Laboratorio di Algoritmi e Strutture Dati

Docenti: M. Trubian, S. Aguzzoli

Appello del 1 luglio 2004

Progetto "Banchetto"
Consegna entro il 19 Luglio 2004

Il problema

In una grande stanza sono disposti dei tavoli imbanditi e su ciascuno di essi è disposta una vivanda prelibata; quando un invitato giunge al banchetto, cerca di piazzarsi nella posizione che gli permette di dominare i cibi di cui è ghiotto.

La *stanza* è rappresentata dall'insieme dei punti del piano

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

dove $\mathbb{R}$ è l'insieme dei numeri reali, ed è suddivisa in celle quadrate di dimensione unitaria aventi come vertici coppie di interi. Più precisamente, dati due interi a e b, la *cella* $C(a, b)$ è il quadrato di lato unitario avente come vertice in basso a sinistra il punto di coordinate (a, b); assumiamo inoltre che i due lati del quadrato adiacente al vertice (a, b) appartengano alla cella, mentre gli altri due lati no. Formalmente:

$$C(a, b) \;=\; \{ (x, y) \in \mathbb{R} \times \mathbb{R} \quad | \quad a \leq x < a + 1 \quad \text{e} \quad b \leq y < b + 1 \}.$$

Si noti che un punto della stanza appartiene a una e una sola cella. Un *tavolo* $T(a, b, s)$, dove a e b sono numeri interi e s (detto *semilato*) è un intero positivo, è il quadrato centrato nel punto (a, b) e avente lato di lunghezza $2s$. Più precisamente, $T(a, b, s)$ è l'unione delle celle $C(m, n)$, dove m e n sono interi tali che:

$$a - s \leq m < a + s \qquad b - s \leq n < b + s$$

Assumiamo che due tavoli non possano sovrapporsi, quindi *un punto della stanza appartiene al più ad un tavolo.*

Ad ogni tavolo sono associati un cibo e la quantità disponibile. Il *cibo* è descritto da una stringa non vuota sull'alfabeto $\mathcal{A}$ delle lettere minuscole dell'alfabeto inglese $\{a, b, c, \ldots, z\}$. La *quantità* è definita da un intero positivo.

Un *invitato* è identificato da una stringa non vuota sull'alfabeto $\mathcal{A}$. Di ogni invitato è definito un profilo in cui compare la lista di cibi di cui è goloso; quando un invitato giunge al banchetto oppure si sposta, tende ad occupare la cella in posizione ottimale rispetto alle sue preferenze, nel modo sotto specificato.

Sia

$$S \;=\; \{ T(a_1, b_1, s_1) \,,\, T(a_2, b_2, s_2) \,,\, \ldots \,,\, T(a_n, b_n, s_n) \}$$

un insieme di tavoli. La *distanza* $d(S)$ fra i tavoli di S è data dalla somma delle distanze dei singoli tavoli:

$$d(S) \;=\; \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} |a_i - a_j| + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} |b_i - b_j|.$$

Il *baricentro* $b(S)$ di S è definito come il baricentro dei tavoli in S, dove la massa di un tavolo è data dalla quantità di cibo disponibile su di esso e si suppone sia concentrata al centro del tavolo. Supponendo che le quantità di cibo dei tavoli in S siano rispettivamente $q_1, \ldots, q_n$, il baricentro $b(S)$ di S è il punto le cui coordinate (X_B, Y_B) soddisfano le uguaglianze:

$$q_1.(X_B - a_1) \;+\; q_2.(X_B - a_2) \;+\; \cdots + q_n.(X_B - a_n) \;=\; 0$$
$$q_1.(Y_B - b_1) \;+\; q_2.(Y_B - b_2) \;+\; \cdots + q_n.(Y_B - b_n) \;=\; 0$$

Un insieme di tavoli S è *adeguato* a un invitato I se, per ogni cibo di cui I è goloso, S contiene esattamente un tavolo recante tale cibo, e nessun altro tavolo appartiene a S.

L'invitato I determina la sua *posizione ottimale* $O(I)$ considerando i baricentri $b(S_1), \ldots, b(S_k)$ degli insiemi $S_1, \ldots, S_k$ che sono adeguati a I e che hanno distanza minima (ossia, $d(S_1) = \cdots = d(S_k) = M$ e, per ogni altro insieme S adeguato per I, si ha che $M < d(S)$) e scegliendo fra questi il baricentro avente coordinate (X_O, Y_O) minime rispetto all'ordine lessicografico (quindi, per ogni $1 \leq i \leq k$, se $b(S_i)$ ha coordinate (x_i, y_i), allora $X_O < x_i$ oppure $X_O = x_i$ e $Y_O < y_i$). Più formalmente:

$$(X_O, Y_O) \;=\; \min \{\, b(S) \mid S \in \mathcal{B}\}$$

dove

$$\mathcal{B} \;=\; \{\, S \,|\, S \text{ è adeguato a } I \text{ e } d(S) = M \,\} \quad \text{e} \quad M \;=\; \min \{\, d(S) \,|\, S \text{ è adeguato a } I \,\}.$$

Si noti che, se esiste almeno un insieme S di tavoli adeguato ad I, allora (X_O, Y_O) è un punto a coordinate razionali ed è univocamente determinato. Altrimenti, il valore $O(I)$ non è determinato: in questo caso l'invitato I, non avendo gradito l'offerta di cibo, esce immediatamente dalla stanza.

Quando un invitato si sposta, tende ad occupare la cella $C(a, b)$ a cui appartiene il punto $O(I)$. Tuttavia, l'invitato può essere impossibilitato a occupare $C(a, b)$ in quanto tale cella è contenuta in un tavolo oppure è già occupata da un altro invitato che non è disposto a cedergliela (le circostanze che determinano o meno questo atto di cortesia sono descritte fra qualche paragrafo). In questo caso, l'invitato cerca la prima cella libera (o cedutagli cortesemente da un altro invitato) nell'ordine descritto dalla visita *a spirale* con *origine* in $C(a, b)$ definita come segue: si parte dall'origine $C(a, b)$ e si prosegue visitando le celle

$$C(a, b+1)\,,\, C(a+1, b+1)\,,\, C(a+1, b)\,,\, C(a+1, b-1)\,,\, C(a, b-1)\,,\, C(a-1, b-1), \ldots$$

come illustrato in Figura 1.

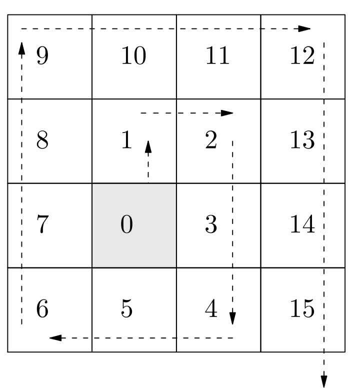

Figura 1: Visita a spirale a partire dalla cella 0

Si possono introdurre o eliminare tavoli dalla stanza. Quando si introduce un nuovo tavolo $T(a, b, s)$, esso non deve sovrapporsi a tavoli o invitati già presenti. Se il quadrato definito dalle coordinate (a, b) e dal semilato s è tale da sovrapporsi a qualche tavolo (o invitato) già presente nella stanza, il tavolo da inserire occuperà una posizione diversa, determinata ponendo il suo centro (a', b') come il primo punto considerato nella visita a spirale con origine in (a, b) tale che $T(a', b', s)$ sia inseribile.

Denotiamo con $A(I)$ la *posizione attuale* dell'invitato I, ossia le coordinate (a, b) della cella $C(a, b)$ cui I appartiene. Si noti che il valore $A(I)$ è determinato se e solo se il valore $O(I)$ lo è. Quando un tavolo viene inserito e/o eliminato, o cambia la quantità di cibo di cui dispone, o qualche invitato esce dalla stanza, possono di conseguenza doversi modificare le posizioni attuali degli invitati presenti al banchetto. Vale a dire, ogni invitato andrà ad occupare la nuova posizione $A(I)$ ridefinita in seguito alla modifica della configurazione dei tavoli e degli invitati. Per buona educazione, gli invitati si spostano uno alla volta nell'ordine in cui sono arrivati al banchetto. L'invitato I_2, giunto dopo I_1 al banchetto, cederà volentieri la propria cella $C(a, b)$ a I_1, se I_1 si volesse posizionare proprio in $C(a, b)$ in seguito alla nuova configurazione della stanza. Se la rimozione di un tavolo rende $O(I)$ indeterminato, allora l'invitato I uscirà dalla stanza.

Riassumendo: supponiamo che sia il turno dell'invitato I. Sia $Pred(I)$ l'insieme di invitati presenti nella stanza e giuntivi prima di I e sia $Post(I)$ l'insieme di invitati presenti nella stanza e giuntivi dopo di I. Dunque, ogni invitato $J \in Pred(I)$ ha già determinato la sua nuova posizione $A(J)$ nella stanza, mentre ogni $J \in Post(I)$ deve ancora determinarla. Per prima cosa I calcola la sua posizione ottimale $O(I)$.

1. Se $O(I)$ è indeterminata, allora I esce dalla stanza.

2. Se la cella $C(a, b)$ contenente $O(I)$ non appartiene ad alcun tavolo e non è occupata da alcun invitato appartenente a $Pred(I)$, allora I si posiziona nella cella $C(a, b)$: dunque $A(I) = (a, b)$.

3. Altrimenti, I occuperà la prima cella $C(c, d)$ incontrata nella visita a spirale con origine nella cella contenente $O(I)$, tale che $C(c, d)$ è libera, oppure è occupata da un invitato appartenente a $Post(I)$. Quindi $A(I) = (c, d)$.

Si noti che, quando tutti gli invitati si sono mossi, non può capitare che due invitati si trovino nella stessa cella. Infatti, supponiamo che la nuova posizione $A(I)$ di I coincida con la posizione $A(K)$ di un invitato $K \in Post(I)$. Quando K dovrà determinare la sua nuova posizione $A(K)$ (*dopo* di I) non può più occupare la cella contenente $A(I)$; infatti, anche se $O(K)$ giacesse nella stessa cella di $A(I)$, K non potrebbe posizionarsi in $A(I)$ in quanto $I \notin Post(K)$, dunque K deve cercare una nuova posizione.

Esempio

Supponiamo che nella stanza siano inseriti nell'ordine i tavoli $T(0, 0, 5)$ avente come cibo *pizza* in quantità 10, $T(20, 20, 5)$ avente come cibo *patatine* in quantità 10 e $T(21, 21, 3)$ avente come cibo *patatine* in quantità 10. I primi due tavoli sono collocati nella posizione richiesta. Il terzo $T(21, 21, 3)$ si sovrappone a $T(20, 20, 5)$, dunque richiede una ricollocazione. Applicando la visita a spirale, il suo centro viene posto in $(15, 28)$. Quindi l'introduzione di $T(21, 21, 3)$ determina la sua collocazione come $T(15, 28, 3)$. A questo banchetto giungono nell'ordine gli invitati *aldo* e *carlo*, che sono entrambi golosi sia di *pizza* che di *patatine*. Gli insiemi adeguati per *aldo* sono i due seguenti: $S_1 = \{T(0, 0, 5), T(20, 20, 5)\}$ e $S_2 = \{T(0, 0, 5), T(15, 28, 3)\}$. Per quanto riguarda le distanze $d(S_1) = 40$, mentre $d(S_2) = 43$. Dunque *aldo* si posiziona nel baricentro $b(S_1) = (10, 10)$. Anche *carlo*, avendo gli stessi gusti, vorrebbe posizionarsi in $(10, 10)$, ma deve invece spostarsi con la visita a spirale, ed andare a occupare la cella $C(10, 11)$. Se adesso eliminiamo il tavolo $T(20, 20, 5)$, i nostri invitati dovranno spostarsi. La nuova posizione di *aldo* (il primo a spostarsi), sarà $(7.5, 14)$, dunque si posizionerà nella cella $C(7, 14)$, mentre *carlo* andrà in $C(7, 15)$ (vedi Figura 2). Supponiamo ora di introdurre il tavolo $T(9, 18, 1)$ contenente *patatine* in quantità 60. La posizione ottimale di *aldo* diventa $(7.714, 15.429)$, quindi *aldo* desidera occupare la cella $C(7, 15)$. Tale cella è attualmente occupata da *carlo* che però è giunto dopo *aldo*, quindi *aldo* può occupare $C(7, 15)$. Ora tocca muoversi *carlo* e anch'egli vorrebbe occupare $C(7, 15)$. Essendo però occupata da *aldo* che è giunto prima al banchetto, *carlo* deve cercare con la visita a spirale una nuova cella e si posiziona nella cella $C(7, 16)$.

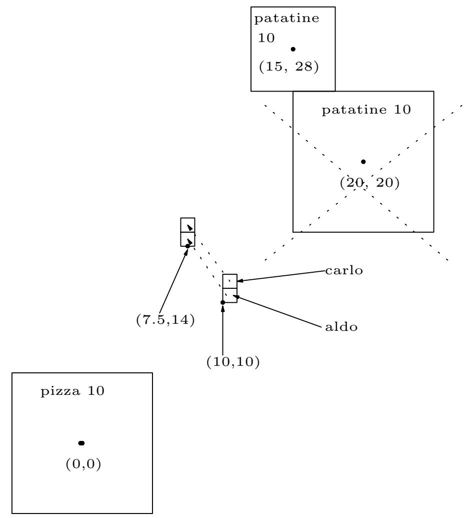

Figura 2: Posizioni di *aldo* e *carlo* prima e dopo l'eliminazione di $T(20, 20, 5)$

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

- **tavolo**(a, b, s, q, γ)

  Inserisce il tavolo $T(a, b, s)$ di centro (a, b) e di semilato s, associandogli una quantità q di cibo γ. Modifica conseguentemente la posizione degli invitati.

- **invitato**$(\alpha, \gamma_1, \ldots, \gamma_n)$

  Se l'invitato I di nome α non è già presente nella stanza, allora gli associa i cibi preferiti $\gamma_1, \ldots, \gamma_n$. Se non esiste alcun insieme di tavoli S adeguato ad I non compie alcun'altra azione, altrimenti inserisce nella stanza l'invitato I, collocandolo nella cella contenente il punto $A(I)$. (Osserviamo esplicitamente che i nomi si possono riutilizzare, a patto che nella stanza non siano mai presenti nello stesso momento più invitati con lo stesso nome. Se in passato è comparso un invitato J di nome α, ma J non è più nella stanza, assumiamo che $I \neq J$).

- **elimina**(a, b)

  Elimina il tavolo contenente la cella $C(a, b)$ e modifica conseguentemente la posizione degli invitati. Se la cella $C(a, b)$ non appartiene ad alcun tavolo, non compie alcuna azione.

- **elimina**(α)

  Elimina l'invitato di nome α e modifica conseguentemente la posizione degli altri invitati. Se non esiste alcun invitato di nome α, non compie alcuna azione.

- **modifica**(a, b, q)

  Imposta al valore q la nuova quantità di cibo associata al tavolo contenente la cella $C(a, b)$ e modifica conseguentemente la posizione degli invitati. Se tale cella non appartiene ad alcun tavolo, non compie alcuna azione.

- **posizione**(α)

  Stampa le coordinate della cella attuale dell'invitato di nome α. Se tale invitato non esiste, non compie alcuna azione.

- **posizione**()

  Stampa i nomi e le coordinate della cella attuale di ogni invitato, in ordine alfabetico, secondo il formato seguente:

  ```
  nome1 a1 b1
  nome2 a2 b2
  ...
  nomeN aN bN
  ```

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 a, b sono numeri interi relativi, q, s sono interi positivi, $\alpha, \gamma, \gamma_1, \ldots, \gamma_n$ sono stringhe finite e non vuote sull'alfabeto $\mathcal{A}$. I vari elementi sulla linea sono separati da uno o più spazi. Quando una linea è letta viene eseguita l'operazione ad essa associata e viene stampato l'eventuale output prodotto dall'esecuzione dell'operazione associata; tutte le operazioni di stampa sono effettuate sullo standard output (`stdout`) e ogni operazione deve iniziare su una nuova linea.

LINEA DI INPUT	OPERAZIONE
`t` a b s q γ	**tavolo** (a, b, s, q, γ)
`i` α γ_1 $\cdots$ γ_n	**invitato** $(\alpha, \gamma_1, \ldots, \gamma_n)$
`e` a b	**elimina** (a, b)
`E` α	**elimina** (α)
`m` a b q	**modifica** (a, b, q)
`p` α	**posizione** (α)
`P`	**posizione** $()$
`q`	Termina l'esecuzione del programma

Tabella 1: Specifiche del programma

Note

1. Non devono essere presenti vincoli sulla dimensione della stanza e sulla dimensione e numero dei tavoli presenti nella stanza. 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()`.

Esempio

Si supponga che le linee di input siano:

```
t 20 0 4 10 birra
t 40 20 4 10 birra
t 20 20 4 10 salsicce
t 40 0 4 10 salsicce
t 18 6 2 10 carpaccio
i aldo birra
i bruno birra salsicce
P
m 16 16 2
p bruno
E aldo
p bruno
e 16 -2
P
i carlo birra salsicce tartine
P
t 40 -10 3 40 tartine
i carlo birra salsicce tartine
P
e 40 -10
e 20 20
e 40 0
e 40 20
e 18 6
P
t -12 -3 2 20 pane
t -5 -7 2  30 coppa
t -3 4 2 70 prosciuttocotto
t 10 4 2 30 coppa
t 3 0 2 40 pane
t -6 4 1 200 salame
i anna pane coppa
p anna
i barbara pane coppa
p barbara
i carlo pane salame
p carlo
i dario pane salame prosciuttocotto
p dario
i ettore pane salame
p ettore
e -3 3
P
```

```
e -12 -3
P
e -5 -7
m  -7 3 400
P
e -6 4
E anna
p barbara
q
```

L'output prodotto dal programma deve essere:

```
aldo 20 4
bruno 20 10
21 4
20 4
bruno 35 19
bruno 35 19
bruno 35 19
carlo 41 -5
-8 -6
-8 -5
-6 2
-6 5
-7 2
anna -8 -6
barbara -8 -5
carlo -6 2
ettore -7 2
anna 6 1
barbara 6 2
carlo -5 3
ettore -5 4
anna 6 1
barbara 6 2
carlo -5 4
ettore -5 3
6 1
```

Presentazione del progetto

Il progetto deve essere inviato per posta elettronica all'indirizzo `aguzzoli@dsi.unimi.it` entro il 19 Luglio 2004 (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/~trubian/studenti.htm`).

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 19 Luglio 2004 (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.

Lo svolgimento del progetto è una prova d'esame da svolgere **individualmente**. I progetti giudicati frutto di **collaborazioni** saranno **estromessi** d'ufficio dalla valutazione.