Laboratorio di Algoritmi e Strutture Dati

Docenti: M. Goldwurm, S. Aguzzoli

Appello del 5 Aprile 2005

Progetto "Recinti"
Consegna entro il 22 Aprile 2005

Il problema

Si studia la resistenza di alcune specie di piante a trattamenti anticrittogamici particolarmente aggressivi e potenzialmente nocivi per le piante stesse.

Formalmente, chiamiamo *piano* l'insieme dei punti

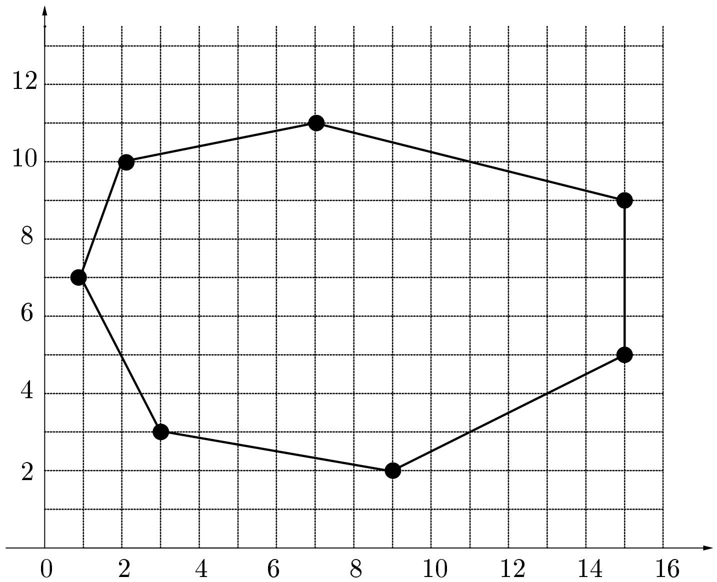
$$\{ (x, y) \in \mathbb{Z} \times \mathbb{Z} \}.$$

Una *pianta* è caratterizzata dalla *posizione* (x, y) in cui è collocata e dalla *specie* w di appartenenza, dove (x, y) è un punto del piano e w è una parola sull'alfabeto $\{a, \ldots, z\}$. In un punto può esservi al più una pianta.

Un *recinto* è un poligono convesso. Usiamo la notazione $R(x_1, y_1, x_2, y_2, \ldots, x_n, y_n)$, dove $n \geq 3$, per definire il recinto i cui vertici elencati in senso orario sono i punti di coordinate (x_1, y_1), (x_2, y_2), $\ldots$, (x_n, y_n). Ad esempio

$$R(7, 11, 15, 9, 15, 5, 9, 2, 3, 3, 1, 7, 2, 10)$$

è il recinto nella figura.



Lo stesso recinto può essere anche definito ad esempio come $R(15, 5, 9, 2, 3, 3, 1, 7, 2, 10, 7, 11, 15, 9)$. Nel seguito si assume che i vertici del recinto siano forniti dall'utente in modo corretto. Ad esempio, non verrà mai chiesto di costruire i recinti

$$R(2, 2, 3, 3, 4, 1, 2, -5, 1, 1) \qquad R(0, 3, 4, -2, -1, -1, -2, -2)$$

Infatti, nel primo caso il punto $(2,2)$ non è un vertice del poligono, nel secondo caso il poligono risultante non è convesso. Nel piano può esservi al massimo un recinto.

Un punto è *nel recinto* se si trova all'interno o sul bordo del recinto. Ad esempio, riferendosi al recinto in figura, i punti (8,9), (7,11), (13, 4), (15,7) sono nel recinto mentre i punti (0,0), (2,4), (12,3) non lo sono. Una pianta è nel recinto se è posizionata in un punto nel recinto. La *capienza* di un recinto è data dal massimo numero di piante che possono esservi nel recinto. Il recinto nella figura ha capienza 103.

Un trattamento anticrittogamico stabilisce fra le specie di piante una gerarchia di resistenza al trattamento stesso, descritta da una *relazione d'ordinamento stretto* $<$. Questo significa che $<$ deve verificare le seguenti proprietà degli ordinamenti stretti:

(P1). Per ogni specie w_1 e w_2, se $w_1 < w_2$, allora non vale $w_2 < w_1$.

(P2). Per ogni specie w_1, w_2 e w_3, se $w_1 < w_2$ e $w_2 < w_3$, allora $w_1 < w_3$.

La gerarchia fra le specie è stabilita dall'utente e il programma deve controllare che le asserzioni del tipo $u < v$ inserite dall'utente non violino le proprietà scritte sopra.

Ad esempio, supponiamo che la relazione fra specie sia così definita:

```
begonia<geranio     geranio<gardenia     primula<oleandro     gardenia<oleandro

oleandro<magnolia     gardenia<azalea     geranio<camelia     camelia<azalea
```

In questa situazione l'utente non può aggiungere a $<$ l'asserzione `oleandro<begonia`. Infatti, per la proprietà (P2) si ha che `begonia<oleandro`, quindi per (P1) non può valere `oleandro<begonia`. È possibile invece inserire `begonia<primula` (oppure: `primula<begonia`), `camelia<gardenia` (oppure: `gardenia<camelia`), ecc.

La somministrazione del trattamento nel recinto determina una selezione alla conclusione della quale rimangono solo le piante appartenenti alle specie più resistenti. La *selezione* fra le piante nel recinto avviene secondo la seguente regola:

- Una pianta della specie w_1 nel recinto viene eliminata se e solo se esiste nel recinto una pianta di specie w_2 tale che $w_1 < w_2$.

Nell'esempio precedente, se nel recinto esiste un oleandro, tutte gli esemplari di primula, di geranio, di begonia e di gardenia nel recinto sono eliminati.

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 tutte le piante può essere molto grande rispetto al numero di piante presenti, quindi *non è sicuramente efficiente rappresentare il piano mediante una matrice*).

- **crea** ()

  Crea un piano vuoto eliminando l'eventuale piano già esistente e le piante in esso presenti, eliminando inoltre il trattamento considerato e dunque ogni asserzione relativa alla relazione $<$.

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

  Se sul punto (x, y) non c'è alcuna pianta, pone su (x, y) una pianta di specie w, altrimenti non compie alcuna operazione.

- **recinto** $(x_1, y_1, \ldots, x_n, y_n)$

  Costruisce il recinto $R(x_1, y_1, \ldots, x_n, y_n)$ $(n \geq 3)$ distruggendo l'eventuale recinto già esistente. Si assume che i punti $(x_1, y_1), \ldots, (x_n, y_n)$ siano i vertici di un poligono convesso, elencati in senso orario.

- **capienza** ()

  Stampa la capienza del recinto sul piano (0 se il recinto non è definito).

- **stampa** ()

  Stampa il contenuto del recinto secondo il formato specificato nell'apposita sezione.

- **minore** (w_1, w_2)

  Aggiunge alla relazione $<$ l'asserzione `w1<w2` fra le specie w_1 e w_2 se non vengono violate le proprietà (P1,P2) definitorie dell'ordine stretto, altrimenti stampa il messaggio: `Impossibile aggiungere l'asserzione w1<w2`.

- **confronta** (w_1, w_2)

  Stampa `w1 < w2` se $w_1 < w_2$, `w2 < w1` se $w_2 < w_1$, `w1 <> w2` altrimenti.

- **selezione** ()

  Somministra il trattamento e compie la selezione fra le piante nel recinto.

- **esemplari** (w)

  Stampa il numero totale di piante di specie w presenti nel piano.

Alcuni suggerimenti

Il determinante della matrice associata a due vettori $(a_1, a_2) \in \mathbb{Z} \times \mathbb{Z}$ e $(b_1, b_2) \in \mathbb{Z} \times \mathbb{Z}$ è

$$\det \begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix} = a_1 b_2 - a_2 b_1 \,.$$

Il valore assoluto di tale determinante fornisce l'area del parallelogramma P i cui vertici sono

$$(0,0), (a_1, a_2), (a_1 + b_1, a_2 + b_2), (b_1, b_2) \,.$$

Inoltre tale valore coincide con il numero di punti di $\mathbb{Z} \times \mathbb{Z}$ contenuti nel parallelogramma P eccettuati quelli giacenti sul segmento di estremi $(a_1, a_2), (a_1 + b_1, a_2 + b_2)$ e quelli giacenti sul segmento di estremi $(a_1 + b_1, a_2 + b_2), (b_1, b_2)$.
Sia $k = a_1 b_2 - a_2 b_1$ il valore del determinante.

- $k > 0$ se la rotazione che porta il segmento di estremi $(0,0), (a_1, a_2)$ a sovrapporsi alla semiretta di origine $(0,0)$ e passante per (b_1, b_2) è effettuata in senso antiorario.
- $k < 0$ se tale rotazione è effettuata in senso orario.
- $k = 0$ se i tre punti $(0,0), (a_1, a_2), (b_1, b_2)$ giacciono sulla stessa retta.

Il numero di punti di $\mathbb{Z} \times \mathbb{Z}$ giacenti sul segmento $(0,0), (a_1, a_2)$ è uguale a $\gcd(a_1, a_2) + 1$, dove gcd denota il massimo comun divisore.

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 $x_1, x_2, \ldots, x_n, y_n$ sono numeri interi e w_1, w_2 sono parole sull'alfabeto $\{a, \ldots, z\}$.

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_1 y_1 w_1	**inserisci** (x_1, y_1, w_1)
`r` x_1 y_1 $\ldots$ x_n y_n	**recinto** $(x_1, y_1, \ldots, x_n, y_n)$
`C`	**capienza** ()
`s`	**stampa** ()
`<` w_1 w_2	**minore** (w_1, w_2)
`?` w_1 w_2	**confronta** (w_1, w_2)
`S`	**selezione** ()
`e` w_1	**esemplari** (w_1)
`f`	Termina l'esecuzione del programma

Tabella 1: Specifiche del programma

Note

1. Non devono essere presenti vincoli sulla dimensione del piano, sulla lunghezza delle parole, sul numero di lati del recinto, sul numero di elementi nel piano (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. Siano $(x_1, y_1), \ldots, (x_k, y_k)$ i punti nel recinto in cui vi è una pianta, dove i punti sono ordinati rispetto all'ordinamento lessicografico[1]. Supponendo che la pianta in (x_i, y_i) appartenga alla specie w_i, l'output del comando `s` deve essere visualizzato nel seguente formato:

   $($
   $x_1, y_1 : w_1$
   $x_2, y_2 : w_2$
   $\vdots$
   $x_k, y_k : w_k$
   $)$

Esempio
Si supponga che le linee di input siano:

```
c
< origano salvia
< camomilla basilico
< ortica prezzemolo
< mirto maggiorana
< rosmarino salvia
< rosmarino origano
< camomilla origano
< salvia camomilla
< ortica mirto
< mirto ginepro
< cicuta sambuco
< menta prezzemolo
< ginepro menta
< prezzemolo mirto
< cipolla prezzemolo
< cicuta prezzemolo
< cicuta cipolla
< cicuta menta
< ginepro cicuta
< maggiorana cicuta
r 4 3 2 8 4 14 10 16 14 6 14 1
C
s
i 2 3 cicuta
i 23 3 ortica
i 29 8 rosmarino
i 33 12 camomilla
i 6 6 origano
i 13 3 prezzemolo
i 10 9 ginepro
i 5 12 basilico
i 24 22 basilico
i 29 19 cipolla
```

[1]In base a tale ordinamento, $(a, b) < (c, d)$ se e solo se $a < c$ oppure $a = c$ e $b < d$.

```
i 29 15 basilico
i 25 10 carota
i 20 10 menta
i 21 15 origano
i 25 17 salvia
i 17 19 cipolla
i 14 12 maggiorana
i 14 9 menta
i 19 5 origano
i 21 19 ginepro
s
? carota basilico
? ortica menta
? menta mirto
? carciofo asparago
e ginepro
S
e ginepro
s
r 17 19 21 22 24 22 29 19 25 10 17 1 15 3
C
s
e origano
S
s
e origano
r 27 24 34 15 25 8 17 7 7 8 13 24
C
s
S
s
i 10 16 prezzemolo
S
s
e ortica
f
```

L'output prodotto dal programma deve essere:

```
Impossibile aggiungere l'asserzione salvia < camomilla
Impossibile aggiungere l'asserzione prezzemolo < mirto
133
(
)
(
5,12:basilico
6,6:origano
10,9:ginepro
13,3:prezzemolo
)
```

```
carota <> basilico
ortica < menta
mirto < menta
carciofo <> asparago
2
1
(
5,12:basilico
6,6:origano
13,3:prezzemolo
)
173
(
17,19:cipolla
19,5:origano
20,10:menta
21,15:origano
21,19:ginepro
24,22:basilico
25,10:carota
25,17:salvia
29,19:cipolla
)
3
(
17,19:cipolla
20,10:menta
24,22:basilico
25,10:carota
25,17:salvia
29,19:cipolla
)
1
341
(
14,9:menta
14,12:maggiorana
17,19:cipolla
20,10:menta
24,22:basilico
25,10:carota
25,17:salvia
29,15:basilico
29,19:cipolla
)
(
14,9:menta
17,19:cipolla
20,10:menta
```

```
24,22:basilico
25,10:carota
25,17:salvia
29,15:basilico
29,19:cipolla
)
(
10,16:prezzemolo
24,22:basilico
25,10:carota
25,17:salvia
29,15:basilico
)
1
```

Presentazione del progetto

Il progetto deve essere inviato per posta elettronica all'indirizzo `aguzzoli@dsi.unimi.it` entro il 22 Aprile 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 22 Aprile 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.