y)$.

- **Subtask 1 [0 punti]**: Casi d'esempio.
- **Subtask 2 [10 punti]**: Il disegno è una *griglia completa*.
- **Subtask 3 [8 punti]**: $N \leq 40$, $L \leq 8$.
- **Subtask 4 [10 punti]**: $N \leq 1000$, $L \leq 500$.
- **Subtask 5 [22 punti]**: $N \leq 30\,000$, $L \leq 10\,000$ e le mosse che realizzano il disegno sono *variegate*.
- **Subtask 6 [17 punti]**: Ogni sequenza di mosse valide che realizza il disegno è *variegata*.
- **Subtask 7 [20 punti]**: $N \leq 30\,000$, $L \leq 10\,000$.
- **Subtask 8 [13 punti]**: Nessuna limitazione aggiuntiva.

Esempi di input/output

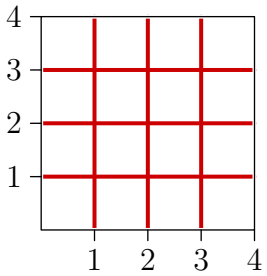
stdin	stdout
21 10 6 6 8 6 0 6 3 6 6 7 8 7 8 1 8 6 6 1 6 3 8 1 10 1 6 1 8 1 3 6 6 6 8 7 10 7 3 3 3 6 8 7 8 10 3 0 3 3 6 3 6 6 6 7 6 10 8 0 8 1 6 6 6 7 8 6 8 7 6 0 6 1 8 6 10 6 3 3 6 3 0 3 3 3	4 6 6 8 1 3 3 8 7
24 4 0 1 1 1 1 1 2 1 2 1 3 1 3 1 4 1 0 2 1 2 1 2 2 2 2 2 3 2 3 2 4 2 0 3 1 3 1 3 2 3 2 3 3 3 3 3 4 3 1 0 1 1 1 1 1 2 1 2 1 3 1 3 1 4 2 0 2 1 2 1 2 2 2 2 2 3 2 3 2 4 3 0 3 1 3 1 3 2 3 2 3 3 3 3 3 4	5 2 2 1 1 3 1 1 3 3 3

stdin	stdout
5 5 2 4 5 4 2 4 2 5 1 0 1 4 1 4 2 4 0 4 1 4	0
11 8 5 0 5 3 5 3 5 6 1 3 5 3 1 0 1 3 5 6 8 6 1 6 5 6 1 6 1 8 0 3 1 3 5 3 8 3 1 3 1 6 0 6 1 6	0

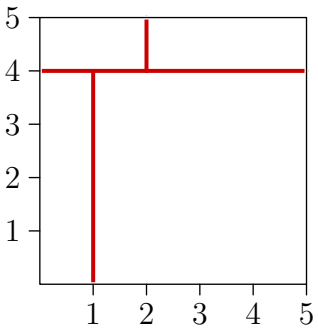
Spiegazione

Il **primo caso di esempio** è quello presentato nella descrizione del problema, e l'output rappresenta la sequenza di mosse lì descritta.

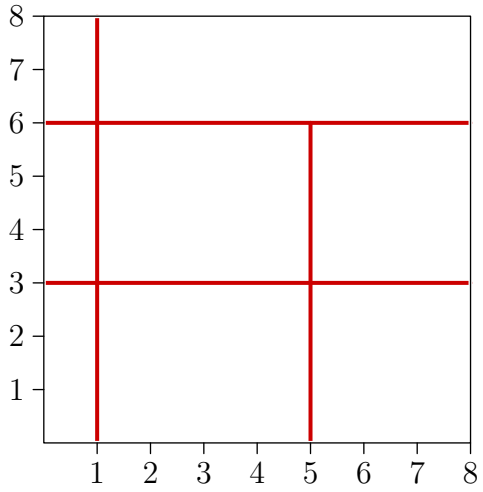
Nel **secondo caso di esempio**, gli $N = 24$ segmenti formano una griglia completa 4×4:



Il **terzo caso di esempio** corrisponde al seguente disegno, che non è ottenibile tramite mosse valide:



Il **quarto caso di esempio** corrisponde al disegno mostrato qui sotto. Anche in questo caso, si può mostrare che esso non è ottenibile tramite una sequenza di mosse valide.



Note

Tra gli allegati del problema troverai alcuni script utili.

- `visualizer.py` è uno script per visualizzare un input. Si aprirà una finestra, contenente una rappresentazione del quadrato e dei segmenti in input. Puoi eseguirlo con:

  ```
  python3 visualizer.py < input.txt
  ```

- `designer.py` è uno script per generare input attraverso un'interfaccia grafica. Si aprirà una finestra e avrai la possibilità di disegnare i segmenti e ottenere l'input corrispondente. Puoi eseguirlo con:

  ```
  python3 designer.py
  ```