Laboratorio di Algoritmi e Strutture Dati

Docenti: M. Torelli, S. Aguzzoli

Progetto "Componenti elettroniche"

valido per gli appelli di giugno e luglio 2010

Premessa

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

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

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.

1 Organizzazione degli appelli e modalità di consegna

Il presente progetto è valido per gli appelli di giugno e luglio 2010 e deve essere consegnato:

- entro l'8 giugno per l'appello del 15 giugno;
- entro il 24 giugno per l'appello dell'1 luglio.

Le discussioni dei progetti per i due appelli si svolgeranno in date e luoghi da specificarsi. Il calendario dei colloqui sarà disponibile sulla pagina del corso `http://homes.dsi.unimi.it/~aguzzoli/algo.htm` qualche giorno dopo il termine di consegna del progetto.

Il progetto va inviato per posta elettronica all'indirizzo `aguzzoli@dsi.unimi.it` entro le date sopra indicate. 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 le scelte implementative, analizzando 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 della forma `cognome_matricola.zip`. La relazione e il codice devono riportare nome, cognome e matricola. Una copia cartacea della relazione e del codice deve inoltre essere consegnata al docente entro le scadenze fissate (lasciandola eventualmente nella sua casella postale presso il dipartimento in via Comelico). Si ricorda infine di presentarsi al colloquio con una copia stampata della relazione e del codice.

2 Il problema

Una fabbrica produce dispositivi elettronici assemblando varie componenti prodotte da terze parti.

Componenti. Formalmente, una *componente* c è caratterizzata da:

- il suo *codice identificativo* $I(c) \in \mathbb{N}$;
- la sua *famiglia di appartenenza* $F(c) \in \mathbb{N}$;
- il suo *costo* $C(c) \in \mathbb{N}^+$;
- la sua *estensione* $E(c)$, dove $E(c)$ è l'area di un rettangolo di lunghezza $L(c)$ e altezza $H(c)$ (con $L(c), H(c) \in \mathbb{N}^+$).

$\mathbb{N}$ è l'insieme dei numeri naturali 0,1,2, ...; $\mathbb{N}^+$ è l'insieme dei numeri naturali positivi 1,2,... .

Dispositivi. Le componenti possono essere assemblate a costituire dei dispositivi. L'assemblaggio avviene posizionando le componenti su un nastro di supporto, in corrispondenza di ganci di fissaggio disposti a distanza di una unità l'uno dall'altro, a partire dall'inizio del nastro; la posizione di una componente all'interno di un dispositivo è quindi indicata da un numero naturale $x \in \mathbb{N}$. L'ordine di fissaggio delle componenti è rilevante; due componenti possono essere affiancate o sovrapposte sul nastro; è possibile fissare più componenti sullo stesso gancio. Dunque, un dispositivo è determinato dal numero totale k di componenti fissate, detto *profondità* del dispositivo, e da una sequenza di coppie $\langle componente, posizione \rangle$, dove la i-esima coppia rappresenta la componente da fissare per i-esima e la posizione dove fissarla.

Formalmente, quindi, un *dispositivo* D è una sequenza finita $(\langle c_1, x_1\rangle, \langle c_2, x_2\rangle, \ldots, \langle c_k, x_k\rangle)$ di coppie, ciascuna delle quali formata da una componente c_i e dalla sua posizione $x_i \in \mathbb{N}$ sul nastro.

Si noti che D non può contenere più copie dello stesso componente: vale a dire che $i \neq j$ implica $c_i \neq c_j$.

Si noti che due sequenze che contengono le stesse componenti ma in posizioni diverse descrivono due dispositivi diversi. Analogamente due sequenze che contengono le stesse coppie componente-posizione ma in ordine diverso descrivono due dispositivi diversi.

Un dispositivo $D = (\langle c_1, x_1\rangle, \langle c_2, x_2\rangle, \ldots, \langle c_k, x_k\rangle)$ si dice *lineare* se $x_1 < x_2 < \ldots < x_k$ e $x_t + L(c_t) \leq x_{t+1}$ per ogni $t = 1, 2, \ldots, k-1$.

L'*estensione* del dispositivo $D = (\langle c_1, x_1\rangle, \langle c_2, x_2\rangle, \ldots, \langle c_k, x_k\rangle)$ è l'area $E(D)$ della figura $\mathcal{F}(D)$ ottenuta come unione

$$\mathcal{F}(D) = \bigcup_{t=1}^{k} \mathcal{R}_t$$

di tutti i rettangoli $\mathcal{R}_t = \{(x, y) \in \mathbb{R}^2 \mid x_t \leq x \leq x_t + L(c_t),\ 0 \leq y \leq H(c_t)\}$. Si noti che se D è un dispositivo lineare, allora $E(D)$ è semplicemente la somma delle estensioni delle sue componenti.

Esempio 1 *Siano $c_1, c_2, \ldots, c_6$ le componenti caratterizzate dai seguenti parametri:*

	I	F	C	L	H
c_1	12	3	6	4	3
c_2	7	3	4	6	2
c_3	100	2	5	3	8
c_4	9	1	2	10	1
c_5	3	2	10	5	5
c_6	74	7	4	2	4

I dispositivi:

$$D_1 = (\langle c_6, 20\rangle, \langle c_5, 19\rangle, \langle c_1, 4\rangle, \langle c_4, 2\rangle, \langle c_2, 7\rangle), \qquad D_2 = (\langle c_2, 1\rangle, \langle c_3, 8\rangle, \langle c_1, 11\rangle),$$

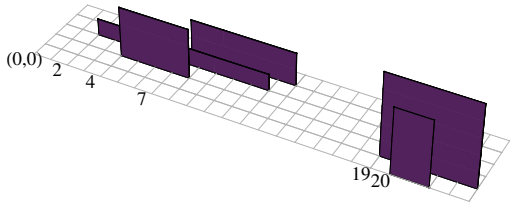

Figura 1: dispositivo D_1

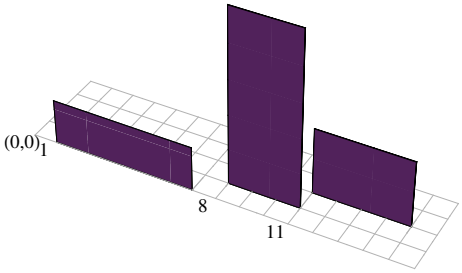

Figura 2: dispositivo D_2

possono essere rappresentati come in Figura 1 e Figura 2.

L'estensione di D_1 *è* $E(D_1) = 49$. *Il dispositivo* D_2 *è lineare e la sua estensione è* $E(D_1) = 6 \times 2 + 3 \times 8 + 4 \times 3 = 48$.

3 Operazioni da implementare

Si richiede di implementare una struttura dati efficiente che permetta di eseguire le operazioni descritte in seguito (si tenga presente che la minima porzione rettangolare di piano contenente tutte le componenti può essere molto grande rispetto al numero di componenti, quindi *non è sicuramente efficiente rappresentare l'insieme dei componenti mediante un'*unica *matrice* contenente i punti interi appartenenti alle varie componenti).

Le operazioni da implementare variano a seconda dell'appello per il quale avviene la consegna: per l'appello del 15 giugno sono obbligatorie le operazioni specificate nelle Sezioni 3.1 e 3.2; per l'appello dell'1 luglio sono obbligatorie le operazioni specificate nelle Sezioni 3.1 e 3.3. L'implementazione delle operazioni specificate nelle altre sezioni è da considerarsi facoltativa.

Si noti che le operazioni richieste sono liberamente implementabili; in particolare, non vanno necessariamente intese come prototipi di funzioni.

3.1 Operazioni di base

- **componente** (i, f, c, l, h)

  Definisce una componente z di codice identificativo i, famiglia f, costo c, estensione data da $L(z) = l$ e $H(z) = h$. Se esiste già una componente di codice identificativo i, sostituisce la nuova definizione a quella esistente.

- **ordina** $(i_1, i_2, \ldots i_n)$

  Ordina le componenti con codice identificativo $i_1, i_2, \ldots, i_n$ in base al costo di produzione. Se, per qualche $t = 1, 2, \ldots, n$, non esiste una componente con codice identificativo i_t, stampa il messaggio

  `Non esiste componente con codice identificativo` i_t`.`

- **stampa** ()

  Stampa tutte le componenti in ordine di identificativo.

- **stampa** (f)

  Stampa tutte le componenti della famiglia f in ordine di identificativo.

3.2 Operazioni per l'appello del 15 giugno (consegna entro l'8 giugno)

Si richiede di implementare la seguente operazione:

- **lineare** $(x_1, x_2, \ldots, x_k, x_{k+1})$

  Trova un dispositivo lineare $D = (\langle c_1, x_1\rangle, \langle c_2, x_2\rangle, \ldots \langle c_k, x_k\rangle)$, contenuto tra l'inizio del nastro di supporto e il gancio di posizione x_{k+1}, tale che $E(D)$ sia massima; stampa estensione e descrizione di tale dispositivo secondo le specifiche riportate nell'apposita sezione. Si noti che tale dispositivo non è necessariamente unico. Se non esiste alcun dispositivo di questo tipo, stampa il messaggio

  `Non esiste un dispositivo lineare adeguato.`

Esempio 2 *Siano $c_1, c_2, \ldots, c_6$ le componenti definite nell'Esempio 1. L'operazione*

$$\textbf{lineare}\ (2, 7, 17, 24, 26)$$

può produrre uno dei due dispositivi seguenti, entrambi di estensione $25 + 12 + 24 + 8 = 69$

$$(\langle c_5, 2\rangle, \langle c_1, 7\rangle, \langle c_3, 17\rangle, \langle c_6, 24\rangle) \qquad \textit{oppure} \qquad (\langle c_5, 2\rangle, \langle c_2, 7\rangle, \langle c_3, 17\rangle, \langle c_6, 24\rangle).$$

3.3 Operazioni per l'appello dell'1 luglio (consegna entro il 24 giugno)

Il costo di produzione di un dispositivo è dato dal costo totale delle sue singole componenti sommato a un costo aggiuntivo di assemblaggio. Il costo di assemblaggio è determinato dal grado di incompatibilità fra famiglie di componenti.

Formalmente, data una famiglia f, l'intero $\delta(f) \in \mathbb{N}$ denota il costo di trasferimento per passare da una componente della famiglia f ad una componente di un'altra famiglia. Il costo di assemblaggio di due componenti c_1 e c_2 è dato da:

$$\delta(c_1, c_2) = \begin{cases} 0 & \text{se } F(c_1) = F(c_2)\ , \\ \delta(F(c_1)) & \text{se } F(c_1) \neq F(c_2)\ . \end{cases}$$

Il *costo di produzione* del dispositivo $D = (\langle c_1, x_1\rangle, \langle c_2, x_2\rangle, \ldots, \langle c_k, x_k\rangle)$ è dato da:

$$C(D) = \sum_{t=1}^{k} C(c_t) + \sum_{t=1}^{k-1} \delta(c_t, c_{t+1})\,.$$

Esempio 3 *Siano $c_1, c_2, \ldots, c_6$ le componenti e D_1 e D_2 i dispositivi definiti nell'Esempio 1. Si assuma che i costi di trasferimento delle famiglie siano come segue:*

F	δ
1	5
2	2
3	3
7	4

Allora i costi di produzione di D_1 e D_2 sono rispettivamente dati da:

$$C(D_1) = (4+10+6+2+4)+(4+2+3+5) = 40 \qquad e \qquad C(D_2) = (4+5+6)+(3+2) = 20\,.$$

Sia D_3 il dispositivo $(\langle c_1, 14\rangle, \langle c_2, 97\rangle, \langle c_3, 19\rangle)$, allora il suo costo è dato da

$$C(D_3) = (6+4+5)+(0+3) = 18\,.$$

La politica della fabbrica è di produrre dispositivi di costo minimo. Per ottenere questo obiettivo, al momento di progettare un nuovo dispositivo, viene preparato un prospetto che contiene tutte le informazioni sulle possibili implementazioni del dispositivo stesso.

Formalmente, un *prospetto* P è costituito da una sequenza $(S_1, S_2, \ldots, S_k)$, dove $k \in \mathbb{N}^+$ e S_t è un sottoinsieme finito di $\mathbb{N} \times \mathbb{N}$ per ogni $t = 1, 2, \ldots, k$.

Un dispositivo $D = (\langle c_1, x_1\rangle, \langle c_2, x_2\rangle, \ldots, \langle c_h, x_h\rangle)$ è *realizzabile* per il prospetto P se e solo se $h = k$ e $\langle I(c_t), x_t\rangle \in S_t$ per ogni $t \in \{1, 2, \ldots, k\}$.

Per semplicità assumiamo che un codice identificativo di componente occorra al più una volta in un dato prospetto.

Un dispositivo di *minimo costo* per il prospetto P è un dispositivo realizzabile per il prospetto P che minimizza il costo di produzione fra tutti i dispositivi realizzabili per P. (Si noti che potrebbero esistere più dispositivi di minimo costo per un dato prospetto P).

Esempio 4 *Siano $c_1, c_2, \ldots, c_6$ le componenti definite nell'Esempio 1 e siano i costi di trasferimento delle famiglie definiti come nell'Esempio 3. Siano inoltre definite le componenti seguenti:*

	I	F	C	L	H
c_7	23	1	6	7	4
c_8	24	1	2	3	3

Sia

$$P = (\{\langle 100, 2\rangle, \langle 12, 2\rangle\}, \{\langle 23, 3\rangle\, \langle 7, 4\rangle, \langle 24, 4\rangle\}, \{\langle 9, 0\rangle, \langle 74, 1\rangle\})$$

un prospetto. Allora un dispositivo di costo minimo per P è il seguente:

$$D = (\langle c_3, 2\rangle, \langle c_8, 4\rangle, \langle c_4, 0\rangle)\,,$$

il cui costo $C(D)$ è $C(c_3)+C(c_8)+C(c_4)+\delta(2)+0 = 5+2+2+2+0 = 11$, mentre il seguente dispositivo

$$D' = (\langle c_1, 2\rangle, \langle c_2, 4\rangle, \langle c_4, 0\rangle)\,,$$

è realizzabile per P, ma il suo costo, che è $C(c_1) + C(c_2) + C(c_4) + 0 + \delta(3) = 6 + 4 + 2 + 0 + 3 = 15$, non è minimo. Si noti inoltre che il dispositivo $(\langle c_2, 2\rangle, \langle c_7, 3\rangle, \langle c_6, 1\rangle)$ non è realizzabile per P in quanto $\langle I(c_2), 2\rangle = \langle 7, 2\rangle \notin \{\langle 100, 2\rangle, \langle 12, 2\rangle\}$.

Si richiede di implementare le operazioni seguenti:

- **trasferimento** (f, x)

  Stabilisce che il costo di trasferimento $\delta(f)$ della famiglia f è x. Se il valore di $\delta(f)$ è già stato definito, sostituisce il nuovo valore a quello precedente.

- **estensione** $(i_1, x_1, i_2, x_2, \ldots, i_k, x_k)$

  Calcola l'estensione del dispositivo $(\langle c_1, x_1\rangle, \langle c_2, x_2\rangle, \ldots, \langle c_k, x_k\rangle)$, dove $I(c_t) = i_t$ per ogni $t = 1, 2, \ldots, k$. Se, per qualche $t = 1, 2, \ldots, k$, non esiste una componente con codice identificativo i_t, stampa il messaggio

  `Non esiste componente con codice identificativo` i_t`.`

- **prospetto** $(nomefile)$

  Definisce il prospetto attuale caricandolo dal file di nome $nomefile$ secondo le specifiche riportate nell'apposita sezione. Se è già stato definito un prospetto, sostituisce la nuova definizione a quella esistente.

- **costominimo** ()

  Calcola il dispositivo di costo minimo realizzabile per il prospetto attuale nel modo seguente:

  - Se non esiste alcun prospetto stampa il messaggio:
    `Non esiste alcun prospetto.`
  - Se non è possibile scegliere alcuna componente per qualche passo, poiché nessun identificativo possibile corrisponde a componenti definite, allora stampa il messaggio:
    `Non esiste alcun dispositivo di costo minimo.`
  - Se esiste almeno un dispositivo di costo minimo, ne stampa costo e descrizione secondo le specifiche riportate nell'apposita sezione (se per qualche famiglia f usata nel prospetto non è stato definito alcun costo di trasferimento, si assuma $\delta(f) = 0$).

- **estensionecostominimo** ()

  Calcola, se esiste, l'estensione di un dispositivo di costo minimo. Se non esiste alcun dispositivo di costo minimo stampa il messaggio:

  `Non esiste alcun dispositivo di costo minimo.`

4 Specifiche di implementazione

Il programma deve leggere dallo standard input (`stdin`) una sequenza di righe (separate da `\n`), ciascuna delle quali corrisponde a una riga della prima colonna della Tabella 1, dove $nomefile$ è il nome di un file, $i, f, i_1, i_2, \ldots, i_k, x, x_1, x_2, \ldots, x_n \in \mathbb{N}$, $c, l, h, n, k \in \mathbb{N}^+$.

I vari elementi sulla riga sono separati da uno o più spazi. Quando una riga 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 riga.

Riga di input	Operazione
`c` $i\ f\ c\ l\ h$	**componente** (i, f, c, l, h)
`o` $i_1\ i_2\ \ldots\ i_n$	**ordina** $(i_1, i_2, \ldots, i_n)$
`s`	**stampa** ()
`s` f	**stampa** (f)
`l` $x_1\ x_2\ \ldots\ x_k$	**lineare** $(x_1, x_2, \ldots, x_k)$
`t` $f\ x$	**trasferimento** (f, x)
`e` $i_1\ x_1\ i_2\ x_2\ \ldots\ i_k\ x_k$	**estensione** $(i_1, x_1, i_2, x_2, \ldots, i_k, x_k)$
`e`	**estensionecostominimo** ()
`p` $nomefile$	**prospetto** $(nomefile)$
`m`	**costominimo** ()
`f`	Termina l'esecuzione del programma

Tabella 1: Specifiche del programma

- La componente z di parametri $I(z) = i$, $F(z) = f$, $C(z) = c$, $L(z) = l$, $H(z) = h$ deve essere visualizzata come riga:

  $i \quad f \quad c \quad l \quad h$

  dove ogni numero è separato dal precedente da un singolo spazio.

- L'output dei comandi di nome `o` ed `s` deve essere visualizzato come

  (
  $componente_1$
  $componente_2$
  $\vdots$
)

  dove $componente_t$ deve essere visualizzato come sopra specificato. L'ordine delle componenti è rilevante.

- Il dispositivo $D = (\langle c_1, x_1\rangle, \langle c_2, x_2\rangle, \ldots, \langle c_h, x_h\rangle)$ deve essere visualizzato come

  [
  $componente_1, x_1$

$componente_2, x_2$
$\vdots$
$componente_k, x_k$
]

dove $componente_t$ deve essere visualizzato come sopra specificato. L'ordine delle componenti è rilevante.

- L'output del comando di nome `l` deve essere della forma

  e :
  dispositivo

  dove e è l'estensione massima calcolata e *dispositivo* deve essere visualizzato come sopra specificato.

- L'output del comando di nome `m` deve essere della forma

  c :
  dispositivo

  dove c è il costo minimo calcolato e *dispositivo* deve essere visualizzato come sopra specificato

- Un file che definisce un prospetto deve essere della forma

  k
  S_1
  S_2
  $\vdots$
  S_k

  dove ogni insieme $S_t = \{\langle i_{t,1}, x_{t,1}\rangle, \langle i_{t,2}, x_{t,2}\rangle, \ldots, \langle i_{t,n_t}, x_{t,n_t}\rangle\}$ è visualizzato a sua volta come

  (
  $i_{t,1} \quad x_{t,1}$
  $i_{t,2} \quad x_{t,2}$
  $\vdots$
  $i_{t,n_t} \quad x_{t,n_t}$
)

  dove ogni numero è separato dal precedente da un singolo spazio.

Note

1. Non devono essere presenti vincoli sul numero di componenti e famiglie, sulle loro dimensioni e collocazione (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()`.

5 Esempi di esecuzione

5.1 Per l'appello del 15 giugno

Si supponga che le righe di input siano:

```
c 12 3 6 4 3
c 7 3 4 6 2
c 100 2 5 3 8
c 9 1 2 10 1
c 3 2 10 5 5
c 74 7 4 2 4
l 2 7 17 24 26
c 24 1 6 3 10
c 23 1 6 7 4
c 24 1 2 3 3
s
s 1
o 23 24 7
l 2 7 17 24 26
l 6 7 17
f
```

L'output prodotto dal programma deve essere il seguente

```
69:
[
3 2 10 5 5, 2
12 3 6 4 3, 7
100 2 5 3 8, 17
74 7 4 2 4, 24
]
(
3 2 10 5 5
7 3 4 6 2
9 1 2 10 1
12 3 6 4 3
23 1 6 7 4
24 1 2 3 3
74 7 4 2 4
100 2 5 3 8
)
(
9 1 2 10 1
23 1 6 7 4
24 1 2 3 3
)
(
24 1 2 3 3
```

```
7 3 4 6 2
23 1 6 7 4
)
85:
[
3 2 10 5 5, 2
100 2 5 3 8, 7
23 1 6 7 4, 17
74 7 4 2 4, 24
]
Non esiste un dispositivo lineare adeguato.
```

5.2 Per l'appello dell'1 luglio

Si supponga che le righe di input siano:

```
t 1 6
t 2 5
t 3 2
t 4 4
c 7 1 6 4 7
c 11 4 5 7 4
c 13 1 4 8 4
c 17 1 5 3 9
c 19 1 8 2 5
c 8 4 6 5 5
c 10 3 6 6 6
c 9 3 5 6 5
c 15 2 7 3 7
c 16 2 7 4 2
c 131 2 8 3 4
c 151 1 8 3 4
c 152 1 6 4 5
c 153 3 6 5 4
c 155 4 4 4 4
c 97 1 4 6 11
e 8 0 6 1
e 8 0 7 0
e 155 12 131 8 152 8 8 6 19 0 17 20 11 22
p prospetto.txt
m
e
p prospettobis.txt
m
e
f
```

dove il contenuto del file `prospetto.txt` è il seguente

```
5
```

```
(
11 5
13 5
155 3
)
(
17 7
16 4
9 21
7 15
)
(
153 12
10 8
131 5
19 44
)
(
97 7
)
(
15 33
152 32
151 11
)
```

mentre il contenuto del file `prospettobis.txt` è il seguente

```
5
(
11 5
13 5
155 3
)
(
17 7
16 4
9 21
7 15
)
(
10 8
131 5
)
(
97 7
)
(
15 33
152 32
```

```
151 11
)
```

L'output prodotto dal programma deve essere il seguente

```
Non esiste componente con codice identificativo 6.
33
107
27:
[
13 1 4 8 4,5
17 1 5 3 9,7
19 1 8 2 5,44
97 1 4 6 11,7
152 1 6 4 5,32
]
104
31:
[
155 4 4 4 4,3
9 3 5 6 5,21
10 3 6 6 6,8
97 1 4 6 11,7
152 1 6 4 5,32
]
138
```