# Laboratorio di Algoritmi e Strutture Dati

Docenti: M. Goldwurm, S. Aguzzoli

Appello del 15 Settembre 2005

Progetto "L'ape bottinatrice"
Consegna entro il 3 Ottobre 2005

## Il problema

Un'ape si muove nel piano alla ricerca di prati fioriti onde poter raccogliere nettare per produrre miele di qualità per l'alveare.

Formalmente, chiamiamo *piano* l'insieme dei punti

$$\{\ (x, y) \in \mathbb{Z} \times \mathbb{Z}\ \}.$$

Dati $x_0, y_0, x_1, y_1$ interi, denotiamo con $R(x_0, y_0, x_1, y_1)$ il rettangolo

$$R(x_0, y_0, x_1, y_1) \ = \ \{(x, y) \in \mathbb{Z} \times \mathbb{Z} \ \mid \ x_0 \leq x \leq x_1,\, y_0 \leq y \leq y_1\}.$$

Un *prato fiorito* è un rettangolo $R(x_0, y_0, x_1, y_1)$ in cui ogni punto $(x, y) \in R(x_0, y_0, x_1, y_1)$ è occupato da un *fiore*, caratterizzato a sua volta da un valore intero $v(x, y)$ (eventualmente negativo) che rappresenta la qualità del nettare del fiore stesso estraibile dall'ape (un valore negativo denota nettare di scarsa qualità, e quindi tendenzialmente trascurabile). Due prati possono sovrapporsi, anche solo parzialmente. Se un punto appartiene a due o più prati, il valore del fiore ivi collocato è dato dalla somma algebrica dei valori che il fiore ha in ciascun prato a cui appartiene.

L'ape pianifica dei viaggi nel piano allo scopo di massimizzare la qualità del nettare raccolto. L'ape può muoversi da un fiore in posizione $p = (x_p, y_p)$ a un fiore in posizione $q = (x_q, y_q)$ se e solo se:

- $y_q = y_p + 1$;
- $x_q = x_p - 1$ oppure $x_q = x_p$ oppure $x_q = x_p + 1$.

Un *viaggio* dai punti $p_1$ a $p_u$ consiste in una sequenza di punti $p_1, p_2, \ldots, p_u$ dove:

- Per ogni $i \in \{1, \ldots, u\}$, $p_i$ è occupato da un fiore.
- Per ogni $i \in \{1, \ldots, u - 1\}$, l'ape può muoversi da $p_i$ a $p_{i+1}$.

La *qualità* $Q$ del viaggio $p_1, p_2, \ldots, p_u$ è data dalla somma dei valori associati ai singoli fiori che lo compongono, vale a dire:

$$Q \ = \ \sum_{i=1}^{u} v(p_i)\,.$$

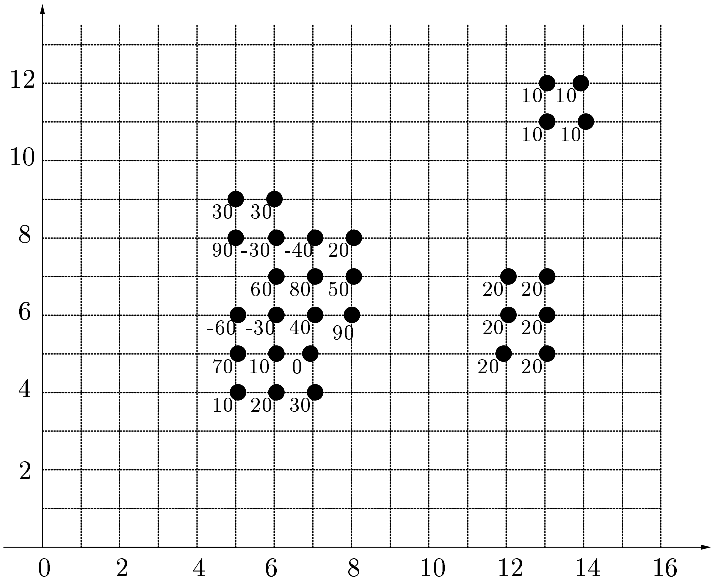
Ovviamente tra due punti $p_1$ e $p_u$ del piano possono esistere zero, uno o più viaggi. Quando più viaggi sono possibili, l'ape dovrà sceglierne uno di qualità massima.

**Esempio**

Si supponga che nel piano vi siano i prati

$P_1 \ = \ R(5, 4, 7, 6) \quad P_2 \ = \ R(6, 6, 8, 8) \quad P_3 \ = \ R(5, 8, 6, 9) \quad P_4 \ = \ R(12, 5, 13, 7) \quad P_5 \ = \ R(13, 11, 14, 12)$

e che i fiori abbiano i valori indicati nella figura.




Dal punto (12,5) al punto (12,7) esistono due viaggi ottimali $V_1$ e $V_2$ di qualità 60:

$$V_1 = (12,5),(12,6),(12,7) \qquad V_2 = (12,5),(13,6),(12,7)$$

Da (6,4) a (7, 8) esiste un unico viaggio ottimale di qualità 150 formato dai punti:

$$(6,4),\ (7,5),\ (8,6),\ (7,7),\ (7,8)$$

Il viaggio ottimale da (5,4) a (6,9) è

$$(5,4),\ (6,5),\ (7,6),\ (6,7),\ (5,8),\ (6,9),$$

e ha qualità 240. Non esiste invece alcun viaggio da (5,4) a (12,6) e neppure da (7,7) a (13,11).

Due prati $A$ e $B$ sono *connessi* se esiste una sequenza $P_1, \ldots, P_n$ di prati tale che $A = P_1$, $B = P_n$ e per ogni $i \in \{1, \ldots, n-1\}$, $P_i \cap P_{i+1} \neq \emptyset$.
Un *campo* $C$ è un insieme $\mathcal{C}$ *massimale* di prati connessi, vale a dire:

- Per ogni coppia di prati $A, B \in \mathcal{C}$, $A$ è connesso con $B$.
- Per ogni $A \in \mathcal{C}$ e ogni prato $P$ del piano tale che $P \notin \mathcal{C}$, $A$ non è connesso con $P$.

Nell'esempio precedente esistono tre campi:

$$C_1 = \{P_1, P_2, P_3\}, \quad C_2 = \{P_4\}, \quad C_3 = \{P_5\}$$

Diciamo che un fiore $(x, y)$ appartiene a un campo $C$ se appartiene ad almeno un prato $P \in C$.

Si richiede di implementare una struttura dati efficiente che permetta di eseguire le operazioni seguenti (si tenga presente che la minima porzione rettangolare di piano contenente tutti i prati può essere molto grande rispetto al numero di prati e fiori presenti nel piano, quindi *non è sicuramente efficiente rappresentare il piano mediante una matrice*).

- **crea** ()

  Crea un piano vuoto eliminando l'eventuale piano già esistente.

- **inserisci** $(x_0, y_0, x_1, y_1, nome)$

  Inserisce nel piano il prato fiorito $R(x_0, y_0, x_1, y_1)$ leggendo il valore dei fiori dal file *nome*, secondo il formato specificato nell'apposita sezione.

- **viaggio** $(x_0, y_0, x_1, y_1)$

  Costruisce un viaggio di qualità massima $Q$ dal fiore $(x_0, y_0)$ al fiore $(x_1, y_1)$ se esso esiste, stampando i fiori di cui è composto e il valore di $Q$, secondo il formato specificato nell'apposita sezione. Altrimenti stampa la frase:

  `Non esiste alcun viaggio da` $(x_0, y_0)$ `a` $(x_1, y_1)$`.`

- **elimina** $(x_0, y_0)$

  Se il punto $(x_0, y_0)$ corrisponde a un fiore, elimina dal piano il campo $C$ a cui il fiore appartiene, cancellando nel contempo ogni prato e ogni fiore di $C$. Altrimenti non compie alcuna operazione.

- **campi** ()

  Restituisce il numero di campi attualmente presenti nel piano.

## 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 $x_0, y_0, x_1, y_1$ sono numeri interi, e *nome* è il nome di un file situato nella directory dove risiede l'eseguibile.

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.

| Linea di input | Operazione |
|---|---|
| `C` | **crea** () |
| `i` $x_0$ $y_0$ $x_1$ $y_1$ *nome* | **inserisci** $(x_0, y_0, x_1, y_1, nome)$ |
| `v` $x_0$ $y_0$ $x_1$ $y_1$ | **viaggio** $(x_0, y_0, x_1, y_1)$ |
| `e` $x_0$ $y_0$ | **elimina** $(x_0, y_0)$ |
| `c` | **campi** () |
| `f` | Termina l'esecuzione del programma |

Tabella 1: Specifiche del programma

## Note

1. Non devono essere presenti vincoli sulla dimensione del piano, sul numero di prati e campi presenti nel piano, sulla dimensione dei prati (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()`.

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. Sia
   `i` $x_0 \quad y_0 \quad x_1 \quad y_1 \quad \mathit{nome}$
   una richiesta di inserimento di un prato. Allora *nome* deve essere il nome di un file contenente $(|x_1 - x_0| + 1) \cdot (|y_1 - y_0| + 1)$ numeri interi (eventualmente negativi) separati da caratteri di spaziatura. L'ordine in cui tali valori vengono assegnati ai fiori del prato è il seguente:

$$\begin{array}{cccc} (x_0, y_0), & (x_0+1, y_0), & \ldots, & (x_1, y_0), \\ (x_0, y_0+1), & (x_0+1, y_0+1), & \ldots, & (x_1, y_0+1), \\ \vdots, & \vdots, & \ddots, & \vdots, \\ (x_0, y_1), & (x_0+1, y_1), & \ldots, & (x_1, y_1) \end{array}$$

   Ad esempio, supponiamo che dopo il comando `C` sia dato il comando

   ```
   i 10 5 12 8 prato.txt
   ```

   e che il contenuto del file `prato.txt` sia:

   ```
   1 2 3 4 5 6 7 8 9 10 11 12
   ```

   Al termine dell'inserimento, nel piano è presente un prato i cui fiori hanno i seguenti valori:

$$\begin{array}{ccc} v(10,5)=1, & v(11,5)=2, & v(12,5)=3, \\ v(10,6)=4, & v(11,6)=5, & v(12,6)=6, \\ v(10,7)=7, & v(11,7)=8, & v(12,7)=9, \\ v(10,8)=10, & v(11,8)=11, & v(12,8)=12. \end{array}$$

4. Sia $(x_0, y_0), (x_1, y_1) \ldots, (x_u, y_u)$ un viaggio $Q$ di qualità massima. Allora l'output del comando
   `v` $x_0 \quad y_0 \quad x_u \quad y_u$
   deve essere visualizzato nel seguente formato:
   $(Q$
   $x_0, y_0$
   $x_1, y_1$
   $\vdots$
   $x_u, y_u$
   $)$

## Esempio

Si supponga che le linee di input siano:

```
C
i 4 5 11 10 primo.txt
v 8 5 7 10
i 8 12 10 14 secondo.txt
v 8 5 10 14
i 7 10 8 12 terzo.txt
v 8 5 10 14
i 21 21 24 23 quarto.txt
i 21 25 24 27 quarto.txt
v 21 21 24 27
c
i 23 23 26 25 quarto.txt
c
v 21 21 24 27
v 21 21 26 25
i -10 -10 -8 -8 secondo.txt
i -6 -10 -4 -8 secondo.txt
c
i -8 -9 -6 -7 secondo.txt
c
e 26 25
v 21 21 24 27
c
f
```

dove il contenuto dei file `.txt` è il seguente:

```
primo.txt:
4 44 444 6 0 3 2 1
1 11 1729 10 20 15 8 4
3223 -15 10 2 -1 3 8 91
-40 10 -1 0 7 7 13 99
7 29 10 9 9 11 5 41
27 28 30 10 14 -20 6 0

secondo.txt:
4 40 1
-13 8 23
5 7 9

terzo.txt:
0 0
7 -200
0 0

quarto.txt:
0 1 1 3
3 1 8 2
7 9 4 1
```

L'output prodotto dal programma deve essere:

```
( 50
8,5
7,6
6,7
5,8
6,9
7,10
)
Non esiste alcun viaggio da (8,5) a (10,14).
( 117
8,5
7,6
6,7
5,8
5,9
6,10
7,11
8,12
9,13
10,14
)
Non esiste alcun viaggio da (21,21) a (24,27).
3
2
( 36
21,21
21,22
22,23
23,24
24,25
23,26
24,27
)
Non esiste alcun viaggio da (21,21) a (26,25).
4
3
Non esiste alcun viaggio da (21,21) a (24,27).
2
```

## Presentazione del progetto

Il progetto deve essere inviato per posta elettronica all'indirizzo `aguzzoli@dsi.unimi.it` entro il 3 Ottobre 2005 (incluso). La discussione del progetto e l'esame orale si svolgeranno in data e luogo da specificarsi (consultare al riguardo il sito: `http://homes.dsi.unimi.it/~goldwurm/algo`).

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 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 `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 3 Ottobre 2005 (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 S204.

## Avvisi

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.

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

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