

TRẦN SĨ TÙNG  
✧★✧



# **BÀI TẬP TIN HỌC**

## **Tin học & Nhà trường**

**Tập III**  
**151 – 200**

**2013**

## Phần 1: ĐỀ BÀI

### Bài 151/2003 – Người nông dân và những quả táo

Một người nông dân đến gặp đức vua và nói: "Xin đức vua ban cho tôi 1 quả táo trong vườn Thượng uyển của ngài". Nhà vua đồng ý. Người nông dân đi đến vườn và thấy muốn vào vườn phải lần lượt qua 3 cổng gác và ở mỗi cổng đều có một người lính canh. Người nông dân tiến đến người lính canh thứ nhất và nói:

– "Đức vua cho tôi lấy một quả táo ở trong vườn"

– "Được, anh hái đi, nhưng khi đi ra phải đưa cho tôi một nửa số táo anh hái và thêm một quả nữa", người lính canh trả lời. Hai người lính canh kia cũng nói như vậy với người nông dân. Hỏi người nông dân phải hái bao nhiêu quả táo để khi ra khỏi vườn và đưa cho những người lính canh số táo đúng như họ yêu cầu, anh ta còn lại một quả?

### Bài 152/2003 – Chia kẹo

Có N gói kẹo, gói thứ i có A[i] cái kẹo.

**Yêu cầu:** Hãy tìm cách chia các gói kẹo này thành 2 phần sao cho độ chênh lệch giữa tổng số kẹo ở hai phần là ít nhất có thể được  $0 < A[i] < 1000, 2 \leq N \leq 10000$

#### Dữ liệu vào:

Cho trong file **Chiakeo.inp**: Một dòng duy nhất gồm các giá trị là số kẹo trong mỗi gói, các số cách nhau bởi một dấu cách (N = số gói kẹo đọc được)

#### Dữ liệu ra:

Ghi ra file **Chiakeo.out**

– Dòng 1 lần lượt ghi 3 số: tổng số kẹo ở phần 1, phần 2 và độ chênh lệch giữa hai phần.

– Dòng 2,3 là giá trị các gói kẹo ở mỗi phần được chia.

Ví dụ:

CHIAKEO.INP	CHIAKEO.OUT
3 4 7 12	14 12 2
	3 4 7
	12

Nguyễn Duy Phi

### Bài 153/2003 – Phân phối hàng

Có N công trường cần vật liệu thi công. Công trường i cần cung cấp D[i] đơn vị hàng. Hàng được cung cấp từ hai kho A và B. Cước vận chuyển một đơn vị hàng từ kho A đến công trường i là A[i]. Cước vận chuyển một đơn vị hàng từ kho B đến công trường i là B[i]. Biết rằng kho A có R đơn vị hàng và tổng số hàng của hai kho đủ cung cấp cho N công trường.

**Yêu cầu:** Hãy phân phối hàng từ hai kho đến các công trường sao cho tổng cước phí vận chuyển là ít nhất.

#### Dữ liệu vào:

File **HANG.INP** có cấu trúc như sau:

Dòng 1: Ghi 2 số N, R ( $N \leq 10000$ )

Dòng 2: Ghi N số D[1], D[2], ..., D[N].

Dòng 3: Ghi N số A[1], A[2], ..., A[N].

Dòng 4: Ghi N số B[1], B[2], ..., B[N].

#### Dữ liệu ra:

File **HANG.OUT** có cấu trúc như sau:

Dòng 1: Ghi một số nguyên dương là tổng chi phí vận chuyển ít nhất.

Dòng 2: Ghi N số nguyên không âm tương ứng số đơn vị hàng mà kho A cung cấp cho các

công trường 1,2...N.

Dòng 3: Ghi N số nguyên không âm tương ứng số đơn vị hàng mà kho B cung cấp cho các công trường 1,2...N.

Ví dụ:

HANG.INP	HANG.OUT
7 22	835
3 9 7 6 11 10 8	3 0 7 6 6 0 0
13 17 20 9 30 10 19	0 9 0 0 5 10 8
25 16 22 9 30 4 11	

Trần Đức Thiện – Xóm Chùa – Xã Hương Lạc – Huyện Lạng Giang – Bắc Giang.

### Bài 154/2003 – Tuổi cha và con

Hai người đàn ông gặp nhau trên đường, một người hỏi "Cậu đi đâu đấy"?

– Mình đến trường mẫu giáo đón con, người kia trả lời

– Cậu có mấy con và các con cậu mấy tuổi?

– Mình có hai đứa. Tuổi của mình gấp 4 lần tuổi đứa lớn và gấp 7 lần tuổi đứa nhỏ, người bạn trả lời.

Bạn hãy cho biết người thứ hai và các con anh ta bao nhiêu tuổi?

### Bài 155/2003 – Bố trí phòng họp

Có N cuộc họp đánh số từ 1 đến N đăng ký làm việc tại một phòng hội thảo. Cuộc họp i cần bắt đầu vào thời điểm Ai và kết thúc vào thời điểm Bi ( $i=1,2,\dots,N$ ). Hai cuộc họp có thể nhận phục vụ nếu các khoảng thời gian làm việc tương ứng của chúng chỉ có thể giao nhau tại đầu mút hoặc tách rời nhau. Hãy tìm một lịch cho phòng hội thảo để có thể phục vụ nhiều cuộc họp nhất.

**Dữ liệu vào** từ file Activity.Inp gồm:

– Dòng đầu tiên ghi giá trị N ( $N \leq 1000000$ )

– Dòng thứ i trong N dòng tiếp theo ghi 2 số Ai và Bi cách nhau ít nhất một dấu cách. ( $A[i], B[i] \leq 32000$  nguyên dương)

**Kết quả:** Ghi ra file Activity.out gồm

– Dòng đầu tiên ghi k là số cuộc họp tối đa có thể bố trí

– Dòng tiếp theo ghi số hiệu của cuộc họp được phục vụ theo trình tự lịch bố trí

Ví dụ:

ACTIVITY.INP	ACTIVITY.OUT
5	3
1 3	1 4 5
2 4	
1 6	
3 5	
7 9	

(Đề ra của Trần Quang Đức)

### Bài 156/2003 – Bày tranh

Cho n bức tranh mã số từ 1..n ( $n \leq 50$ ). Người ta cần chọn ra một bức để đặt ở cửa phòng tranh, số còn lại được treo thẳng hàng trong phòng trên m vị trí định sẵn có mã số 1..m từ trái qua phải. Các bức tranh phải được treo theo trật tự nghiêm ngặt sau đây: tranh có số hiệu nhỏ phải treo ở trên tranh có số hiệu lớn.

Biết các thông tin sau về mỗi bức tranh:

- Tranh thứ  $i$  treo tại cửa sẽ đạt trị thẩm mỹ  $c[i]$ ;
- Tranh thứ  $i$  treo tại vị trí  $j$  sẽ đạt trị thẩm mỹ  $v[i,j]$ .
- $m+1 \geq n$ .

– Các giá trị thẩm mỹ là những số tự nhiên không vượt quá 50.

**Yêu cầu:** Hãy xác định một phương án treo tranh để có tổng trị thẩm mỹ là lớn nhất.

**Dữ liệu vào:** Tập văn bản 'Picture.INP'

- Dòng thứ nhất ghi  $n, m$  (cách nhau 1 dấu cách)
- Dòng tiếp theo là  $n$  giá trị  $c$ .
- Tiếp đến là  $n$  dòng, dòng  $i$  gồm  $m$  vị trí  $v[i,1], v[i,2], \dots, v[i,m]$ .

**Dữ liệu ra:** Tập văn bản 'Picture.OUT'

- Dòng thứ nhất ghi giá trị thẩm mỹ lớn nhất tìm được
- Dòng thứ hai: ghi mã số hiệu bức tranh treo ở cửa phòng tranh.
- Dòng thứ 3 ghi  $n-1$  số tự nhiên sắp tăng chặt cho biết mã số các vị trí được chọn để treo tranh

Ví dụ:

PICTURE.INP	PICTURE.OUT
3 4	40
1 2 0 1	2
1 10 1 3	2 4
2 1 2 2	
1 3 0 10	

Vũ Mạnh Hùng – lớp 10 A (Tin) – Trường THPTNK Ngô Sỹ Liên – tx Bắc Giang – Tỉnh Bắc Giang

### Bài 157/2003 – Biến đổi xâu

(Dành cho học sinh THPT)

Với một xâu ký tự  $S$  cho trước, ta có thể thực hiện các phép biến đổi sau:

- D: xoá một ký tự của xâu  $S$ . Ký hiệu D  $i$  trong đó  $i$  là vị trí ký tự cần xoá.
- I: chèn trước vị trí  $t$  của xâu  $S$  một ký tự  $c$  nào đó. Ký hiệu I  $t c$ .

Quy định thêm về vị trí chèn: nếu  $S$  có độ dài  $k$ , vị trí chèn có thể là  $1, 2, 3, \dots, k+1$ , chèn ở vị trí  $k+1$  có nghĩa là viết thêm vào cuối xâu  $S$ .

- R: thay ký tự thứ  $t$  của  $S$  bởi ký tự  $c$  nào đó. Ký hiệu R  $t c$ .

Giả sử  $X$  và  $Y$  là hai xâu ký tự. Độ dài xâu  $X$  là  $n$ , độ dài xâu  $Y$  là  $m$ .

**Yêu cầu:** Hãy tìm một dãy gồm ít nhất các phép biến đổi xâu  $X$  thành xâu  $Y$ . (Số phép biến đổi ít nhất gọi là khoảng cách giữa hai xâu).

Dữ liệu vào ghi trong file 'XAU.INP' gồm hai dòng:

- Dòng thứ nhất là xâu  $X$ .
- Dòng thứ hai là xâu  $Y$ .

Kết quả ghi ra file 'XAU. OUT'

- Dòng thứ nhất là  $K$ , đó là khoảng cách hai xâu.
- $K$  dòng tiếp theo mỗi dòng ghi ký hiệu một phép biến đổi theo trình tự thực hiện để biến xâu  $X$  thành xâu  $Y$ .

Ví dụ:

XAU.INP	XAU.OUT
ertyui	6
tyuhi	D 1
	D 1
	D 1
	D 1
	I h 4
	R 5 j

**Bài 158/2003 – Tuổi của hai anh em**

Các em nhỏ bắt đầu đi học khi được 6 tuổi (tính theo năm âm lịch). Sơn học ở lớp mà số của lớp bằng tuổi của em Dũng. Hỏi khi anh Sơn học xong phổ thông (hết lớp 12) thì em Dũng học xong lớp mấy?

**Bài 159/2003 – Dây con lỏi**

Dãy giá trị nguyên  $A=(A_1, A_2, \dots, A_N)$  được gọi là lỏi, nếu nó giảm dần từ  $A_1$  đến một  $A_i$  nào đó, rồi tăng dần tới  $A_N$ .

Ví dụ dãy lỏi: 10 5 4 2 -1 4 6 8 12

**Yêu cầu:** Lập trình nhập vào một dãy số nguyên, bằng cách xóa bớt một số phần tử của dãy và giữ nguyên trình tự các phần tử còn lại, ta nhận được dãy con lỏi dài nhất.

**Dữ liệu vào** trong file: Dayloi.inp có dạng

– Dòng đầu là N ( $N \leq 2000$ )

– Dòng tiếp theo là N số nguyên của dãy số (các số kiểu integer)

**Kết quả** ra file: Dayloi.out gồm:

– Dòng đầu tiên ghi số phần tử lớn nhất của dãy con tìm được

– Dòng tiếp theo ghi các số thuộc dãy con (không thay đổi trật tự các phần tử trong dãy ban đầu)

Ví dụ

DAYLOLINP	DAYLOLOUT
10	6
1 2 3 4 2 5 1 2 3 4	4 2 1 2 3 4

**Bài 160/2003 – Truyền tin trên mạng**

Trong một mạng gồm n máy tính đánh số từ 1 đến N. Sơ đồ nối mạng được cho bởi hệ thống gồm M kênh nối trực tiếp giữa một số cặp máy tính trong mạng. Biết chi phí truyền một đơn vị thông tin theo mỗi kênh nối của mạng.

Người ta cần chuyển một bức thông điệp từ máy S đến T. Để đảm bảo an toàn, người ta muốn chuyển bức thông điệp này theo K đường truyền tin khác nhau. Hai đường truyền tin được gọi là khác nhau nếu không có bất cứ kênh nối trực tiếp nào được dùng chung trên cả hai đường truyền tin. Chi phí của một đường truyền tin được hiểu là chi phí trên các kênh của nó.

**Yêu cầu :** Giả sử bức thông điệp có độ dài là 1 đơn vị thông tin, hãy tìm cách chuyển thông tin từ S đến T sao cho tổng chi phí chuyển thông tin (bằng tổng chi phí theo cả K đường truyền tin) là nhỏ nhất.

**Dữ liệu:** Vào từ file văn bản Ttin.INP:

– Dòng đầu tiên ghi năm số N,M,S,T,K cách nhau bởi dấu cách ( $N \leq 100$ ).

– M dòng sau mỗi dòng ghi ba số  $d_i, c_i, g_i$ : trong đó  $d_i, c_i$  là chỉ số của hai máy tương ứng có kênh nối và  $g_i$  (nguyên dương) là chi phí để truyền một đơn vị thông tin từ máy  $d_i$  đến máy  $c_i$  và ngược lại ( $i=1..n$ ).

**Kết quả:** Ghi ra file văn bản TTIN.OUT:

– Dòng đầu tiên ghi chi phí truyền thông điệp theo cách tìm được

– K dòng tiếp theo, mỗi dòng ghi đường truyền tin dưới dạng dãy có thứ tự các máy bắt đầu từ máy S và kết thúc ở máy T.

– Nếu không tìm đủ K đường đưa ra một dòng duy nhất: NO SOLUTION.

Ví dụ:

TTIN.INP	TTIN.OUT
5 7 1 5 2	24
1 2 3	1 2 3 5
1 4 8	1 4 5
2 3 5	
2 4 4	
3 5 5	
4 3 8	
4 5 3	

### Bài 161/2003 – Người bạn cũ

Gặp lại người bạn cũ đã có hai đứa con, tôi hỏi tuổi ba mẹ con. Là một nhà toán học, bạn tôi trả lời rằng tích của tuổi bạn tôi với tuổi hai đứa nhỏ là 2.450.

Sau khi suy nghĩ một lát, tôi nói "Điều bạn nói đưa ra quá nhiều khả năng. Thế tổng tuổi của ba mẹ con là bao nhiêu?"

"Bằng số tuổi của bạn" bạn tôi trả lời.

Điều đó vẫn khiến tôi phải lựa chọn nhưng quan sát kỹ tôi thấy hai đứa nhỏ không thể là anh em sinh đôi và tôi đã tìm ra câu trả lời.

Hỏi tuổi của bạn tôi là bao nhiêu? Biết rằng tôi năm nay 64 tuổi.

### Bài 162/2003 – Dây chuyền thông báo

Các học sinh trong một lớp học quyết định lập một dây chuyền thông báo như sau. Mỗi học sinh chọn một học sinh duy nhất khác làm **người kế tiếp** để truyền trực tiếp thông báo. Khi mỗi học sinh nhận được thông báo, anh ta sẽ truyền ngay cho người kế tiếp của mình. Dây chuyền thông báo được gọi là tốt nếu nó thỏa mãn điều kiện: Khi một học sinh  $A_1$  bắt kỳ gửi thông báo cho người kế tiếp  $A_2$ ,  $A_2$  lại gửi cho người kế tiếp  $A_3, \dots$ , cứ như vậy thì cuối cùng thông báo sẽ đến mọi người trong lớp kể cả người ban đầu ( $A_1$ ) đã phát ra thông báo. Không nhất thiết mọi dây chuyền thông báo là tốt.

**Bài toán đặt ra là:** Cho trước một dây chuyền thông báo, hãy tìm số ít nhất việc thay đổi người kế tiếp để có thể nhận được một dây chuyền thông báo tốt.

**Dữ liệu vào** được cho bởi file văn bản TB.INP trong đó dòng thứ nhất ghi số  $N < 10000$  là số học sinh trong lớp, các học sinh này có tên từ 1 đến  $N$ . Trong dòng tiếp theo ghi  $N$  số, số thứ  $i$  là tên người kế tiếp của học sinh  $i$ .

**Kết quả** ghi ra file TB.OUT như sau: dòng thứ nhất ghi số  $K$  là số thay đổi cần tiến hành (nếu dây chuyền thông báo đã cho là tốt thì  $K=0$ ). Nếu  $K>0$ , trong  $K$  dòng tiếp theo, mỗi dòng ghi hai tên học sinh, người sau là người kế **tiếp mới được thay đổi** của người trước.

**Ví dụ**

TB.INP	TB.OUT
10	3
6 9 2 7 3 1 10 3 6 9	1 4
	10 8
	8 5

### Bài 163/2003 – Bán hàng

Một người đi bán hàng dọc theo các chợ trên một trục đường. Nhà anh ta ở đầu trục đường, ngay cạnh cái chợ thứ nhất. Các chợ được đánh số từ 1 đến  $n$  (chợ  $n$  ở cuối trục đường). Thời gian để đi từ chợ thứ  $i$  đến chợ thứ  $(i+1)$  là  $t(i)$ . Thời gian để bốc dỡ hàng là không đáng kể cho nên người ta bỏ qua. Tại một chợ số hàng bán được thay đổi theo thời gian.

Chẳng hạn tại chợ thứ  $i$ , với  $i=1,...,n$ , số hàng có thể bán được ở mỗi phút trong  $g(i,1)$  phút đầu là  $K(i,1)$ , mỗi phút trong  $g(i,2)$  phút tiếp theo là  $K(i,2),...,$  và sau  $q(i)$  phút, số hàng bán được là  $h(i)=k(i,1) + k(i,2) + ... + k(i,q(i))$   
 $q(i) \leq 5, h(i) \leq 100$ .

Viết chương trình chỉ ra thời gian dừng lại ở các chợ để bán hàng sao cho số hàng bán được là nhiều nhất.

**Dữ liệu vào** được cho trong file văn bản BANHANG.INP.

Dòng đầu tiên là số chợ  $n$  ( $n \leq 100$ ), và số phút người này có thể đi và bán hàng trong các chợ. Số phút này có thể không quá 1800.

Dòng thứ hai chứa  $n-1$  số  $t(i)$ .

Trong dòng thứ  $2+i$ , với  $i=1,...,n$ , chứa  $q(i)+1$  số; số đầu tiên là  $q(i)$  và sau đó là  $q(i)$  số  $g(i,1), g(i,2),..., g(i,q(i))$ .

Từ dòng thứ  $n+3$  là thông tin về số hàng có thể bán được ở tại mỗi chợ ở các thời điểm. Trên dòng thứ  $n+2+i$ , với  $i=1,2,...,n$  chứa  $q(i)+1$  số, số đầu tiên là  $q(i)$  và sau đó là  $q(i)$  số  $K(i,1), K(i,2),..., K(i,q(i))$ . Trên mỗi dòng, các số cách nhau ít nhất một dấu cách.

**Kết quả** đưa ra trong file BANHANG.OUT có dạng như sau:

Dòng đầu ghi  $n$  số, chỉ số phút dừng lại ở mỗi chợ.

Dòng thứ hai là số lượng hàng lớn nhất có thể bán được

**Ví dụ:**

**BANHANG.INP**

```
4 5
1 2 3
2 1 2
4 1 1 2 2
4 2 1 1 1
3 2 2 1
2 10 9
4 15 13 12 9
4 50 46 42 8
3 30 30 8
```

**BANHANG.OUT**

```
0 0 2 0
100
```

### **Bài 164/2003 – Rán bánh**

Một cái chảo có thể rán 6 miếng bánh mì. Để rán 1 mặt của mỗi miếng bánh cần 30 giây. Hỏi thời gian ít nhất cần để rán 9 miếng bánh, 15 miếng bánh, 33 miếng bánh là bao nhiêu.

### **Bài 165/2003 – Tìm vị trí**

Một đại lý kinh doanh xăng dầu có  $n$  trạm bán xăng dầu (gọi tắt là cây xăng) đánh số từ 1 đến  $n$  trên một đường cao tốc muốn tìm vị trí đặt  $k$  bể chứa xăng để cung ứng xăng cho các cây xăng. Trên đường cao tốc người ta đặt các cột mốc cây số, bắt đầu từ cột số 0. Biết vị trí của cây xăng thứ  $i$  là ở cột cây số  $d_i$  ( $i=1,2,...,n$ ):  $d_1 < d_2 < ... < d_n$

**Yêu cầu:** Tìm vị trí đặt  $k$  bể chứa xăng tại  $k$  trong số  $n$  cây xăng sao cho khoảng cách lớn nhất từ cây xăng không có bể chứa đến cây xăng có bể chứa gần nó nhất là nhỏ nhất.

**Dữ liệu:** Vào từ file văn bản VITRI.INP

– Dòng đầu tiên ghi 2 số nguyên dương  $n, k$  ( $n < 200, k < 30, k$

– Dòng thứ 2 ghi các số  $d_1, d_2, ..., d_n$  ( $d_i$  là các số nguyên dương không quá 320000). Các

số trên cùng một dòng ghi cách nhau ít nhất một dấu trắng.

**Kết quả:** Ghi ra file văn bản VITRI.OUT

File gồm k dòng, mỗi dòng ghi chỉ số cây xăng đặt bể chứa xăng.

Ví dụ:

VITRI.INP	VITRI.OUT
6 3	2
5 6 12 19 20 27	4
	6

### Bài 166/2003 – Đếm cây trên đồi thông

Trong một chuyến du lịch bằng trực thăng tới đồi thông Đà Lạt, một vị khách đến từ Nhật rất ngạc nhiên trước vẻ đẹp kỳ thú của rừng thông nơi đây. Ông rất thích thú và muốn đếm xem có bao nhiêu cây thông (để đem về so với số cây ít ỏi ở đất nước công nghiệp của ông!), nhưng việc đếm chúng chẳng dễ dàng chút nào. Rất may trên cùng chuyến bay có một sinh viên Việt Nam đã cho ông mượn một máy chụp Digital đặc biệt để chụp toàn cảnh đồi thông vào một tệp có tên là **doithong.fit** với quy cách đặc biệt như sau: Mỗi cây thông được biểu diễn bằng các ký tự '@' xếp thành hình tam giác cân có thêm 1 chữ '@' ở giữa đáy làm thân cây, chẳng hạn trong tệp **doithong.fit** có 1 cây thông kích thước bằng 3 (là chiều cao tán lá, không tính thân):

		@		
	@	@	@	
@	@	@	@	@
		@		

Mọi cây thông trong **doithong.fit** đều có đỉnh quay lên trên và trong đó không có bất kỳ loại cây nào khác, các cây thông không dính vào nhau, chẳng hạn 2 cây thông sau là dính vào nhau (một cây thông kích thước 3 và một cây thông kích thước 1):

		@		
	@	@	@	
@	@	@	@	@
		@		@

Khi về nước vị khách người Nhật quên không đem về chương trình giải mã tệp **doithong.fit** và ông ấy không thể mở tệp ra đếm một số lượng cây thông lớn như vậy được. Bạn hãy giúp vị khách viết chương trình đọc tệp **doithong.fit** và đưa ra tệp **doithong.hut** số lượng cây thông trên đồi thông, lưu ý là số lượng dòng trong tệp **doithong.fit** rất lớn ( $\leq 999$ ).

Ví dụ:

doithong.fit									
		@			@			@	
	@	@	@		@			@	
@	@	@	@	@					
		@						@	
								@	
	@				@				
@	@	@			@	@	@		
	@			@	@	@	@	@	
			@	@	@	@	@	@	@
					@				

doithong.hut
6

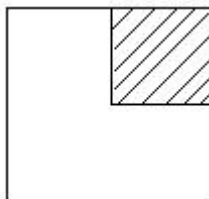
Đinh Mạnh Đạt – [dingmanhdat@vol.vnn.vn](mailto:dingmanhdat@vol.vnn.vn)



**Bài 167/2004 – Chia đất**

Một người cha có 4 đứa con trai. Ông ta có 1 mảnh vườn hình vuông và ông để lại cho mình 1 phần 4 mảnh vườn như hình vẽ.

Ông hứa sẽ cho 4 đứa con phần đất còn lại nếu họ biết cách chia đất thành 4 phần bằng nhau về diện tích và hình dáng. Những người con phải chia như thế nào để được hưởng đất?

**Bài 168/2004 – Đoán số**

Bạn Tin nghĩ một số (gọi là số S) gồm 4 chữ số (không nhất thiết phải khác nhau) trong 6 chữ số từ 1 đến 6. Để tìm số đó, máy lần lượt đưa ra các số dự đoán (gọi là M), mỗi số gồm 4 chữ số (không nhất thiết phải khác nhau). Với mỗi số dự đoán, máy nhận được hai câu trả lời của bạn Tin cho hai câu hỏi sau:

- 1) Có bao nhiêu chữ số trong số M là chữ số trong S nhưng vị trí xuất hiện của mỗi chữ số đó là sai?
- 2) Có bao nhiêu chữ số trong số M là chữ số trong S và đồng thời vị trí xuất hiện của mỗi chữ số đó đều đúng?

**Ví dụ:**

Bạn Tin nghĩ số 4655 (số S). Một cách để tìm số đó là:

TT	Số dự đoán của máy (Số M)	Trả lời của bạn Tin	
1	1234	1) 1	2) 0
2	5156	2	1
3	6165	2	1
4	5625	1	2
5	5653	1	2
6	4655	0	4

**Yêu cầu:**

- a) Hãy thực hiện lên màn hình lần lượt các số máy dự đoán và với mỗi số đó nhận hai câu trả lời (từ bàn phím) của bạn Tin cho đến khi đưa ra được số đúng như bạn Tin nghĩ
- b) Hãy thực hiện yêu cầu a) với điều kiện số lần hỏi – đáp không nhiều quá 6. Nếu không làm được như vậy, thì hãy cố gắng để số lần hỏi – đáp càng ít càng tốt.

**Bài 169/2004 – Món quà đầu năm**

Huda là vùng nổi tiếng trên thế giới với sự hiện đại kết hợp với sự huyền bí của mình. Mỗi người ở đây có một con số riêng của mình và không ai giống ai cả (như "số hiệu" vậy!). Ở Huda 3, 5, 7 được coi là những con số đem lại sự may mắn, vì vậy mỗi khi năm mới về mọi người đều tặng và nhận quà tập thể bằng cách rất độc đáo mang "dấu ấn" của những con số này.

Các món quà được đánh số "may mắn" từ nhỏ đến lớn và đặt ở quảng trường lớn. Số "may mắn" là số nguyên dương chỉ chia hết cho 3, 5 và 7 mà không chia hết cho số nguyên tố nào khác (nghĩa là có dạng  $3^i \cdot 5^j \cdot 7^k$ ). Số món quà đúng bằng số người ở đây nên mỗi người được nhận đúng một món quà tương ứng với "số hiệu" của mình về thứ tự từ nhỏ đến lớn (người có "số hiệu" lớn thứ N thì sẽ nhận món quà có số lớn thứ N). Những người có "số hiệu" 1, 2, 3, 4.. sẽ nhận những món quà "may mắn" tương ứng có số là 3, 5, 7, 9.. Giao

thừa nào cả Huda cũng nhận nhịp, mọi người đổi "số hiệu" cho nhau chỉ trước vài giờ nhưng rồi ai cũng nhanh chóng tìm ra món quà may mắn cho mình. Tuy nhiên những người mới đến Huda thì rất bối rối không biết từ "số hiệu" của mình bằng cách nào tìm ra số may mắn của món quà (hàng năm có trên dưới 1000 người mới tới Huda).

Bạn có thể giúp họ nhanh chóng tìm ra món quà may mắn của mình không? Bằng một chiếc máy tính với ngôn ngữ lập trình bạn ưa thích! Trong email họ gửi HelpUs.txt có chứa "số hiệu" của họ (nhỏ hơn 30000), có dạng như sau: số đầu tiên là số người mới đến Huda các số tiếp theo là "số hiệu" của họ. Bạn hãy lập trình đưa ra file ForYou.txt chứa các số may mắn trên món quà của họ, các số cách nhau bởi dấu trắng.

Một năm mới của Huda sẽ may mắn hơn nhờ chương trình của bạn đó!

Ví dụ:

HelpUs.txt
4 5 6 7 8

ForYou.txt
15 21 25 27

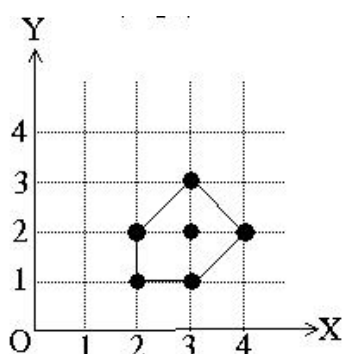
### Bài 170/2004 – Xây dựng hàng rào

Khu giải trí mới xây Princessland bao gồm rất nhiều điểm vui chơi trên một vùng đất rộng lớn. Người ta cần xây dựng hàng rào bảo vệ hình đa giác lồi có diện tích nhỏ nhất quanh khu giải trí có các cạnh hướng song song, vuông góc hoặc nghiêng  $45^\circ$  so với xích đạo (4 hướng). Một vệ tinh chụp được toàn cảnh Princessland, ở độ cao trên 900 km nên các điểm vui chơi chỉ coi như một điểm, thậm chí có nhiều điểm vui chơi như trùng thành một điểm, chúng có thể ở bên trong, trên cạnh hay trùng đỉnh của "đa giác hàng rào". Các cặp tọa độ của chúng theo bản đồ phẳng Princessland (trục OX song song với xích đạo) được gửi về trái đất để tính toán. Bạn hãy lập trình tìm phương án xây dựng hàng rào bảo vệ tốt nhất cho Princessland. Tập **Princess.inp** dòng đầu chứa N là số điểm vui chơi (giả sử rằng  $N \leq 32000$ !), N dòng tiếp chứa N cặp tọa độ (x,y) của chúng ( $0 \leq x, y \leq 1000$ ) có thể trùng nhau. Bạn cần đưa ra tập **Princess.out**, dòng đầu ghi M là số cạnh của hàng rào, M dòng tiếp theo ghi M cặp tọa độ nguyên (x,y) của hàng rào bảo vệ theo trình tự ngược chiều kim đồng hồ. (Bài toán này được gọi là bài toán "Bao lồi chuẩn")

Ví dụ:

Princess.inp
6
3 3
3 1
2 2
4 2
3 2
2 1

Princess.out
5
3 1
4 2
3 3
2 2
2 1



**Bài 171/2004 – Từ nhà tới trường**

Tuấn đi từ nhà tới trường hết 20 phút. Có một lần đang đi trên đường, Tuấn chợt nhớ rằng mình để quên bút ở nhà. Tuấn biết rằng nếu cứ tiếp tục đi đến trường với vận tốc đang đi thì sẽ đến sớm 8 phút trước khi có trống bắt đầu vào lớp, còn nếu quay về nhà lấy bút vẫn với vận tốc ấy, thì sẽ bị muộn giờ 10 phút. Hỏi Tuấn đã đi được bao nhiêu phần quãng đường.

**Bài 172/2004 – Nhà gương cười**

Ban quản lý nhà gương cười muốn thay đổi lại toàn bộ các tấm gương phủ tường của nhà gương để phục vụ tốt hơn khách tham quan. Nhà gương hiện tại được mô tả bởi **bảng ký tự** kích thước  $N \times N$  ( $3 \leq N \leq 33$ ). Một số ô của bảng chứa dấu chấm '.' để ký hiệu ô trống, một số ô khác chứa dấu thăng '#' để ký hiệu ô vuông được **bao bọc bởi các bức tường**. Tất cả các ô vuông đều có kích thước  $3 \times 3$  mét.

Người ta đặt gương xung quanh nhà gương, ngoại trừ ô ở góc trên trái và ô ở góc dưới phải tương ứng với lối vào và ra của nhà gương. Giả thiết rằng ô ở góc trên trái và dưới phải của bảng luôn chứa dấu chấm. Hệ thống gương cũng được đặt bao quanh các ô có tường, tức là các ô có dấu thăng.

Bạn cần giúp ban quản lý tính diện tích gương cần mua hay diện tích của các bức tường ở phía trong của nhà gương là phần nhìn thấy được bởi du khách vào chơi. Biết rằng chiều cao mỗi bức tường đều là 3 mét.

**Dữ liệu:** từ tệp MIRROR.INP

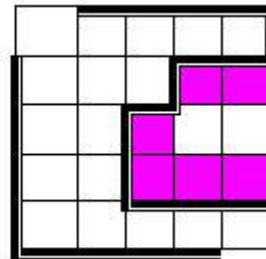
– Dòng đầu tiên chứa số  $N$ .

–  $N$  dòng tiếp là các dấu chấm hay dấu thăng mô tả nhà gương.

**Kết quả:** ghi ra tệp MIRROR.OUT diện tích gương cần mua.

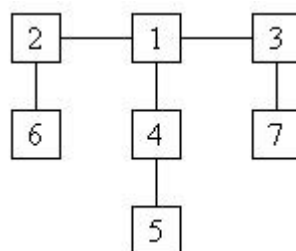
**Ví dụ và hình vẽ minh họa:** Với ví dụ này thì đường nét đậm trong hình minh họa chính là các vị trí cần đặt gương. Các ô được bôi đen biểu thị các ô chứa dấu thăng.

MIRROR.INP	MIRROR.OUT
<pre> 5 ..... ...## ...#.. ...### ..... </pre>	198

**Bài 173/2004 – Cân bằng**

Coi một cây  $T$  với  $N$  ( $1 \leq N \leq 20000$ ) nút được đánh số từ  $1..N$ . Hai nút hoặc là nối với nhau bởi một cạnh duy nhất hoặc không nối với nhau. Xóa bất cứ nút nào trong cây sẽ sinh ra một rừng: rừng là một tập hợp một hoặc nhiều cây. Định nghĩa cân bằng của một nút là kích cỡ của cây lớn nhất trong rừng  $T$  được tạo bởi bằng cách xóa nút  $T$ .

**Ví dụ:** cho một cây:



Xóa nút 4 tạo ra hai cây với các nút của chúng là  $\{5\}$  và  $\{1,2,3,6,7\}$ . Cây lớn hơn trong hai cây có năm nút, do đó cân bằng của nút 4 là năm...

Dữ liệu vào là một **cây**, tính xem **nút** nào có **cân bằng** nhỏ nhất. Nếu nhiều **nút** có cùng **cân bằng**, hãy in ra **nút** có thứ tự bé nhất.

**INPUT:** file BALANCE.INP

Dòng đầu: N Mỗi dòng trong N-1 dòng tiếp theo có hai số chỉ hai điểm của của một cạnh trong **cây**. Không có cạnh nào xuất hiện hai lần trong file, tất cả các cạnh có trong **cây** đều được thông báo. **OUTPUT** file BALANCE.OUT

Dòng đầu là số thứ tự của **nút** có cân bằng nhỏ nhất Dòng tiếp là cân bằng của **nút** đó

**Ví dụ:**

INPUT	OUTPUT
7	1
2 6	2
1 2	
1 4	
4 5	
3 7	
3 1	

(Đề ra của Phạm Cường – CTV)

### Bài 174/2004 – Cân cặp

Hai học sinh Tuấn và An nhìn thấy một cái cân và lần lượt đặt cặp của mình lên cân. Cân chỉ ra rằng 1 cặp nặng 3 kg, cặp kia nặng 2kg. Sau đó 2 cậu bé đặt cả hai cặp lên cân thì thấy cân chỉ 6kg.

– Sao lại thế được – An hỏi bạn – Chẳng lẽ 3 cộng 2 bằng 6.

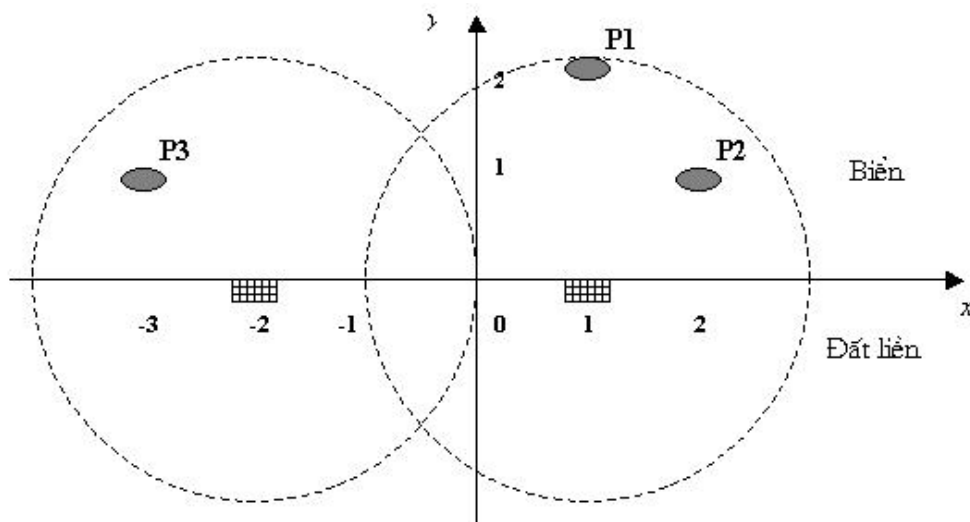
– Cậu không thấy là kim trên mặt đĩa cân hơi bị lệch hay sao, Tuấn trả lời.

Vậy thực tế cặp của các bạn cân nặng bao nhiêu kg?

### Bài 175/2004 – Trạm Radar

Coi như bờ biển là một đường thẳng chia mặt phẳng làm hai phần: Đất liền và biển. Mỗi hòn đảo nhỏ là một điểm nằm bên phía biển. Mỗi trạm Radar nằm trên bờ biển có khoảng phủ sóng là  $d$ , do đó mỗi hòn đảo trên biển có thể bị theo dõi bởi một trạm Radar nếu khoảng cách giữa chúng không quá  $d$ .

Chúng ta sử dụng hệ toạ độ Đề các với bờ biển là trục  $x$ . Phần biển nằm phía trên trục  $x$ , còn phần đất liền ở phía dưới. Cho vị trí các đảo và khoảng bao phủ của Radar, xác định số Radar tối thiểu cần thiết để theo dõi được tất cả các đảo. Vị trí của mỗi hòn đảo cho bởi toạ độ  $(x,y)$ .



**Input: RADAR.INP**

File input là một bộ gồm nhiều test. Dòng đầu mỗi test gồm 2 số nguyên  $n$  ( $n \leq 1000$ ) và  $d$ , trong đó  $n$  là số đảo trên biển,  $d$  là khoảng bao phủ của Radar. Mỗi dòng trong  $n$  dòng tiếp theo là hai số nguyên chỉ toạ độ của từng đảo. Các test cách nhau một dòng trống. Dữ liệu vào kết thúc bởi một dòng gồm hai số 0

**Output: RADAR.OUT**

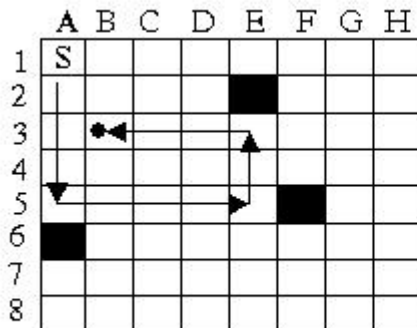
Mỗi dòng là một số nguyên thể hiện số Radar tối thiểu tìm được cho mỗi test. Ghi -1 nếu không có cài đặt được

**Ví dụ:**

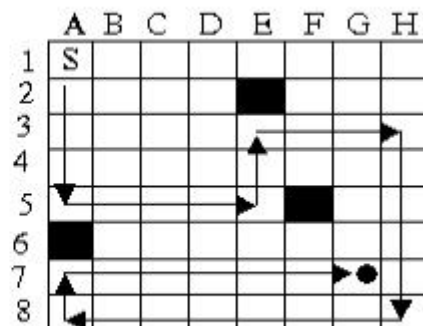
RADAR.INP	RADAR.OUT
3 2	2
1 2	1
-3 1	
2 1	
1 2	
0 2	
0 0	

**Bài 176/2004 – Đường đi của con rắn**

Một con rắn sống trên một lưới vuông  $8 \times 8$  có chứa một vài vật cản (là các ô màu đen trong hình vẽ dưới đây). Con rắn di chuyển theo đường thẳng (khi còn có thể di chuyển được). Khi nó gặp vật cản hay gặp biên của lưới vuông, nó sẽ quay sang ô bên phải hay bên trái của nó. Nếu nó đối diện với ô đã đi qua, nó dừng hẳn.



Câu A: Đường đi khi luôn quay sang trái (13 ô)



Câu B: Đường đi qua nhiều ô nhất

Con Rắn luôn bắt đầu tại ô A1, và luôn bắt đầu việc di chuyển bằng cách đi hướng xuống dưới. Vị trí của các vật cản (tức là các ô đen) được cho biết trước.

**Yêu cầu:**

Tìm số ô mà con rắn đi qua cho đến khi không đi được nữa và thỏa mãn:

- Con rắn luôn đi về phía bên trái của nó.
- Con rắn đi sao cho số ô đi qua là lớn nhất

**Ví dụ:** Hình vẽ trên cho ta hình ảnh của một lưới với ba ô đen (tại các vị trí A6, E2 và F5). Đường đi thỏa câu a) là: đi từ A1 đến A5, quay sang trái đến E5. Từ E5 quay trái đến E3, quay trái, đi thẳng và dừng ở B3. Con rắn đã đi qua 13 ô.

Đường đi thỏa câu b) được cho ở lưới vuông bên phải. Đường đi nói trên đi qua 32 ô.

**Dữ liệu vào:** Ghi trên file SNAL.INP gồm  $N+1$  dòng:

– Dòng đầu ghi số nguyên dương  $N$  ( $1 \leq N \leq 32$ ) chỉ số các ô đen.

– Trên N dòng tiếp theo, mỗi dòng ghi tọa độ một ô đen trên dưới dạng Xn, trong đó X (là các chữ cái la tinh viết hoa từ A đến Z) chỉ cột và n ( $1 \leq n \leq 8$ ) chỉ dòng.

Chú ý rằng không có ô đen tại các ô A1 và A2.

**Dữ liệu ra:** Ghi trên file SNAL.OUT gồm 2 dòng:

- Dòng đầu ghi số bước thoả yêu cầu a).
- Dòng thứ nhì ghi số bước thoả yêu cầu b).

**Ví dụ:**

SNAIL.INP	SNAIL.OUT
3	13
A6	32
E2	
F5	

### **Bài 177/2004 – Đàn gia súc**

Một người chăn gia súc chăn 100 con vật và anh ta nhận được tiền công là 200 đồng. Chăn một con trâu anh ta nhận được 20 đồng, chăn một con bò anh ta nhận được 10 đồng, chăn một con nghé anh ta nhận được 1 đồng. Hỏi trong đàn gia súc, mỗi loại có bao nhiêu con?

### **Bài 178/2004 – Sự lựa chọn**

Trong một kỳ thi dự báo kết quả thi đấu trận chung kết tranh giải bóng đá vô địch Seagames có N ( $N < 101$ ) người có kết quả dự đoán đúng. Để lựa chọn người được giải thưởng, ban tổ chức quyết định xếp hàng N người vào một vòng tròn đánh số từ 1 đến N theo chiều ngược chiều kim đồng hồ (tức là người đánh số N đứng bên trái người đứng số 1), sau đó lựa chọn một cách ngẫu nhiên 4 số nguyên dương m, k, p, q, trong đó m,  $k < N$ , còn p, q là số nguyên dương tùy ý. Tiếp theo người được nhận giải thưởng sẽ được xác định nhờ thủ tục loại bỏ dần sau đây:

Ban tổ chức cử ra 2 nhân viên, nhân viên thứ nhất bắt đầu đếm từ người đứng ở vị trí m theo chiều ngược với kim đồng hồ từ 1 đến p, còn nhân viên thứ 2 bắt đầu đếm từ người đứng ở vị trí thứ k theo chiều kim đồng hồ từ 1 đến q. Hai người đứng ở vị trí kết thúc đếm của hai nhân viên sẽ rời khỏi hàng. Đến đây kết thúc một lượt đếm. Lưu ý rằng, nhân viên thứ 2 đếm đồng thời với nhân viên thứ 1, do đó anh ta có thể đếm cả người ở vị trí kết thúc đếm của nhân viên thứ nhất, ngoài ra, nếu vị trí kết thúc đếm của hai nhân viên là trùng nhau thì chỉ có một người phải rời khỏi hàng. Sau khi những người không may mắn đã rời khỏi hàng, mỗi nhân viên lại tiếp tục lượt đếm mới bắt đầu từ người đứng cạnh người vừa bị loại theo vòng đếm của họ. Người phải rời khỏi hàng cuối cùng theo thủ tục lựa chọn trên là người được nhận giải. Nếu ở lượt đếm cuối cùng, có hai người phải đồng thời rời khỏi hàng thì cả hai người này đều được nhận giải.

**Dữ liệu vào** trong file văn bản có tên **CHON.INP** trong đó ghi 5 số N, m, k, p, q cách nhau bởi ít nhất một dấu cách.

**Kết quả** ghi ra file văn bản **CHON.OUT** có cấu trúc như sau:

- Dòng đầu tiên ghi số L là số lần thực hiện việc đếm.
- Mỗi dòng thứ i trong số L dòng tiếp theo ghi chỉ số của những người rời khỏi hàng ở lượt đếm thứ i ( $i = 1, 2, \dots, L$ ).

**Ví dụ:**

CHON.INP	CHON.OUT
10 1 10 4 3	6 4 8 9 5 3 1 2 6 10 7

### Bài 179/2004 – Toạ độ tam giác

Cho các điểm trên lưới tam giác đều vô hạn như hình dưới đây:

```

      *
    * *
  * * *
* * * *
* * * * *
* * * * *

```

.....

Các điểm này được đánh số theo thứ tự từ trái qua phải, từ trên xuống dưới (mỗi dấu \* biểu thị một số). Một nhóm những điểm này sẽ tạo thành đỉnh của một hình học nhất định. Ví dụ như tập hợp các điểm {1,2,3} là đỉnh của một hình tam giác, tập hợp {11,13,26,24} là đỉnh của một hình bình hành, tập hợp {8,10,17,21,32,34} là đỉnh của một lục giác đều.

**Yêu cầu:** Viết chương trình đọc từng tập hợp điểm trên lưới tam giác, xét xem chúng có là tập đỉnh của một trong những loại hình sau không: tam giác đều, hình thoi, lục giác đều. Hình đó phải thỏa mãn cả hai điều kiện sau:

1. Các cạnh của hình phải song song với cạnh của lưới.
2. Các cạnh của hình phải có độ dài bằng nhau.

**Input:** File VERTICES.INP gồm nhiều tập hợp điểm. Mỗi tập hợp được ghi trên một dòng và không có quá 6 số trong khoảng từ 1..2.000.000.000

**Output:** File VERTICES.OUT.

Với mỗi tập hợp điểm nhận được, phải xét xem nếu là đỉnh của hình tam giác đều thì ghi 3, là đỉnh của hình thoi thì ghi 4, là đỉnh của lục giác đều thì ghi 6. Nếu không thuộc các trường hợp trên thì ghi 0. Mỗi số ghi trên một dòng.

**Ví dụ:**

VERTICES.INP
1 2 3
11 13 29 31
26 11 13 24
4 5 9 13 12 7
1 2 3 4 5
47
8 10 17 21 32 34

VERTICES.OUT
3
0
4
6
0
0
6

Đề ra Phạm Mạnh Cường – CTV

### Bài 180/2004 – Gard

Một toán công nhân gồm N người đánh số hiệu từ 1 đến N được bố trí làm việc trên một toà

nhà có  $N$  tầng. Do tính chất của công việc nên các công nhân phải được bố trí làm việc theo các tầng liên tục từ tầng 1 và mỗi tầng không quá  $K$  người ( $K \leq N$ ) (như vậy trên các tầng cao có thể không có công nhân làm việc).

**Yêu cầu:** Xác định xem có bao nhiêu cách khác nhau bố trí công nhân làm việc trên các tầng thoả mãn yêu cầu nói trên. Hai cách bố trí xem như khác nhau nếu trong hai cách bố trí đó có ít nhất một công nhân được phân công khác nhau.

Ví dụ  $N = 3$ ,  $K = 2$  có tất cả 12 cách bố trí theo bảng dưới đây: trong bảng, từ cột thứ hai các ô ghi tên các công nhân làm việc trên tầng ghi ở cột 1

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

**Dữ liệu:** Trong file văn bản GARD.INP gồm một dòng ghi hai số  $N$   $K$  ( $1 \leq K \leq N \leq 50$ ).

**Kết quả:** Đưa ra file văn bản GARD một dòng duy nhất chứa số  $M$  – số cách bố trí khác nhau tìm được.

**Ví dụ:**

GARD.INP
3 2

GARD.OUT
12

### Bài 181/2004 – Những quả cam trong giỏ

Có 3 cái giỏ, một màu nâu, một màu đỏ và một màu hồng, chứa tất cả 10 quả trứng. Trứng trong giỏ màu nâu nhiều hơn 1 quả so với trong giỏ màu đỏ. Trứng trong giỏ màu đỏ ít hơn 3 quả so với giỏ màu hồng.

Hỏi có bao nhiêu quả trứng trong mỗi giỏ?

### Bài 182/2004 – Dây số hạnh phúc

Tại vương quốc Ba Tư xa xưa, người ta thường tổ chức các cuộc thi tìm dãy số hạnh phúc: Các chàng trai, cô gái thông minh trong thời gian ngắn nhất phải tìm ra được một dãy số hạnh phúc có nhiều phần tử nhất.

Dãy số tự nhiên  $a_1, a_2, \dots, a_k$  được gọi là hạnh phúc nếu nó thoả mãn các điều kiện sau:

– Dãy trên là một dãy giảm dần.

– Với mọi  $i$ ,  $a_i$  hoặc là số nguyên tố, hoặc phải là ước của một trong các số  $a_1, a_2, \dots, a_{i-1}$ .

**Ví dụ:** 8 5 3 2 là dãy hạnh phúc.

**Yêu cầu:** Hãy viết chương trình giúp các chàng trai, cô gái Ba Tư để: Nhập một số tự nhiên  $N$  từ bàn phím và in ra màn hình một dãy số hạnh phúc càng dài càng tốt với số hạng đầu tiên là  $N$ .

### Bài 183/2004 – Số Catalan

Với  $N$  cho trước, xét các dãy số  $A = (A_0, A_1, A_2, \dots, A_{2N})$ , trong đó:

–  $A_i$  – nguyên, không âm.

–  $A_0 = A_{2N} = 0$ .

–  $|A_i - A_{i+1}| = 1$

Số lượng các dãy số  $A$  thoả mãn các tính chất trên là số Catalan. Các dãy số  $A$  có thể được sắp xếp theo thứ tự từ điển.

Ví dụ, với  $N = 3$  ta có 5 dãy số:



- 1) 0 1 0 1 0 1 0
- 2) 0 1 0 1 2 1 0
- 3) 0 1 2 1 0 1 0
- 4) 0 1 2 1 2 1 0
- 5) 0 1 2 3 2 1 0

**Yêu cầu:**

Tìm số Catalan theo  $N$ ,  $N \leq 60$ .

Cho thứ tự từ điển, tìm dãy số  $A$ .

Cho dãy số  $A$ , tìm thứ tự từ điển.

**Dữ liệu:** Vào từ file CATALAN.INP:

– Dòng đầu tiên – số nguyên  $N$ .

– Các dòng sau có dạng: 1  $K$  – tìm dãy số  $A$  có thứ tự từ điển là  $K$ .

– Hoặc 2 *Dãy số  $A$*  – tìm thứ tự từ điển của  $A$ , các số trên một dòng cách nhau một dấu cách.

– Kết thúc là dòng chứa một số 0.

**Kết quả:** Đưa ra file CATALAN.OUT:

– Dòng đầu tiên: Số Catalan.

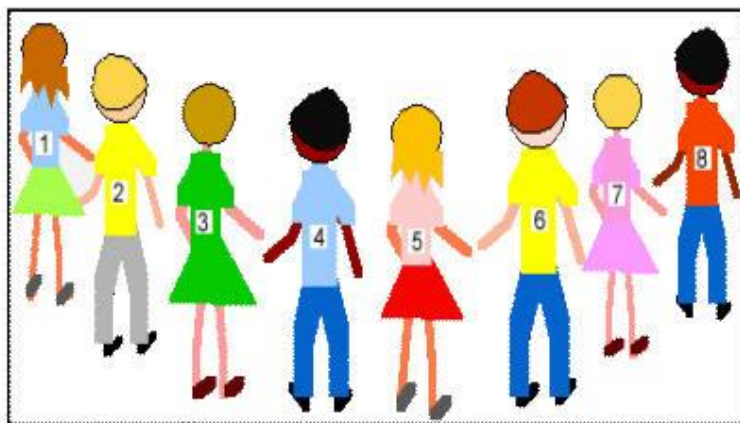
– Các dòng sau có dạng: 1 *Dãy số  $A$*  hoặc 2  $K$ .

**Ví dụ:**

CATALAN.INP	CATALAN.OUT
3	5
1 2	1 0 1 0 1 2 1 0
2 0 1 2 3 2 1 0	2 5
1 4	1 0 1 2 1 2 1 0
0	

**Bài 184/2004 – Các học sinh trong một lớp học**

Một lớp học gồm 30 học sinh, các học sinh được đánh số từ 1 đến 30. Đây là hình ảnh một vài học sinh của lớp:



Thầy giáo của họ dùng 6 chiếc ghế xếp thành một vòng tròn và xếp các học sinh quanh vòng tròn theo cách sau: Học sinh mang số 1 đứng sau chiếc ghế thứ nhất, học sinh mang số 2 đứng sau chiếc ghế thứ 2, học sinh mang số 3 đứng sau ghế thứ 3, cứ tiếp tục như vậy cho tới học sinh mang số 6. Học sinh mang số 7 sẽ được xếp đứng sau học sinh mang số 1, học sinh mang số 8 xếp đứng sau học sinh mang số 2, vv... Các học sinh còn lại tiếp tục được xếp theo cách này cho đến khi tất cả các học sinh trong lớp hoặc là đứng sau ghế hoặc đứng sau một học sinh khác. Bạn hãy cho biết:

– Có bao nhiêu học sinh đứng sau mỗi chiếc ghế?

– Những học sinh nào đứng sau học sinh mang số 3?

### **Bài 185/2004 – Ông Ngâu bà Ngâu**

Hắn các bạn đã biết ngày "ông Ngâu bà Ngâu" hàng năm, đó là một ngày đầy mưa và nước mắt. Tuy nhiên, một ngày trước đó, nhà Trời cho phép 2 "ông bà" được đoàn tụ. Trong vũ trụ vùng thiên hà nơi ông Ngâu bà Ngâu ngự trị có N hành tinh đánh số từ 1 đến N, ông ở hành tinh Adam (có số hiệu là S) và bà ở hành tinh Eva (có số hiệu là T). Họ cần tìm đến gặp nhau.

N hành tinh được nối với nhau bởi một hệ thống cầu vồng. Hai hành tinh bất kỳ chỉ có thể không có hoặc duy nhất một cầu vồng (hai chiều) nối giữa chúng. Họ luôn đi tới mục tiêu theo con đường ngắn nhất. Họ đi với tốc độ không đổi và nhanh hơn tốc độ ánh sáng. Điểm gặp mặt của họ chỉ có thể là tại một hành tinh thứ 3 nào đó.

**Yêu cầu:** Hãy tìm một hành tinh sao cho ông Ngâu và bà Ngâu cùng đến đó một lúc và thời gian đến là sớm nhất. Biết rằng, hai người có thể cùng đi qua một hành tinh nếu như họ đến hành tinh đó vào những thời điểm khác nhau.

**Dữ liệu** Trong file văn bản ONBANGAU.INP gồm:

Dòng đầu là 4 số N M S T ( $N \leq 100$ ,  $1 \leq S \neq T \leq N$ ), M là số cầu vồng. M dòng tiếp, mỗi dòng gồm hai số I J L thể hiện có cầu vồng nối giữa hai hành tinh I, J và cầu vồng đó có độ dài là L ( $1 \leq I \neq J \leq N$ ,  $0 < L \leq 200$ ).

**Kết quả** Ra file văn bản ONBANGAU.OUT, do tính chất cầu vồng, mỗi năm một khác, nên nếu như không tồn tại hành tinh nào thỏa mãn yêu cầu thì ghi ra một dòng chữ CRY. Nếu có nhiều hành tinh thỏa mãn thì ghi ra hành tinh có chỉ số nhỏ nhất.

**Ví dụ:**

ONBANGAU.INP	ONBANGAU.OUT
4 4 1 4 1 2 1 2 4 1 1 3 2 3 4 2	2

### **Bài 186/2004 – Đường đi trên đồng hồ cát**

Hình chiếu ngang của đồng hồ cát có dạng 2 tam giác đều chung đỉnh. Hình này được chia thành  $2N - 1$  dòng, dòng trên cùng được chia thành N ô, dòng tiếp theo  $- N - 1$  ô, ..., mỗi ô chứa một số nguyên (Xem hình bên). Trên một hàng các ô được đánh số từ trái sang phải, bắt đầu từ 0.

Mỗi ô (trừ các ô ở hàng cuối cùng) đều kề cạnh với 1 hoặc 2 ô ở hàng dưới. Xét các đường đi xuất phát từ một ô ở hàng trên cùng xuống một ô ở hàng dưới cùng, đi qua các ô kề cạnh và chỉ đi xuống. Tổng giá trị các ô trên đường đi là trọng số của nó. Đường đi đánh dấu trên hình có trọng số là 41. Từ một ô ta có thể xuống ô tiếp theo bên trái hoặc phải, vì vậy đường đi có thể mô tả bằng cách chỉ ra ô xuất phát và chuỗi ký tự L, R. Đường ở hình bên có mô tả là 2 **RRLLRRRLR**.

**Yêu cầu:** Cho giá trị các ô và số nguyên S. Hãy xác định số đường đi khác nhau có trọng số S và mô tả đường đi có thứ tự từ điển nhỏ nhất nếu có đường đi, tức là đường đi xuất phát từ ô có trọng số nhỏ nhất và xâu ký tự tiếp theo có thứ tự từ điển nhỏ nhất.

**Dữ liệu:** Vào từ file văn bản PATHS.INP gồm nhiều Tests:

Dòng thứ nhất chứa 2 số nguyên N S ( $2 \leq N \leq 20$ ,  $0 \leq S < 500$ ),

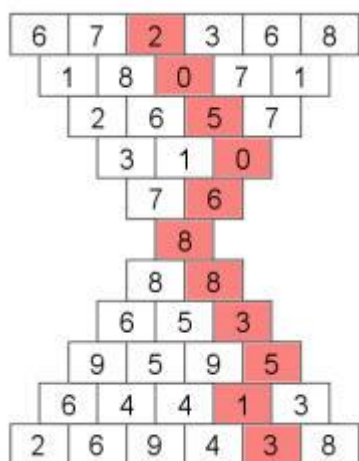
–  $2N - 1$  dòng sau: mỗi dòng chứa các số nguyên của các ô trên một hàng, bắt đầu từ hàng trên cùng.

Kết thúc là một dòng chứa 2 số 0

**Kết quả** Đưa ra file văn bản PATHS.OUT: Kết quả ứng với mỗi Test đưa ra trên 1 hoặc 2

dòng: dòng đầu đưa ra số nguyên  $M$  – số đường đi. Nếu  $M > 0$  thì dòng thứ 2 mô tả đường đi theo yêu cầu. Kết quả các Test cách nhau một dòng trống.

**Ví dụ**



PATHS.INP	PATHS.OUT
6 4 1	1
6 7 2 3 6 8	2 RRRLRRRLR
1 8 0 7 1	0
2 6 5 7	
3 1 0	5
7 6	2 RLLRRRLR
8	
8 8	
6 5 3	
9 5 9 5	
6 4 4 1 3	
2 6 9 4 3 8	
2 7	
3 1	
2	
3 5	
5 2 6	
2 8 7 2 5	
3 6 0 2	
1 3 4	
2 5	
3	
7 2	
2 9 3	
1 0 4 4	
4 8 7 2 3	
0 0	

### Bài 187/2004 – Gà và Thỏ

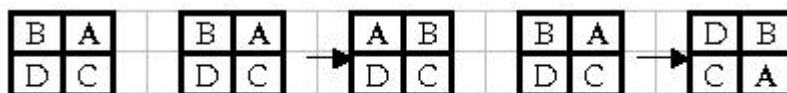
Những chú gà và thỏ đang dạo ngoài sân. Người ta đếm được tất cả 700 cái đầu và 1800 cái chân. Bạn hãy cho biết có bao nhiêu gà, bao nhiêu thỏ?

### Bài 188/2004 – Rút gọn

Cho một hình vuông  $H$  kích thước  $2 \times 2$  được ghép từ 4 hình vuông đơn vị. Mỗi hình vuông đơn vị chứa một chữ cái khác nhau trong 4 chữ cái A, B, C, D. **Trạng thái ban đầu của  $H$  là một trạng thái bất kỳ nào đó** (hình 1 là một ví dụ).

Xét hai phép biến đổi trên  $H$ :

– Phép S – Đổi chỗ 2 hình vuông bên trên (xem hình 2). – Phép R – Đổi chỗ vòng tròn theo chiều kim đồng hồ các hình vuông đi một vị trí (xem hình 3).



Hình 1      Hình 2: Phép biến đổi S      Hình 3: Phép biến đổi R

**Yêu cầu:** Cho dãy phép biến đổi, hãy tìm dãy biến đổi ngắn nhất cho cùng kết quả. Nếu có nhiều dãy kết quả thỏa mãn thì đưa ra dãy có thứ tự từ điển nhỏ nhất.

**Dữ liệu:** Trong file văn bản AUT.INP gồm nhiều bộ dữ liệu, mỗi dòng là một dãy biến đổi, đó là một xâu (khác rỗng) các thao tác trên có độ dài không quá 10. Số bộ dữ liệu trong file không quá  $10^5$ .

**Kết quả:** Ra file văn bản AUT.OUT ứng với mỗi bộ dữ liệu vào là xâu biến đổi ngắn nhất theo yêu cầu tương ứng. Nếu không phải biến đổi, ghi ra dòng trống.

Ví dụ:

AUT.INP	AUT.OUT
RRRRSRRRR	S

### Bài 189/2004 – Khu đất

Công ty tư vấn nhà đất Microsofter đang muốn bán một khu đất hình chữ nhật chia làm  $M \times N$  lô đất nhỏ hình vuông đơn vị. Các lô đất đánh số liên tiếp từ trên xuống dưới bắt đầu từ 1 tới M, từ trái sang phải bắt đầu từ 1 tới N. Lô đất (i, j) có trị giá là một số nguyên dương  $A_{ij}$ .

Mr Linuxer muốn mua một mảnh đất hình vuông có trị giá đúng bằng T. Trị giá một mảnh đất bằng tổng trị giá tất cả các lô đất đơn vị thuộc nó.

**Yêu cầu:** Hãy giúp công ty Microsofter tư vấn cho khách hàng (Mr Linuxer) một mảnh đất vừa ý sao cho diện tích của nó là lớn nhất có thể.

**Dữ liệu:** Trong file văn bản LAND.INP gồm

– Dòng đầu chứa ba số M N T ( $1 \leq M, N \leq 200, 0 \leq T \leq 10^9$ ).

– M dòng tiếp theo, dòng thứ i gồm N số, số thứ j là  $A_{ij}$ . Hai số liên tiếp nhau trên cùng một dòng ghi cách nhau dấu cách ( $1 \leq A_{ij} \leq 10000$ ).

**Kết quả:** Ra file văn bản LAND.OUT gồm một số duy nhất là độ dài cạnh của mảnh đất (hình vuông con) tìm được. Nếu không có mảnh đất nào thỏa mãn thì ghi ra -1.

Ví dụ:

LAND.INP	LAND.OUT									
<table><tr><td>2</td><td>3</td><td>4</td></tr><tr><td>1</td><td>1</td><td>4</td></tr><tr><td>1</td><td>1</td><td>2</td></tr></table>	2	3	4	1	1	4	1	1	2	2
2	3	4								
1	1	4								
1	1	2								

Ghi chú: ở ví dụ trên, có 2 hình vuông thỏa mãn trị giá bằng T và mảnh đất diện tích lớn nhất có cạnh bằng 2. Hạn chế thời gian 1s

### Bài 190/2004 – Những đứa trẻ trong gia đình John Smith

Vợ chồng John Smith có 4 người con, 2 con trai tên là Tom và Ben, 2 con gái tên là Kate và Sally. Tom lớn hơn Ben 2 tuổi. Tổng tuổi của 2 con gái bằng tổng tuổi của 2 con trai. Kate gấp đôi tuổi của Sally. Bạn hãy cho biết tuổi của 4 đứa trẻ? Biết rằng một năm trước đây tuổi của Tom gấp đôi tuổi của Sally.

### Bài 191/2004 – Chia đôi

Trong mặt phẳng, cho toạ độ  $2 \times N$  điểm khác nhau đôi một.

**Yêu cầu:** Hãy xác định một đường thẳng có phương trình  $ax + by + c = 0$  không đi qua bất cứ điểm nào trong các điểm đã cho và mỗi nửa mặt phẳng bờ là đường thẳng đó, chứa đúng N điểm đã cho.

**Dữ liệu:** Trong file văn bản HALF.INP gồm

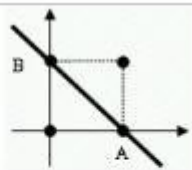
– Dòng đầu là N ( $1 \leq N \leq 5000$ ).

–  $2 \times N$  dòng tiếp theo, dòng thứ i chứa hai số thực  $x_i y_i$  có trị tuyệt đối không quá  $10^6$  là toạ độ của điểm thứ i.

**Kết quả:** Ra file văn bản HALF.OUT gồm 1 dòng ghi 3 số a, b, c lấy chính xác tới 5 chữ số thập phân.

Ví dụ:

HALF.INP	HALF.OUT
1 0 0 1 1	1.00000 1.00000 -1.00000



Test	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10
Kích	Case 1	4000	20	53	100	300	1000	2000	5000	5000
Thuốc	Case 2	10	20	50	100	200	2500	2000	3000	4500

### Bài 192/2004 – IP

Tập đoàn Công nghệ Amser nổi tiếng hiện có  $N$  nhân viên. Dưới thời tổng giám đốc cũ, Huluxima gán cho mỗi nhân viên  $i$  ( $i = 1, 2, \dots, N$ ) một mã IP là một số nguyên dương  $A_i$  (tất nhiên 2 nhân viên khác nhau sẽ có số IP khác nhau). Hiện nay, Tổng giám đốc mới lên nắm quyền là Buđata rất chú trọng tới sự ngăn nắp và qui củ nên ông muốn ngay cả các số IP của nhân viên cũng phải được cân nhắc kỹ càng. Khi có thêm một nhân viên mới, ông ta muốn biết số IP nhỏ nhất còn có thể gán cho nhân viên đó.

**Yêu cầu:** Cho  $N$  mã IP của các nhân viên trong tập đoàn, hãy tìm mã IP nhỏ nhất chưa xuất hiện cho tập  $N$  mã đã cho.

**Dữ liệu:** Trong file văn bản IP.INP gồm

– Dòng đầu là  $N$  ( $0 \leq N \leq 10^8$ ).

–  $N$  dòng tiếp theo, dòng thứ  $i$  ghi số  $A_i$  ( $i = 1..N$ ,  $1 \leq A_i \leq 10^9$ ).

**Kết quả:** Ra file văn bản IP.OUT gồm một số duy nhất là mã IP tìm được.

**Ví dụ:**

IP.INP	IP.OUT
3 3 1 4	2

### Bài 193/2004 – Bộ Cờ

Có một số quân cờ nhảy, mỗi quân được sơn một trong số  $K$  màu khác nhau ( $K \leq 25$ ). Trên mỗi quân cờ ghi một số nguyên không âm bằng một trong số  $N$  số nguyên không âm cho trước ( $N \leq 101$ ).  $K$  quân cờ có màu khác nhau từng đôi một tạo thành một bộ cờ. Hai bộ cờ gọi là giống nhau, nếu số ghi trên các quân cờ cùng màu là như nhau. Trong trường hợp ngược lại hai bộ cờ gọi là khác nhau. Tổng các số ghi trên các quân của một bộ cờ gọi là trọng số của nó.

**Yêu cầu:** Hãy tìm số bộ cờ khác nhau có cùng trọng số  $S$  ( $0 < S \leq 300$ ).

**Dữ liệu:** Vào từ file văn bản COMPLETE.INP:

Dòng đầu tiên chứa số nguyên  $K$ ,

**Kết quả:** Đưa ra file văn bản COMPLETE.OUT số lượng tìm được (nguyên).

**Ví dụ:**

COMPLETE.INP
6
2
2
0 1

COMPLETE.OUT
15

**Bài 194/2004 – Cửa hàng bán kẹo**

Sau khi tan học Judy, Kenny, Liam, Mandy và Nathan cùng vào một cửa hàng bán kẹo. Chúng xem tất cả các loại kẹo có giá dưới 50p. Gồm có choco bar, kẹo chew, bánh trứng và kẹo mút. Mỗi đứa trẻ có một đô la, chúng không cần thiết mua hết số tiền của mình.

Judy mua một choco bar, một bánh trứng, một kẹo chew hết 61p.

Kenny mua một choco bar, một bánh trứng, một kẹo mút còn lại 46p.

Liam mua 3 kẹo mút hết 84p.

Mandy mua một bánh trứng, một kẹo mút, một kẹo chew còn lại 20p.

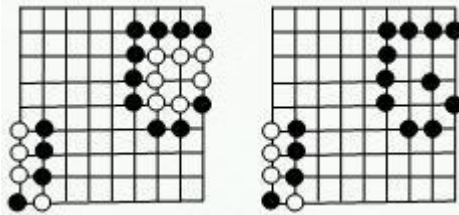
Nathan mua mỗi loại một cái. Bạn hãy cho biết Nathan còn lại bao nhiêu tiền.

*Chú ý: 1 đô la = 100p*

**Bài 195/2004 – Cờ vây**

Trên lưới ô vuông tạo thành từ N đường dọc và N đường ngang, hai người lần lượt đặt quân của mình trên các nút của lưới. Các đường được đánh số từ trên xuống dưới và từ trái qua phải bắt đầu từ 1. Một người cầm quân đen, người kia cầm quân trắng. Nhóm quân cùng màu là các quân trên các nút tạo thành một miền liên thông khi chuyển động theo chiều dọc hoặc ngang. **Một quân hoặc nhóm quân cùng màu gọi là bị vây nếu mọi nút kề với các quân này đều có quân khác màu.** Những quân bị vây sẽ bị đối phương ăn (lấy ra khỏi bàn cờ).

Trên hình bên trái có 2 nhóm quân trắng có **nguy cơ bị vây**. Người cầm quân đen có thể đặt quân của mình vào vị trí (5,1) và ăn được 3 quân hoặc đặt vào vị trí (4,8) và ăn được 7 quân. Hình bên phải là kết quả quân đen đặt vào vị trí (4,8).



**Yêu cầu:** Cho N và cấu hình quân trên bàn cờ. Hãy xác định một vị trí đặt quân đen để có thể ăn được nhiều quân nhất các quân trắng tại nước đi đó, giả thiết rằng luôn tồn tại ít nhất một nước đi như vậy.

**Dữ liệu:** Trong file văn bản ENCIRCLE.INP gồm

– Dòng đầu tiên chứa số nguyên N ( $8 \leq N \leq 100$ ).

– N dòng sau, mỗi dòng chứa một xâu N ký tự xác định trạng thái dòng tương ứng của bàn cờ, quân đen được ký hiệu bằng ký tự b, trắng – w, vị trí trống – dấu chấm ‘.’.

*Chú ý: Không xét những nhóm quân trắng đã bị vây từ trước.*

**Kết quả:** Ra file văn bản ENCIRCLE.OUT một dòng chứa số nguyên M là số quân trắng nhiều nhất có thể ăn được tại nước đi tiếp theo của quân đen.

**Ví dụ:**

ENCIRCLE.INP	ENCIRCLE.OUT
9	7
.....	
..... bbbb	
..... bwww	
..... bw.w	
wb... bwwb	
wb... bb.	
wb.....	
bw.....	
.....	

(Trích đề thi HAOI – 2004)

**Bài 196/2004 – Các nền văn minh cổ đại**

Các nền văn minh cổ đại có những ảnh hưởng qua lại tới nhau trực tiếp hoặc gián tiếp nếu có khoảng thời gian cùng tồn tại. ảnh hưởng sẽ càng lớn nếu thời gian tồn tại đồng thời lớn. Ví dụ, nếu nền văn minh A tồn tại từ 600 năm trước công nguyên (CN) đến trước năm 400 trước CN, còn nền văn minh B xuất hiện vào năm 450 trước CN và tồn tại đến trước năm 300 trước CN, thì giữa chúng có 50 năm cùng tồn tại. Nếu nền văn minh C xuất hiện vào năm 400 trước CN và tồn tại đến trước năm 50 trước CN thì giữa A và C không có tác động qua lại, trong lúc đó giữa B và C có đến 100 năm cùng tồn tại và ảnh hưởng qua lại. Để đánh giá mức độ ảnh hưởng theo thời gian, các nhà khảo cổ học quyết định chọn 2 nền văn minh có khoảng thời gian cùng tồn tại khác 0 nhỏ nhất trong số N nền văn minh mà tài liệu còn được ghi chép và tìm thấy ( $1 \leq N \leq 100\,000$ ).

**Dữ liệu:** Vào từ file văn bản ANCIENT.INP:

– Dòng đầu tiên chứa số nguyên N, – N dòng sau mỗi dòng chứa 2 số nguyên Si và Ei, trong đó Si – năm xuất hiện, Ei – năm diệt vong, giá trị âm hoặc 0 thể hiện trước công nguyên, giá trị tuyệt đối của năm không vượt quá  $10^9$ .

**Kết quả:** Đưa ra file văn bản ANCIENT.OUT hai số nguyên trên một dòng xác định các nền văn minh có thời gian cùng tồn tại khác 0 nhỏ nhất. Nếu không tìm thấy 2 nền văn minh nào cùng tồn tại thì đưa ra một số 0.

**Ví dụ:**

ANCIENT.INP	ANCIENT.OUT
3	1 3
-10 80	
-80 10	
60 90	

(Đề ra của Phạm Mạnh Cường)

**Bài 197/2004 – Xếp hàng chào cờ**

Trong buổi chào cờ, các học sinh lớp 7 và lớp 8 đứng thành 2 hàng theo khối của mình. Đứng trước mỗi học sinh lớp 8 là một học sinh lớp 7 có chiều cao thấp hơn. Hãy chứng minh rằng, nếu hai khối học sinh lớp 7 và lớp 8 xếp thành hàng (theo từng khối) theo chiều cao tăng dần, thì mỗi học sinh lớp 8 vẫn đứng sau một học sinh lớp 7 thấp hơn.

**Bài 198/2004 – Chuỗi nhị phân**

Một chuỗi chỉ gồm toàn các kí tự 0 và 1 được gọi là chuỗi nhị phân. Một đoạn liên tiếp có k kí tự của chuỗi được gọi là **chuỗi con độ dài k**.

**Yêu cầu:** Cho trước số nguyên dương k ( $k < 16$ ). Hãy lập trình xác định chuỗi nhị phân dài nhất sao cho mỗi chuỗi con độ dài k chỉ xuất hiện 1 lần.

**Dữ liệu vào** chỉ là một số nguyên k trong tệp **binstr.inp**

**Kết quả** tìm được ghi vào tệp **binstr.out** gồm 2 dòng. Dòng đầu là độ dài của chuỗi tìm được. Dòng 2 chứa chuỗi này.

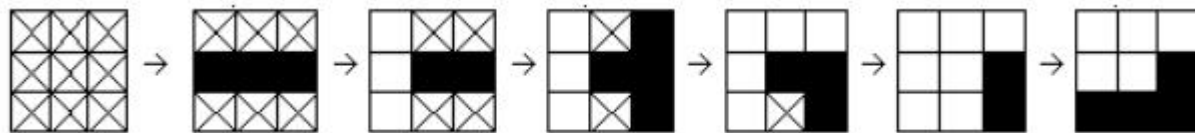
**Ví dụ:**

Binstr.inp	Binstr.out
3	10 0001110100



**Bài 199/2004 – Băng giấy**

Cho tờ giấy kẻ ô vuông màu xám kích thước  $N \times N$  ô. Các dòng và cột được đánh số từ 1 đến  $N$ . Ngoài ra còn có  $2N$  băng giấy kích thước  $N \times 1$  ô,  $N$  băng có màu trắng (ký hiệu W),  $N$  băng – màu đen (ký hiệu B). Người ta lần lượt dán các băng giấy lên tờ giấy, sao cho mỗi lần dán, một dòng hoặc cột của tờ giấy ban đầu bị phủ kín. Sau khi dán hết  $2N$  băng mỗi ô của tờ giấy sẽ có một trong hai màu: trắng hoặc đen. Ví dụ, sau khi dán các băng lên tờ giấy kích thước  $3 \times 3$ , ta có thể có bức tranh sau (các ô màu xám được đánh dấu chéo):



**Yêu cầu:** Hãy xác định cách dán các băng giấy để tờ giấy có trạng thái theo yêu cầu cho trước.

**Dữ liệu:** Vào từ file văn bản PAPER.INP:

- Dòng đầu tiên chứa số nguyên  $N$  ( $0 < N \leq 2\,000$ ),
- $N$  dòng sau: mỗi dòng chứa  $N$  ký tự W hoặc B xác định màu cần có ở mỗi ô, các ký tự cách nhau 1 dấu cách.

**Kết quả:** Đưa ra file văn bản PAPER.OUT:  $2N$  dòng, mỗi dòng xác định một cách dán băng giấy, các dòng đưa ra theo trình tự dán và có quy cách:

X K Y, trong đó X – ký tự W hoặc B – màu băng giấy, Y – Ký tự L (dòng) hoặc C (cột), K – số thứ tự của dòng hoặc cột. K Y xác định cách dán.

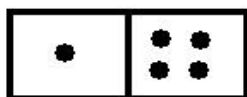
**Ví dụ:**

PAPER.INP		PAPER.OUT
3		B 2 L
W W W		W 1 C
W W B		B 3 C
B B B		W 1 L
		W 2 C
		B 3 L

**Bài 200/2004 – Cờ domino**

Một bộ cờ DOMINO tiêu chuẩn gồm có 28 quân, mỗi quân thể hiện hai số trong khoảng từ 0 (quân bỏ trống) đến 6. Không có 2 quân nào có cùng bộ hai số.

Bộ cờ DOMINO 28 quân											
STT			STT			STT			STT		
1	0	0	8	1	1	15	2	3	22	3	6
2	0	1	9	1	2	16	2	4	23	4	4
3	0	2	10	1	3	17	2	5	24	4	5
4	0	3	11	1	4	18	2	6	25	4	6
5	0	4	12	1	5	19	3	3	26	5	5
6	0	5	13	1	6	20	3	4	27	5	6
7	0	6	14	2	2	21	3	5	28	6	6



1 | 4 (STT 11)



0 | 3 (STT 4)



Xếp bộ DOMINO vào vừa khít một lưới ô  $7 \times 8$ . Mỗi quân cờ nằm trên hai ô liền kề nhau (chung cạnh).

Cho bảng số  $7 \times 8$  là hình ảnh bàn cờ DOMINO nhìn từ trên xuống.

**Yêu cầu:** Đưa ra cách xếp các quân cờ DOMINO lên lưới ô để nhận được bảng số đã cho.

**INPUT:** DOMINO.INP

File dữ liệu vào là một bảng  $7 \times 8$  gồm các số từ 0 đến 6, các số cách nhau ít nhất một dấu cách. Bảng chắc chắn thể hiện một cách xếp cờ DOMINO.

**OUTPUT:** DOMINO.OUT

Thể hiện vị trí các quân cờ bằng cách điền số thứ tự của quân cờ vào các vị trí tương ứng sao cho khi xếp các quân cờ lên ta được bảng số như đã cho. Trong trường hợp có nhiều cách xếp, chỉ cần hiển thị một cách.

**Ví dụ 1:**

DOMINO.INP	DOMINO.OUT
5 4 3 6 5 3 4 6	6 20 20 27 27 19 25 25
0 6 0 1 2 3 1 1	6 18 2 2 3 19 8 8
3 2 6 5 0 4 2 0	21 18 28 17 3 16 16 7
5 3 6 2 3 2 0 6	21 4 28 17 15 15 5 7
4 0 4 1 0 0 4 1	24 4 11 11 1 1 5 12
5 2 2 4 4 1 6 5	24 14 14 23 23 13 13 12
5 5 3 6 1 2 3 1	26 26 22 22 9 9 10 10

## Phần II: LỜI GIẢI

### ***Giải bài 151/2003 – Người nông dân và những quả táo***

Bài toán được giải khá đơn giản nếu các bạn để ý một chút trong cách lập luận.

Đề còn lại 1 quả táo sau khi qua cổng thứ 3 trước khi qua cổng này (tức là sau khi qua cổng thứ 2), người nông dân phải có  $2(1+1) = 4$  quả.

Đề còn lại 4 quả táo sau khi qua cổng thứ 2, trước khi qua cổng này, người nông dân phải có  $2(4+1) = 10$  quả.

Đề còn lại 10 quả táo sau khi qua cổng 1, người nông dân phải hái  $2(10+1) = 22$  quả.

Như vậy đề còn đúng một quả và đáp ứng các yêu cầu của lính canh thì người nông dân phải hái 22 quả táo.

### ***Giải bài 152/2003 – Chia kẹo***

**Tư tưởng thuật toán:** Giả sử ta chia N gói kẹo thành 2 phần P1, P2 với số chênh lệch là  $C = \text{Abs}(Tp1 - Tp2)$ . Ta chỉ có 2 cách chia P1, P2 như sau:

(1): Chuyển một gói kẹo A[i] từ phần này sang phần khác

(2): Chuyển gói kẹo A[i] ở P1 sang P2 và chuyển gói A[j] từ p2 sang p1 (tức là đổi chỗ hai gói đã được chia ở hai phần cho nhau)

Nếu còn có thể chia N gói kẹo một cách tốt hơn tức là p1, p2 có độ chênh lệch  $C' < C$  ta có thể khẳng định còn một số cách chia lại (1) hoặc (2) sao cho sau khi chia p1, p2 có độ chênh lệch  $C' < C$  ban đầu.

Nếu các gói kẹo đã được chia một cách tốt nhất thì ta có thể khẳng định không còn cách chia lại nào trong hai cách (1), (2) sao cho sau khi chia p1, p2 có độ chênh lệch ít hơn.

#### ***Thuật toán:***

Từ một cách chia ban đầu p1, p2 có độ chênh lệch  $C = \text{Abs}(Tp1 - Tp2)$  ta tìm xem có còn tồn tại cách chia lại nào làm cho p1, p2 tốt hơn không, nếu còn thì chia lại theo cách đó và cập nhật lại C, Tp1, Tp2 nếu không còn khẳng định p1, p2 đã được chia tốt nhất. Lặp lại cho đến khi p1, p2 được chia tốt nhất.

Từ thuật toán trên ta có thể cài đặt chương trình như sau:

```
Program Chia_keo;
const   fi='chiakeo.inp';
        fo='chiakeo.out';
        max=10000;
var     p1,p2:array[1..max] of integer;
        tp1,tp2,c,n:longint;
        f:text;
{*_____*}
procedure input;
var i:integer;
begin
  fillchar(p1,sizeof(p1),0);
  p2:=p1;
  assign(f,fi); reset(f);
  tp1:=0;tp2:=0;
  c:=0;i:=0;
  while not eof(f) do
  begin
```

```

while not eof(f) do
  if tp2>=tp1 then {doc vao phan nao it keo hon}
  begin
    inc(i);
    read(f,p1[i]);
    tp1:=tp1+p1[i];
  end
  else
  begin
    inc(i);
    read(f,p2[i]);
    tp2:=tp2+p2[i];
  end;
  readln(f);
end;
n:=i; c:=abs(tp1-tp2);
close(f);
end;
{ * _____ * }
Procedure Divice;
var ok:boolean;
i,t,j:integer;
begin
  ok:=true;
  while ok do
  begin
    ok:=false;
    for i:=1 to n do
      for j:=1 to n do
        if abs((tp1+p2[i]-p1[j])-(tp2+p1[j]-p2[i])) then
          begin
            tp1:=tp1-p1[j]+p2[i];
            tp2:=tp2+p1[j]-p2[i];
            c:=abs(tp1-tp2);
            t:=p2[i];
            p2[i]:=p1[j];
            p1[j]:=t;
            ok:=true;
          end;
        end;
      end;
    end;
  end;
end;
{ * _____ * }
Procedure output;
var i:integer;
begin
  assign(f,fo); rewrite(f);
  writeln(f,n,' ',tp1,' ',tp2,' ',c);
  for i:=1 to n do if p1[i]>0 then write(f,p1[i],' ');

```

```

writeln(f);
for i:=1 to n do if p2[i]>0 then write(f,p2[i], ' ');
close(f);
end;
{ * _____ * }
BEGIN
  input;
  device;
  output;
END.

```

*{Bài giải của bạn Nguyễn Phi Duy}*

Với thuật toán này ta có thể chia với số lượng gói kẹo lớn đến giới hạn khai báo mảng một chiều của Pascal: 20000. Chú ý tổng số kẹo phải nhỏ hơn Maxlongint.

Các bạn có thấy rằng thuật toán thật đơn giản và hiệu quả không. Vậy mà toà soạn chỉ nhận được duy nhất 5 lời giải từ các bạn độc giả – một con số ít ỏi lắm. Và đáng buồn hơn là cả 5 bạn đều đi sai hướng, vì thế không bạn nào giải quyết được trọn vẹn bài toán. Vậy có thể kết luận đây là một bài toán "khó"!

Cả 5 bạn đều chọn giải thuật Quy hoạch động (QHĐ) để giải quyết bài toán, tư tưởng chung là: tính tất cả các tổng có thể lập được từ số lượng kẹo của các gói kẹo đã cho, sau đó chọn tổng có lợi nhất (là tổng nhất gần với một nửa tổng số kẹo). Để cài đặt thuật toán QHĐ trên, các bạn cần ít nhất một mảng để đánh dấu những tổng có thể lập được (chưa tính đến một mảng lưu vị trí để ghi kết quả). Giải thuật trên không sai, nhưng bị giới hạn bởi tổng số kẹo (bằng GioiHanSoLuong\*GioiHanKhoiLuong), trong trường hợp này là:

$$1.000 \cdot 10.000 = 10.000.000 \text{ (10 triệu).}$$

Rõ ràng dù khéo léo đến đâu các bạn cũng khó lòng thoả mãn yêu cầu bài toán nếu sử dụng QHĐ.

Phải bàn thêm rằng, đây là một bài toán không hề mới mà lại là mới! Các bạn có thể bắt gặp bài toán này trong khá nhiều tài liệu Bài tập tin học (còn có dạng khác là bài về các quả cân), nhưng trong các tài liệu đó số lượng gói kẹo và số kẹo trong mỗi gói thường được giới hạn rất nhỏ (chỉ khoảng trên dưới 200), rất thuận tiện cho việc cài đặt thuật toán QHĐ (thực tế là phân các hướng dẫn giải đều định hướng cho các bạn dùng QHĐ). Bài toán lại trở thành mới khi giới hạn lên tới 10.000 cho số gói kẹo và 1.000 cho số kẹo trong một gói, điểm yếu của QHĐ là tổ chức dữ liệu công kênh đã lộ rõ.

Theo chúng tôi, bài toán 'khó' là ở chỗ nó quá quen, cách giải quyết đã trở thành đường mòn. Các bạn chỉ loay hoay cố gắng đi trên con đường mòn đó, tìm cách tổ chức dữ liệu để tăng được giới hạn về tổng số kẹo lên càng nhiều càng tốt. Không có độc giả nào đề xuất các giải khác...

### ***Giải bài 153/2003 – Phân phối hàng***

#### ***Thuật toán:***

Ta có tổng chi phí vận chuyển:

$$S = \sum_{i=1}^n A_i * F_i + B_i * (D_i - F_i) = \sum_{i=1}^n B_i * D_i + \sum_{i=1}^n C_i * F_i$$

với  $C_i = A_i - B_i$  và  $F_i$  là số đơn vị hàng từ kho A đến công trường i ( $i=1..n$ ), do  $\sum B_i * F_i$  không thay đổi nên ta có bài toán tối ưu quen thuộc và đơn giản hơn: Cực tiểu hàm  $\sum C_i * F_i$  với ràng buộc  $0 \leq F_i \leq D_i$  và  $\sum F_i \leq R$ . Bài toán này được giải quyết dễ dàng chỉ bằng thao tác sắp xếp mảng C:  $C_i$  càng nhỏ thì càng được ưu tiên chọn trước và chọn nhiều hơn. Hầu hết các

bạn đã nhận thấy điều này và đưa ra thuật toán sau đây là một thuật toán tối ưu (có thể kiểm nghiệm bằng Toán học hay chỉ bằng "Trực giác Tin học"):

– Sắp xếp mảng C tăng dần (chính là thay đổi thứ tự ưu tiên với các công trường) có lưu lại chỉ số ban đầu.

– Lần lượt cung cấp cho: công trường A[1] (không phải là công trường số 1 ban đầu) số đơn vị hàng là D[1], A[2] là D[2],... A[k] là D[k] ( $k \geq 0$ ),... cho đến khi không đủ để cung

cấp nữa, tức là  $\sum_{i=1}^k D_i \leq R$ . Lúc này đối với công trường k+1 ta phải lấy từ kho B để cung

cấp nốt cho đủ, số đơn vị hàng còn lại này là  $D[k+1] - (R - \sum_{i=1}^k D_i)$ . Tất cả các công trường còn lại ta cung cấp từ kho B "thoải mái" cho đủ. Như vậy chỉ có nhiều nhất 1 công trình được nhận hàng từ cả 2 kho.

Thuật toán quả thực rất đơn giản – chỉ hơn bài toán sắp xếp một chút. Đa số các bạn dùng Quick Sort do yêu cầu dữ liệu  $N \leq 10000$ , rất tiếc là có bạn vẫn dùng Bubble Sort chạy rất lâu hay một số bạn đã làm phức tạp hoá bài toán trở thành Quy hoạch động trên mảng 1 chiều, 2 chiều... (như trong một số sách BT Tin học giải mẫu!) với  $N \leq 100$ .

Tuy nhiên Đề ra kỳ này lại tập trung vào việc tổ chức dữ liệu nhiều hơn. Thực tế là hầu hết các bạn làm không đạt yêu cầu! Các bạn có lẽ đã hạn chế tối đa khi khai báo biến, và cuối cùng dùng khoảng 5–10 mảng nguyên 10000 phần tử (!) – tất nhiên là dùng nhiều mảng động (Như thế là chưa "đầu tư công sức" để tìm ra cách cài đặt tối ưu hơn). Kích thước dữ liệu khá lớn nên việc dùng chỉ số trong sắp xếp là phải có, một số bạn đã làm được điều này, nhưng chưa bạn nào đạt được sự tinh tế của "phương pháp chỉ số" – khi xuất kết quả các bạn vẫn dùng 1–2 mảng kết quả trong khi chúng ta không hề cần! Các bạn hãy xem kỹ chương trình mẫu để thấy được điều đó.

Một vấn đề quan trọng nữa trong việc cài đặt chương trình đối với tất cả các bạn kỳ này đó là các bạn thường lạm dụng dùng mảng – từ thuật toán đã nêu chúng ta thấy chỉ có công trường k là nhận từ cả 2 kho A và B (còn lại "một loạt 0" từ kho A và "một loạt 0" từ kho B) vậy thì chúng ta chỉ cần ghi nhớ vị trí k này mà thôi, hầu như các bạn làm thêm một mảng kết quả như mảng F mô tả tổng chi phí, mảng này chỉ khác mảng D đúng một phần tử – các bạn có thấy mình lãng phí không nào?

**Chú ý:** Cùng một file input có thể có nhiều file output đúng, vì ở trường hợp  $A_i = B_i$  có thể lấy từ A hay B đều được. Các bạn làm Quick Sort file output khác file output của test đề bài nhưng vẫn đúng (file output của test đề bài có lẽ được tác giả Trần Đức Thiện làm bằng Bubble Sort!)

**Chương trình mẫu hoàn chỉnh:**

```
{A+,B+,D+,E+,F-,G-,I+,L+,N+,O-,P-,Q+,R+,S+,T-,V+,X+}
{$M 16384,0,655360}
```

```
program Bai153;
```

```
const fi='hang.inp';
```

```
fo='hang.out';
```

```
type mang=array[0..10000] of integer;
```

```
var s: real;
```

```
  n,r: integer;
```

```
  k,w,x,y: integer; {Cac bien lưu lại}
```

```
  a,b,cs: mang;
```

```
  d: ^mang;
```

```
{*-----*}
```

```
procedure Nhap;
```

```

var i: integer;
    f: text;
begin
    assign(f,fi); reset(f);
    readln(f,n,r);
    new(d);
    fillchar(d^,sizeof(mang),0);
    for i:=1 to n do
        begin
            cs[i]:=i; {Mang chi so}
            read(f,d^[i]);
        end;
    for i:=1 to n do read(f,a[i]);
    for i:=1 to n do read(f,b[i]);
    close(f);
end;
{ * _____ * }
{ Quick sort theo chi so }
procedure Qsort(l,r: integer);
var i,j,mid,tg: integer;
begin
    i:=l; j:=r;
    mid:=a[cs[(l+r) div 2]]-b[cs[(l+r) div 2]];
    repeat
        while a[cs[i]]-b[cs[i]]
        while a[cs[j]]-b[cs[j]]>mid do dec(j);
        if i<=j then
            begin
                tg:=cs[i]; cs[i]:=cs[j]; cs[j]:=tg;
                inc(i); dec(j);
            end;
    until i>j;
    if l
    if i
end;
{ * _____ * }
procedure TimK; {Tim vi tri K dac biet – nhan hang tu ca 2 kho!}
var i,t: integer;
begin
    s:=0; t:=0; k:=0;
    while t+d^[cs[k]]
    begin
        inc(k);
        t:=t+d^[cs[k]];
        s:=s + a[cs[k]]*d^[cs[k]];
        a[cs[k]]:=-a[cs[k]];
    end;
    x:=r-t; y:=d^[cs[k+1]]-(r-t);

```

```

s:=s + a[cs[k+1]]*x + b[cs[k+1]]*y;
for i:=k+2 to n do s:=s + b[cs[i]]*d^[cs[i]];
w:=cs[k+1]; {Luu lai vi tri dac biet nay trong mang chi so}
end;
{ * _____ * }
procedure Xuat;
var i: integer;
    g: text;
begin
    assign(g,fo); rewrite(g);
    writeln(g,s:0:0);
    {Lay tu mang a;}
    for i:=1 to w-1 do
        if a[i]<0 then write(g,d^[i], ' ') else write(g,0, ' ');
        write(g,x, ' '); {Vi tri lay tu ca 2 kho}
    for i:=w+1 to n do
        if a[i]<0 then write(g,d^[i], ' ') else write(g,0, ' ');
    writeln(g);
    {Viet b;}
    for i:=1 to w-1 do
        if a[i]>0 then write(g,d^[i], ' ') else write(g,0, ' ');
        write(g,y, ' '); {Vi tri lay tu ca 2 kho}
    for i:=w+1 to n do
        if a[i]>0 then write(g,d^[i], ' ') else write(g,0, ' ');
        close(g);
        dispose(d);
end;
{ * _____ * }
BEGIN
    Nhap;
    Qsort(1,n);
    TimK;
    Xuat;
END.

```

### ***Giải bài 154/2003 – Tuổi cha và con***

**Ta lập luận như sau:** Theo giả thiết của bài toán thì tuổi của người cha phải chia hết cho 7 và 4, do đó sẽ là bội số của 28. Vậy tuổi của người cha có thể là 28, 56, 84, 112... Nhưng nếu như tuổi của người cha lớn hơn 28, nghĩa là bằng 56 hoặc 84 thì tuổi của đứa con nhỏ sẽ là 8 hoặc 12, mà theo giả thiết đứa con này đang học mẫu giáo. Vậy tuổi của cha chỉ có thể là 28 và tuổi của các con là 7 và 4.

Giải đúng bài này có bạn *Trương Quỳnh Nhi* – [Nghi.pham@gannon.com.vn](mailto:Nghi.pham@gannon.com.vn), tuy nhiên bạn chưa đưa ra lập luận lời giải của bài toán. Hy vọng lần sau bạn sẽ tham gia nhiệt tình hơn nữa.

### ***Giải bài 155/2003 – Bố trí phòng họp***

#### ***Tư tưởng thuật toán:***

Dễ dàng nhận thấy cuộc họp có thời gian kết thúc sớm nhất sẽ là cuộc họp thứ nhất, giả

sử cuộc họp này bắt đầu tại thời điểm  $t_1$  và kết thúc tại thời điểm  $kt_1$ . Cuộc họp có thời điểm bắt đầu  $\geq kt_1$  và thời điểm kết thúc sớm nhất sẽ là cuộc họp thứ 2, giả sử cuộc họp này bắt đầu tại thời điểm  $t_{u2}$  và kết thúc tại thời điểm  $kt_{u2}$  ... cuộc họp có thời điểm bắt đầu  $\geq kt_{i-1}$  và thời điểm kết thúc sớm nhất sẽ là cuộc họp thứ  $i$  ... Thuật toán kết thúc nếu tại cuộc họp thứ  $u$  với thời điểm kết thúc  $kt_u$  nào đó ta không tìm được một cuộc họp nào có thời điểm bắt đầu  $\geq kt_u$ . Như vậy  $u$  sẽ là số cuộc họp lớn nhất mà ta có thể bố trí được.

Tuy nhiên vấn đề của bài toán này lại là việc xử lý dữ liệu, với số lượng cuộc họp đăng kí là 1000.000 không phải một con số nhỏ. Chúng ta không thể lưu vào 2 mảng **đau** và **cuoi** ( $đau[i]$ ,  $cuoi[i]$  thời gian bắt đầu và kết thúc của cuộc họp  $i$ ). Tuy nhiên rất may mắn là thời gian muộn nhất của cuộc họp là 32.000, vì vậy ta sẽ dùng 1 mảng **S** ( $s[i]$  là thời điểm bắt đầu cuộc họp, còn  $i$  là thời điểm kết thúc). Như vậy sẽ có nhiều cuộc họp cùng kết thúc tại một thời điểm, khi đó ta sẽ chỉ lưu cuộc họp có thời điểm bắt đầu sớm nhất. Sau khi có được mảng **S**, việc bố trí các cuộc họp sẽ trở lên dễ dàng, đầu tiên ta sẽ đi tìm  $s[i] < 0$  đầu tiên (tức là tìm thời điểm kết thúc sớm nhất của cuộc họp thứ 1), Giả sử là *Now*, sau đó ta lại tìm  $s[i] < 0$  và  $s[i] \geq Now$  đầu tiên (tức là ta đã tìm được  $i$ : là thời điểm kết thúc của cuộc họp thứ 2), gán  $now=i$ , tiếp tục cuộc tìm kiếm cho đến khi  $i = \text{Max}$  (là thời điểm kết thúc muộn nhất của các cuộc họp)

Vấn đề trở ngại lớn nhất với chúng ta ở đây là việc in kết quả, làm thế nào để lấy được chỉ số của các cuộc họp. ở đây chúng tôi xin nêu ra 2 cách, cách một chúng ta dùng 2 mảng **Thuong** và **Du** (**kiểu integer hoặc byte**) ( $Thuong[i]$ ,  $du[i]$  là thương và dư của phép chia  $i$  (**kiểu longint**) cho một số nào đó ) với cách quản lý này chúng ta hoàn toàn có thể lưu được chỉ số của cuộc họp có thời điểm kết thúc là  $i$ . Tuy nhiên chúng ta phải dùng đến mảng động, cách thứ 2 sẽ tránh dùng mảng động đó là cách đọc file 2 lần, để làm được điều đó trong quá trình tìm lịch trên mảng  $s$ , với mỗi  $s[i]$  tìm được chúng ta đánh dấu nó bằng  $s[i] := -s[i]$ . Như vậy trong mảng  $s$  những phần tử  $s[i] < 0$  thì  $i$  sẽ là thời điểm kết thúc của một cuộc họp được chọn, còn  $ABS(s[i])$  là thời điểm bắt đầu cuộc họp đó. Sau khi đọc lại file với mỗi giá trị  $d, c$  ta kiểm tra nếu ( $s[c] < 0$ ) và  $ABS(s[c]) = d$  thì ghi nhận chỉ số cuộc họp đó

### **Chương trình mẫu:**

```
const MaxN = 32000;
    fi='activity.inp';
    fo='activity.out';
var    f1,f2:text;
        s:array[0..MaxN] of integer;
        d,c,n,max,count:longint;
{*_____*}
procedure int;
    var i:longint;
        begin
            assign(f1,fi); reset(f1);
            max:=0;
            readln(f1,n);
            fillchar(s,sizeof(s),0);
            for i:=1 to n do
                begin
                    readln(f1,d,c);
                    if s[c] < d then s[c]:=d;
                    if c > max then max:=c;
                end;
```



```

        close(f1);
        assign(f2,fo);
        rewrite(f2);
    end;
{*-----*}
procedure process;
    var i,j,now:longint;
    begin
        count:=0; now:=0;
        for i:=1 to max do
            if s[i] > 0 then
                if s[i] >= now then
                    begin
                        inc(count);
                        s[i]:=-s[i];
                        now:=i;
                    end;
                end;
        end;
{*-----*}
procedure print;
    var i,j:longint;
    begin
        assign(f1,fi); reset(f1);
        readln(f1,n);
        writeln(f2,count);
        for i:=1 to n do
            begin
                readln(f1,d,c);
                if s[c] < 0 then
                    if ABS(s[c]) = d then
                        begin
                            writeln(f2,i);
                            s[c]:=0;
                        end;
            end;
        close(f1);
        close(f2);
    end;
{*-----*}
BEGIN
    int;
    process;
    print;
END.

```

***Giải bài 156/2003 – Bài tranh***

Đây là một biến thể của bài toán quy hoạch động.

***Thuật toán như sau:***

Đầu tiên ta sẽ thử treo bức tranh  $k$  ở cửa ( $k = 1..n$ ), sau đó ứng với  $k$  ta sẽ tìm cách treo  $n-1$  bức còn lại vào trong phòng, sau đó ta tìm giá trị thẩm mỹ lớn nhất ứng với  $k$ . Nếu nó lớn hơn các giá trị thẩm mỹ ứng với  $k$  khác thì ta ghi nhận, cuối cùng ta sẽ có được giá trị lớn nhất.

Như vậy bài toán trên thực ra là bài toán tìm giá trị thẩm mỹ lớn nhất khi treo  $n-1$  bức tranh vào  $m$  vị trí.

Để giải bài toán trên chúng ta có nhận xét sau: vì các bức tranh phải xếp theo thứ tự số hiệu nên với mỗi bức tranh  $i$  thì chúng ta chỉ có thể đặt tại  $sl = m - (n-1) + 1$  vị trí mà thôi, cụ thể với bức tranh  $i$  có thể đặt tại vị trí start đến finish trong đó  $start = i$ ,  $finish = i + sl - 1$  nếu  $i < k$  ( $k$  là bức tranh treo ở cửa)  $start = i-1$ ;  $finish = i + sl - 2$  nếu  $i > k$

Gọi  $s[i,j]$  là độ thẩm mỹ lớn nhất nếu treo bức tranh  $i$  ở vị trí  $j$  (như vậy  $j = start..finish$ ).

Với mỗi vị trí có thể đặt được của bức tranh  $i$  ta sẽ so sánh với bức tranh tri (tri là bức tranh treo trước bức tranh  $i$ ; tri =  $i-1$  nếu  $i < k+1$  và tri =  $i-2$  nếu  $i = k+1$ ), để tìm  $\max s[i,j]$ , khi  $i$  được treo ở  $j$  thì tri chỉ có thể được treo ở  $start-1$  tới  $j-1$ , vì vậy hàm quy hoạch động của ta sẽ là:

**$s[i,j] := \max(s[i,j], s[tri,u] + v[i,j])$ ; trong đó  $u = start-1$  tới  $j-1$**

Để tìm lại kết quả chúng ta dùng mảng **Tr** (tr[i,j] tức là vị trí treo bức tranh tri) Cách làm trên sẽ được trình bày ở bài mẫu (ở đây giải quyết với  $\max M = 150$ , vì đề bài không nói rõ  $\max M$ , chúng ta hoàn toàn có thể tăng  $\max M$  bằng việc sử dụng thêm mảng động).

Ngoài cách xử lý trên chúng ta có thể viết chương trình quy hoạch động dễ dàng bằng cách, ứng với mỗi bức tranh  $k$  được treo ở cửa thì ta sẽ loại bức tranh đó ra khỏi mảng  $v$ , tức là trong mảng  $v$  chỉ còn  $n-1$  hàng. Việc làm này được thực hiện đơn giản bằng mảng  $cs[i]$   $i = 1..n-1$  trong đó  $cs[i]$  là chỉ số của bức tranh sẽ được bố trí ở trong phòng, như vậy trong mảng  $cs$  sẽ không có giá trị  $cs[i] = k$ .

Sau đây là toàn bộ văn bản chương trình mẫu:

```
const MaxN = 51;
      MaxM = 150;
      fi='picture.inp3p';
      fo='picture.out';
type mang =array[0..MaxN,0..maxM] of byte;
      mang1 =array[0..MaxN,0..maxM] of integer;
var f:text;
      n,m,lvtr,luumax,lk,ln:longint;
      c:array[0..maxN] of byte;
      v,tr,ltr:mang;
      s,ls:mang1;
{*-----*}
procedure int;
var i,j:integer;
begin
  assign(f,fi); reset(f);
  readln(f,n,m);
  for i:=1 to n do read(f,c[i]);
  for i:=1 to n do
    for j:=1 to m do
      read(f,v[i,j]);
  close(f);
```

```

    assign(f,fo); rewrite(f);
    luumax:=0;
end;
{ * _____ *}
procedure check_result(k:byte);
var j,max,vtr,so:longint;
begin
    max:=0;
    so:=n;
    if n=k then dec(so);
    for j:=1 to m do
        if s[so,j] > max then
            begin max:=s[so,j]; vtr:=j;end;
    if max+c[k] > LuuMax then
        begin
            LuuMax:=max+c[k];
            ls:=s;
            ltr:=tr;
            lvtr:=vtr;
            lk:=k;
            ln:=so;
        end;
    end;
end;
{ * _____ *}
procedure process;
var i,j,sl,k,u,start,finish,tri:integer;
begin
    sl:=m-n+2; {so luong}
    for k:=1 to n do
        begin
            fillchar(s,sizeof(s),0);
            for i:=1 to n do
                if i<>k then
                    begin
                        start:=i;
                        finish:=i+sl-1;
                        if i>k then
                            begin
                                dec(start);
                                dec(finish);
                            end;
                        for j:=start to finish do
                            for u:=start-1 to j-1 do
                                begin
                                    tri:=i-1;
                                    if i = k+1 then dec(tri);
                                    if s[i,j]
                                        begin

```

```

        s[i,j]:=s[tri,u]+v[i,j];
        tr[i,j]:=u;
    end;
end;
end;
check_result(k);
end;
end;
{ *-----* }
procedure inkq(n,m:byte);
var trn:byte;
begin
    if n > 0 then
        begin
            trn:=n-1;
            if n = lk+1 then dec(trn);
            inkq(trn,ltr[n,m]);
            write(f,m,' ');
        end;
    end;
end;
{ *-----* }
procedure print;
var i,j:integer;
begin
    writeln(f,luumax);
    writeln(f,lk);
    inkq(ln,lvtr);
    close(f);
end;
{ *-----* }
BEGIN
    int;
    process;
    print;
END.

```

### ***Giải bài 157/2003 – Biến đổi xâu***

#### ***Thuật toán:***

Đây là một bài toán Quy hoạch động thuần túy do đó cách giải tốt nhất là phương pháp Quy hoạch động.

Ta thiết lập hàm Quy hoạch động trên mảng 2 chiều:  $L[i,j]$  cho khoảng cách ngắn nhất giữa 2 xâu con của  $X$  và  $Y$  là  $X_{1..i}$  và  $Y_{1..j}$ . Dễ thấy:

- $L[i,0]=i$ : Bắt buộc ta phải xoá cả  $i$  ký tự ở  $X_{1..i}$  để thành  $Y_{1..j}$  rỗng
- $L[0,j]=j$ : Bắt buộc ta phải chèn cả  $j$  ký tự từ  $Y_{1..j}$  sang  $X_{1..i}$  rỗng
- $L[i,j]$  với  $i,j>0$  có 2 khả năng xảy ra:

+ Nếu  $X_i = Y_j$  thì hiển nhiên  $L[i,j]=L[i-1,j-1]$

+ Nếu  $X_i \neq Y_j$  ta có 3 cách biến đổi: Xoá  $X_i$  đi, Thay  $X_i$  bằng  $Y_j$  và Chèn  $Y_j$  vào sau  $X_i$ , ta lấy cách nào tối ưu nhất:  $L[i,j]=\min(L[i-1,j]+1; L[i-1,j-1]+1; L[i,j-1]+1)$

Công việc tiếp theo đó là xuất dữ liệu. Để ý thấy nhờ chính mảng L ta biết được ngay phép biến đổi tại từng bước – là các bước ngược lại, ta không cần lưu trữ kết quả tại từng bước Quy hoạch động như các bài toán khác. Chẳng hạn:  $L[i,j]=L[i-1,j]+1$  tức là tại bước này dùng phép xoá,  $L[i,j]=L[i,j-1]$  tức là tại bước này dùng phép chèn...v.v...

Đề bài không có yêu cầu lớn về dữ liệu, hai xâu được hiểu là không quá 255 ký tự để các bạn có thể dùng kiểu String, tuy nhiên như cách cài đặt của chương trình mẫu các bạn sẽ thấy chiều dài xâu có thể lớn hơn nhiều cũng được. Với hàm Quy hoạch động ở trên chúng ta không thể tránh khỏi việc dùng mảng động (cho xâu 255 ký tự) tuy nhiên đa số các bạn chưa khéo khi khai báo mảng động, các bạn khai báo ngay mảng 2 chiều trên Heap nên tối đa là  $[0..254,0..254]$ , bằng cách khai báo một mảng 1 chiều trên Heap và một mảng của mảng trên Heap đó chúng ta coi như có một mảng 2 chiều mà kích thước có thể lớn hơn nhiều. Một số bạn đã làm được việc này tốt.

*Chương trình mẫu:*

Program BienDoiXau;

Const      fi=&rsquo;xau.inp&rsquo;;  
            fo=&rsquo;xau.out&rsquo;;

Type mang=array[0..255] of byte;

Var a,b: string;

    n,m: byte;

    l: array[0..255] of ^mang;

    Mark\_Pointer: pointer;

{\*\_\_\_\_\_\*

Procedure Nhap;

Var f: text;

    Begin

        assign(f,fi); reset(f);

        readln(f,a);

        readln(f,b);

        close(f);

        n:=length(a);

        m:=length(b);

    End;

{\*\_\_\_\_\_\*

Procedure QHD;

Var i,j: byte;

    Begin

        for i:=0 to n do new(l[i]);

        for i:=0 to n do l[i]^0:=i;

        for i:=0 to m do l[0]^i:=i;

        for i:=1 to n do

            for j:=1 to m do

                if a[i]=b[j] then l[i]^j:=l[i-1]^j-1] else

                begin

                    l[i]^j:=l[i-1]^j-1]+1;

                    if l[i]^j>l[i]^j-1]+1 then l[i]^j:=l[i]^j-1]+1;

                    if l[i]^j>l[i-1]^j]+1 then l[i]^j:=l[i-1]^j]+1;

                end;

    End;

```

{*_____ *}
Procedure Xuat;
Var f: text;
    i: byte;
Begin
    assign(f,fo); rewrite(f);
    writeln(f,l[n]^m);
    while (n>0) and (m>0) do
        if l[n]^m=l[n-1]^m+1 then
            begin
                writeln(f,'I',n+1,' ',b[m]);
                dec(m);
            end
        else if l[n]^m=l[n-1]^m+1 then
            begin
                writeln(f,'D ',n); dec(n);
            end
        else
            if l[n]^m=l[n-1]^m+1 then
                begin
                    writeln(f,'R',n,' ',b[m]);
                    dec(m); dec(n);
                end
            else
                begin
                    dec(n); dec(m);
                end;
        if (n=0) and (m>0) then
            for i:=m downto 1 do writeln(f,'I ',b[i])
        else
            for i:=n downto 1 do writeln(f,'D ');
        close(f);
    End;
{*_____ *}
BEGIN
    Mark(Mark_Pointer);
    Nhap;
    QHD;
    Xuat;
    Release(Mark_Pointer);
END.

```

### ***Giải bài 158/2003 – Tuổi của hai anh em***

Tuổi của học sinh bước vào lớp 1 là 6, nghĩa là tuổi hơn số thứ tự của lớp là 5. Do đó Sơn nhiều hơn em 5 tuổi. Vậy khi Sơn học hết lớp 12 thì Dũng học hết lớp 7.

### ***Giải bài 159/2003 – Dãy con lùi***

*Phân tích bài toán:* Theo định nghĩa của đề bài: "Dãy giá trị nguyên  $A = (A_1, A_2, A_3, \dots, A_N)$  được gọi là lùi, nếu nó giảm dần từ  $A_1$  đến một  $A_i$  nào đó, rồi tăng dần tới  $A_N$ " ta thấy rằng

phần tử đặc biệt trong một *dãy lồi* là *điểm gãy*  $A_i$ , dãy đơn điệu tăng về hai phía của *điểm gãy*. Do không được định nghĩa rõ nên ở đây chúng ta tự ngầm định:  $1 < i < N$  (nếu trùng với 1 hoặc  $N$  thì dãy trở thành đơn điệu tăng hoặc giảm).

Từ nhận xét trên, ta giải quyết bài toán như sau:

– Xây dựng mảng  $U$  thỏa mãn:  $U[i]$  số phần tử của dãy giảm dài nhất là dãy con của dãy  $X_1, X_2, \dots, X_i$ , với  $X_i$  là phần tử nhỏ nhất của dãy con.

$$\text{Vậy: } i=1: U[i]=1$$

$$i>1: U[i] = \max_{1 \leq j < i}^{X[j] \geq X[i]} \{U[j] + 1\}$$

– Xây dựng mảng  $D$  thỏa mãn:  $D[i]$  số phần tử của dãy tăng dài nhất là dãy con của dãy  $X_i, X_{i+1}, \dots, X_n$ , với  $X_i$  là phần tử nhỏ nhất của dãy con.

$$\text{Vậy: } i=n: D[i] = 1$$

$$i<n: D[i] = \max_{i < j \leq n}^{X[j] < X[i]} \{D[j] + 1\}$$

– *Điểm gãy* được chọn là *điểm vt* thỏa mãn:

$$U[vt] + D[vt] = \max_{1 \leq i < n}^{D[i] > 1, D[j] > 1} \{U[i] + D[j]\}$$

( $D[i] > 1$  và  $D[j] > 1$  để đảm bảo *điểm gãy* không nằm ở đầu mút)  
Độ dài của *dãy lồi* dài nhất là:  $U[vt] + D[vt] - 1$ .

*Chương trình mẫu hoàn chỉnh:*

```
const fin='DAYLOI.INP';
      fou='DAYLOI.OUT';
      nmax=2000;
type arr= Array[0..nmax+1] of integer;
var A: arr;
    U,Tru,D,TrD: arr; {Up, Down}
    n, KyLuc, Vitri: integer;
    f: text;
{*_____*}
procedure Nhap;
var i: integer;
begin
  assign(f,fin); reset(f);
  readln(f,n);
  for i:=1 to n do read(f,A[i]);
  close(f);
end;
{*_____*}
procedure GhiLen(i: integer);
begin
  if TrU[i]<>0 then GhiLen(TrU[i]);
  Write(f,A[i], ' ');
end;
{*_____*}
procedure GhiXuong(i: integer);
```

```

begin
    Write(f,A[i], ' ');
    if TrD[i]<>0 then GhiXuong(TrD[i]);
end;
{*-----*}
procedure Xuat;
var i: integer;
begin
    assign(f,fou); rewrite(f);
    writeln(f,KyLuc);
    if KyLuc >0 then
        begin
            GhiLen(ViTri);
            GhiXuong(TrD[ViTri]);
        end;
    close(f);
end;
{*-----*}
procedure ChuanBi;
var i: integer;
begin
    for i:=1 to n do
        begin
            U[i]:=1;
            D[i]:=1;
            TrU[i]:=0;
            TrD[i]:=0;
        end;
end;
{*-----*}
procedure Up;
var i,j: integer;
begin
    for i:=1 to n do
        for j:=1 to i-1 do
            if (A[i] then
                begin
                    U[i]:=U[j]+1;
                    TrU[i]:=j;
                end;
        end;
end;
{*-----*}
Procedure Down;
var i,j :integer;
begin
    for i:=n downto 1 do
        for j:=n downto i+1 do
            if (A[i]

```



```

begin
    D[i]:=D[j]+1;
    TrD[i]:=j;
end;
end;
{*_____ *}
procedure TimDiemGay;
var i: integer;
begin
    KyLuc:=0;
    for i:=1 to n do
        if (U[i]>1) and (D[i]>1) and (U[i]+D[i]-1>KyLuc) then
            begin
                KyLuc:=U[i]+D[i]-1;
                Vitri:=i;
            end;
    end;
end;
{*_____ *}
BEGIN
    Nhap;
    ChuanBi;
    Up;
    Down;
    TimDiemGay;
    Xuat;
END.

```

**Lưu ý:** Do đề bài không chặt chẽ nên một số bài của các bạn cho kết quả là những dãy đơn điệu, còn một số bạn trả lời là không có dãy lời, cả hai kết quả trên đều được chấp nhận. Bài giải mẫu sẽ cho kết quả là 0 với những dãy chỉ có điểm gây trùng với một trong hai điểm đầu mút.

### ***Giải bài 160/2004 – Truyền tin trên mạng***

Đây là một bài toán hay và khá khó về Đồ thị, rất ít bạn tham gia giải bài này, có lẽ do đề bài hơi dài và số lượng dữ liệu đầu vào có vẻ phức tạp nên các bạn đã không muốn làm đến cùng. Tạp chí dành thêm 1 tháng nữa để các bạn tiếp tục suy nghĩ và giải quyết bài toán này.

Qua 4 năm "Đề ra kỳ này" đã đăng hầu hết các thuật toán cơ bản của "Toán – Tin". Sắp tới "Đề ra kỳ này" sẽ nâng cao hơn độ phức tạp của các bài toán, đề cập đến các thuật toán phức tạp hơn, đồng thời các bài toán cũng hay và lý thú hơn với các bạn, sẽ có cả bài khó và dễ để tất cả các bạn cùng làm. Toà soạn sẵn sàng nhận các lời giải của các bạn cho dù chương trình chưa hoàn chỉnh hay chỉ cần thuật giải của bạn, chúng tôi sẽ nhận xét kỹ lưỡng và giúp các bạn học "Toán – Tin" được tốt hơn. Hy vọng các bạn sẽ nhiệt tình tham gia giải bài nhiều hơn nữa, với những chương trình ngày càng tốt hơn!

### ***Giải bài 161/2004 – Người bạn cũ***

Để giải bài toán này ta sử dụng dấu hiệu chia hết cho các số 2, 5, 7. Ta có thể phân tích như sau: Dùng dấu hiệu chia hết cho 2 ta có  $2450 = 2 \times 1225$

Dùng dấu hiệu chia hết cho 5 ta có  $1225 = 5 \times 5 \times 49$

Và  $49 = 7 \times 7$ .

Như vậy  $2450 = 2 \times 5 \times 5 \times 7 \times 7$ . Do tuổi của ba mẹ con là 64 nên tuổi mẹ chỉ có thể là: 35, 49, 50. Xét các trường hợp:

Nếu tuổi mẹ là 35 thì tuổi của 2 con là 10, 7 (loại vì  $35 + 10 + 7$  khác 64)

Nếu tuổi mẹ là 49: có 2 các trường hợp xảy ra: 10 và 5 (thỏa mãn); 2 và 25 (loại vì  $49 + 2 + 25$  khác 64)

Nếu tuổi mẹ là 50: thì tuổi của hai con chỉ có thể là 7 và 7 (loại vì hai đứa nhỏ không phải là anh em sinh đôi).

Kết luận tuổi của mẹ (người bạn) là 49.

### **Giải bài 162/2004 – Dây chuyền thông báo**

**Tư tưởng thuật toán:** Thực chất đây là một bài toán đồ thị, nếu ta coi mỗi học sinh là một đỉnh, thì đỉnh  $i$  nối với đỉnh  $j$  khi học sinh  $j$  là người kế tiếp của học sinh  $i$ .

Ta thấy một học sinh  $j$  có thể là người kế tiếp của rất nhiều học sinh khác, hoặc sẽ không là người kế tiếp của bất kì học sinh nào (tức là sẽ không có đỉnh  $i$  nào nối tới  $j$ ). Giả sử ta có một học sinh  $j_0$  như vậy,  $j_0$  sẽ chọn cho mình một người kế tiếp  $j_1$ , sau đó  $j_1$  lại chọn cho mình người kế tiếp  $j_2, \dots$  quá trình này cứ tiếp diễn cho đến khi người kế tiếp mà  $j_n$  chọn trùng với người kế tiếp của  $j_i$  nào đó. Ta có thể hình dung  $j_0 \dots j_n$  đã tạo nên một cây với gốc là  $j_0$  (cây không phân nhánh có gốc là  $j_0$  ngọn là  $j_n$ ). Như vậy mỗi học sinh  $j$  không được ai chọn là người kế tiếp đều tạo ra một cây, giả sử có  $a$  cây như thế. Những học sinh không tham gia ở bất kì một cây nào sẽ tạo ra những chu trình (vì mỗi học sinh của nhóm này đều được một học sinh của nhóm chọn là người kế tiếp). Giả sử có  $b$  chu trình, bây giờ ta sẽ tìm cách ghép  $a$  cây và  $b$  chu trình (ta coi chu trình là một cây khép kín, đỉnh  $i$  bất kì của nó là ngọn, đỉnh  $i+1$  là gốc) để tạo ra một chu trình duy nhất. Rất đơn giản ta chỉ việc lấy ngọn của cây này ghép với gốc của cây kia như vậy ta cần ít nhất  $a+b$  cách thay đổi để nhận được một dây chuyền thông báo tốt.

Từ thuật toán trên ta có thể cài đặt chương trình như sau:

```
uses crt;
const max=10001;
    fi='tb.inp';
    fo='tb.out';
    fu='tb.luu';
var next:array[1..max] of integer;
    tt,ok:array[1..max] of byte;
    count,n:integer;
    f,g:text;
{*-----*thnt*-----*}
procedure init;
var i,j:integer;
begin
    assign(f,fi); reset(f);
    fillchar(tt,sizeof(tt),0);
    readln(f,n);
    for i:=1 to n do
        begin
            read(f,next[i]);
            tt[next[i]]:=1;
        end;
    close(f);
```

```

    assign(f,fu); rewrite(f);
    count:=0;
end;
{*_____*thnt*_____*}
procedure get_tree(k:integer);
var i,finish:integer;
begin
    inc(count);
    i:=k;
    while ok[i]=0 do
        begin
            ok[i]:=1;
            finish:=i;
            i:=next[i];
        end;
    writeln(f,k,' ',finish);
end;
{*_____*thnt*_____*}
procedure main;
var i,j:integer;
begin
    fillchar(ok,sizeof(ok),0);
    for i:=1 to n do
        if tt[i]=0 then get_tree(i); {cay}
        for i:=1 to n do
            if ok[i]=0 then get_tree(i); {chu trinh}
    close(f);
end;
{*_____*thnt*_____*}
procedure show;
var start,finish,store_start,i:integer;
begin
    assign(f,fu);
    reset(f);
    assign(g,fo);
    rewrite(g);
    writeln(g,count);
    read(f,store_start);
    for i:=1 to count-1 do
        begin
            readln(f,finish);
            read(f,start);
            writeln(g,finish,' ',start);
        end;
    readln(f,finish);
    writeln(g,finish,' ',store_start);
    close(f);
    close(g);

```

```

end;
{ * _____ * thnt * _____ * }
BEGIN
  init;
  main;
  show;
END.

```

**Nhận xét:**

Nhìn chung các bạn đều nhìn nhận bài toán trên bằng đồ thị, vì vậy các thuật toán đưa ra đều cho kết quả đúng. Tuy nhiên phần cài đặt lại chưa tốt, cách tổ chức dữ liệu chưa hợp lý. Chúng tôi rất hoan nghênh những bạn viết thuật toán ra một file riêng, như vậy sẽ tiện theo dõi và đánh giá chính xác được bài làm của các bạn.

***Giải bài 163/2004 – Bán hàng***

**Thuật toán:** Gọi  $A[i,j]$  là số hàng hoá lớn nhất bán được khi tới chợ  $i$  tại thời điểm  $j$ .

Ta có hàm quy hoạch động của  $A[i,j]$ :  $A[i,j] = \text{Max}\{A[i-1, j-t[i-1]-k] + B[i,k]\}$

*Trong đó:*

–  $t[i-1]$ : là khoảng thời gian di chuyển từ chợ thứ  $(i-1)$  đến chợ thứ  $i$ .

–  $k$ : thời gian bán hàng tại chợ thứ  $i$ ;  $k$  nhận các giá trị từ 0 đến  $\sum_{\ell=1}^{q[i]} g[i, \ell]$  (tổng số thời gian có thể bán hàng ở chợ  $i$ ).  $k=0$  khi đi qua chợ  $i$  mà không dừng lại bán hàng.

–  $B[i,k]$ : Số hàng hoá bán được trong  $k$  phút đầu tại chợ thứ  $i$ .

*Từ thuật toán trên, ta cài đặt như sau:*

Với yêu cầu của đề bài là số chợ  $n \leq 100$ , tổng số phút đi bán hàng nhỏ hơn 1800, ta không thể lưu toàn bộ giá trị của mảng quy hoạch động cũng như những mảng liên quan đến việc ghi kết quả. Mặt khác ta thấy rằng, để tính toán số lượng hàng hoá tối ưu tại một chợ, ta chỉ cần quan tâm đến giá trị tối ưu của chợ ngay trước đó. Do vậy ta dùng kĩ thuật "quay vòng" hai mảng một chiều  $a1, a2$ .

*Chương trình mẫu hoàn chỉnh:*

```

Program BanHang;
const fin='BANHANG.INP';
      fou='BANHANG.OUT';
      TimeMax=1800;
      nmax=100;
      qmax=5;
type m1int= array[0..TimeMax+1] of integer;
      m1byte= array[0..nmax+1] of byte;
      m2int= array[0..nmax+1,0..qmax] of integer;
      TGian= array[0..nmax] of integer;
var a1, a2, trx, try: m1int;
    g,K : m2int; q: m1byte;
    t,t1,KetQua: TGian;
    n,tmax : integer;
    vt,Lx,Ly,KyLuc : integer;
    f: text;
{ * _____ * }
procedure Nhap;

```

```

var i,j: integer;
begin
  assign(f,fin); reset(f);
  readln(f,n,tmax);
  t[0]:=0;
  for i:=1 to n-1 do read(f,t[i]);
  for i:=1 to n do
    begin
      read(f,q[i]);
      for j:=1 to q[i] do read(f,g[i,j]);
    end;
  for i:=1 to n do
    begin
      read(f,q[i]);
      for j:=1 to q[i] do read(f,K[i,j]);
    end;
  close(f);
end;
{ * _____ * }

```

```

procedure InKetQua;
var i: integer;
begin
  assign(f,fou); rewrite(f);
  for i:=1 to n do write(f,KetQua[i],');
  writeln(f);
  write(f,KyLuc);
  close(f);
end;
{ * _____ * }

```

```

procedure ChuanBi;
var i,j: integer;
begin
  fillchar(KetQua,sizeof(KetQua),0);
  t1:=t;
  for i:=1 to n do
    begin
      t[i]:=t[i]+t[i-1];
      g[i,0]:=0;
      for j:=1 to q[i] do inc(g[i,0],g[i,j]);
    end;
  end;
{ * _____ * }

```

```

function min(x,y: integer): integer;
begin
  if x
  end;

```

*{Hàm tính số hàng hoá bán được trong y phút đầu tại chợ x }*

```

function Ban(x,y: integer): integer;

```

```

var i,j,m,tg: integer;
begin
  i:=0;
  j:=1;
  m:=0;
  tg:=0;
  repeat
    inc(i);
    inc(m);
    inc(tg,k[x,j]);
    if m=g[x,j] then
      begin
        inc(j);
        m:=0;
      end;
  until (i=y) or (j>q[x]);
  Ban:=tg;
end;
{ * _____ * }
procedure CapNhap(i,j,k,tg: integer);
begin
  a2[j]:=tg+a1[j-k-t1[i-1]];
  trx[j]:=i-1;
  try[j]:=k;
end;
{ * _____ * }
procedure GhiKyLuc;
var i,tg: integer;
begin
  tg:=0;
  for i:=1 to tmax do
    if a2[tg]
      if KyLuc
        begin
          vt:=tg;
          KyLuc:=a2[vt];
          Lx:=Trx[vt];
          Ly:=Try[vt];
        end;
  end;
end;
{ * _____ * }
procedure BanToiUu(n,tmax: integer);
var i,j,k,m,tg: integer;
begin
  fillchar(a1,SizeOf(a1),0);
  a2:=a1;
  trx:=a1;
  try:=a1;

```

```

    for i:=1 to n do
    begin
        m:=min(tmax-t[i-1], g[i,0]);
        for j:=1 to t[i-1]+1 do a2[j]:=0;
        for j:=t[i-1]+1 to tmax do
            begin
                CapNhap(i,j,0,0);
                for k:=1 to m do
                    if j-k-t1[i-1]>=0 then
                        begin
                            tg:=Ban(i,k);
                            if a2[j]
                                CapNhap(i,j,k,tg);
                        end
                    else break;
                end;
            end;
        a1:=a2;
        GhiKyLuc;
    end;
end;
{*****}
procedure XuLi;
var i, SoHang, x, y: integer;
begin
    BanToiUu(n,tmax);
    x:=Lx+1;
    y:=Ly;
    SoHang:=KyLuc-Ban(x,y);
    while SoHang>0 do
    begin
        KetQua[x]:=y;
        dec(tmax,y+t1[x-1]);
        BanToiUu(x-1,tmax);
        x:=trx[tmax]+1;
        y:=try[tmax];
        SoHang:=SoHang-Ban(x,y);
    end;
    KetQua[x]:=y;
end;
{*****}
BEGIN
    Nhap;
    ChuanBi;
    XuLi;
    InKetQua;
END.

```

**Giải bài 164/2003 – Rán bánh**

Đề rán 9 miếng bánh, lúc đầu ta đặt 6 miếng vào chảo, một mặt đã chín, ta trở mặt khác của 3 miếng, bỏ ra 3 miếng và đặt 3 miếng còn lại vào chảo. Sau 1 phút, 3 miếng đã chín cả 2 mặt được lấy ra và thay bằng 3 miếng đã chín 1 mặt để ở ngoài, 3 miếng mới chín một nửa ở trong chảo cũng được trở sang mặt kia. Sau 30 giây, 6 miếng trong chảo đã chín đều hai mặt. Tất cả hết 1 phút 30 giây.

Như vậy để rán **9** miếng bánh cần **1 phút 30 giây**. Trên cơ sở này ta có: Để rán 15 miếng bánh, ta rán 6 miếng cả 2 mặt hết 1 phút, 9 miếng còn lại hết 1 phút 30 giây, tổng cộng hết 2 phút 30 giây.

Để rán 33 miếng, ta rán 24 miếng hết 4 phút, 9 miếng còn lại hết 1 phút 30 giây, tổng cộng hết 5 phút 30 giây.

Đáp án: 1 phút 30 giây; 2 phút 30 giây; 5 phút 30 giây.

**Giải bài 165/2003 – Tìm vị trí****Tư tưởng thuật toán:**

Đây là một bài toán tối ưu, nên phương pháp tốt nhất mà chúng ta chọn để giải quyết là Quy Hoạch Động. Gọi  $S[i,j]$  là khoảng cách nhỏ nhất cần tìm khi đã bố trí được  $i$  bể xăng, trong đó bể xăng thứ  $i$  được bố trí ở cây xăng  $j$ . **Để thấy  $s[1,j] = d[j]-d[1]$** , và  **$s[i,i] = 0$**  (vì  $i$  bể xăng được bố trí ở  $i$  cây xăng liên tiếp lên). Ta sẽ tìm cách bố trí lần lượt từng bể xăng một. Nhận thấy bể xăng  $i$  có thể được bố trí từ cây xăng  $i$  đến cây xăng  $n - (k-i)$ . **Công thức quy hoạch động:  $s[i,j] := \text{Max}(s[i-1,h], ff(h,j))$** ; trong đó:  $h$  là vị trí có thể bố trí được của bể xăng  $i-1$  khi bố trí bể xăng  $i$  tại vị trí  $j$ , như vậy  **$h: i-1 \dots j-1$** . Còn  **$ff(h,j)$**  là khoảng cách lớn nhất của các cây xăng tới bể xăng gần nó nhất (các cây xăng này nằm giữa **bể xăng  $i-1$**  và  $i$ , nói cách khác là các cây xăng này nằm giữa **cây xăng  $h$**  và **cây xăng  $j$** ). Việc tính  **$ff(h,j)$**  rất đơn giản (tham khảo chương trình mẫu). Sau khi xây dựng được bảng  $s$ , ta sẽ tìm ngược lại vị trí đặt các bể xăng, để làm được điều này ta cần có mảng  **$Vtr[i,j]$** : bể xăng  $i-1$  đặt tại vị trí cây xăng  $vtr[i,j]$ .

**Chương trình mẫu:**

```
const fi='vitri.inp';
fo='vitri.out';
MaxN=201;
MaxK=31;
type mang = array[0..MaxN] of longint;
var s,vtr: array[0..MaxK,0..MaxN] of longint;
a : mang;
f : text;
n,k : longint;
{*_____*thnt*_____*}
procedure read_file;
var i: integer;
begin
  assign(f,fi); reset(f);
  readln(f,n,k);
  for i := 1 to n do read(f,a[i]);
  close(f);
end;
{*_____*thnt*_____*}
function max(x,y: longint): longint;
begin
```



```

    if x > y then max := x
    else max:= y;
end;
{*_____*thnt*_____*}
function min(x,y: longint): longint;
begin
    if x > y then min:= y
    else min := x;
end;
{*_____*thnt*_____*}
procedure init;
var i: integer;
begin
    fillchar(s,sizeof(s),0);
    vtr:=s;
    for i:= 2 to n do s[1,i]:=a[i]-a[1];
    for i:=1 to k do vtr[i,i]:=i-1;
end;
{*_____*thnt*_____*}
function ff(u,v:integer):integer;
var m:longint; i:byte;
begin
    m:=(a[u]+a[v]-1) div 2+1; {diem o giua}
    i:=u1;
    while a[i] diem giua}
    ff:=max(a[i-1]-a[u],a[v]-a[i]);
end;
{*_____*thnt*_____*}
procedure work;
var i,j,h,value: longint;
begin
    for i:=2 to k do
    for j:=i+1 to n-(k-i) do
    begin
        s[i,j]:=maxlongint;
        for h:=i-1 to j-1 do
        begin
            value:=max(s[i-1,h],ff(h,j));
            if value < s[i,j] then
            begin
                s[i,j]:=value;
                vtr[i,j]:=h;
            end;
        end;
    end;
end;
{*_____*thnt*_____*}
procedure try(k,i:integer);

```

```

begin
  if k <> 0 then
    begin
      try(k-1,vtr[k,i]);
      write(f,i,' ');
    end;
  end;
  {*_____ *thnt*_____ *}
procedure show_result;
var v,i,result,tg: longint;
begin
  assign(f,fo); rewrite(f);
  result := maxlongint;
  for i := n downto k do
    begin
      tg:= max(s[k,i],a[n]-a[i]);
      if tg < result then
        begin
          result := tg;
          v := i;
        end;
    end;
  try(k,v);
  close(f);
end;
{*_____ *thnt*_____ *}
begin
  read_file;
  init;
  work;
  show_result;
end.

```

### ***Giải bài 166/2003 – Đếm cây trên đôi thông***

#### ***Tư tưởng:***

Đây là một bài toán thật sự đơn giản, xuất phát từ lớp các bài toán "Chữ Xiếc" nhưng đã được đơn giản hoá đi rất nhiều – đồng thời tăng thêm khả năng "đánh lừa". "Đề ra kỳ này" muốn thử xem sự tinh táo của các bạn khi nghĩ giải thuật như thế nào, kết quả là có một số bạn đã dễ dàng làm được ngay với chương trình chưa đến 20 dòng, các bạn còn lại thì đã phức tạp bài toán lên với các giải thuật và cài đặt cực kỳ dài dòng, phức tạp. Bài toán này nhắc các bạn nhớ lại châm ngôn "**Best of Simple**"!

Từ giả thiết đề bài là không có bất kỳ loại cây nào khác và không có hai cây thông nào "dính" vào nhau chúng ta thấy đặc điểm để nhận dạng một cây chỉ đơn giản là ngọn cây và gốc cây có dấu '@' nằm giữa 2 dấu cách. Ta đếm số ký tự '@' như vậy rồi chia cho 2 là được số cây. File sẽ được xử lý trên từng dòng một. Rất nhiều bạn đã làm máng động hay một số cách khác để lưu lại toàn bộ file sau đó xử lý (Loang chẳng hạn!) do đề bài đã cho giới hạn số dòng  $\leq 999$  là một con số "đánh lừa" – nếu không cho giới hạn đó chắc hẳn các bạn đã nghĩ ra cách xử lý từng dòng và bài toán trở thành đơn giản! Một số bạn thì xử lý 2

dòng (hay như thầy Nguyễn Xuân Huy gọi là "chuẩn bị trước một nhịp") tuy nhiên việc đó không cần thiết ở bài toán này, các bạn đã rập khuôn mà không nghĩ đến giải thuật đơn giản hơn.

Về mặt tổ chức dữ liệu, mỗi dòng của tệp không thể lưu trong mảng hay String như các bạn được vì chưa có giới hạn về số ký tự trên một dòng, ta đọc từng ký tự trong tệp bình thường mà không quan tâm tới việc xuống dòng (thực chất là 2 ký tự #13 và #10). Tuy nhiên các bạn làm String được xem là đạt yêu cầu, mặc dù thời gian tăng gấp đôi so với cách trước. Khi đọc mỗi dòng vào 1 String như vậy ta phải cộng vào đầu và cuối xâu một dấu cách để dễ đếm các cây "sát biên" có kích thước 1.

### **Chương trình:**

#### **Theo cách 1:**

```
const fi='doithong.fit';
      fo='doithong.hut';
var n: longint; g: text;
      c1,c2,c3: char;
      f: file of char;
procedure main;
begin
  assign(f,fi); reset(f);
  n:=0;
  c1:=#32; read(f,c2);
  while not eof(f) do
  begin
    read(f,c3);
    if ((c1=#32) or (c1=#10)) and (c2='@') and ((c3=#32) or (c3=#13)) then inc(n);
    c1:=c2; c2:=c3;
  end;
  if ((c1=#32) or (c1=#10)) and (c2='@') then inc(n);
  close(f);
  assign(g,fo); rewrite(g);
  write(g,n div 2); close(g);
end;
BEGIN
main;
END.
```

#### **Theo cách 2: (Chương trình của bạn Đặng Duy Nhân)**

```
VAR i:byte;s:string;n:longint;
      f:text;
BEGIN
  assign(f,'doithong.fit'); reset(f);
  n:=0;
  while not eof(f) do
  begin
    readln(f,s);
    s:=' '+s+' ';
    for i:=2 to length(s)-1 do
      if (s[i]='@')and(s[i-1]<>'@')and(s[i+1]<>'@') then inc(n);
```

```

end;
n:=n div 2;
close(f);
assign(f,'doithong.hut'); rewrite(f);write(f,n);close(f);

```

END.

### Nhận xét lời giải của một số bạn:

Bạn *Đặng Duy Nhân* – Lớp 12A12 – Trường THPT Sơn Tịnh 1 – **Quảng Ngãi**: Thuật giải đúng đắn, chương trình đạt yêu cầu. Bạn đã biết cách xử lý trên từng dòng, cộng vào đầu và cuối xâu các dấu cách. Mặc dù thời gian chạy chương trình lâu hơn chương trình mẫu nhưng đây là một trong các lời giải tốt nhất kỳ này.

Bạn "*ABBA*" – **Email**: [popcapfan@yahoo.com](mailto:popcapfan@yahoo.com): Chương trình chạy đúng tuy nhiên bạn cài đặt hơi dài, chúng ta không cần dùng đến 2 mảng và kiểm tra như vậy. Chương trình của bạn có một điểm rất tốt đó là xử lý được số ký tự lớn trên một dòng, mặc dù chưa phải là vô hạn như chương trình mẫu.

Bạn *Duy Hưng*: Bạn đọc dữ liệu không đúng với yêu cầu đề bài, đề bài không cho dữ liệu dạng bảng trong file và không cho trước kích thước bảng. Khi sửa lại tệp vào "theo" chương trình của bạn thì chương trình chạy vẫn sai do tư tưởng của giải thuật chưa đúng, chương trình rất phức tạp, có vẻ bạn muốn tìm các miền liên thông trên bảng mà mình đọc vào.

Bạn *Hoàng Tuấn Hải* – Lớp 12A2 Chuyên Vĩnh Phúc – Tỉnh **Vĩnh Phúc**; *Huỳnh Quang Hiếu* – Lớp 12 Lý – Trường Lê Khiết – **Quảng Ngãi**: Các bạn đều xử lý 2 dòng liên nhau để đếm số cây tuy nhiên chương trình chạy chưa đúng, ngay với test có 3 cây: 2 cây kích thước 1, 1 cây kích thước 3 chương trình bị sai trong khi giải thuật của bài toán cực kỳ đơn giản. Chương trình của bạn Hiếu có lẽ chưa được bạn test (thiếu khai báo biến i).

Bạn *Lê Đình Thuận* – Lớp 11 Trường PTTH Lê Quý Đôn – Tp Quy Nhơn – **Bình Định**: Thuật giải của bạn chưa xử lý được biên bên trái, các test này chương trình đều sai cho kết quả 0.

Bạn *Nguyễn Hữu Phương* – Lớp 11A2 – Trường PTTH Chuyên Vĩnh Phúc – Tỉnh **Vĩnh Phúc**; *Nguyễn Khả Tâm* – Lớp 11 Toán – Trường Chuyên Lương Văn Chánh – Tx Tuy Hoà – **Phú Yên**; *Phan Nguyễn Hoài Anh* – Lớp 11 Toán Tin – Trường THPT Chuyên Lương Thế Vinh – Biên Hoà – **Đồng Nai**: Chương trình của các bạn chạy đều cho kết quả đúng nhưng giải thuật thì hơi phức tạp. Các bạn nên dành nhiều thời gian cho giải thuật hơn là chương trình – chương trình của các bạn đều khá công phu (và rắc rối!), thói quen suy nghĩ đơn giản về mọi việc là rất cần thiết.

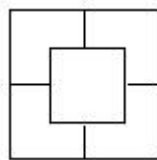
Bạn *Phí Hồng Thái* – Lớp 11A2 – Trường PTTH Chuyên Vĩnh Phúc – Tỉnh **Vĩnh Phúc**: Chương trình bị lỗi "Range Check Error", giải thuật phức tạp và chưa đúng.

Bạn *Trần Quang Đức* – Lớp 12A2 – Trường THPT Chuyên Phan Bội Châu – **Nghệ An**: Chương trình có thuật giải tốt, chạy được số lượng lớn ký tự trên một dòng do bạn sử dụng mảng động. Chương trình xử lý từng dòng một nên số ký tự vẫn bị giới hạn. Các bạn chú ý bộ nhớ Heap đối với Turbo Pascal ta chỉ sử dụng dưới 400K là an toàn.

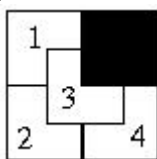
Bạn *Xgenius*: Chương trình chạy tốt, tư tưởng đúng với việc xử lý 2 dòng liên tiếp. Có thể nói bạn đã hoàn thành tốt yêu cầu đề bài.

### Giải bài 167/2004 – Chia đất

Bài toán được giải một cách đơn giản như sau: Chia mảnh vườn hình vuông thành 4 phần bằng nhau, lấy một hình vuông có diện tích bằng hình vuông vừa được chia, sau đó đặt hình vuông này vào tâm của mảnh vườn (như hình vẽ dưới)



Đánh dấu phần đất của ông bố để lại, khi đó ta thu được kết quả:



Với cách chia này sẽ đảm bảo mảnh đất được chia thành 4 phần có hình dáng và kích thước giống nhau.

### ***Giải bài 168/2004 – Đoán số***

#### ***Tư tưởng thuật toán:***

Để giải quyết bài toán trên ta sẽ dùng thuật toán " **Loại trừ** ", giống với phương pháp Sàng số nguyên tố của *Eratosthenes*. Ta nhận thấy có  $6^4 = 1296$  số có 4 chữ số tạo ra từ 6 chữ số 1 đến 6. Đầu tiên ta đoán đại khái một số, thường là 1234, sau khi nhận được 2 câu trả lời. Ta sẽ loại bỏ tất cả các số không có cùng 2 câu trả lời trên với số 1234. Như vậy số lượng số thoả mãn điều kiện sẽ giảm đi rất nhiều, ta lại chọn đại khái một số, rồi tiếp tục làm như trên... Thuật toán dừng lại khi nhận được câu trả lời 2 bằng 4.

Ví dụ số Tin nghĩ là: **4655**. Ta đoán: 1234 Nhận được 2 câu trả lời: 1) 1 2) 0

Như vậy những số như 1111, 1112, 1113... sẽ bị loại vì khi so sánh với 1234 ta sẽ không có câu trả lời là: 1) 1 2) 0. Còn những số như: 2555, 2556, 2566 ... sẽ thoả mãn. Tiếp tục chọn một số trong các số thoả mãn cho đến khi cho được kết quả đúng.

Việc cài đặt bài toán trên không có gì khó khăn.

#### ***Chương trình mẫu:***

```
var n,count,wrong,right,m:integer;
    so:array[1..1300,1..4] of byte;
    tt:array[1..3000] of boolean;
    finish:boolean;
{*-----*ISM*-----*}
procedure prepare;
var i1,i2,i3,i4:integer;
begin
    n:=0;
    for i1:=1 to 6 do
        for i2:=1 to 6 do
            for i3:=1 to 6 do
                for i4:=1 to 6 do
                    begin
                        inc(n);
                        so[n,1]:=i1;
                        so[n,2]:=i2;
                        so[n,3]:=i3;
                        so[n,4]:=i4;
                    end;
                end;
            end;
        end;
    fillchar(tt,sizeof(tt),true);
end;
```

```

{*-----*ISM*-----*}
procedure guess;
var i,j:integer;
begin
  i:=1;
  while not tt[i] do inc(i);
  m:=i;
  if count = 1 then m:=52;
  Write('Doan lan thu ',count:9,' la so: ');
  for j:=1 to 4 do write(so[m,j]);
  writeln;
end;
{*-----*ISM*-----*}
procedure answer;
begin
  Write('Cs co mat sai vi tri la : ');
  readln(wrong);
  Write('Cs dung vi tri la: ');
  readln(right);
end;
{*-----*ISM*-----*}
procedure exclude;
var đ,đ1:array[1..4] of boolean;
    i,j,k,x,y:integer;
begin
  for i:=1 to n do
    if tt[i] then
      begin
        x:=0;
        fillchar(đ,sizeof(đ),false);
        for j:=1 to 4 do
          if so[i,j] = so[m,j] then
            begin
              inc(x);
              đ[j]:=true;
            end;
        y:=0;
        fillchar(đ1,sizeof(đ1),false);
        for j:=1 to 4 do
          if not đ[j] then
            for k:=1 to 4 do
              if (not đ[k])and(not đ1[k]) then
                if so[i,j] = so[m,k] then
                  begin
                    đ1[k]:=true;
                    inc(y);
                    k:=4;
                  end;
      end;
end;

```

```

    if (y<>Wrong) or (x<>right) then
        tt[i]:=false;
    end;
end;
{*-----*ISM*-----*}
procedure check;
begin
    finish:=((wrong=0) and (right=4))or(count>10);
end;
{*-----*ISM*-----*}
procedure play_game;
begin
    finish:=false;
    count:=0;
    repeat
        inc(count);
        guess; {doan}
        answer; {tra loi}
        exclude; {loai tru}
        check; {kiem tra}
    until finish;
    If count>10 then writeln('Finish!, I Win!!!!')
    Else writeln('Computer wins!!!!!!')
end;
{*-----*ISM*-----*}
begin
    prepare;
    play_game;
    readln;
end.

```

### **Nhận xét bài giải một số bạn:**

Đây là một bài toán khá phức tạp, đòi hỏi phải tư duy vì vậy chúng tôi chỉ nhận được bài làm của một số ít bạn. Nhìn chung các bạn đều giải bài toán bằng phương pháp "Loại trừ", tuy nhiên vẫn còn một số sai sót nhỏ nên bài của các bạn đều chạy không chính xác.

### ***Giải bài 169/2004 – Món quà đầu năm***

#### ***Nhận xét về giải thuật của bài toán:***

Chúng ta phân tích lại đề bài một cách đơn giản hơn như sau: Mỗi người ở Huda có một "số hiệu" là  $N$  trong khoảng từ 1 đến 30000 và tương ứng với số hiệu đó là món quà may mắn có số lớn thứ  $N$  trong dãy số quà may mắn có dạng  $3^i \cdot 5^j \cdot 7^k$  ( $i, j, k \geq 0$ ;  $i \cdot j \cdot k > 0$ ). Như vậy vấn đề của bài toán là tìm số lớn thứ  $N$  trong dãy số dạng  $3^i \cdot 5^j \cdot 7^k$  (dãy 3, 5, 7, 9, 15, ...). Một trong các cách giải quyết đơn giản nhất là sinh ra dãy một cách bất kỳ sao cho các phần tử của dãy có dạng  $3^i \cdot 5^j \cdot 7^k$  này và sắp tăng dần dãy, chẳng hạn ta tạo được dãy  $A[1], A[2], \dots$  khi đó người có số hiệu  $N$  sẽ nhận món quà may mắn là  $A[N]$ . Với  $N \leq 30000$ , con số may mắn lớn nhất sẽ là một con số rất lớn với 37 chữ số

(027127527401674550420284271240234375) điều đó có thể là điều đáng sợ nhất do phải xử lý số lớn và lại phải sắp xếp 30000 số nếu như bạn không nghĩ đến một cách khác tốt hơn, đó có lẽ là lý do bài 169 kỳ này ít bạn tham gia giải mặc dù đây là một bài toán hay

(và đơn giản!).

Có một phương pháp rất hay được dùng cho một số lớp bài toán dãy số đó là **phương pháp Sinh**. Trong bài toán này phương pháp Sinh được áp dụng một cách riêng đơn giản và hiệu quả như các bạn sẽ thấy, giải thuật riêng đó được sử dụng như một bài toán con cho khá nhiều bài toán khác và được các bạn lập trình gọi tên là giải thuật "Sức mạnh", trong các đề thi HSG một số năm trước đây ở Hải Phòng, Ninh Bình... có sử dụng giải thuật này (theo các bạn học sinh cho biết thì cái tên "Sức mạnh" được các bạn HSG Hải Phòng và Ninh Bình gọi đầu tiên, sau được dùng trong các tài liệu nội bộ, còn một số giải thuật khác nữa giống như vậy mà Đề ra kỳ này sẽ giới thiệu với các bạn trong các số tới). Giải thuật Sinh đơn giản đó như sau:

– Mục đích: Tạo ra dãy  $A[1], A[2], A[3], \dots$  tăng dần đến  $N=30000$

– Nguyên tắc: Sinh ra số  $A[N]$  dựa vào các số trước  $A[1], A[2], \dots, A[N-1]$  sao cho  $A[N]$  đảm bảo là gần  $A[N-1]$  nhất.

**Nhận xét:**  $A[N]$  bằng tích của 3, 5 hoặc 7 với một trong các số đã biết, sao cho  $A[N]$  nhỏ nhất. Hay  $A[N] = \min\{3 \cdot A[i], 5 \cdot A[i], 7 \cdot A[i]\}, i=1..n$ .

Tuy nhiên chúng ta không cần phải thử  $i$  từ 1 đến  $n$  như vậy, chúng ta dùng 3 chỉ số  $i3, i5, i7$  cho việc tìm min này – chúng được tăng lên theo từng bước tìm  $A[N]$  mà không cần kiểm tra từ 1 như sau:

$A[0] := 1;$

$i3 := 0; i5 := 0; i7 := 0;$

For  $i := 1$  to  $N$  do

    Begin

        While  $3 \cdot A[i3] \leq A[i-1]$  do Inc( $i3$ );

        While  $5 \cdot A[i5] \leq A[i-1]$  do Inc( $i5$ );

        While  $7 \cdot A[i7] \leq A[i-1]$  do Inc( $i7$ );

$A[i] := \min(3 \cdot A[i3], 5 \cdot A[i5], 7 \cdot A[i7]);$

    End;

Đó là một giải thuật rất đơn giản và nhanh gọn. Sau khi đã có dãy  $A[1..30000]$  được "tính sẵn" (việc này được gọi là Lookup) chúng ta dễ dàng làm phần còn lại. Chương trình sẽ không đơn giản như trên vì chúng ta còn phải Lookup dữ liệu vào mảng động mới đủ, phải xử lý số nguyên lớn, nếu bạn dùng kiểu Extended giả nguyên thì  $N$  tối đa khoảng 3000 nhưng chương trình sẽ đơn giản với đoạn mã y như trên. Các bạn có lời giải tốt đều dùng giải thuật "Sức mạnh" và xử lý mảng động, số lớn tốt. Có bạn dùng cách sắp xếp, đó là một giải thuật không hay và tốn nhiều thời gian. Đề ra kỳ này đưa chương trình của bạn *Cao Minh Anh* để các bạn tham khảo, đây là một lời giải tốt.

{Cao Minh Anh – lớp 12Lý – trường PTTH chuyên Le Khiết – **Quang Ngai**.

**Thuật toán:** Dùng 3 mảng tu 0.30000 để lưu số mu của số may man có số hiệu thu  $i$ .}

program Mon\_qua\_dau\_nam;

uses crt;

const fi='HelpUs.txt';

fo='ForYou.txt';

mn=30000;

type arr=array[0..mn+1] of byte;

var m3,m5,m7: ^arr;

n,v3,v5,v7,m,m1,du,k :integer;

p: pointer;

T1,T2,T3,r,s :real;

L,H :string;



```

    f,g :text;
    {*_-----*}
procedure openf;
begin
    assign(f,fi); reset(f);
    assign(g,fo); rewrite(g);
end;
{*_-----*}
procedure Closef;
begin
    close(f);
    close(g);
end;
{*_-----*}
procedure Solve;
begin
    r:=ln(3)/ln(5);
    S:=ln(7)/ln(5);
    New(m3);
    New(m5);
    New(m7);
    fillchar(m3^,sizeof(m3^),0);
    fillchar(m5^,sizeof(m5^),0);
    fillchar(m7^,sizeof(m7^),0);
end;
{*_-----*}
procedure Findmin(i:integer);
begin
    if (T1<=T2) and (T1<=T3) then
        begin
            m3^[i]:=m3^[v3]+1; m5^[i]:=m5^[v3]; m7^[i]:=m7^[v3];
        end else
            if (T2<=T1) and (T2<=T3) then
                begin
                    m3^[i]:=m3^[v5]; m5^[i]:=m5^[v5]+1; m7^[i]:=m7^[v5];
                end else
                    if (T3<=T1) and (T3<=T2) then
                        begin
                            m3^[i]:=m3^[v7]; m5^[i]:=m5^[v7]; m7^[i]:=m7^[v7]+1;
                        end;
                    end;
end;
{*_-----*}
procedure Sinh;
var i:integer;
begin
    v3:=0;v5:=0;v7:=0;
    for i:=1 to mn do
        begin

```

```

    T1:=(m3^[v3]+1)*r+m5^[v3]+m7^[v3]*s;
    T2:=m3^[v5]*r+(m5^[v5]+1)+m7^[v5]*s;
    T3:=m3^[v7]*r+m5^[v7]+(m7^[v7]+1)*s;
    if T1<=m3^[i-1]*r+m5^[i-1]+m7^[i-1]*s then inc(v3);
    if T2<=m3^[i-1]*r+m5^[i-1]+m7^[i-1]*s then inc(v5);
    if T3<=m3^[i-1]*r+m5^[i-1]+m7^[i-1]*s then inc(v7);
        T1:=(m3^[v3]+1)*r+m5^[v3]+m7^[v3]*s;
        T2:=m3^[v5]*r+(m5^[v5]+1)+m7^[v5]*s;
        T3:=m3^[v7]*r+m5^[v7]+(m7^[v7]+1)*s;
        Findmin(i);
    end;
end;
{ * _____ * }
procedure nhan(H:string;k:byte);
var i:integer;
begin
    L:="";du:=0;
    for i:=length(H) downto 1 do
        begin
            m:=ord(H[i])-48;
            m1:=(m*k+du) mod 10;
            du:=(m*k+du) div 10;
            L:=chr(m1+48)+L;
        end;
    if du<>0 then L:=chr(du48)+L;
    while L[1]=' ' do delete(L,1,1);
end;
{ * _____ * }
procedure Xuat(k:integer);
var i:integer;
begin
    H:='1';
    for i:=1 to m3^[k] do begin nhan(H,3); H:=L; end;
    for i:=1 to m5^[k] do begin nhan(H,5); H:=L; end;
    for i:=1 to m7^[k] do begin nhan(H,7); H:=L; end;
    write(g,H,#32);
end;
{ * _____ * }
procedure Main;
var i:integer;
begin
    read(f,n);
    for i:=1 to n do
        begin
            read(f,k);
            Xuat(k);
        end;
end;
end;
```

```

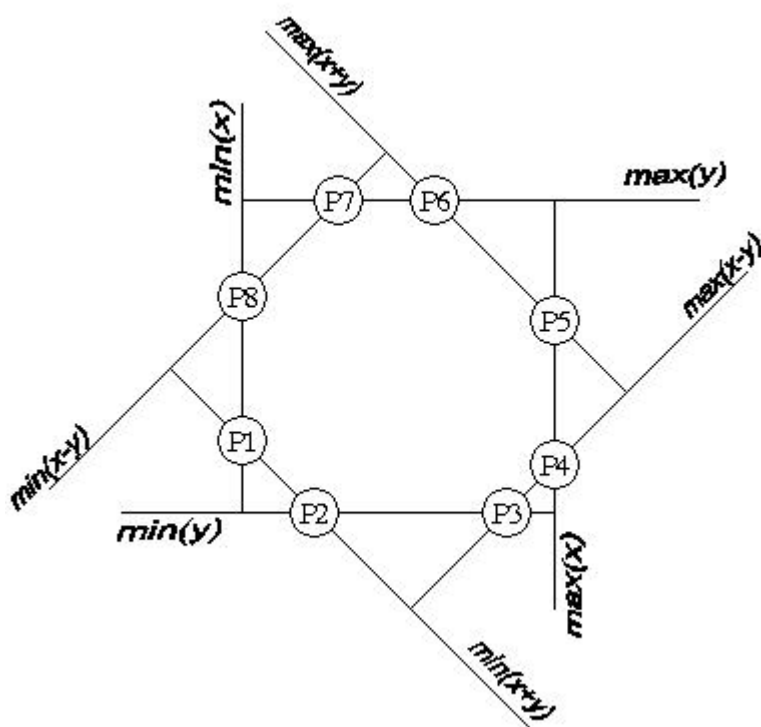
{*-----*}
BEGIN
  clrscr;
  mark(p);
  Openf;
  Solve;
  Sinh;
  Main;
  Closef;
  release(p);
end.

```

### ***Giải bài 170/2004 – Xây dựng hàng rào***

**Phân tích bài toán:** Bài toán được gọi là bài toán "bao lồi chuẩn", các cạnh của đa giác lồi được chọn làm hàng rào có phương song song hoặc tạo với các trục toạ (xích đạo) độ góc  $45^0$ ). Ta có các nhận xét sau:

- Đa giác bao lồi chuẩn sẽ có tối đa 8 cạnh, tương đương với 8 đỉnh.
- Do các điểm giải trí nằm bên trong, trên cạnh hoặc trùng đỉnh "đa giác hàng rào" nên các đỉnh của hàng rào có thể ngoài các điểm giải trí.
- Xét trên hệ trục toạ độ Đề các, các điểm trên cạnh song song với trục tung có hoành độ không đổi, song song với trục hoành có tung độ không đổi, song song với phân giác góc phần tư thứ I có hiệu tung độ và hoành độ không đổi, song song với phân giác góc phần tư thứ II có tổng tung độ và hoành độ không đổi.



Từ nhận xét trên, ta thấy rằng đa giác hàng rào diện tích nhỏ nhất sẽ có các cạnh nằm trên các đường thẳng có phương trình:

- (1)  $x-y=\min(x-y)$
- (2)  $x=\min(x)$
- (3)  $y=\max(y)$
- (4)  $x+y=\max(x+y)$

- (5)  $x = \max(x)$
- (6)  $x - y = \max(x - y)$
- (7)  $y = \min(y)$
- (8)  $x + y = \min(x + y)$

(Các kí hiệu **max()** (hay **min()**) ở đây để chỉ giá trị lớn nhất (hay nhỏ nhất) trong tập hợp các toạ độ đã cho)

Đỉnh của hàng rào là giao điểm của hai đường thẳng liên tiếp theo thứ tự trên (kể cả cặp (1)–(8)).

Trong một số trường hợp đặc biệt, các đỉnh của hàng rào trùng nhau, số cạnh của hàng rào sẽ nhỏ hơn 8.

*Ta có toàn bộ văn bản chương trình.*

```

Program PrincessLand;
const fin = 'PRINCESS.INP';
fou = 'PRINCESS.OUT';
var maxX, maxY, minX, minY,
    maxX_Y, minX_Y, maxXY, minXY: integer;
    n: integer;
    kqX, kqY: array[0..9] of integer;
    f: text;
{*_____*}
procedure KhoiTao;
begin
    maxX := -maxint;
    maxY := -maxint;
    minX := maxint;
    minY := maxint;
    maxX_Y := -maxint;
    minX_Y := maxint;
    maxXY := -maxint;
    minXY := maxint;
end;
{*_____*}
procedure Max(Var T: integer; S: integer);
begin
    if T < S then T := S;
end;
{*_____*}
procedure Min(Var T: integer; S: integer);
begin
    if T > S then T := S;
end;
{*_____*}
procedure Nhap;
var i: integer;
    x, y: integer;
begin
    assign(f, fin); reset(f);
    readln(f, n);

```

```

for i:=1 to n do
begin
  readln(f, x, y);
  Max(maxX, x);
  Max(maxY, y);
  Max(maxXY, x+y);
  Max(maxX_Y, x-y);
  Min(minX, x);
  Min(minY, y);
  Min(minXY, x+y);
  Min(minX_Y, x-y);
end;
close(f);
end;
{ * _____ *}
procedure Xuat;
var i, dem: integer;
begin
  dem:=0;
  for i:=1 to 8 do
    if (kqX[i]<>kqX[i+1]) or (kqY[i]<>kqY[i+1]) then inc(dem);
  assign(f,fou); rewrite(f);
  writeln(f,dem);
  for i:=8 downto 1 do
    if (kqX[i]<>kqX[i+1]) or (kqY[i]<>kqY[i+1]) then writeln(f,kqX[i], ' ', kqY[i]);
  close(f);
end;
{ * _____ *}
procedure XuLi;
var i, j: integer;
begin
  kqX[1]:=minX;
  kqY[1]:=minX-minX_Y;
  kqX[2]:=maxY+minX_Y;
  kqY[2]:=maxY;
  kqX[3]:=maxXY-maxY;
  kqY[3]:=maxY;
  kqX[4]:=maxX;
  kqY[4]:=maxXY-maxX;
  kqX[5]:=maxX;
  kqY[5]:=maxX-maxX_Y;
  kqX[6]:=minY+maxX_Y;
  kqY[6]:=minY;
  kqX[7]:=minXY-minY;
  kqY[7]:=minY;
  kqX[8]:=minX;
  kqY[8]:=minXY-minX;
  kqX[9]:=kqX[1];

```

```

    kqY[9]:=kqY[1];
end;
{ * _____ *}
BEGIN
    KhoiTao;
    Nhap;
    XuLi;
    Xuat;
END.

```

**Nhận xét chung bài giải của các bạn:** Nhìn chung các bạn đều phát hiện ra điểm mấu chốt của bài toán là tìm các tọa độ cực trị (*8 tọa độ đặc biệt như đã trình bày ở trên*), vì thế phần lớn các bạn đều giải tốt bài toán.

### ***Giải bài 171/2004 – Từ nhà tới trường***

Nếu như Tuấn không rời khỏi nhà sớm 8 phút, thì khi quay về nhà, Tuấn đã bị muộn không phải là 10 phút mà là 18 phút. Đây là quãng thời gian để Tuấn đi 2 lần quãng đường đi được. Như vậy, khi Tuấn nhớ rằng mình bị quên bút, Tuấn đã đi được 9 phút, bằng  $\frac{9}{20}$  quãng đường.

### ***Giải bài 172/2004 – Nhà gương cười***

#### ***Tư tưởng thuật toán:***

Thực chất đây là một bài toán loang đơn thuần. Vì chúng ta chỉ có một cửa vào duy nhất là ô (1,1), nên ta sẽ loang bắt đầu từ (1,1). Giả sử ta đang ở ô (i,j), chúng ta sẽ nhìn ra 4 ô kề cạnh (trên dưới, trái phải). Nếu ô kề cạnh là "#" hoặc biên thì chúng ta sẽ phải lắp gương (tăng biến đếm). Nếu ô kề cạnh là "." chúng ta sẽ chuyển vị trí sang ô đó và tiếp tục quá trình loang. Vì chúng ta phải lắp kính cho cả các bức tường lớn nên phải rào xung quanh mảng a là các bức tường (kí tự #). Như vậy ở góc trên trái nhất và góc dưới phải nhất cũng sẽ được lắp gương. Nhưng do yêu cầu đề bài 2 ô này là cửa ra và cửa vào nên trong biến đếm **count (đếm số bức tường được lắp gương)** chúng ta phải trừ đi bốn. Và do mỗi bức tường cần  $3 \times 3 = 9m^2$  gương nên kết quả ra là:  $(count - 4) \times 9$ . Để tiện cho việc xây hàng rào chúng ta quy định 0: là ô bao bọc bởi bức tường, 1: là ô trống.

#### ***Từ thuật toán trên ta có thể cài đặt chương trình như sau:***

```

const fi='Mirror.inp';
      fo='Mirror.out';
      MaxN = 101;
      tu:array[1..4] of integer = (-1,0,1,0);
      tv:array[1..4] of integer = (0,1,0,-1);
var a:array[0..MaxN,0..MaxN] of byte;
      n,count:longint;
      f:text;
{ * _____*thnt* _____ *}
procedure init;
var i,j:integer; kt:char;
begin
    count:=0;
    assign(f,fi); reset(f);
    readln(f,n);

```

```

fillchar(a,sizeof(a),0);
for i:=1 to n do
begin
  for j:=1 to n do
  begin
    read(f,kt);
    if kt = '.' then a[i,j]:=1;
  end;
  readln(f);
end;
close(f);
end;
{ *-----*thnt*-----* }
procedure spread(u,v:byte);
var i:integer;
begin
  a[u,v]:=2;
  for i:=1 to 4 do
  begin
    if a[utu[i],v+tv[i]] = 0 then inc(count)
    else
      if a[utu[i],v+tv[i]] = 1 then spread(utu[i],v+tv[i]);
  end;
end;
{ *-----*thnt*-----* }
procedure ShowResult;
var i,j:integer;
begin
  assign(f,fo); rewrite(f);
  count:=count - 4;
  write(f,count*9);
  close(f);
end;
{ *-----*thnt*-----* }
begin
  init;
  spread(1,1);
  showResult;
end.

```

### ***Giải bài 173/2004 – Cân bằng***

Đây là một bài toán khá hay trong bộ đề thi mẫu được hội đồng IOI 2003 (Mỹ) giới thiệu. Nội dung bài toán và bộ testcase kiểm tra các bạn có thể dễ dàng download ở địa chỉ <http://www.ioi2003.com> để tự kiểm tra.

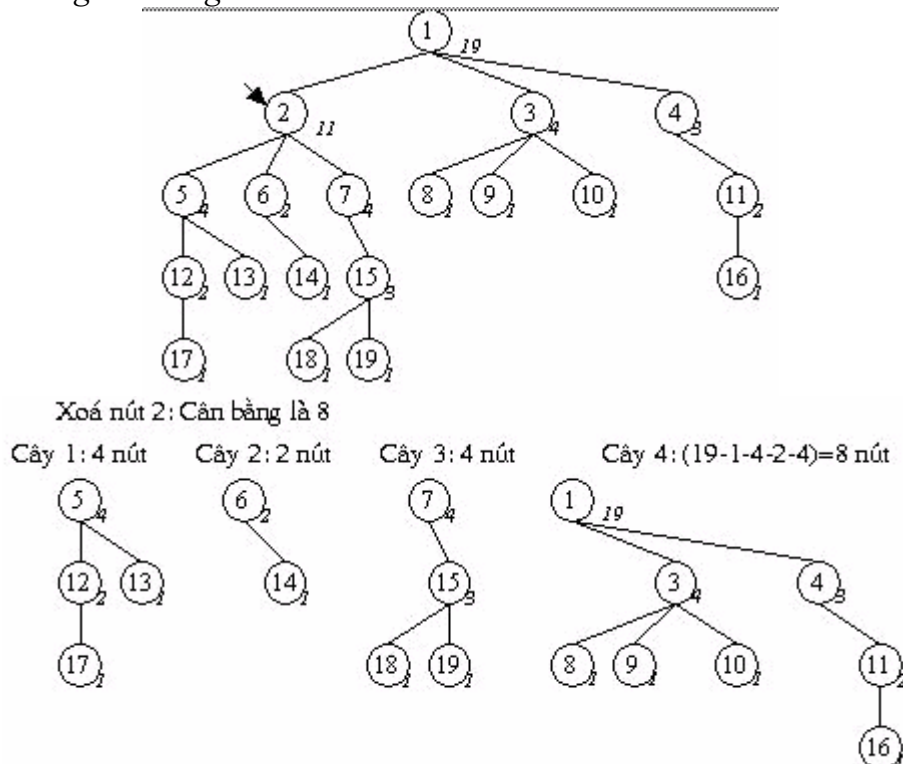
#### **Sau đây, xin được trình bày thuật toán:**

- Gán nhãn cho từng nút, giá trị của nhãn sẽ được tính như sau:
- Các lá (các nút ở biên, không có nút con nào) có nhãn là 1.
- Các nút còn lại có nhãn là tổng giá trị các nhãn con trực tiếp của nó cộng thêm 1 (tính

cho nút đó).

Như vậy, nhận được xây dựng ngược từ các nút lá đi lên.

– Tính cân bằng cho từng nút:



Theo đề bài "Xóa bất cứ nút nào trong cây sẽ sinh ra một rừng: rừng là một tập hợp một hoặc nhiều cây". Theo cách tính nhãn như trên, số lượng cây trong rừng khi xoá bất kì một nút nào đó bằng số nút con của nó cộng thêm 1. Ví dụ nút 2 trong hình vẽ nếu bị xoá đi sẽ sinh ra 4 cây, số nút trong mỗi cây bằng chính nhãn của nút con trực tiếp của 2, trừ cây thứ tư có số nút bằng tổng số nút ban đầu trừ đi nút 2 và tổng số nút của các cây con còn lại.

**Thuận toán được cài đặt cụ thể như sau:**

Program Balacing;

const fin='BALANCE.INP';

fou='BALANCE.OUT';

nmax=20000;

{\*\_\_\_\_\_\*

type m1= array[0..nmax+1] of integer;

var

n: integer;

a: m1;

d,c,tr,t: ^m1;

f: text;

{\*\_\_\_\_\_\*

procedure nhap;

var i: integer;

begin

new(d);

new(c);

new(tr);

new(t);

assign(f,fin); reset(f);



```

    readln(f,n);
    for i:=1 to n-1 do readln(f,d^[i],c^[i]);
    close(f);
end;
{ *-----* }
procedure inkq;
var i,min:integer;
begin
    dispose(d);
    dispose(c);
    dispose(tr);
    dispose(t);
    min:=1;
    for i:=2 to n do
    if a[i]
    assign(f,fou); rewrite(f);
        writeln(f,min);
        writeln(f,a[min]);
        close(f);
end;
{ *-----* }
procedure xuli;
var i,j,dem,bac: word;
    x,y: word;
begin
    fillchar(a,sizeof(a),0);
    for i:=1 to n do t^[i]:=1;
    dem:=0;
    while dem
    begin
        for i:=1 to n-1 do
        begin
            x:=d^[i];
            y:=c^[i];
            if (a[x]<>-1) and (a[y]<>-1) then
            begin
                inc(a[x]);
                tr^[x]:=y;
                inc(a[y]);
                tr^[y]:=x;
            end;
        end;
        for i:=1 to n do
        if a[i]=1 then
        begin
            a[i]:=-1;
            inc(t^[tr^[i]],t^[i]);
            inc(dem);
        end;
    end;
end;

```

```

    end
    else
        if a[i]<>-1 then a[i]:=0;
    end;
    for i:=1 to n do
    begin
        if a[i]=0 then tr^[i]:=0;
        a[i]:=n-t^[i];
    end;
    for i:=1 to n do
        if t^[i]>a[tr^[i]] then a[tr^[i]]:=t^[i];
    end;
    {*-----*}
BEGIN
    nhap;
    xuli;
    inkq;
END.

```

Đây là đáp án cho 10 bộ testcase (download tại địa chỉ <http://www.ioi2003.com>):

Testcase.1	Testcase.2	Testcase.3	Testcase.4	Testcase.5
3	82	305	315	209
3	35	499	673	1627
Testcase.6	Testcase.7	Testcase.8	Testcase.9	Testcase.10
1033	1381	2856	2927	9246
2600	3746	4266	5325	9438

\* **Lưu ý khi sử dụng bộ testcase:** để chương trình chạy đúng, các bạn phải xóa số đầu tiên trong mỗi testcase, đó là số chỉ thứ tự bộ testcase.

### ***Giải bài 174/2004 – Cân cặp***

Bài toán được giải một cách đơn giản như sau: Giả sử cái kim trên cân bị lệch đi x kg. Khi đó theo đề bài ta có  $(2+x) + (3+x) = 6+x$ , do đó  $x=1$ ; Nghĩa là 1 cái cặp nặng 4 kg, một cái cặp nặng 3 kg.

### ***Giải bài 175/2004 – Trạm Radar***

#### ***Thuật toán:***

Với tọa độ (x,y) của mỗi đảo và bán kính bao phủ R của Radar, ta có thể tìm được đoạn [a,b] là hoành độ đầu cuối trên bờ biển, radar đặt ở bất kì điểm nào trên đoạn đó đều có thể "theo dõi" được đảo.

Trong trường hợp  $y > r$  – tức là đảo đó không thể theo dõi. Đây là trường hợp duy nhất phải xuất -1 ở file kết quả (vô nghiệm!).

Vậy mỗi đảo lúc này được biểu diễn bởi một đoạn [a,b] để tiện xử lý. Bài toán trở thành: "Cho n đoạn thẳng có tọa độ đầu và cuối, song song với nhau, hỏi cần ít nhất bao nhiêu đường thẳng vuông góc với n đoạn thẳng trên và cắt hết n đoạn thẳng trên." (*Phát biểu của bạn Cao Minh Anh*).

#### ***Giải bài toán bằng cách:***

– Sắp xếp các đoạn tăng theo hoành độ đầu. Nếu cùng hoành độ, ưu tiên những đảo có tung độ nhỏ hơn.

– Để tìm số Radar ít nhất, ta dùng thuật toán "tham lam" – thuật toán rất hiệu quả và

chính xác trong trường hợp này (độc giả có thể dễ dàng tự chứng minh!): Một radar sẽ bao phủ những đoạn liên tiếp nhau trong dãy đã sắp xếp.

Trong khi giải bài toán, các bạn cần để ý rằng yêu cầu bài toán chỉ là số lượng Radar cần thiết để theo dõi được tất cả n đảo, vì thế tọa độ chính xác để đặt từng Radar là không cần thiết. Hầu hết bài giải của các bạn đều cố gắng tìm số lượng Radar dựa trên việc xác định chính xác vị trí của chúng – điều này là hoàn toàn không cần thiết!

**Toàn bộ văn bản chương trình:**

```
const fi='radar.inp';
      go='radar.out';
      maxn=1000;

Type Square=record
      x,y:real;
end;

var L :array[1..maxn] of square;
    b :array[1..maxn] of boolean;
    T :square;
    n,dem,d :longint;
    find,thoat :boolean;
    f,g :text;
{*_____*}
procedure openf;
begin
    assign(f,fi);
    reset(f);
    assign(g,go);
    rewrite(g);
end;
{*_____*}
procedure Closef;
begin
    close(f);
    close(g);
end;
{*_____*}
procedure Swap(var A,B:square);
var T:square;
begin
    T:=A;
    A:=B;
    B:=T;
end;
{*_____*}
procedure Solve;
var i,j:longint;
begin
    for i:=1 to n-1 do
```

```

    for j:=i+1 to n do
        begin
            if L[i].x>L[j].x then swap(L[i],L[j]);
            if (L[i].x=L[j].x) and (L[i].y>L[j].y) then swap(L[i],L[j]);
        end;
    end;
{*_____*}
procedure Giao(var Q,P:square);
    var i:longint;
    begin
        if Q.y
        else
            begin
                Q.x:=P.x;
                if Q.y>P.y then Q.y:=P.y;
            end;
        end;
    end;
{*_____*}
procedure Xuli;
    var i,j:longint;
    begin
        dem:=0;
        fillchar(b,sizeof(b),1);
        for i:=1 to n do
            if b[i] then
                begin
                    inc(dem); T := L[i];
                    j := i;
                    find := true;
                    while find do
                        begin
                            j := j+1;
                            if j>n then exit;
                            giao(T,L[j]);
                            if find then b[j] := false;
                        end;
                    end;
                end;
        end;
    end;
{*_____*}
procedure Print;
    begin
        writeln(g,dem);
    end;
{*_____*}
procedure Process;
    begin
        Solve;
        Xuli;
    end;

```

```

    Print;
end;
{ * _____ * }
procedure Main;
var i,x,y :longint;
    k :real;
begin
    while not eof(f) do
        begin
            readln(f,n,d);
            if n=0 then exit;
            fillchar(L,sizeof(L),0);
            thoat:=false;
            for i:=1 to n do
                begin
                    readln(f,x,y);
                    K:=sqr(d)-sqr(y);
                    if k<0 then thoat:=true else k:=sqrt(k);
                    L[i].x:=x-k;
                    L[i].y:=x+k;
                end;
            if thoat then writeln(g,-1) else Process;
            readln(f);
        end;
    end;
{ * _____ * }
BEGIN
    Openf;
    Main;
    Closef;
END.

```

*(Lời giải của bạn Cao Minh Anh – có sửa chữa)*

### ***Giải bài 176/2004 – Đường đi của con rắn***

#### ***Giải thuật:***

Đây là bài toán đệ quy đơn giản với kích thước nhỏ. Để giải quyết câu A ta chỉ cần một vòng lặp cho con rắn đi theo quy tắc đã cho đến lúc dừng. Để giải quyết câu B ta dùng phương pháp duyệt toàn bộ, khi gặp tường hay chướng ngại ta cho con rắn thử rẽ cả 2 hướng trái và phải, cuối cùng ta thu được đường đi dài nhất.

Ta có thể kết hợp cả 2 câu A, B để giải quyết cùng một lúc. Để ý rằng nếu khi duyệt toàn bộ ta luôn cho con rắn rẽ trái thì lần duyệt đầu tiên kết thúc sẽ cho ta lời giải của câu A. Cách giải quyết này có nhược điểm là phải luôn kiểm tra tại mọi lần duyệt có phải là lần duyệt đầu tiên không điều đó gây ra sự lãng phí về tốc độ. Tuy nhiên số bước duyệt không lớn nên có thể dùng cách này thay cho việc làm 2 câu rời nhau và đây cũng là một ý tưởng hay, bạn *Cao Minh Anh* đã áp dụng ý tưởng này với một lời giải tốt. Dưới đây là lời giải của bạn:

```

uses crt;
const fi='Snail.inp';

```

```

        go='Snail.out';
        hx:array[1..4] of integer=(1,0,-1,0);
        hy:array[1..4] of integer=(0,1,0,-1);
        maxn=8;
type arr=array[0..maxn+1,0..maxn+1] of integer;
var a :arr;
    max,left,m,dem :integer;
    s :string;
    f,g :text;
{*_____*}
procedure Openf;
begin
    assign(f,fi); reset(f);
    assign(g,go); rewrite(g);
end;
{*_____*}
procedure Closef;
begin
    close(f);
    close(g);
end;
{*_____*}
procedure Input;
var i:integer;
begin
    readln(f,m);
    for i:=1 to m do
        begin
            readln(f,S);
            a[ord(S[2])-ord('0'),ord(S[1])-64]:=1;
        end;
    end;
{*_____*}
procedure Khoitao;
var i:integer;
begin
    for i:=0 to 9 do
        begin
            a[0,i]:=1; a[9,i]:=1;
        end;
    for i:=1 to 8 do
        begin
            a[i,0]:=1; a[i,9]:=1;
        end;
    end;
{*_____*}
procedure Getmax;
begin

```

```

    if max=0 then
        begin
            left:=dem; max :=dem;
        end
    else
        if dem>max then max:=dem;
    end;
{*_____*}
procedure Run(x,y,k:integer);
var k1,k2:integer;
begin
    a[x,y]:=2;
    dem:=dem+1;
    if a[x+hx[k],y+hy[k]]=0 then Run(x+hx[k],y+hy[k],k)
    else
        begin
            if a[x+hx[k],y+hy[k]]=2 then getmax
            else
                if a[x+hx[k],y+hy[k]]=1 then
                    begin
                        k1:=k+1;k2:=k-1;
                        if k1=5 then k1:=1;
                        if k2=0 then k2:=4;
                        if a[x+hx[k1],y+hy[k1]]=0 then Run(x+hx[k1],y+hy[k1],k1)
                        else Getmax;
                        if a[x+hx[k2],y+hy[k2]]=0 then Run(x+hx[k2],y+hy[k2],k2);
                    end;
                end;
            dem:=dem-1;
            a[x,y]:=0;
        end;
{*_____*}
procedure Print;
var i,j:integer;
begin
    writeln(g,left-1);
    writeln(g,max-1);
end;
{*_____*}
BEGIN
    openf;
    input;
    Khoitao;
    Run(1,1,1);
    Print;
    closef;
END.

```

**Giải bài 177/2004 – Đàn gia súc**

Gọi x,y,z là số trâu, bò, nghé tương ứng, theo điều kiện ta phải tìm nghiệm nguyên không âm của hệ phương trình:

$$\begin{cases} x+y+z=100 \\ 20x+100y+z=200 \end{cases} \Leftrightarrow \begin{cases} 19x+9y=100 \\ y=9+\frac{19(1-x)}{9} \end{cases}$$

Vì y là số nguyên không âm,  $1-x$  phải chia hết cho 9 và do đó  $1-x=9k$ , k là số nguyên. Nếu  $x=1$  thì  $y=9$  và  $z=90$

Nếu  $x>1$  thì  $1-x$  chia hết cho 9, x phải bằng 10, 19, ... nhưng khi đó  $19x>100$  và y sẽ là số âm.

Vậy trong đàn gia súc có: 1 con trâu, 9 con bò, 90 con nghé.

**Giải bài 178/2004 – Sự lựa chọn**

Đây là một bài toán rất đơn giản dành cho các bạn học sinh THCS, tuy nhiên kỳ này có nhiều các bạn THPT tham gia giải nên lời giải của các bạn đều tốt. Đối với các bạn học sinh THCS thì bài toán chỉ hơi phức tạp một chút, còn đối với các bạn THPT thì không có gì là khó khăn cả, nhưng vẫn có các bạn có lời giải chưa tốt. Dù sao Đề ra kỳ này vẫn khuyến khích các bạn tham gia giải bài, đặc biệt khuyến khích các bạn học sinh THCS giải bài THPT!

**Tư tưởng:**

Ta thực hiện lặp việc loại những người "không may mắn" ra khỏi vòng, mỗi bước lặp có thể loại 1 hoặc 2 người đến khi còn lại 1 hoặc 2 người "may mắn". Tại mỗi bước ta phải làm các công việc:

- Tìm ra vị trí của người bị loại nhờ việc đếm. Để ý rằng nếu vòng chơi còn N người thì đếm tới vị trí m nghĩa là người thứ  $m \bmod N$  bị loại tại vòng hiện tại.
- Loại 1 hoặc 2 người đó ra khỏi vòng, tuy nhiên chúng ta cần lưu lại chỉ số của họ lúc đầu tiên để ghi ra file kết quả.
- Giảm số người đi.
- Lặp lại công việc đến khi kết thúc.

**Chương trình:**

Cùng ý tưởng trên các bạn đều có những chương trình chạy tốt, đúng kết quả. Dưới đây là chương trình gọn gàng và tốt nhất của bạn Đỗ Minh Phương:

```
Program Lua_Chon_Seagame;
Const Max = 100;
      Fi = 'CHON.INP';
      Fo = 'CHON.OUT';
Var A : Array[1..Max] of Boolean;
      N,m,k : Byte;
      p,q,c : Longint;
      F,G : Text;
{*_____*}
Procedure Enter;
Begin
  Assign(F,Fi);
  Reset(F);
  Read(F,N,m,k,p,q);
  Close(F);
End;
```



```

Function P1(i : Byte) : Byte;
  Var j : Longint;
  Begin
    j:=0; i:=i-1;
    Repeat
      inc(i);
      if i=N+1 then i:=1;
      if A[i] then inc(j);
    Until j=p;
    P1:=i;
  End;
  {*_____ *}

```

```

Function P2(i : Byte) : Byte;
  Var j : Longint;
  Begin
    j:=0; i:=i+1;
    Repeat
      dec(i);
      if i=0 then i:=N;
      if A[i] then inc(j);
    Until j=q;
    P2:=i;
  End;
  {*_____ *}

```

```

Procedure Select;
  Var people : Byte; i,j : Byte;
  Begin
    people:=N; c:=0;
    Repeat
      inc(c);
      i:=P1(m);
      j:=P2(k);
      if i<>j then
        Begin
          Writeln(F,i,' ',j);
          A[i]:=False;
          A[j]:=False;
          people:=people-2;
        End
      Else
        Begin
          Writeln(F,i);
          A[i]:=False;
          dec(people);
        End;
      if i
        if j>1 then k:=j-1 else k:=N;
    Until people=0;

```

```

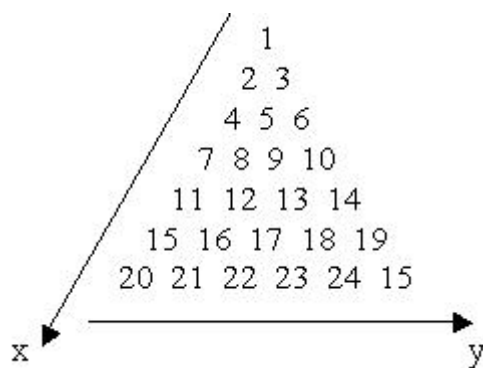
End;
{*_____*}
Procedure Run;
Begin
  Enter;
  Fillchar(A,Sizeof(A),True);
  Assign(F,'Temp.out');
  Rewrite(F);
  Select;
  Close(F);
  Assign(F,'Temp.out'); Reset(F);
  Assign(G,Fo); Rewrite(G);
  Writeln(G,c);
  For p:=1 to c do
    Begin
      While not seekeoln(F) do
        Begin
          Read(F,m);
          Write(G,m,'');
        End;
      Writeln(G);
      Readln(F);
    End;
  Close(F); Close(G);
End;
{*_____*}
BEGIN
  Run;
END.

```

### ***Giải bài 179/2004 – Toạ độ tam giác***

Để giải bài toán thuận tiện, chúng ta chuyển từ "toạ độ tam giác" sang hệ toạ độ (x,y) như hình vẽ, x là số dòng trong toạ độ tam giác, y là số thứ tự tính từ trái sang phải của điểm trên dòng đó. Ví dụ điểm 1 tương đương với toạ độ (1,1), điểm 9 tương đương với toạ độ (4,3)...

*Cách xây dựng:*



Xét một điểm A trong toạ độ tam giác. Giả sử toạ độ của A trong hệ toạ độ mới là (x,y).

Dễ dàng chứng minh được công thức sau:  $x*(x-1)/2 + y = A$

Bài toán ở đây là tìm (x,y) từ A.

Có nhiều phương pháp, nhưng ta chọn phương pháp đơn giản bằng nhận xét sau: Do

$y \leq x$  nên  $\text{sqrt}(2 \cdot A)$  chỉ có thể là  $x$  hoặc  $x-1$ . Ta thử chọn hai trường hợp của  $x$  là  $\text{sqrt}(2 \cdot A)$  và  $\text{sqrt}(2 \cdot A)+1$ .

Phương pháp giải này được mô tả chi tiết trong thủ tục **Chuyen**. Từ tọa độ  $(x, y)$  ta dễ dàng xác định được một cạnh của đa giác song song với cạnh của tọa độ tam giác khi hai điểm đầu mút của cạnh có cùng tọa độ  $x$  hoặc tọa độ  $y$  hoặc hiệu tọa độ  $(x-y)$ .

Chiều dài của một cạnh song song với trục  $x$  bằng hiệu tọa độ  $y$ , chiều dài của cạnh song song với hai cạnh bên của tam giác bằng hiệu tọa độ  $x$ .

Dựa vào những nhận xét trên, ta có chi tiết văn bản chương trình sau:

```

program ToaDoTamGiac;
const fin='VERTICES.INP';
      fou='VERTICES.OUT';
type td = record
      x,y: longint;
end;

var h: array[0..7] of td;
    r: array[0..7] of longint;
    n, vt, dai: integer;
    fi, fo: text;
{*_____*}
procedure moFile;
begin
  assign(fi,fin); reset(fi);
  assign(fo,fou); rewrite(fo);
end;
{*_____*}
procedure dongFile;
begin
  close(fi);
  close(fo);
end;
{*_____*}
procedure InKetQua;
begin
  writeln(fo,n);
end;
{*_____*}
function KhoangCach(a,b:td): integer;
begin
  with a do
    if x <> b.x then KhoangCach:=abs(x-b.x)
    else KhoangCach:=abs(y-b.y);
end;
{*_____*}
function SongSong(a,b: td): Boolean;
begin

```

```

    SongSong:=(a.x=b.x) or (a.y=b.y) or ((a.x-a.y)=(b.x-b.y));
end;
{*****}
function Trung: boolean;
var i,j: integer;
begin
    Trung:=true;
    for i:=1 to n-1 do
        for j:=i+1 to n do
            if (h[i].x=h[j].x) and (h[i].y=h[j].y) then break;
        Trung:=false;
    end;
{*****}
procedure Chuyen(sou: longint; var des: td);
begin
    with des do
        begin
            x:=trunc(sqrt(sou*2));
            y:=sou-(x*(x-1) div 2);
            if y>x then
                begin
                    inc(x);
                    y:=sou-(x*(x-1) div 2);
                end;
        end;
end;
{*****}
procedure Chuan;
var i, min: integer;
begin
    for i:=1 to n do Chuyen(r[i],h[i]);
    min:=1;
    for i:=2 to n do
        if r[i]
        vt:=0;
        dai:=0;
        for i:=1 to n do
            with h[min] do
                if (n=4) or (n=3) then
                    begin
                        if (i<>min) and ((h[i].x>x) and ((h[i].y=y) or (h[i].x-h[i].y=x-y))) then
                            begin
                                vt:=i;
                                dai:=h[i].x-x;
                                break;
                            end;
                    end
                end
            else

```

```

        if (i<>min) and ((h[i].x>x) and (h[i].y=y)) then
            begin
                vt:=i;
                dai:=h[i].x-x;
                break;
            end;
        end;
    {*_____ *}
procedure XetTiep;
var i, j, dem: integer;
begin
    Chuan;
    if Trung then
        begin
            n:=0;
            exit;
        end;
    for i:=1 to n do
        begin
            dem:=0;
            for j:=1 to n do
                if i<>j then
                    if SongSong(h[i],h[j]) and (KhoangCach(h[i],h[j])=dai) then
                        inc(dem);
                if dem<2 then
                    begin
                        n:=0;
                        exit;
                    end;
            end;
        end;
    {*_____ *}
procedure XuLy;
begin
    while not EOF(fi) do
        begin
            n:=0;
            while not EOLn(fi) do
                begin
                    inc(n);
                    read(fi,r[n]);
                end;
            readln(fi);
            case n of
                3,4,6: XetTiep;
                else n:=0;
            end;
            InKetQua;
        end;
end;

```

```

        end;
    end;
    {*-----*}
BEGIN
    moFile;
    XuLy;
    dongFile;
END.

```

### **Giải bài 180/2004 – Gard**

*Tư tưởng thuật toán:*

Thực chất đây là một bài toán quy hoạch động. Ta gọi  $L[i]$  là số cách bố trí khi có  $i$  người. Đầu tiên ta sẽ xếp vào tầng thứ nhất, giả sử ta bố trí ở đây  $T$  người, rõ ràng  $T=1..Min(i,k)$ , khi xếp  $T$  người ở tầng 1 sẽ còn  $(i-t)$  người và sẽ có  $L[i-t]$  cách bố trí  $(i-t)$  người còn lại vào các tầng tiếp theo. Mặt khác ta có  $C_i^t$  cách chọn ra  $T$  người ở tầng 1.

Như vậy  $L[i] = C_i^u * L[i-u]$ ; trong đó  $u=1..t$  ( $t=\min(i,k)$ );  **$L[i] = S(C_i^u * L[i-u])$  trong đó  $u = 1..t$**

Sau khi xây dựng xong mảng  $L$ , kết quả bài toán là  $L[n]$ . Vấn đề ở chỗ giả thiết cho  $K < N < 50$ , nên kết quả thu được có thể sẽ rất lớn, vì vậy ta sẽ dùng bài toán với số lớn để giải quyết. Ta làm chương trình với phép cộng trừ số lớn (xâu).

*Chương trình mẫu:*

```

const  fi = 'grad.inp';
        fo = 'grad.out';
        max = 50;
        ch:array [0.. 9] of char = ('0','1','2','3','4','5','6','7','8','9');
        nu:array ['0'..'9'] of byte = ( 0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 );
type num = String;
var n,k : byte;
    Cnk : array [1..max] of num;
    L : array [0..max] of num;
    f : text;
    {*-----*thnt*-----*}
procedure readf;
begin
    assign(f,fi); reset(f);
    readln(f,N,K);
    close(f);
end;
    {*-----*thnt*-----*}
procedure Tang(var n1:num;n2:Num);
var l,i,nho,l1,l2:Byte;
begin
    {them 0 de hai chuoai cung do dai}
    l1:=length(n1);
    l2:=length(n2);
    if l1 < l2 then l:=l2 else l:=l1;
    for i:=l1+1 to l do n1:=ch[0]+n1;
    for i:=l2+1 to l do n2:=ch[0]+n2;

```

```

    {cong chuoi n2 vao chuoi n1}
    nho:=0;
    for i:=l downto 1 do
        begin
            nho := nho + nu[n1[i]] + nu[n2[i]];
            n1[i]:= ch[nho mod 10];
            nho := nho div 10;
        end;
    if nho > 0 then n1:=ch[nho] + n1;
end;
{*-----*thnt*-----*}
Function Nhan(n1,n2:num):Num;
var n,nn : num;
    i,j,l,l1,l2,nho,x: byte;
begin
    n:=ch[0]; l1:=length(n1); l2:=length(n2);
    for i:=l2 downto 1 do
        begin
            {nhan n1 voi chu so thu i}
            x:=nu[n2[i]]; nn:=''; nho:=0;
            for j:=l1 downto 1 do
                begin
                    nho:=nho + nu[n1[j]]*x;
                    nn :=ch[nho mod 10] + nn;
                    nho:=nho div 10;
                end;
            if nho > 0 then nn:=ch[nho] + nn;
            {them 0 de hai chuoi cung do dai}
            x:=l2-i;
            if Length(n) < Length(nn)+x then l:=length(nn)+x
            else l:=length(n);
            for j:=length( n) +1 to l do n :=ch[0] + n;
            for j:=length(nn)+x+1 to l do nn:=ch[0] + nn;
            {cong them chuoi nn vao ket qua}
            nho:=0;
            for j:=l-x downto 1 do
                begin
                    nho :=nho + nu[n[j]] + nu[nn[j]];
                    n[j]:=ch[nho mod 10];
                    nho :=nho div 10;
                end;
            if nho > 0 then n:=ch[nho] + n;
        end;
    Nhan:=n;
end;
{*-----*thnt*-----*}
procedure QHD;
var i,j,jj : Byte;

```

```

    sol,so2 : num;
begin
    for i:=1 to K do Cnk[i]:=ch[0];
    L[0]:=ch[1];
    for i:=1 to N do L[i]:=ch[0];
    for i:=1 to N do
        begin
            if k < i then jj:=k else jj:=i;
            sol:=ch[1];
            for j:=1 to jj do
                begin
                    {Tinh to hop chap j cua i}
                    so2:=cnk[j];
                    Tang(cnk[j],sol);
                    sol:=so2;
                    Tang(L[i],Nhan(Cnk[j],L[i-j]) );
                end;
            end;
        end;
    end;
    {*_____*thnt*_____*}
procedure writef;
var f : text;
begin
    assign(f,fo); rewrite(f); write(f,L[n]); close(f);
end;
{*_____*thnt*_____*}
BEGIN
    readf;
    QHD;
    writef;
END.

```

(Lời giải của bạn *Lê Đình Thuận* – Lớp 11 Trường PTTH Lê Quý Đôn – Tp Quy Nhơn – Bình Định)

### ***Giải bài 181/2004 – Những quả trứng trong giỏ***

Gọi số trứng trong giỏ màu đỏ là  $n$  quả. Khi đó trứng trong trong giỏ màu nâu là  $n+1$ . trong giỏ màu hồng là  $n+3$ .

Vì tổng số trứng trong ba giỏ là 10 nên ta có:  $n + (n+1) + (n+3) = 10$ ,  $3n + 4 = 10$ ,  $3n = 6$ ,  $n = 2$

Vậy trong giỏ màu nâu có 3 quả trứng, giỏ màu đỏ 2 quả, giỏ màu hồng 5 quả.

### ***Giải bài 182/2004 – Dãy số hạnh phúc***

Đây là một bài toán khá cơ bản, theo định nghĩa dãy hạnh phúc là một dãy thoả mãn 2 điều kiện:

+ Là một dãy giảm

+ Các phần tử là số nguyên tố hoặc là ước của một trong các số trước nó.

Phần tử đầu tiên của dãy là  $N$ , ta kiểm tra các phần tử  $i$ :  $n-1$  đến 1

Vì  $N$  là số lớn nhất của dãy và  $N$  có thể không phải là một số nguyên tố. Nên các phần tử



của hạnh phúc nếu không phải là số nguyên tố thì nó phải là ước của N.

Vì vậy với  $i = n - 1$  đến 1 đầu tiên ta kiểm tra nếu  $i$  là ước của N hoặc  $i$  là số nguyên tố thì nó sẽ là một phần tử của dãy.

Để chương trình chạy tốt ta cần phải viết tốt hàm kiểm tra một số có phải là số nguyên tố không. Đây là một cách khá hay:

```
function So_Nguyen_To(so:longint):boolean;
var i:longint;
begin
  So_Nguyen_To:=False;
  for i:=2 to TRUNC(SQRT(so)) do
    if so mod i = 0 then exit;
  So_Nguyen_To:=True;
end;
```

*Dưới đây là toàn bộ chương trình:*

```
uses crt;
var i,n,dem:longint;
{*_____*}
function So_Nguyen_To(so:longint):boolean;
var i:longint;
begin
  So_Nguyen_To:=False;
  for i:=2 to TRUNC(SQRT(so)) do
    if so mod i = 0 then exit;
  So_Nguyen_To:=True;
end;
{*_____*}
BEGIN
  clrscr;
  write('Nhập vào một số n: '); readln(n);
  dem := 0;
  for i:=n downto 1 do
    if n mod i = 0 then
      begin
        write(i:8); inc(dem);
        if dem mod 10 = 0 then writeln;
      end
    else
      if So_Nguyen_To(i) then
        begin
          write(i:8); inc(dem);
          if dem mod 10 = 0 then writeln;
        end;
      writeln;
      writeln('Đã hạnh phúc có ',dem,' phần tử');
      readln;
  END.
```

**Giải bài 183/2004 – Catalan**

Theo định nghĩa, dãy Catalan là dãy:

- $A_i$  nguyên, không âm.
- $A_0 = A_{2N} = 0$ .
- $|A_i - A_{i+1}| = 1$

Để cho tường minh, ta xét với trường hợp cụ thể  $N=3$ . Có mảng số sau:

0			
0	1		
0	1	2	
0	1	2	3
0	1	2	
0	1		
0			

Nhận thấy rằng ta sẽ thu được một dãy Catalan nếu "di chuyển" trên mảng số theo quy tắc sau:

- Bắt đầu từ số 0 của dòng 1
- Không được di chuyển ra ngoài mảng
- Mỗi bước di chuyển xuống một dòng, đến ô sai khác với ô vừa rời đi đúng một đơn vị

0			
0	1		
0	1	2	
0	1	2	3
0	1	2	
0	1		
0			

0			
0	1		
0	1	2	
0	1	2	3
0	1	2	
0	1		
0			

0			
0	1		
0	1	2	
0	1	2	3
0	1	2	
0	1		
0			

0			
0	1		
0	1	2	
0	1	2	3
0	1	2	
0	1		
0			

0			
0	1		
0	1	2	
0	1	2	3
0	1	2	
0	1		
0			

0 1 0 1 0 1 0

0 1 0 1 2 1 0

0 1 2 1 0 1 0

0 1 2 1 2 1 0

0 1 2 3 2 1 0

Từ nhận xét trên, ta thấy bài toán trở nên quen thuộc: Tìm đường đi trên mảng theo quy tắc đã cho.

Để thực hiện hai yêu cầu của bài toán: Cho dãy tìm thứ tự và cho thứ tự tìm dãy, ta lập bảng quy hoạch động dựa theo đường đi như trên:

Gọi  $L[i,j]$  là số đường đi có thể đi khi đã đi đến dòng  $i$  cột  $j$  ( $0 \leq i; 0 \leq j$ ).

Vậy:

$$L[n*2,0]=1$$

$$L[i,j]=L[i+1,j-1]+L[i+1,j+1]$$

Dễ dàng thấy được  $L[0,0]$  chính là số lượng dãy Catalan cùng độ dài  $n$ .

Với  $n=3$  ta xây dựng được bảng  $L$  như sau:

5			
0	5		
2	0	3	
0	2	0	1
1	0	1	
0	1		
1			

1. Bài toán tìm dãy theo thứ tự từ điển  $tt$  cho trước:

Dùng quy nạp để xây dựng dãy:

Gọi dãy cần tìm là  $kq$ ;  $dau=1$ ;  $cuoi=L[0,0]$

Bước 0:  $kq[0]=0$ ; Thứ tự từ điển của dãy nằm trong khoảng từ  $dau$  đến  $cuoi$

Bước 1:  $kq[1]=1$ ; Thứ tự từ điển của dãy nằm trong khoảng từ *dau* đến *cuoi*

Bước 2:

+)  $kq[2]=kq[1]-1$  nếu  $tt \leq L[2,kq[1]-1]$  hay  $cuoi=L[2,kq[1]-1]$  ( $dau \leq tt \leq cuoi$ )

+)  $kq[2]=kq[1]+1$  nếu  $tt > L[2,kq[1]-1]$  hay  $dau=L[2,kq[1]-1]$  ( $dau \leq tt \leq cuoi$ )

...

Bước k:

+)  $kq[k]=kq[k-1]-1$  nếu  $tt \leq L[k,kq[k-1]-1]$  hay  $cuoi=L[k,kq[k-1]-1]$  ( $dau \leq tt \leq cuoi$ )

+)  $kq[k]=kq[k-1]+1$  nếu  $tt > L[k,kq[k-1]-1]$  hay  $dau=L[k,kq[k-1]-1]$  ( $dau \leq tt \leq cuoi$ )

...

Bước  $n*2$ :  $kq[n*2]=0 \rightarrow dau=tt=cuoi$

2. Bài toán tìm thứ tự từ điển của dãy  $d$  cho trước: Tìm thứ tự từ điển chính là tìm số lượng những dãy đứng trước dãy đã cho. Cũng với cách làm như trên, tại mỗi bước ta có hai lựa chọn: Nếu  $d[i] > d[i-1]$  thì tức là tại bước đó ta có thể xây dựng được  $L[i, d[i]-2]$  dãy Catalan có thứ tự nhỏ hơn dãy đã cho.

Tuy nhiên bài toán không chỉ dừng lại ở việc tìm ra quy luật của dãy số. Với dãy Catalan  $n=60$ , số lượng dãy là một số rất lớn (khoảng 35 chữ số). Như vậy bài toán không thể cài đặt với các số nguyên thông thường, và cũng không thể "phung phí" dữ liệu. Phương pháp giải quyết ở đây là sử dụng cách cài đặt "số khổng lồ" và dùng dữ liệu động. Để ý rằng không phải phần tử nào trong mảng  $L$  cũng được sử dụng, vì thế để tối ưu, ta chỉ khởi tạo những phần tử cần dùng đến.

*Toàn bộ văn bản chương trình:*

Program So\_Catalan;

Const fi = 'Catalan.Inp';

fo = 'Catalan.Out';

maxN = 60;

ch : array [ 0.. 9 ] of char = ('0','1','2','3','4','5','6','7','8','9');

nu : array ['0'..'9'] of byte = ( 0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 );

bl = #32;

mLs = 40;

type num = String[mLs];

Ar = Array [0..2\*maxN] of Byte;

var f,g : Text;

L : array [0..2\*maxN,-1..maxN+1] Of ^Num;

N,NN : Byte;

P : Pointer;

{\*\_\_\_\_\_\*

procedure init;

begin

assign(f,fi); reset(f);

assign(g,fo); rewrite(g);

Mark(P);

end;

{\*\_\_\_\_\_\*

{Kiểm tra n1 đã khởi tạo chưa.(Nếu chưa thì khởi tạo n1=0)}

Function KiểmTrần1:Num):boolean;

var i: integer;

begin

```

    KiemTra:=false;
    for i:=1 to length(n1) do
        if (n1[i]<'0') or (n1[i]>'9') then exit;
    KiemTra:=true;
end;
{*_____*}
Function Cong(n1,n2:Num):Num;
var
    l : Byte;
    i,nho : byte;
    S : Num;

Begin
    if not KiemTrăn1) then n1:='0';
    if not KiemTrăn2) then n2:='0';
    {them 0 de hai chuoi cung do dai}
    if Length(n1) < Length(n2) then l:=length(n2)
    else l:=length(n1);
    for i:=length(n1)+1 to l do n1:=ch[0]+n1;
    for i:=length(n2)+1 to l do n2:=ch[0]+n2;
    {cong chuoi n2 vao chuoi n1}
    S:='';
    nho:=0;
    for i:=l downto 1 do
        begin
            nho := nho + nu[n1[i]] + nu[n2[i]];
            S := ch[nho mod 10] + S;
            nho := nho div 10;
        end;
    if nho > 0 then S:=ch[nho] + S;
    Cong:=S;
end;
{*_____*}
Function tru(n1,n2:Num):Num; {voi n1 > n2}
var
    l,i : Byte;
    nho,x : byte;
    S : Num;

Begin
    if not KiemTrăn1) then n1:='0';
    if not KiemTrăn2) then n2:='0';
    {them 0 de hai chuoi cung do dai}
    l:=length(n1);
    for i:=length(n2)+1 to l do n2:=ch[0]+n2;
    {chuoi n1 tru chuoi n2}
    S := '';
    nho := 0;
    for i:=l downto 1 do
        begin
            x := nho + nu[n2[i]];
            if nu[n1[i]] < x then

```

```

begin
  x := nu[n1[i]] + 10 - x;
  nho := 1;
end
else
begin
  x := nu[n1[i]] - x;
  nho := 0;
end;
S := ch[x] + S;
end;
{Xoa cac ky tu 0 o dau}
While S[1] = Ch[0] do delete(s,1,1);
Tru:=S;
end;
{*_____*}
function SS(n1,n2:Num):byte; {so sanh n1 va n2 }
begin { ss = 0 —> n1 = n2}
  if length(n1) = length(n2) then { = 1 —> n1 > n2}
  if n1 = n2 then ss := 0 { = 2 —> n1 < n2}
  else
    if n1 > n2 then ss := 1
    else ss := 2
    else
      if length(n1) > length(n2) then ss := 1
      else ss := 2;
  end;
end;
{*_____*}
Procedure QHD;
Var i,j : integer;
Begin
  NN:=N*2;
  New(L[0, 0]); L[0, 0]^ := ch[1];
  For i:=1 to N Do
    begin
      j := i mod 2;
      repeat
        New(L[i,j]);
        L[i,j]^ := Cong(L[i-1,j-1]^ , L[i-1,j+1]^);
        inc(j,2);
      Until j>i;
    end;
  For i:=N+1 To NN Do
    Begin
      j := NN - i;
      repeat
        New(L[i,j]);
        L[i,j]^ := Cong(L[i-1,j-1]^ , L[i-1,j+1]^);

```

```

        dec(j,2);
        until j<0;
    End;
End;
{ *-----* }
Procedure findARR;
    var    i,j : Byte;
           K,m : Num;
           A : Ar;
    begin
        read(f,K);
        While K[1] = bl do delete(K,1,1);
        While K[1] = ch[0] do delete(K,1,1);
        While K[length(K)] = bl do delete(K,length(K),1);

        A[0] := 0; A[1] := 1;
        i := 2;
        While SS(K,ch[1]) = 1 do
            begin
                m := L[NN-i,A[i-1]-1]^;
                if SS(K,m) = 1 then
                    begin
                        K:=Tru(K,m);
                        A[i] := A[i-1] + 1;
                    end
                else
                    begin
                        A[i] := A[i-1] - 1;
                        if A[i] = 0 then
                            begin
                                inc(i);
                                A[i]:=1;
                            end;
                        end;
                    end;
                inc(i);
            end;
        if i <= nn then
            for j:=i to nn do
                if A[j-1] > 0 then A[j] := A[j-1] - 1
                else A[j] := 1;
            for i:=0 to nn do write(g,bl,A[i]);
            writeln(g);
        end;
    { *-----* }
Procedure FindNum;
    var    i,j : Byte;
           K : Num;
           A : Ar;

```

```

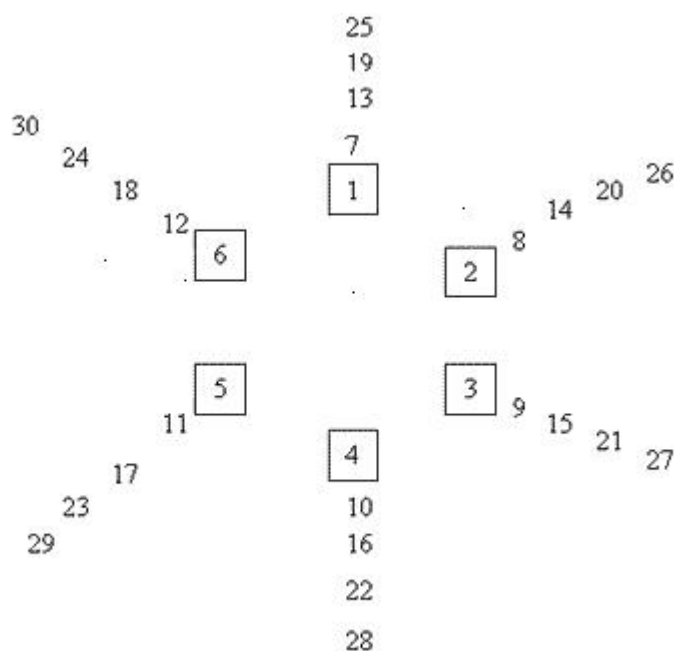
begin
  for i:=0 to nn do read(f,A[i]);
  K:=ch[1];
  for i:=2 to nn do
    if (A[i] > A[i-1]) and (A[i] > 1) then
      K := Cong(K, L[nn-i,A[i]-2]^);
  writeln(g,bl,K);
end;
{ *-----* }
Procedure Run;
var cs : Byte;
Begin
  readln(f,N);
  QHD;
  Writeln(g,L[Nn,0]^);
  repeat
    read(f,cs);
    if cs <> 0 then
      begin
        write(g,cs);
        if cs = 1 then findArr
        else findNum;
      end;
    readln(f);
  until cs = 0;
End;
{ *-----* }
procedure finish;
begin
  close(f);
  close(g);
  release(P);
end;
{ *-----* }
BEGIN
Init;
  Run;
  Finish;
END.
{Bài giải của bạn Lê Đình Thuận có sửa chữa}

```

### ***Giải bài 184/2004 – Các học sinh trong một vòng tròn***

Đây là một bài toán khá dễ, có rất nhiều cách giải khác nhau, sau đây là một cách giải các bạn có thể tham khảo:

Xếp từng học sinh vào vòng tròn thoả mãn điều kiện của bài toán. Sau khi xếp tất cả các học sinh, ta thu được biểu đồ sau:



Quan sát hình trên ta rút ra các kết quả:

- Sau mỗi chiếc ghế có 4 học sinh.
- Các học sinh mang số 9, 15, 21, 27 đứng sau học sinh mang số 3.

### ***Giải bài 185/2004 – Ông Ngâu bà Ngâu***

#### ***Tư tưởng thuật toán:***

*Chúng ta có một số nhận xét sau:*

- + Hai hành tinh bất kì chỉ được nối đến nhau bởi nhiều nhất một cầu vòng
- + Ông Ngâu và bà Ngâu luôn đi tới mục tiêu theo con đường ngắn nhất
- + Họ đi với vận tốc không đổi và nhanh hơn vận tốc ánh sáng

Thực chất đây là một bài toán đồ thị: Từ hành tinh S (nơi ông Ngâu ở) ta xây dựng bảng SP. Trong đó  $SP[i]$  là đường đi ngắn nhất từ hành tinh S đến hành tinh i (do ông Ngâu luôn đi tới mục tiêu theo con đường ngắn nhất).  $SP[i] = 0$  tức là không có đường đi từ hành tinh S đến hành tinh i. Tương tự ta sẽ xây dựng bảng TP, trong đó  $TP[i]$  là đường đi ngắn nhất từ hành tinh T đến hành tinh i. Và  $TP[i] = 0$  tức là không có đường đi từ hành tinh T đến hành tinh i.

Do yêu cầu của bài toán là tìm hành tinh khác S và T mà 2 ông bà Ngâu cùng đến một lúc và trong thời gian nhanh nhất. Tức là ta sẽ tìm hành tinh h sao cho (h khác S và T) và  $(SP[h] = ST[h])$  đạt giá trị nhỏ nhất khác 0. Nếu không có hành tinh h nào thỏa mãn thì ta thông báo CRY

Để xây dựng mảng SP và ST ta có rất nhiều giải thuật khác nhau. ở đây ta chọn giải thuật Dijkstra tìm đường đi ngắn nhất giữa 2 đỉnh đồ thị.

*Chương trình như sau:*

```
uses crt;
const MaxN = 101;
      fi= 'ONBANGAU.inp';
      fo= 'ONBANGAU.out';
var n,m,s,t,h:byte;
    a:array[0..MaxN,0..MaxN] of byte;
    SP,ST,B:array[0..MaxN] of integer;
    f:text;
{*_____*thnt*_____*
```



```

procedure Init;
  var i,u,v,ts:byte;
  begin
    fillchar(a,sizeof(a),0);
    assign(f,fi);
    reset(f);
    readln(f,n,m,s,t);
    for i:=1 to m do
      begin
        readln(f,u,v,ts);
        a[u,v]:=ts;
        a[v,u]:=ts;
      end;
    close(f);
  end;
{ *_____ *thnt* _____ *}
procedure Build(s:byte);
  var      tt:array[0..maxN] of byte;
          min,i,vtr:integer;
  begin
    fillchar(tt,sizeof(tt),0);
    fillchar(b,sizeof(b),0);
    for i:=1 to n do
      b[i] := a[s,i];
    tt[s]:=1;
    min:=0;
    while min <> maxint do
      begin
        min:=maxint; vtr:=0;
        for i:=1 to n do
          if tt[i] = 0 then
            if (b[i] <> 0) and (b[i]
              begin min:=b[i]; vtr:=i; end;
          if vtr <> 0 then tt[vtr]:=1;
          for i:=1 to n do
            if (tt[i] = 0) then
              if a[vtr,i] <> 0 then
                if (b[vtr] + a[vtr,i]
                  b[i]:=b[vtr] + a[vtr,i];
          end;
        end;
      end;
{ *_____ *thnt* _____ *}
procedure FindWay;
  var i:integer;
  begin
    build(s); {xay dung mang SP }
    SP:=B;
    build(t); {xay dung mang ST}

```

```

ST:=B;
h:= 0; {hanh tinh can tim}
sp[0]:= Maxint;
for i:=1 to n do
  if (SP[i] = ST[i]) then
    if (SP[i]<>0) then
      if (SP[i] < SP[h]) then
        h:=i;
end;
{*-----*thnt*-----*}
procedure ShowWay;
begin
  assign(f,fo);
  rewrite(f);
  if h <> 0 then writeln(f,h)
  else writeln(f,'CRY');
  close(f);
end;
{*-----*thnt*-----*}
BEGIN
  Init;
  FindWay;
  ShowWay;
END.

```

### **Bài 186/2004 – Đường đi trên đồng hồ cát**

#### ***Tư tưởng thuật toán:***

Đây là một bài toán dạng Quy hoạch động đã có sự biến đổi và không kém phần phức tạp. Vấn đề chính để giải quyết được bài toán chính là ở khâu giải thuật, tuy nhiên với kích thước khá lớn cũng yêu cầu các bạn phải chú ý đến khâu cài đặt. Phần lớn các bạn đều không tìm ra được giải thuật tối ưu nhất, các bạn thường đi vào Đề quy hay Duyệt toàn bộ – đó là một giải thuật chưa tối ưu với lớp các bài toán dạng này vì không thỏa mãn được về mặt kích thước. Để rõ hơn chúng ta cùng xem xét một số điểm xung quanh giải thuật của bài toán.

#### ***– Tại sao không thể Duyệt toàn bộ?***

Bài toán cho giới hạn  $2 \leq N \leq 20$ ,  $0 \leq S < 500$  có nghĩa là nếu duyệt toàn bộ, trong trường hợp xấu nhất tính một cách đơn giản chúng ta sẽ có hơn 2 mũ 20 đường duyệt. Cùng với các công việc in đường đi và tìm đường đi tối ưu, thuật toán sẽ tốn rất nhiều thời gian. Hơn nữa bộ nhớ của máy tính không thể đủ để cài đặt. Với giải thuật này chúng ta có thể làm đúng nhưng không tối ưu về mặt thời gian và kích thước yêu cầu của bài toán.

#### ***– Thuật giải Quy hoạch động là tốt nhất?***

Nếu để ý các bạn sẽ thấy tư tưởng Quy hoạch động trong bài toán này ở việc tích lũy các giá trị qua từng bước. Tuy nhiên đây không chỉ là bài toán tích lũy thông thường, chính là ở chỗ "Đồng hồ cát" – với một hình dạng không dễ dàng gì để cài đặt. Việc tìm đường đi được chia làm 2 nửa Đồng hồ cát là tốt nhất, các bạn đã nhận thấy điều này. Bài toán còn yêu cầu đưa ra kết quả đường đi có thứ tự từ điển nhỏ nhất (nếu có). Thuật toán QHĐ là tốt nhất – nhưng việc cài đặt nó ở đây không dễ dàng như "Bài toán cái Balô", "Chia kẹo",... Lý do nằm ở kích thước đã cho của bài toán. Sau đây là thuật giải của bài toán và những con

số để bạn thấy tại sao không dễ dàng cài đặt bình thường như các bài toán khác được.

– *Thuật giải:*

Gọi  $L[i,j,k]$  là số cách để đi từ hàng đầu đến ô  $[i,j]$ ,  $k$  là một trọng số nào đó, trong đó có thể tồn tại đường đi có trọng số bằng  $k$  hoặc không tồn tại. Nếu không tồn tại hiển nhiên:  $L[i,j,k] = 0$ .

Từ các ô trên cùng dễ thấy  $L[1,j,k] = 0$  nếu  $k < a[1,j]$ ,  $L[1,j,k] = 1$  nếu  $k = a[1,j]$ . Với các ô bên dưới trừ các ô biên, để xuống được nó có 2 ô ở trên để xuống, do đó số cách đến ô  $[i,j]$  bằng tổng số cách 2 ô trên, ta có:

$$L[i,j,k] = L[i-1,j,k-A[i,j]] + L[i-1,j+1,k-A[i,j]]$$

Theo bạn Lê Đình Thuận đánh giá: Mảng  $L$  là mảng 3 chiều và số cách tối đa khá lớn nên dùng mảng động cũng không đủ, cần khoảng  $(20 \times 21 - 1) \times 500 \times 11 = 2.304.500$  bytes, bạn đã đề xuất cách dùng xâu thay làm số lớn. Lại thêm một sự phức tạp của bài toán, nhưng đó là vấn đề cài đặt, chúng ta quay lại với giải thuật.

Ô ở giữa của Đồng hồ cát luôn được đi qua nên bài toán có tính đối xứng trong lúc duyệt. Do đó chúng ta có thể chia việc duyệt thành nửa trên và nửa dưới (các bạn có thể mở rộng bài toán với một Đồng hồ cát có nhiều ô hơn ở giữa!). Điều này đã làm cho bài toán trở nên dễ dàng hơn rất nhiều. Bạn Thuận còn có nhận xét việc duyệt một nửa như vậy làm cho mảng  $L$  không vượt quá 2 mũ 19 bytes.

Để đường đi có thứ tự từ điển nhỏ nhất (ưu tiên 'L' hơn 'R') khi duyệt các đường đi xuống chúng ta đi về phía trên trái trước. Còn khi duyệt các đường đi lên chúng ta làm ngược lại.

Với mảng  $L$  thu được cùng các mảng phụ khác chúng ta hoàn tất bài toán với việc tìm ra kết quả từ  $N$  kết quả bên dưới, nghĩa là trong các số  $L[N,j,S]$  hoặc không có kết quả nào.

### ***Toàn văn chương trình:***

Program Duong\_di\_tren\_dong\_ho\_cat;

Const     fi = 'Paths.Inp';

          fo = 'Paths.Out';

          maxN = 20;

          maxS = 500;

          LL = 'L';

          RR = 'R';

          Bl = #32;

          ch : Array [ 0.. 9 ] Of Char = ('0','1','2','3','4','5','6','7','8','9');

          nu : Array ['0'..'9'] Of Byte = ( 0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 );

Type   StringN = String[maxN];

          Num = String[16];

          arB = Array [0..maxS] Of Byte;

          arL = Array [0..maxS] Of Longint;

Var N,S : Integer;

      NN1,N1 : Integer;

      a : Array [1..maxN,0..maxN-1] Of Integer;

      M1,M2 : Array [1..maxN,0..maxN-1] Of Integer;

      Đ : Array [1..maxN,0..maxN-1] Of ^arB;

      L1,L2 : Array [0..maxN-1] Of ^arL;

      SL : Array [1..2] Of arL;

      Path : Array [1..2,0..maxS] Of StringN;

      P : Pointer;

      f,g : Text;

```

{*_____*}
Procedure Init;
  Begin
    Assign(f,fi);Reset(f);
    Assign(g,fo);ReWrite(g);
    Mark(P);
  End;
{*_____*}
Procedure Prepare;
  Var i,j : Integer;
  Begin
    For i:=N1 DownTo 0 Do
      Begin
        If L1[i] = Nil Then New(L1[i]);
        If L2[i] = nil Then New(L2[i]);
        Fillchar(L1[i]^,SizeOf(L1[i]^),0);
        For j:=0 To i Do
          Begin
            If Đ[N-i,j] = Nil Then New(Đ[N-i,j]);
            M1[N-i,j] := S;
          End;
        End;
        Fillchar(M2,SizeOf(M2),0);
      End;
    End;
  End;
{*_____*}
Procedure Readf1;
  Var i,j : Byte;
  Begin
    For i:=N1 DownTo 0 Do
      Begin
        For j:=0 To i Do Read(f,a[N-i,j]);
        Readln(f);
      End;
    End;
  End;
{*_____*}
procedure Readf2;
  Var i,j : Byte;
  Begin
    For i:=1 To N1 Do
      Begin
        For j:=0 To i Do Read(f,a[N-i,j]);
        Readln(f);
      End;
      a[N,0]:=0;
      Readln(f);
    End;
  End;
{*_____*}
Function Min(x,y:Integer):Integer;

```

```

    Begin
        If x
            else Min:=y;
        End;
    {*-----*}
Function Max(x,y:Integer):Integer;
    Begin
        If x
            else Max:=x;
        End;
    {*-----*}
procedure QHD;
var i,j,k : Integer;
Begin
    For j:=0 To N1 Do
        Begin
            L1[j]^a[1,j] := 1;
            M1[1,j] := a[1,j];
            M2[1,j] := a[1,j];
        End;
    For i:=2 To N Do
        For j:=0 To N-i Do
            Begin
                Fillchar(L2[j]^,SizeOf(L2[j]^),0);
                if a[i,j] <= S then
                    begin
                        M1[i,j] := Min(M1[i-1,j],M1[i-1,j+1]) + a[i,j];
                        M2[i,j] := Min(Max(M2[i-1,j],M2[i-1,j+1])+a[i,j] , S);
                        For k:=M1[i,j] To M2[i,j] Do
                            Begin
                                If L1[j]^k-a[i,j] > 0 Then Đ[i,j]^k := 1
                                Else Đ[i,j]^k := 0;
                                L2[j]^k := L1[j]^k-a[i,j] + L1[j+1]^k-a[i,j];
                            End;
                        end;
                        L1[j]^ := L2[j]^;
                    End;
            End;
        End;
    {*-----*}
Procedure FindPath(x : Byte);
var i,j,k,kk : Integer;
Begin
    SL[x]:=L2[0]^;
    For k:=M1[N,0] To M2[N,0] Do
        If SL[x,k] > 0 Then
            Begin
                Path[x,k] := '';
                j:= 0;
            End;
        End;
    End;
    {*-----*}

```

```

        kk := k;
        For i:=N DownTo 2 Do
            Begin
                kk := kk - a[i,j];
                If Đ[i,j]^[kk + a[i,j]] = 1 Then Path[x,k] := RR + Path[x,k]
                Else
                    Begin
                        Path[x,k] := LL + Path[x,k];
                        Inc(j);
                    End;
            End;
        End;
    End;
End;
{*_____*}
Procedure Ad(Var n1 : Num;n2 : Num);
    var l: Byte;
        i,nho : byte;
    Begin
        l := Max(Length(n1),Length(n2));
        For i:=Length(n1)+1 To l Do n1:=ch[0]+n1;
        For i:=Length(n2)+1 To l Do n2:=ch[0]+n2;
        nho := 0;
        For i:=l DownTo 1 Do
            Begin
                nho:= nho + nu[n1[i]] + nu[n2[i]];
                n1[i] := ch[nho Mod 10];
                nho:= nho Div 10;
            End;
            If nho > 0 Then n1:=ch[nho] + n1;
        End;
    End;
{*_____*}
Function Mutliply(n1,n2 : Longint) : Num;
    Var kq: String;
        nho,tich : Longint;
    Begin
        nho := 0;
        kq:= '';
        While n1 > 0 do
            begin
                tich := n2*(n1 mod 10) + nho;
                nho:= tich Div 10;
                kq:= ch[tich Mod 10] + kq;
                n1:= n1 div 10;
            end;
        While nho > 0 Do
            Begin
                kq:= ch[nho Mod 10] + kq;
                nho := nho Div 10;
            End;
        End;
    End;

```



```

        QHD;
        FindPath(2);
        Writef;
    End;
    Until (N=0) And (S=0);
End;
{*_____*}
Procedure Finish;
Begin
    Release(P);
    Close(f);
    Close(g);
End;
{*_____*}
BEGIN
    Init;
    Run;
    Finish;
END.

```

*(Chương trình của bạn Lê Đình Thuận – Lớp 11 Trường PTTH Lê Quý Đôn – Tp Quy Nhơn – Bình Định)*

Như các bạn thấy, về tư tưởng Quy hoạch động, Đề quy... ai cũng nắm được cả nhưng ở mỗi bài toán lại có một đặc điểm riêng. Bài toán này khá hay, các bạn hãy cố gắng suy nghĩ nhiều hơn đến thuật giải để tìm được lời giải đẹp nhất trong các kỳ sau.

### ***Giải bài 187/2004 – Gà và Thỏ***

Gọi số gà là x, số thỏ là y theo đầu bài ta có:

$$\begin{cases} x + y = 700 \\ 2x + 4y = 1800 \end{cases}$$

Từ đây ta có số thỏ là 200 con, số gà là 500 con

### ***Giải bài 188/2004 – Rút gọn***

Đây thực chất là một bài toán loang thuần túy, ta nhận thấy rằng số trạng thái tối đa của hình vuông H là  $4! = 24$  trạng thái. Như vậy xuất phát từ một trạng thái bất kì, không mất tổng quát ta giả sử là trạng thái S1 luôn là "1234". Với mỗi dãy biến đổi trong file AUT.INP ta thực hiện các thao tác để thu được trạng thái S2 tương ứng. Sau đó ta dùng thuật toán loang theo chiều rộng để tìm bước biến đổi ngắn nhất từ trạng thái S1 sang trạng thái S2. Do phải đưa ra xâu biến đổi có thứ tự từ điển thấp nhất trong các xâu thoả mãn đề bài nên ta sẽ ưu tiên phép biến đổi R trước phép biến đổi S. Để đánh dấu trạng thái của một hình vuông H ta quy định như sau:

$L[GT(h)] = 0$ :

Nếu chưa có trạng thái H=1:

Nếu có trạng thái H

Trong đó  $GT(H) = a[1,1]*64 + a[1,2]*16 + a[2,2]*4 + a[2,1]*1$

Bài giải cụ thể:

const fi='AUT.inp';

go='AUT.out';



```

type arr=array[1..2,1..2] of byte;
var Q :array[1..200] of arr;
    B,E,T :array[1..1000] of byte;
    a,u :arr;
    l,luu,last,first,kq :integer;
    s,y :string;
    thoat,stop,path :boolean;
    f,g :text;
{*-----*thnt*-----*}
procedure Openf;
begin
    assign(f,fi);
    reset(f);
    assign(g,go);
    rewrite(g);
end;
{*-----*thnt*-----*}
procedure Closef;
begin
    close(f);
    close(g);
end;
{*-----*thnt*-----*}
function Tiãvar s:string):string;
var k,r :integer;
    p :string;
begin
    repeat
        P:=S;
        k:=0;
        repeat
            inc(k);
            r:=k;
            while (S[k]=S[k+1]) and (k<=length(S)-1) do inc(k,1);
            case S[r] of
                'S':delete(S,r,((k-r+1) div 2)*2);
                'R':delete(S,r,((k-r+1) div 4)*4);
            end;
        until k>length(S);
        if S<>P then thoat:=false else thoat:=true;
    until thoat;
    Tia:=S;
end;
{*-----*thnt*-----*}
procedure Swap(var a,b:byte);
var tg:integer;
begin
    Tg:=a; a:=b; b:=Tg;

```

```

    end;
    {*_____*thnt*_____*}
    procedure Biendoi(var S:arr;k:integer);
    begin
        case k of
            1:begin
                swap(S[1,1],S[1,2]);
                Swap(S[1,1],S[2,1]);
                swap(S[2,2],S[2,1]);
                end;
            2:swap(S[1,1],S[1,2]);
        end;
    end;
    {*_____*thnt*_____*}
    procedure Run;
    var i,j,k:integer;
    begin
        j:=last;
        for i:=first to last do
            begin
                for k:=1 to 2 do
                    begin
                        u:=Q[i];
                        biendoi(u,k);
                        l:=u[1,1]*64+u[1,2]*16+u[2,2]*4+u[2,1]*1;
                        if (B[l]=0) or (l=kq) then
                            begin
                                inc(j);
                                Q[j]:=u;
                                E[j]:=k;
                                T[j]:=i;
                                B[l]:=1;
                            end;
                        if L=kq then begin path:=true;luu:=j;end;
                    end;
                end;
            end;
        first:=last+1;
        last:=j;
        if first>last then stop:=true;
    end;
    {*_____*thnt*_____*}
    procedure Empty;
    var i:integer;
    begin
        first:=1;
        last:=1;
        fillchar(b,sizeof(b),0);
        fillchar(e,sizeof(e),0);
    end;

```

```

    fillchar(t,sizeof(t),0);
    a[1,1]:=1;a[1,2]:=2;a[2,2]:=3;a[2,1]:=4;
    Q[1]:=a;
  end;
  {*-----*thnt*-----*}
  procedure Findtrangthaidich;
    var i:integer;
  begin
    for i:=1 to length(S) do biendo(a,ord(S[i])-81);
    kq:=a[1,1]*64+a[1,2]*16+a[2,2]*4+a[2,1]*1;
  end;
  {*-----*thnt*-----*}
  procedure Process;
  begin
    stop:=false;
    Path:=false;
    repeat
      Run;
    until Stop or Path;
  end;
  {*-----*thnt*-----*}
  procedure Print;
  begin
    Y:='';
    while luu<>1 do
      begin
        if E[luu]=1 then Y:='R'+Y else Y:='S'+Y;
        luu:=T[luu];
      end;
    writeln(g,Y);
  end;
  {*-----*thnt*-----*}
  procedure Main;
  begin
    while not eof(f) do
      begin
        readln(f,S);
        S:=TiãS);
        Empty;
        findtrangthaidich;
        process;
        if S<>' ' then Print else writeln(g);
      end;
    end;
  {*-----*thnt*-----*}
  BEGIN
    Openf;
    Main;

```

closef;

END.

(Lời giải của bạn Cao Minh Anh)

### **Giải bài 189/2004 – Khu đất**

#### **Giải thuật:**

Đây là một bài toán duyệt đơn giản nhưng có kích thước lớn vì vậy đòi hỏi các bạn cần có sự khéo léo trong lúc duyệt. Thay vì duyệt toàn bộ chúng ta cần phải có được các nhận xét chính xác trước khi bắt tay vào làm:

– Để mảnh đất hình vuông thu được có kích thước lớn nhất ta sẽ duyệt các mảnh đất hình vuông có kích thước từ lớn về nhỏ thay vì làm ngược lại, mặc dù việc này có vẻ phức tạp hơn.

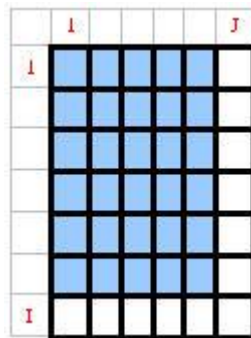
– Hơn nữa, khi chuyển từ hình vuông lớn về hình vuông nhỏ bằng cách loại bớt các lô đất xung quanh đi chúng ta có thể sử dụng nguyên lý của Quy hoạch động.

– Kích thước của bài toán là khá lớn và chúng ta phải dùng mảng động. Thuật toán mang một chút dấu ấn Quy hoạch động của chúng ta như sau:

– Tính giá trị của các mảnh đất vuông có kích thước lớn nhất, đương nhiên có kích thước  $M \times M$  nếu  $M < N$  hoặc  $N \times N$  nếu  $N < M$ . Chắc chắn ở bước này chúng ta phải cộng tất các lô đất trong mảnh đất lại.

– Tiếp theo chúng ta thu nhỏ mảnh đất lại bằng cách loại bỏ các ô xung quanh mảnh đất to này. Diện tích của nó sẽ tính bằng cách đem diện tích mảnh đất vuông to (đã tính ở bước trước) trừ đi phần bỏ đi như vậy sẽ nhanh hơn rất nhiều lần cộng từng lô đất nhỏ trong nó.

Hình vẽ minh họa:



Hình vuông tối màu là mảnh đất có "tọa độ"  $(1,1)$   $(i-1,j-1)$ , nếu mảnh đất vuông to  $(1,1)$   $(i,j)$  và hai mảnh đất hình chữ nhật có tọa độ  $(1,1)$   $(i-1,j)$  và  $(1,1)$   $(i,j-1)$  đã biết giá trị thì mảnh đất tối màu này dễ dàng tính được giá trị:

$$F[i-1,j-1] = F[i-1,j] + F[i,j-1] + A[i,j] - F[i,j]$$

Với  $F[i,j]$  cho giá trị của mảnh đất có tọa độ  $(1,1)$   $(i,j)$ ,  $A[i,j]$  là giá trị lô đất  $(i,j)$ . Từ đó ta có giá trị của mảnh đất hình vuông có đỉnh trên trái  $(i,j)$  kích thước  $k$  là (đây cũng là nhận xét chính xác của bạn Lê Đình Thuận):

$$S_{i,j,k} = L[i+k-1,j+k-1] - L[i-1,j+k-1] - L[i+k-1,j-1] + L[i-1,j-1]$$

Việc duyệt giá trị chính là kiểm tra giá trị của biểu thức  $S_{i,j,k}$  này xem có bằng  $T$  không, đến đây bài toán coi như đã được giải quyết xong. Các bạn tham khảo lời giải của bạn Lê Đình Thuận (có cả biến tính thời gian chạy chương trình – nhưng với cách làm như đã nêu, thời gian là rất nhỏ!):

Program Land;

Const fi = 'Land.Inp';

fo = 'Land.Out';

MN = 200;

VN = -1;

```

Type    ar = array [0..MN] of Longint;
Var     M,N : Byte;
        T : Longint;
        L : array [0..mn] of ^Ar;
        p : Pointer;
        Nhup : Longint absolute $0000:$046C;
        Time : Longint;
        f,g : Text;
{*-----*}
Procedure ChuanBi;
  Var i : Byte;
  Begin
    Assign(f,fi); reset(f);
    Assign(g,fo); rewrite(g);
    readln(f,M,N,T);
    mark(p);
    For i:=0 to M do
      Begin
        New(L[i]);
        Fillchar(L[i]^,Sizeof(L[i]^),0);
      end;
  End;
{*-----*}
Procedure Tong;
  Var   i,j : Byte;
        A : Longint;
  Begin
    for i:=1 to M do
      begin
        for j:=1 to N do
          begin
            Read(f,A);
            L[i]^j := L[i-1]^j + L[i]^j-1 + A - L[i-1]^j-1;
          end;
        readln(f);
      end;
  End;
{*-----*}

Function Min(x,y:byte):Byte;
  Begin
    if x < y then min:=x else min:=y;
  end;
{*-----*}

Procedure Tim;
  Var i,j,k : Byte;
  Begin
    for k:=Min(M,N) downto 1 do

```

```

        for i:=0 to N-k do
            for j:=0 to M-k do
                if  $L[i+k]^j + L[i]^j - L[i]^j - L[i+k]^j = T$  then
                    Begin
                        Write(g,k);
                        exit;
                    End;
            Write(g,VN);
        End;
    { * _____ * }
Procedure KetThuc;
Begin
    Release(p);
    close(f);
    Close(g);
end;
{ * _____ * }
BEGIN
    time:=Nhip;
    Chuanbi;
    Tong;
    Tim;
    Ketthuc;
    Writeln(' Thoi gian chay: ',(nhip-time)/18.2 :0:3,' s. ');
END.

```

### ***Giải bài 190/2004 – Những đứa trẻ trong gia đình John Smith***

Chúng ta ký hiệu như sau: Kate – K, Sally – S, Tom – T, Ben – B.

Dựa vào điều kiện của bài toán ta có các phép tính như sau:

$$T = B + 2 \quad (1)$$

$$K + S = T + B \quad (2)$$

$$K = 2S \quad (3)$$

$$T - 1 = 2(S - 1) \quad (4)$$

Phép tính (4) có thể được viết lại như sau:  $T - 1 = 2S - 2$  hay  $T = 2S - 1$  (5).

Thay K từ (3) vào (2) ta có:  $3S = T + B$  (6)

Bây giờ ta thay T từ (1) vào (6) khi đó ta có:  $3S = 2B + 2$  (7)

Tiếp tục thay T từ (5) vào (6) ta có:  $3S = 2S - 1 + B$ . Vì vậy  $B = S + 1$  (8)

Thay B từ (8) vào (7):  $3S = 2(S + 1) + 2$

$$3S = 2S + 4 \text{ và cuối cùng ta có } S = 4.$$

Và từ (8) ta có  $B = 5$ .

Từ (1) ta tính được  $T = 7$ .

Từ (3) ta suy ra  $K = 8$ .

Vậy tuổi của những đứa trẻ như sau:

Kate: 8 tuổi

Sally: 4 tuổi

Tom: 7 tuổi

Ben: 5 tuổi.

***Giải bài 191/2004 – Chia đôi***

Đây là một bài toán hình học khá thú vị, giải thuật của nó như sau:

+ Sắp xếp các điểm theo chiều tăng (giảm) của tung độ, nếu  $2*n$  điểm này có tung độ khác nhau thì ta chọn tung độ (ký hiệu  $yy$ ) có giá trị nằm giữa tung độ 2 điểm  $n$  và  $n+1$ , tức là  $yy = (y[n] + y[n+1])/2$ . Khi đó đường thẳng  $y = yy$  sẽ chia mặt phẳng thành 2 phần mỗi phần có  $n$  điểm khác nhau.

+ Trong trường hợp các điểm này có tung độ trùng nhau, giả sử ở mặt phẳng có  $y > yy$  đã có  $u$  điểm, mặt phẳng có  $y < yy$  có  $v$  điểm. Khi đó số điểm nằm trên đường thẳng  $y = yy$  sẽ là:  $(2*n - u - v)$  điểm. Ta lại sắp xếp các điểm này theo chiều tăng của  $x$ . Sau đó chọn  $n-u$  điểm có hoành độ nhỏ nhất trong số các điểm này, tức là chúng có hoành độ nhỏ hơn  $xx$ , trong đó  $xx = (x[ol[n-u]] + x[ol[n-1]]) / 2$ .  $ol[n-u]$  là điểm thứ  $n-u$  trong số  $(2*n-u-v)$  điểm nằm trên đường thẳng  $y = yy$ . Như vậy với  $u$  điểm có hoành độ  $y > yy$  và  $n-u$  điểm có hoành độ  $x < xx$  ta đã chọn được  $u + n - u = n$  điểm cho mặt phẳng thứ nhất và mặt phẳng kia sẽ gồm những điểm còn lại.

+ Để xác định đường thẳng  $ax + by + cy = 0$  ta gọi điểm  $A (xx,yy)$  làm gốc, ta thử các đường thẳng đi qua  $A$  xem nó có thoả mãn yêu cầu đề bài hay không. Mỗi lần thử ta xoay đường thẳng qua  $A$  này một góc rất nhỏ, ta sẽ thử cho đến khi đạt được yêu cầu của bài toán.

***Dưới đây là toàn bộ chương trình:***

```
Const   fi = 'Half.inp';
        fo = 'Half.out';
type    ii = integer;
        ll = longint;
        rr = real;
        ar = array [1..10000] of rr;
        aa = array [1..10000] of ii;
var     n, nn, k, ok: ii;
        a, b, c, xx, yy: rr;
        x, y: ^ar;
        ol: aa;

{*_____*}
procedure mo_file;
begin
  assign(input, fi);
  reset(input);
  assign(output, fo);
  rewrite(output);
end;
{*_____*}
procedure dong_file;
begin
  Write(a:0:6,' ',b:0:6,' ',c:0:6);
  close(input);
  close(output);
end;
{*_____*}
procedure doicho(var x,y: real);
var temp:real;
```

```

begin
    temp:=x;
    x:=y;
    y:=temp;
end;
{*****}
procedure nhap;
var i: ii;
begin
    readln(n);
    nn:= n*2;
    for i:= 1 to nn do readln(x^[i], y^[i]);
end;
{*****}
procedure qSort(l, r: Integer);
var i, j: integer;
    x1: rr;
begin
    i := l;
    j := r;
    x1 := y^[(l+r) DIV 2];
    repeat
        while y^[i] < x1 do inc(i);
        while x1 < y^[j] do dec(j);
        if i <= j then
            begin
                doicho(x^[i],x^[j]);
                doicho(y^[i],y^[j]);
                inc(i);
                dec(j);
            end;
        until i > j;
        if l < j then qSort(l, j);
        if i < r then qSort(i, r);
    end;
{*****}
procedure chuanbi;
var i, j, tg, hi, lo: ii;
begin
    hi:= 1;
    lo:= nn;
    qSort(hi, lo);
    yy:= (y^[n] + y^[n+1])/2;
    k:= 0;
end;
{*****}
procedure xacdin(x1, y1, x2, y2: rr);
begin

```



```

    a:= y1-y2;
    b:= x2-x1;
    c:= -(a*x1+b*y1)
end;
{ * _____ *}
procedure xuli;
  var dem1, dem2: ii;
      es, x1, y1: rr;
begin
  a:= 0;
  b:= -1;
  c:= yy;
  bangnhau(a, b, c);
  if ok=1 then exit;
  qsort2(1,k);
  x1:= x^[ol[n-dem2+1]];
  y1:= y^[ol[n-dem2+1]];
  xx:= (x^[ol[n - dem2]]+x1)/2;
  es:= 0.5;
  repeat
    es:= es/2;
    xacdinhh(xx, yy, x1+es, y1+es);
    bangnhau(a,b,c);
  until ok=1;
end;
{ * _____ *}
procedure bangnhau(a, b, c: rr);
  var tp: rr;
      i: ii;
begin
  k:= 0;
  dem1:= 0;
  dem2:= 0;
  ok:= 0;
  for i:= 1 to nn do
    begin
      tp:= a*x^[i]+b*y^[i]+c;
      if tp>0 then inc(dem1);
      if tp<0 then inc(dem2);
      if tp=0 then
        begin
          inc(k);
          ol[k]:= i;
        end;
    end;
  if dem1<>dem2 then exit;
  if dem1=n then ok:=1;
end;

```

```

{ * _____ *}
procedure qSort2(l, r: Integer);
  var i, j, y1: integer;
      x1: rr;
  begin
    i := l;
    j := r;
    x1 := x^[ol[(l+r) DIV 2]];
    repeat
      while x^[ol[i]] < x1 do inc(i);
      while x1 < x^[ol[j]] do dec(j);
      if i <= j then
        begin
          y1 := ol[i];
          ol[i] := ol[j];
          ol[j] := y1;
          inc(i);
          dec(j);
        end;
      until i > j;
      if l < j then qSort2(l, j);
      if i < r then qSort2(i, r);
    end;
  { * _____ *}
procedure init;
  begin
    new(x);
    new(y);
  end;
{ * _____ *}
procedure clear;
  begin
    dispose(x);
    dispose(y);
  end;
{ * _____ *}
BEGIN
  init;
  mo_file;
  nhap;
  chuanbi;
  xuli;
  clear;
  dong_file;

```

END.

(Bài giải của bạn Phi Hồng Thái – Lớp 12 A2 trường THPT Chuyên Vĩnh Phúc – Vĩnh Phúc)

**Bài 192/2004 – IP*****Những vấn đề xung quanh bài toán***

Đây sẽ thực sự là một bài toán rất đơn giản nếu kích thước của nó không quá lớn như đã cho. Mặc dù đề bài "hơi dài" nhưng nội dung của nó có thể phát biểu như sau: *Cho  $N$  số nguyên dương  $A[1..N]$  phân biệt. Tìm số nguyên dương bé nhất chưa có mặt trong dãy này.*

Với kích thước xếp hạng "kinh khủng" về độ lớn trong các bài toán tin học:  $0 \leq N \leq 10^8$ ,  $1 \leq A[i] \leq 10^9$ . Thật vậy, file dữ liệu vào tối đa sẽ có  $10^8$  số có 10 chữ số trên mỗi dòng nghĩa là có kích thước lên đến hàng GB. Một bạn nào đó có thể đã bị tràn ổ đĩa khi tạo file dữ liệu!

Việc xử lý số lượng số lớn như vậy khiến chương trình sẽ mất rất nhiều thời gian. Bạn chỉ cần thử một vòng lặp FOR từ 1 đến 1 tỉ để tăng một số thôi đã mất khá nhiều thời gian. Đề bài cho  $N$  tối đa là  $10^8$  có tính tham khảo với các bạn, với cách làm tốt nhất (nhiều bạn đã làm được) chương trình cũng sẽ mất khá nhiều thời gian, khi dịch chương trình ra file EXE chạy trên các máy có cấu hình cao nhất hiện nay thì thời gian tạm chấp nhận được. Nhìn chung kỳ này bạn nào chương trình chạy được với  $N=10^7$  là đạt yêu cầu – trừ một số vấn đề khác về cài đặt và thậm chí sai kết quả!

Để khắc phục vấn đề kích thước các bạn đã nghĩ ra rất nhiều cách mới lạ, độc đáo, như ghi nhiều file, chia đoạn, tính tổng, mảng BIT,... nhưng không phải cách nào mới lạ cũng hiệu quả – đôi khi là ngược lại – một số cách rất "tối ý", máy móc khiến chương trình không chạy được với cả các test nhỏ. Cuối cùng, cách tốt nhất lại dựa trên ý tưởng đơn giản nhất:

*Nếu chúng ta có được một mảng đánh dấu các số từ số bé nhất đến lớn nhất để mỗi khi đọc một số chúng ta đánh dấu nó lại thì công việc còn lại chỉ là duyệt lại mảng đánh dấu để lấy phân tử đầu tiên chưa được đánh dấu.*

Chẳng hạn nếu số bé nhất là 1, lớn nhất là 10000, chúng ta dùng mảng danh dau[1..10001] khởi tạo bằng 0. Khi đọc được số  $X$  ( $1 \leq X \leq 1000$ ) ta "đánh dấu"  $X$ : danh dau[X]:=1. Đọc hết dữ liệu đầu vào, duyệt lại mảng danh dau, phần tử đầu tiên trong mảng danh dau bằng 0 là phần tử cần tìm. Đây là thuật giải nhanh nhất cho bài toán.

Nhưng với  $N \leq 10^8$ , không thể đủ bộ nhớ để bạn đánh dấu kể cả bạn có dùng bộ nhớ động. Các bạn đã nhận ra điều này và không ai dùng bộ nhớ động. Các bạn lưu ý một lần nữa rằng bộ nhớ động với Turbo Pascal có thể sử dụng ở giới hạn an toàn ở 300K, cần thiết có thể lên 400K. Đây là lời khuyên của các giáo sư dạy các bạn ạ!

Có một cách để nâng cao kích thước và tốc độ xử lý cho mảng đó là sử dụng các "mảng BIT" – một khái niệm lạ nhưng lại rất chính xác, mỗi byte là 8 bit, coi mỗi bit là một phần tử, như vậy 1 byte có thể đánh dấu được cho tới 8 số! Ngay cùng số báo có đề bài có bài viết "Xử lý BIT" đã gợi ý cho bạn cách làm, các bạn tìm đọc lại. Chi tiết hơn về giải thuật xử lý BIT (cùng với các giải thuật độc đáo khác) bạn có thể tham khảo duy nhất tại cuốn sách nổi tiếng "Bản tàu trên biển" – TS Nguyễn Xuân Huy.

Đến đây không còn vấn đề gì cần phải bàn về mặt giải thuật của bài toán nữa. Chính vì chung ý tưởng mà các bạn dùng mảng BIT đều có chương trình khá giống nhau, kết quả thì y hệt như nhau tới phần trăm giây! Dưới đây là một trong các chương trình ngắn gọn và tốt nhất của bạn *Nguyễn Quốc Tú* – 11A2 – chuyên Vĩnh Phúc:

```
{A+,B-,D+,E+,F-,G-,I+,L+,N-,O-,P-,Q-,R+,S+,T-,V+,X+}
{$M 16384,0,655360}
program Ma_Ip;
const fi='ip.inp';
      fo='ip.out';
type mang=array[0..31249] of word;
```

```

var ip,n :longint;
    a :mang;
{*-----*}
procedure turnon(var x:word;y:byte);
begin
    x:=x or (1 shl y);
end;
{*-----*}
function getbit(x:word;y:byte):byte;
begin
    getbit:=(x shr y) and 1;
end;
{*-----*}
procedure main;
var s,i,min,max :longint;
    kx,j :byte;
    f :text;
begin
    for kx:=0 to 200 do
    begin
        fillchar(a,sizeof(a),0);
        turnon(a[0],0);
        min:=kx*500000;
        max:=(kx+1)*500000;
        assign(f,fi);reset(f);
        readln(f,n);
        for i:=1 to n do
            begin
                read(f,s);
                if (s>=min) and (s
                    begin
                        s:=s-min;
                        turnon(a[s shr 4],s and 15);
                    end;
            end;
        close(f);
        assign(f,fo);rewrite(f);
        for i:=0 to 31249 do
            if a[i]<>65535 then
                for j:=0 to 15 do
                    if getbit(a[i],j)=0 then
                        begin
                            ip:=min+i*16+j;
                            writeln(f,ip);
                            close(f);halt;
                        end;
            end;
        end;
        close(f);

```

```

    end;
    { * _____ * }
BEGIN
    main;
END.

```

### **Giải bài 193/2004 – Bộ cờ**

Ta có nhận xét đầu tiên là đây là một bài toán đếm. Để giải quyết bài toán đếm ta có rất nhiều phương pháp quen thuộc như duyệt tất cả các cấu hình, phương pháp quy hoạch động, phương pháp bù trừ... Vì thế, với mỗi bài toán cụ thể, một công việc rất quan trọng là phải phân lớp bài toán để lựa chọn được phương pháp, giải thuật tối ưu nhất.

Như chúng ta đều biết, phương pháp duyệt tất cả các cấu hình (thuật toán quy lui) là phương pháp vạn năng cho các bài toán đếm. Cũng chính vì "vạn năng" nên nó lại là phương pháp kém tối ưu nhất, và thường là giải pháp cuối cùng được suy xét tới. Những giải pháp có độ phức tạp thấp phải cần được quan tâm đầu tiên.

Cụ thể trong trường hợp bài toán "Bộ cờ", chúng ta hoàn toàn có thể sử dụng **phương pháp quy hoạch động** như sau:

**Gọi  $L[i,j]$  là số bộ cờ khác nhau có  $i$  màu và có trọng số là  $j$**

Giả sử quân cờ nhảy màu thứ  $i$  có ghi số là  $arN[t]$  trong  $N$  số cho trước. Khi đó  $L[i,j] := L[i,j] + L[i-1, j-arN[t]]$  với  $arN[t] \leq j$

Vậy số bộ cờ cần tìm là  $L[K,S]$  với giả thiết của đề bài  $K$  là số màu khác nhau,  $S$  là trọng số của bộ cờ.

Ngoài ra ta còn có thể áp dụng **phương pháp hàm sinh** để giải quyết bài toán đếm như sau:

Ta thấy tồn tại tương ứng 1-1 giữa một bộ cờ trọng số  $S$  với một nghiệm của phương trình:  $x_1 + x_2 + \dots + x_n = S$  thỏa mãn điều kiện  $x_i$  nhận một trong  $n$  giá trị nguyên không âm đã cho  $a_1, a_2, \dots, a_n$

Số các nghiệm này chính là hệ số  $x^S$  của đa thức  $(x^{a_1} + x^{a_2} + \dots + x^{a_n})^k$

Do đó để giải bài toán ta chỉ cần xử lý phép nhân đa thức, với lưu ý là ta không cần tính các hệ số có bậc lớn hơn  $s$  của bất cứ đa thức nào

Về cơ bản phương pháp hàm sinh là cách trình bày lý thuyết hoá – toán học hóa của phương pháp quy hoạch động (phương pháp vốn rất được ưu chuộng trong tin học).

Hai phương pháp trên đều có độ phức tạp nhỏ ( $O(n^3)$ )

Thêm một vấn đề cần quan tâm khi giải bài toán này là với giới hạn của đề bài thì liệu có thể chỉ sử dụng các kiểu dữ liệu chuẩn của PASCAL mà giải quyết được bài toán? Vấn đề này đã được trình bày ở nhiều số trước, ở đây chỉ xin được nhắc lại như một lưu ý với các bạn. ở đây có sự "bùng nổ" số cấu hình đếm được với các test khác nhau, vì thế các kiểu dữ liệu chuẩn của PASCAL không thể đáp ứng được, các bạn phải áp dụng các phép toán trên "số không lồ"

Sau đây là văn bản chương trình sử dụng thuật toán quy hoạch động (chương trình chỉ có mục đích mô phỏng giải thuật, không phải là chương trình hoàn chỉnh):

```

Const  fi = 'COMPLETE.INP';
        fo = 'COMPLETE.OUT';
        maxK = 25;
        maxN = 101;
        MaxS = 300;

Var  N,K : Byte;
      S : Integer;

```

```

    ArN : Array [1..maxN] of Longint;
    L : Array [1..maxK,0..maxS] of Longint;
{*_____*}
Procedure Readf;
  Var f : Text;
      i : Byte;
  Begin
    Assign(f,fi);
    Reset(f);
    Readln(f,K);
    ReadLn(f,S);
    ReadLn(f,N);
    For i:=1 To N Do Read(f,ArN[i]);
    Close(f);
  End;
{*_____*}
Procedure QHD;
  Var i,j,t : Integer;
  Begin
    FillChar(L,SizeOf(L),0);
    For j:=1 To N Do L[1,arN[j]] := 1;
    For i:=2 To K Do
      For j:=0 To S Do
        For t:=1 To N Do
          If ArN[t] <= j Then L[i,j] := L[i,j] + L[i-1,j-arN[t]];
    End;
  End;
{*_____*}
Procedure WriteF;
  Var f : Text;
  Begin
    Assign(f,fo);
    ReWrite(f);
    Write(f,L[K,S]);
    Close(f);
  End;
{*_____*}
BEGIN
  Readf;
  QHD;
  Writef;
END.

```

(Bài giải của bạn Lê Đình Thuận)

### ***Giải bài 194/2004 – Cửa hàng bán kẹo***

Liam mua 3 kẹo mút hết 84p vì vậy mỗi kẹo mút giá 28p

Kenny mua một choco bar, một bánh trứng và một kẹo mút hết 54p, vì vậy choco bar + bánh trứng = 54p – 28p = 26p.

Ta có một choco + bánh trứng + một kẹo chew = 61p từ đó một kẹo chew giá 61p – 26p =

35p.

Mady mua một bánh trứng, một kẹo mút và một kẹo chew hết 80p vậy một bánh trứng giá  $80p - 35p - 28p = 17p$ . Từ đó kẹo choco + bánh trứng = 26p suy ra một bánh trứng là 17p, một choco bar giá 9p. Vậy giá của các loại kẹo như sau:

Kẹo mút = 28p

Kẹo chew = 35p

Bánh trứng 17p

Choco bar = 9p.

Vì vậy nếu Nathan mua mỗi loại một cái thì tổng số tiền phải trả là 89p, vậy Nathan còn lại 11p.

### **Giải bài 195/2004 – Cờ vây**

*Tư tưởng thuật toán:*

Đây là một bài toán **Loang** khá đơn giản tuy nhiên cũng có một số vấn đề nhạy cảm khiến chúng ta phải suy nghĩ. Đối với các bài toán có hình tượng hình học như thế này hầu hết chúng ta đều nghĩ đến một giải thuật cao cấp và đặc biệt nào đó để giải quyết, nhưng cuối cùng lại lựa chọn cách tốt nhất là cách đơn giản nhất! ở bài toán này, thuật toán mẫu mực là thuật toán Loang.

Các bạn chú ý rằng Loang nghĩa là duyệt theo chiều rộng (**BFS – Breadth First Search**) chứ không phải theo chiều sâu (**DFS – Depth First Search**), nhiều bạn đã hiểu nhầm khái niệm này và đã đưa ra thuật toán có tên là Loang nhưng lại duyệt theo chiều sâu. Thực vậy, nếu hình dung một giọt nước đang lan ra nền phẳng đồng thời theo mọi hướng các bạn sẽ thấy đó là hình ảnh của BFS chứ không phải DFS.

Giải thuật của bài toán nhằm mục đích tìm ra số quân trắng ăn được nhiều nhất nhưng cũng chính là để tìm ra vị trí đặt quân đen hiệu quả nhất – và cách của chúng ta là duyệt mọi vị trí có thể đặt quân đen, đặt thử vào đó và tìm xem có bao nhiêu quân trắng được ăn, đồng thời lưu kết quả tốt nhất. Cách này hoàn toàn có thể chấp nhận được về mặt thời gian do kích thước của bài toán không lớn và chúng ta cũng sẽ có một số thủ thuật đặc biệt để tăng hiệu quả. Không phải ai cũng làm được điều này và trong số các bạn giải đề kỳ này nhiều bạn đã không giải quyết được bài toán ở kích thước khá lớn.

*Giải thuật như sau:*

– Thử tại mọi vị trí (i, j) có thể đặt quân đen: Vấn đề là vị trí như thế nào thì đặt được, hầu hết các bạn đều cho rằng có dấu '.' (hay ô trống trên bàn cờ) là đặt và thử ngay, đó là một nhận định thô sơ, chưa sát. Có bạn đưa ra nhận định sắc hơn: nếu xung quanh ô trống có ít nhất 3 ô khác không trống thì đặt, cũng chưa sát nhất. Nhận định tốt nhất để đặt thử quân đen vào đó là: ô đó trống, 3 ô ngay bên trên có quân cờ, quân ở giữa 3 quân đó là trắng – và 3 trường hợp tương tự cho bên dưới, bên trái và bên phải. Hình vẽ minh họa trường hợp đầu, ô có dấu 'x' là phải có quân, ô có dấu '.' là ô trống, các ô còn lại không cần quan tâm đến.

x	w	x
	.	

Có thể nhận thấy các trường hợp trên tương ứng với các trường hợp "đầy nắp" cho vùng quân cờ đen. Nhờ nhận xét này chúng ta đã giảm được số ô thử đặt đi rất nhiều.

for i:=1 to n do

for j:=1 to n do

if (a[i,j]='.') and

((a[i-1,j]='w') and (a[i-1,j-1]>'.') and (a[i-1,j+1]>'.')) or

((a[i+1,j]='w') and (a[i+1,j-1]>'.')) and (a[i+1,j+1]>'.')) or  
 ((a[i,j-1]='w') and (a[i-1,j-1]>'.')) and (a[i+1,j-1]>'.')) or  
 ((a[i,j+1]='w') and (a[i-1,j+1]>'.')) and (a[i+1,j+1]>'.')) then Put(i,j);

– Sau khi đặt quân đen theo cách trên chúng ta bắt đầu thực hiện Loang vùng quân cờ trắng ở mọi phía xung quanh quân mới đặt, một số bạn đã sai trong trường hợp đặt một quân đen ăn hai vùng quân trắng. Đây là vấn đề nhạy cảm nhất. Nếu khi loang đến các ô trắng gặp ô trống bên cạnh nó, vùng loang này không được tính.

– Khi thử đặt một quân đen, các ô trắng loang được không cần xét lại nữa vì vậy ta thay đổi giá trị biểu diễn cho ô này, chẳng hạn ký tự '@'.

– Khi thử đặt một ô đen xong, chúng ta cần khôi phục tình trạng bàn cờ như cũ để xét các vị trí khác.

**Chương trình chi tiết thực hiện giải thuật này như sau:**

```
{ $M 32768,0,655360 }
Const fi='encircle.inp';
      fo='encircle.out';
      dx: array[1..4] of integer=(-1,1,0,0);
      dy: array[1..4] of integer=(0,0,-1,1);
var a: array[0..101,0..101] of char;
    n: byte;
    result: integer;
{ *_____ * }
procedure ReadInput;
var f: text;
    i,j: byte;
begin
  assign(f,fi);
  reset(f);
  readln(f,n);
  for i:=1 to n do
    begin
      for j:=1 to n do read(f,a[i,j]);
      readln(f);
    end;
  for i:=0 to n+1 do
    begin
      a[0,i]:='b';
      a[n+1,i]:='b';
      a[i,0]:='b';
      a[i,n+1]:='b';
    end;
  close(f);
end;
{ *_____ * }
procedure WriteOutput;
var f: text;
begin
  assign(f,fo);
  rewrite(f);
```



```

        write(f,result);
        close(f);
    end;
    {*_____*}
    procedure Put(i,j: byte);
        var qx,qy: array[1..10000] of byte;
            x,y,l,k: byte;
            fq,rq,count: integer;
            stillMH: boolean;
    begin
        a[i,j]:='b';
        count:=0;
        for l:=1 to 4 do
            if a[i+dx[l],j+dy[l]]='w' then
                begin
                    stillMH:=true;
                    fq:=1; rq:=1;
                    qx[fq]:=i+dx[l]; qy[fq]:=j+dy[l];
                    a[i+dx[l],j+dy[l]]='@';
                    while (fq<=rq) and stillMH do
                        begin
                            x:=qx[fq]; y:=qy[fq]; inc(fq);
                            inc(count);
                            for k:=1 to 4 do
                                case a[x+dx[k],y+dy[k]] of
                                    '.': begin
                                        count:=0;
                                        stillMH:=false;
                                        break;
                                    end;
                                    'w': begin
                                        inc(rq); qx[rq]:=x+dx[k]; qy[rq]:=y+dy[k];
                                        a[x+dx[k],y+dy[k]]='@';
                                    end;
                                end;
                            end;
                        end;
                end;
            end;
        if count>result then result:=count;
        a[i,j]:='.';
    end;
    {*_____*}
    procedure MainTH;
        var i,j: byte;
    begin
        result:=0;
        for i:=1 to n do
            for j:=1 to n do
                if (a[i,j]='.') and

```

```

      (([a[i-1,j]='w') and (a[i-1,j-1]>'.')) and (a[i-1,j+1]>'.')) or
      ((a[i+1,j]='w') and (a[i+1,j-1]>'.')) and (a[i+1,j+1]>'.')) or
      ((a[i,j-1]='w') and (a[i-1,j-1]>'.')) and (a[i+1,j-1]>'.')) or
      ((a[i,j+1]='w') and (a[i-1,j+1]>'.')) and (a[i+1,j+1]>'.'))) then Put(i,j);
    end;
  {*_}
BEGIN
  ReadInput;
  MainTH;
  WriteOutput;
END.

```

### ***Giải bài 196/2004 – Các nền văn minh***

#### ***Tư tưởng thuật toán:***

Đây là một bài toán khá hay và điển hình về "mô hình xử lý ngoài". Độc giả THNT đã rất khá quen thuộc với các bài toán dữ liệu lớn, và để khắc phục chúng ta thường tìm đến với những thuật toán linh hoạt, để đảm bảo rằng trong quá trình xử lý chỉ phải dùng đến bộ nhớ trong.

Nhưng, sẽ như thế nào nếu không có giải thuật nào đủ thông minh để với dung lượng hạn chế của bộ nhớ trong, chúng ta có thể quản lý hết khối dữ liệu khổng lồ cần phải xử lý? Lúc đó, chúng ta bắt buộc phải sử dụng đến một kỹ thuật khá phức tạp: "kỹ thuật xử lý ngoài".

Trước khi bàn thêm về các phép xử lý ngoài, chúng ta hãy phân tích giải thuật của bài toán:

Thấy rằng, cách đơn giản nhất để giải quyết bài toán là đem từng nền văn minh so sánh với các nền văn minh khác. Kết quả là hiển nhiên đúng! Tuy nhiên, độ phức tạp của thuật toán là  $O(n^2)$  (đã là không thể chấp nhận được với  $n=10^5$ ), thêm vào đó để phục vụ thuật toán này, hoặc ta cần có 2 mảng  $10^5$  phần tử kiểu longint, hoặc ta phải thực hiện đọc file nhiều lần. Cả hai phương án này đều khó chấp được.

Như vậy, cần có phương án khả thi hơn:

Ta sắp xếp dãy S, E theo chiều giảm dần. Sau đó, với mỗi số  $S[i]$  thuộc S, ta xét các số  $E[j]$  thuộc dãy E, nếu  $S[i], E[j]$  thỏa mãn là thời gian tồn tại chung của cả 2 nền văn minh thì cập nhật, ngược lại ta xét tiếp  $S[i+1]$  với các số thuộc dãy E từ  $E[j+1]$  trở đi. Để sắp xếp hai dãy S, E ta dùng giải thuật Merge Sort (hay còn gọi là sắp xếp hòa nhập hai đường) rất thường xuyên được sử dụng trong xử lý ngoài. Giải thuật Merger Sort được xếp vào bộ những giải thuật sắp xếp nhanh (cùng với quick sort, heap sort) có độ phức tạp  $O(n \log_2 n)$ .

Sau đây là toàn bộ văn bản chương trình giải bài toán "Các nền văn minh"

```

{$A+,B-,D+,E+,F-,G-,I+,L+,N+,O-,P-,Q+,R+,S+,T-,V+,X+}
{$M 16384,0,655360}

```

```

Program cac_nen_van_minh_co_dai;

```

```

uses crt;

```

```

const fi='ancient.inp';

```

```

      fo='ancient.out';

```

```

      ft1='ancien1.tmp';

```

```

      ft2='ancien2.tmp';

```

```

      ft3='ancien3.tmp';

```

```

      m = 10000; vc=maxlongint div 2;

```

```

type km = array[1..m]of longint;

```

```

var n,s,e,nn,l,r : longint;
    d : array[1..5]of ^km;
    p : pointer;
    f : text;
{*-----*}
procedure quick_sort(l,r:longint);
var i,j,t,u,v,x,y : longint;
    tg , a : longint;
begin
    i:=l; j:=r;
    t:=(l+r) div 2;
    u:=(t-1)div m + 1;
    v:=(t-1)mod m + 1;
    a:=d[u]^v;
    repeat
        x:=(i-1)div m + 1;
        y:=(i-1)mod m + 1;
        while d[x]^y > a do
            begin
                inc(i);
                x:=(i-1)div m + 1;
                y:=(i-1)mod m + 1;
            end;
        u:=(j-1)div m + 1;
        v:=(j-1)mod m + 1;
        while d[u]^v < a do
            begin
                dec(j);
                u:=(j-1)div m + 1;
                v:=(j-1)mod m + 1;
            end;
        if i<=j then
            begin
                tg:=d[x]^y;
                d[x]^y:=d[u]^v;
                d[u]^v:=tg;
                inc(i);
                dec(j);
            end;
    until i>j;
    if l
    if i
end;
{*-----*}
procedure merge_sort(ft:string);
var i,j,x,y,a,b : longint;
    f1, f2 : text;
begin

```

```

    assign(f1,ft1); reset(f1);
    assign(f2,ft); rewrite(f2);
    i:=1; readln(f1,a);
    while (i<=(n-nn)) do
        begin
            x:=(i-1) div m +1;
            y:=(i-1) mod m+1;
            if d[x]^>[y]>a then
                begin
                    b:=d[x]^>[y];
                    writeln(f2,b);
                    inc(i);
                end
            else
                begin
                    b:=a;
                    writeln(f2,b);
                    if eof(f1) then break;
                    readln(f1,a);
                end;
            end;
        if i>n-nn then
            begin
                writeln(f2,a);
                while not eof(f1) do
                    begin
                        readln(f1,a);
                        writeln(f2,a);
                    end;
            end
        else
            if eof(f1) then
                begin
                    while i<=n-nn do
                        begin
                            x:=(i-1) div m +1;
                            y:=(i-1) mod m +1;
                            writeln(f2,d[x]^>[y]);
                            inc(i);
                        end;
                    end;
                close(f1); close(f2);
            end;
        {*-----*}
    procedure write_temp(n:longint;ft:string);
        var i,x,y : longint;
            f : text;
        begin

```

```

    assign(f,ft); rewrite(f);
    for i:=1 to n do
        begin
            x:=(i-1) div m + 1;
            y:=(i-1) mod m + 1;
            writeln(f,d[x]^[y]);
        end;
    close(f);
end;
{ * _____ *}
procedure start;
var i,j,x,y : longint;
begin
    assign(f,fi); reset(f);
    readln(f,n);
    nn:=n div 2;
    mark(p);
    for i:=1 to 5 do new(d[i]);
    { doc file lan 1 }
    for i:=1 to n do
        begin
            readln(f,s);
            if i<=nn then
                begin
                    x:=(i-1) div m + 1;
                    y:=(i-1) mod m + 1;
                    d[x]^[y]:=s;
                end;
            if i=nn then
                begin
                    quick_sort(1,nn);
                    write_temp(nn,ft1);
                    release(p);
                    mark(p);
                    for j:=1 to 5 do new(d[j]);
                end;
            if i>nn then
                begin
                    x:=(i-nn-1) div m + 1;
                    y:=(i-nn-1) mod m + 1;
                    d[x]^[y]:=s;
                end;
        end;
    quick_sort(1,n-nn);
    merge_sort(ft2);
    release(p);
    close(f);
    { doc file lan 2 }

```

```

    mark(p);
    assign(f,fi); reset(f); readln(f);
    for i:=1 to 5 do new(d[i]);
    for i:=1 to n do
        begin
            readln(f,s,e);
            if i<=nn then
                begin
                    x:=(i-1) div m +1;
                    y:=(i-1) mod m +1;
                    d[x]^[y]:=e;
                end;
            if i=nn then
                begin
                    quick_sort(1,nn);
                    write_temp(nn,ft1);
                    release(p);
                    mark(p);
                    for j:=1 to 5 do new(d[j]);
                end;
            if i>nn then
                begin
                    x:=(i-nn-1) div m +1;
                    y:=(i-nn-1) mod m +1;
                    d[x]^[y]:=e;
                end;
            end;
        quick_sort(1,n-nn);
        merge_sort(ft3);
        release(p);
        close(f);
    end;
{*_____*}
procedure make;
    var i,j,k,x,y : longint;
        a,h : longint;
        daxet : boolean;
        f,fl : text;
    begin
        assign(f,ft3); reset(f);
        assign(fl,ft2); reset(fl);
        mark(p);
        l:=vc; r:= vc;
        i:=0; j:=2; h:=1;
        daxet:=false;
        for k:=1 to 5 do new(d[k]);
        for k:=1 to nn do
            begin

```

```

        x:=(k-1) div m +1;
        y:=(k-1) mod m +1;
        readln(f,d[x]^[y]);
    end;
while not eof(f1) do
    begin
        readln(f1,a);
        inc(i); k:=0;
        while j<=n do
            begin
                if (j=nn+1)and (not daxet) then
                    begin
                        release(p); mark(p);
                        for k:=1 to 5 do new(d[k]);
                        for k:=1 to n- nn do
                            begin
                                x:=(k-1) div m +1;
                                y:=(k-1) mod m +1;
                                readln(f,d[x]^[y]);
                            end;
                        daxet:=true;
                    end;
                if j<=nn then
                    begin
                        x:=(j-1) div m +1;
                        y:=(j-1) mod m +1;
                    end
                else
                    begin
                        x:=(j-nn-1) div m +1;
                        y:=(j-nn-1) mod m +1;
                    end;
                if a
                    begin
                        if ((d[x]^[y]-a<R-L)AND(I<=H))OR(K>0) then
                            begin
                                l:=a;
                                r:=d[x]^[y];
                            end;
                        inc(j); inc(k);
                    end
                else
                    begin
                        h:=j-1;
                        break;
                    end;
            end;
        end;
    end;
end;

```

```

    release(p);
    close(f);
end;
{ * _____ *}
procedure result;
    var i,s,e,min : longint;
        d,c,c1,c2,c3 : longint;
        ok1,ok2,ok3 : boolean;
        f1,f2 : text;
begin
    assign(f1,fi); reset(f1);
    assign(f2,fo); rewrite(f2);
    if r<>vc then
        begin
            min:=r-l; d:=0;
            c1:=0; c2:=0; c3:=0;
            ok1:=false;;ok2:=false; ok3:=false;
            writeln(min);
            readln(f1);
            for i:=1 to n do
                begin
                    readln(f1,s,e);
                    if (c1=0)and(s<=l)and(e>=r) then c1:=i;
                    if (c2=0)and(e=r)and(s<=l) then c2:=i;
                    if (c3=0)and(s=l)and(e>=r) then c3:=i;
                    if (not ok1)and(not ok2)and(not ok3) then
                        if (s=l)and(e=r)and(i<>c1) then
                            begin
                                d:=i;
                                ok1:=true;
                            end;
                    if (not ok1)and(not ok2)and(not ok3) then
                        if (s=l)and(e-s>min)and(i<>c2) then
                            begin
                                d:=i;
                                ok2:=true;
                            end;
                    if (not ok1)and(not ok2)and(not ok3) then
                        if (e=r)and(e-s>min)and(i<>c3) then
                            begin
                                d:=i;
                                ok3:=true;
                            end;
                    if (ok1)and(c1<>0) then
                        begin
                            c:=c1; break;
                        end;
                    if (ok2)and(c2<>0) then

```



```

begin
    c:=c2; break;
end;
if (ok3)and(c3<>0) then
begin
    c:=c3; break;
end;
end;
writeln(f2,d,' ',c);
end
else writeln(f2,0);
close(f1); close(f2);
end;
{*_____*}
BEGIN
    clrscr;
    start;
    make;
    result;
END.

```

{Bài giải của bạn **Ngô Quốc Hoàn** – Lớp 11A – Trường Năng Khiếu Ngô Sỹ Liên – **Bắc Giang**}

Công việc đọc mã nguồn quả là vất vả? Không, nó sẽ rất thú vị nếu bạn là một người yêu tin học, yêu lập trình. Qua đây, bạn sẽ rút ra được nhiều bài học thú vị cho bản thân. Bạn có thấy rằng chương trình trên khá linh hoạt về giải thuật. Bạn Hoàn đã phá được tư tưởng gò bó rằng các cặp  $E_{u_i}$  Si luôn phải đi cùng nhau, chỉ cần xử lý khéo léo, chương trình của bạn sẽ hiệu quả hơn rất nhiều.

### ***Giải bài 197/2004 – Xếp hàng chào cờ***

Đầu tiên ta xếp lại các học sinh lớp 8 theo chiều cao giảm dần và xếp lại các học sinh lớp 7. Bắt đầu từ học sinh lớp 7 cao nhất. Ta đưa cậu bé này về vị trí số 1 và đưa cậu bé đang đứng ở vị trí số 1 về vị trí của cậu bé cao nhất. Khi đó điều kiện của bài toán (mỗi học sinh lớp 8 đều cao hơn học sinh lớp 7 đứng phía trước) vẫn được thực hiện. Sau đó ta đưa cậu bé lớp 7 cao thứ 2 về vị trí số 2 và đưa cậu bé đang đứng ở vị trí số 2 vào vị trí mà cậu bé cao thứ 2 đang đứng. Tiếp tục làm như vậy ta sẽ có được mỗi học sinh lớp 8 đều cao hơn học sinh lớp 7 đứng phía trước.

### ***Giải bài 198/2004 – Chuỗi nhị phân***

Đây là một bài toán khá hay, đòi hỏi bạn đọc kết hợp cả tư duy toán học và suy duy lập trình để giải bài.

Trước hết ta nhận thấy rằng, mỗi chuỗi nhị phân độ dài  $k$  có thể biểu diễn bằng một số hệ thập phân và ngược lại. Theo công thức tổ hợp đơn giản ta có tất cả  $2^k$  dãy nhị phân độ dài  $k$ . Như vậy với một hoán vị tốt nhất của  $2^k$  dãy con ta sẽ được một dãy dài nhất thỏa mãn yêu cầu đề bài. Vậy với một số nguyên dương  $k$  ( $k \leq 16$ ) thì dãy thỏa mãn yêu cầu mà mỗi dãy con độ dài  $k$  chỉ xuất hiện đúng một lần sẽ có độ lớn nhất là  $2^k + k - 1$ .

Vấn đề còn lại ở đây của chúng ta sẽ chỉ là tìm cách xây dựng dãy có độ dài  $2^k + k - 1$  thỏa mãn yêu cầu đề bài.

Về cơ bản, để xây dựng dãy trên ta có thể duyệt toàn bộ các hoán vị của  $2^k$  dãy! Nhưng

điều đó sẽ gây phí phạm tài nguyên máy một cách vô ích! Vậy có cách nào xây dựng dãy trên một cách tuần tự hay không?

Để tiện giải thích, trước hết chúng tôi xin đưa ra cách xây dựng rồi chứng minh. Xây dựng xây dựng dựa trên tư tưởng quy nạp toán học.

Các bước xây dựng:

- Khởi tạo một xâu gồm  $k$  số 0.
- Tại mỗi bước tiếp theo, mỗi lần ta sinh thêm một số 1 hoặc 0. Ưu tiên số 1 trước. Nếu việc sinh số 1 sẽ tạo ra một dãy con trùng với một dãy con có trước thì bước đó sẽ sinh ra số 0. Kết thúc việc sinh khi độ dài xâu đạt  $2k + k - 1$ . Chứng minh các bước xây dựng là đúng đắn.

- Dễ dàng nhận thấy xâu gồm  $k$  số 0 thỏa mãn yêu cầu đề bài.

- Giả sử ta đã xây dựng được một xâu XY có độ dài  $n$  ( $n < 2k + k - 1$ ) thỏa mãn yêu cầu đề bài. Trong đó Y là xâu nhị phân con độ dài  $k - 1$ , X là xâu nhị phân con còn lại.

Trường hợp 1: xâu Y1 không trùng với một xâu con nào của XY thì ta xây dựng được xâu có độ dài  $n + 1$ . Bài toán được chứng minh

Trường hợp 2: Xâu Y1 trùng với một xâu con của XY. Ta chứng minh rằng với cách xây dựng trên Y0 sẽ không trùng với xâu con nào của XY.

Thật vậy. Giả sử cả Y1 và Y0 đều đã xuất hiện trong XY. Như vậy chắc chắn cả 0Y và 1Y cũng đều đã xuất hiện trong XY. Như vậy cả 4 trường hợp của xâu độ dài  $k - 1$  Y đều đã xuất hiện trong XY (và cả 4 trường hợp này Y không thể xuất hiện ở cuối cùng do có Y0 và Y1). Như vậy việc Y ở cuối xâu XY là vô lý. Điều giả sử là sai. Vậy bài toán được chứng minh.

Từ thuật toán trên, ta dễ dàng viết chương trình với một nhận xét nhỏ nữa liên quan đến kỹ thuật lập trình: Ta sẽ sử dụng một mảng các số nguyên để đánh dấu những xâu đã xuất hiện thay bằng việc phải sử dụng một mảng các xâu. Toàn bộ văn bản chương trình:

program Day\_nhi\_phan;

Const

    fi = 'binstr.inp';

    fo = 'binstr.out';

    max = 32800;

type

    ll = longint;

    ma = array [0..max] of 0..1;

Var   a: ma;

        n, lk, k : ll;

{\*-----\*}

Procedure Openf;

    Begin

        assign(input,fi);

        reset(input);

        assign(output,fo);

        rewrite(output);

    End;

{\*-----\*}

Procedure closef;

    Begin

        close(input);

        close(output);

```

    End;
  { * _____ * }
Procedure khoitao;
  Begin
    readln(k);
    fillchar(a, sizeof(a), 0);
    lk := 1 shl k + (k-1);
    n := 1 shl (k-1);
  End;
  { * _____ * }
Procedure Xuly;
  var  luu, i : ll;
  Begin
    writeln(lk);
    luu := 0;
    for i := 1 to k do write('0');
    for i := k+1 to lk do
      if a[luu+1] = 1 then
        begin
          write('0');
          a[luu] := 1;
          if luu > n then luu := luu - n;
          luu := luu * 2;
        end
      else
        begin
          a[luu+1] := 1;
          write('1');
          if luu >= n then luu := luu - n;
          luu := (luu+1) * 2;
        end;
    End;
  { * _____ * }
BEGIN
  openf;
  khoitao;
  xuly;
  closef;
END.

```

*(Chương trình cài đặt của bạn Nguyễn Thị Kim Thanh – Lớp 11A2 – Trường chuyên Vĩnh Phúc – Vĩnh Phúc)*

### ***Giải bài 199/2004 – Bảng giấy***

Từ đề bài chúng ta có những nhận xét sau:

- + Có N băng màu trắng và N băng màu đen
- + Mỗi một cột, hoặc một hàng sẽ được phủ bằng một băng duy nhất (Trắng hoặc Đen).

Để giải bài toán trên chúng ta sử dụng phương pháp “Bóc lớp”, tức là chúng ta sẽ bóc các lớp băng trên cùng ra, đến khi tất cả các băng đều được bóc. Do tờ giấy này luôn tồn tại một

bảng dán không bị băng gián nào khác đè lên.

Vì  $n \leq 2000$  nên ở đây ta cần phải tổ chức dữ liệu hợp lý, ta dùng 3 mảng một chiều HB, HW, HR, trong đó HB[i], HW[i], HR[i] tương ứng cho biết số ô màu đen, màu trắng và màu bất kì của hàng i (ô mang màu bất kì nghĩa là ô này có thể là màu đen hoặc trắng). Tương tự chúng ta sẽ có 3 mảng một chiều CB, CW, CR để dùng cho các cột, giá trị của 6 bảng này sẽ được tính sau khi chúng ta đọc lần lượt các ô trong file PAPER.inp, ví dụ nếu ô [i,j] là màu 'W' thì HW[i] và CW[j] sẽ được tăng lên một ... Vì mỗi ô của tờ giấy sẽ có một trong 2 màu trắng hoặc đen, nên sau khi đọc hết giá trị các ô trong file PAPER.inp ta có  $HB[i] + HW[i] = n$ ; và  $CB[i] + CW[i] = n$ ;  $HR[i] = CR[i] = 0$ ; với  $i = 1..n$ ;

Đầu tiên ta sẽ đi tìm xem hàng hoặc cột nào chỉ toàn ô màu đen hoặc màu trắng, tức là ta tìm băng dính phía trên cùng, hay ta đi tìm i thỏa mãn  $HW[i] = n$ , hoặc  $HB[i] = n$ , hoặc  $CW[i] = n$ , hoặc  $CB[i] = n$ . Sau đó ta sẽ bóc băng dính này ra, đánh dấu hàng hoặc cột bị bóc (nếu là hàng thì gán  $H[i]:=1$ ; nếu là cột thì gán  $C[i]:=1$ ). Nếu băng được bóc là hàng thì ta phải tính lại các mảng CB, CW, CR. Nếu băng được bóc là cột thì ta phải tính lại các mảng HB, HW, HR. Với chú ý các ô thuộc băng được bóc sẽ trở thành các ô có màu bất kì. Bước tiếp theo ta sẽ đi tìm hàng hoặc cột thỏa mãn tính chất các ô trên nó chỉ có nhiều nhất 2 màu là màu bất kì và màu đen hoặc màu trắng. Sau đó ta sẽ bóc các băng này đi, đánh dấu và tính toán lại các mảng HB, HW, HR, CB, CW, CR... cứ làm như vậy cho đến hết.

Sau đây là toàn bộ chương trình:

```
const fi = 'PAPER.inp';
      fo = 'PAPER.out';
      max = 2001;
type bg = record
  X:char;
  k:integer;
  Y:char;
end;
var f:text;
    HB,HW,HR,CB,CW,CR,H,C:array[1..max] of integer;
    Kq:array[1..2*max] of Bg;
    n,d,sw,sb:integer;
{*-----*}
procedure ReadFile;
  var i,j:integer; kt:char;
  begin
    assign(f,fi);
    reset(f);
    {*Khoi tao cac tap ve 0*}
    fillchar(HB,sizeof(HB),0);
    HW:=HB; HR:=HB;
    CB:=HB; CW:=HB; CR:=HB;
    H:=HB; C:=CB; {Chua Hang, Cot nao duoc chon}
    SW:=0; SB:=0;
    readln(f,n);
    for i:=1 to n do
      for j:=1 to n do
        begin
          read(f,kt);
```

```

        if kt = 'W' then
            begin
                inc(HW[i]);
                inc(CW[j]);
            end
        else if kt = 'B' then
            begin
                inc(HB[i]);
                inc(CB[j]);
            end;
        if j < n then read(f,kt) else readln(f);
    end;
    close(f);
end;
{ * _____ * }
procedure GhiKq(XX:char; KK:integer; YY:char);
begin
    inc(d);
    Kq[d].X:=xx;
    Kq[d].K:=kk;
    kq[d].Y:=yy;
end;
{ * _____ * }
procedure Xuli(chon:integer; mau:char);
var i:integer;
begin
    {Ghi nhan ket qua}
    if chon > 0 then GhiKq(mau,abs(chon),'L')
    else GhiKq(mau,abs(chon),'C');
    {Thay doi trang thai bang a}
    if mau = 'W' then inc(SW) else inc(SB);
    if chon > 0 then
        begin
            H[chon]:=1;
            for i:=1 to n do
                if C[i] = 0 then
                    begin
                        if mau = 'W' then dec(CW[i]) else dec(CB[i]);
                        inc(CR[i]);
                    end;
        end
    else
        if chon < 0 then
            begin
                C[-chon]:=1;
                for i:=1 to n do
                    if H[i] = 0 then
                        begin

```

```

        if mau = 'W' then dec(HW[i]) else dec(HB[i]);
        inc(HR[i]);
    end;
end
end;
{*_____*}
procedure Process;
var chon,i:integer; mau:char;
begin
    repeat
        chon:=0;
        for i:=1 to n do
            begin
                {Hang: chon > 0; Cot: Chon < 0}
                if (HW[i] > 0) and (HW[i] + HR[i] = n) then
                    begin chon:=i; mau:='W'; HW[i]:=0; break; end
                else
                    if (HB[i] > 0) and (HB[i] + HR[i] = n) then
                        begin chon:=i; mau:='B'; HB[i]:=0; break; end
                    else
                        if (CW[i] > 0) and (CW[i] + CR[i] = n) then
                            begin chon:=-i; mau:='W'; CW[i]:=0; break; end
                        else
                            if (CB[i] > 0) and (CB[i] + CR[i] = n) then
                                begin chon:=-i; mau:='B'; CB[i]:=0; break; end
                            end;
                        if chon <> 0 then Xuli(chon,mau);
                    until chon = 0;
                end;
            end;
        end;
    end;
    {*_____*}
end;
procedure Output;
var i:integer;
begin
    {Dien not cac hang, cac cot con` trong'}
    for i:=1 to n do
        begin
            if H[i] = 0 then
                if SW < n then begin GhiKq('W',i,'L'); inc(SW); end
                else if SB < n then begin GhiKq('B',i,'L'); inc(SB); end;
            if C[i] = 0 then
                if SW < n then begin GhiKq('W',i,'C'); inc(SW); end
                else if SB < n then begin GhiKq('B',i,'C'); inc(SB); end;
            end;
        end;
    {kiem tra xem co' xep dan duoc bang giay khong}
    assign(f,fo);
    rewrite(f);
    if (SW = n) and (SB=n) then
        begin

```

```

    for i:=d downto 1 do
        Writeln(f,KQ[i].X,' ',KQ[i].K,' ',KQ[i].Y);
    end
    else writeln('Khong phu duoc');
    close(f);
end;
{*-----*}
BEGIN
    ReadFile;
    Process;
    Output;
END.

```

### ***Giải bài 200/2004 – Cờ domino***

#### ***Tư tưởng thuật toán:***

Đây là một bài toán Quay lui – Đề quy thuộc loại cơ bản. “Bề mặt làm việc” của thuật toán là một bảng hay một lưới các nút và nhiệm vụ của ta là “gắn” các đối tượng vào đó sao cho thích hợp. Nhiều năm nay “lưới” là một dạng cơ bản và thông dụng cho Đề quy trong mọi kỳ thi Tin học, “bề mặt làm việc” mới chỉ dừng ở mức đó chứ chưa đến mức độ khó hơn. Chúng ta hy vọng sẽ sớm có các bài toán với các điểm nút trong không gian 3 hay 4 chiều!

Quay lại với bài toán, có thể thấy không có điểm nào đặc biệt hay nhạy cảm đối với chương trình các bạn viết, chính vì vậy hầu hết các bạn đều làm đúng. Có thể nói bài Domino kỳ này là dễ, không có các điểm “sát thủ” như các bài kỳ trước (như “Cờ vây” chẳng hạn). Cũng do đó sẽ là rất tiếc cho các bạn làm chưa đúng.

ĐRKN chỉ nêu thuật toán của bài toán và một vài kỹ xảo nhỏ khi viết chương trình, không có nhiều điều để nhận xét thêm về bài toán cơ bản này! Tại mỗi vị trí của bàn cờ chưa được đặt quân, ta lấy một quân cờ trong các quân chưa đặt có trùng số với bàn, “thử” vào đó. Quá trình “thử” được cài đặt bằng Đề quy – Quay lui và việc chọn vị trí trên bàn cờ, lấy quân, đặt quân... được thực hiện thông qua các mảng đánh dấu. Các quân cờ được quy ước số sẵn, chẳng hạn quân cờ có 2 số 2–4 là quân cờ 16... Tốt hơn hết là chúng ta cũng “quy ước” lại chúng trong chương trình, có bạn khai báo hằng, có bạn dùng vòng lặp tạo quân cờ... Các cách đó đều tốt và mục đích chính là để tạo ra sự thuận tiện trong chương trình khi thực hiện Đề quy. Nếu giỏi toán (!) các bạn có thể tìm ra một cách liên hệ bằng biểu thức từ 2 số trên quân cờ thành số quân cờ, hãy thử xem!

Các mảng chính chúng ta dùng gồm có một mảng để lưu đầu vào yêu cầu, một mảng đánh dấu bàn cờ và một mảng đánh dấu các quân, đó đều là các mảng 2 chiều có kích thước nhỏ. Riêng mảng đánh dấu bàn cờ khi đề quy chúng ta nên có các phần nổi rộng mảng gọi là “lính canh”. Các bạn có thể thấy các mảng này trong chương trình của bạn Thuận đăng ở bên dưới.

Chú ý cuối cùng là trong trường hợp bài toán không có nghiệm – tất cả các bạn đều đúng vì đề bài đã nói không có test nào như thế (!). Có điều thú vị là một số bạn cho ra một số kết quả “kỳ lạ”, một số bạn không viết file kết quả, một số bạn để file rỗng và tốt nhất là một số bạn ghi số -1 ra file kết quả (đương nhiên bạn thích ghi hay làm gì cũng được).

Chương trình của bạn Lê Đình Thuận – một nhân vật quen thuộc của Đề ra kỳ này:

```

Program Domino;
Const fi = 'Domino.Inp';
      fo = 'Domino.Out';

```

```

Var a : Array [1..7,1..8] of byte;
    b : Array [1..8,1..9] of byte;
    V : Array [0..6,0..6] of Byte;
    T : array [1..28] of boolean;
{ * _____ *}
Procedure Readf;
    Var f : text;
        i,j : Byte;
    Begin
        Assign(f,fi); Reset(f);
        For i:=1 to 7 do
            begin
                for j:=1 to 8 do read(f,A[i,j]);
                Readln(f);
            end;
        Close(f);
    End;
{ * _____ *}
procedure Init;
    var i,j,k : Byte;
    begin
        fillchar(b,sizeof(b),0);
        for i:=1 to 7 do b[i,9] := 29; {linhcanh}
        for j:=1 to 8 do b[8,j] := 29;
        {tao bang ma so}
        k:=0;
        for i:=0 to 6 do
            for j:=i to 6 do
                begin
                    inc(k);
                    V[i,j]:=k;
                end;
            for i:=1 to 6 do for j:=i-1 downto 0 do v[i,j]:=v[j,i];
            fillchar(t,sizeof(t),true);
        end;
Procedure Writef;
Var
    f : Text;
    i,j : byte;
Begin
    Assign(f,fo); Rewrite(f);
    For i:=1 to 7 do
        begin
            for j:=1 to 8 do write(f,b[i,j]:3);
            writeln(f);
        end;
    Close(f);
Halt;

```



```

End;
{ *-----* }
Procedure Try(i,j:Byte);
  var   k : byte;
  Begin
    {tim o tiep theo chua duyet}
    While b[i,j] <> 0 do
      Begin
        inc(j);
        if j>8 then
          begin
            inc(i);
            j:=1;
          end;
        if i>7 then Writef;
      End;
    for k:=0 to 1 do
      if (b[i+k,j+1-k] = 0) and t[v[a[i,j],a[i+k,j+1-k]]] then
        begin
          b[i,j]:=v[a[i,j],a[i+k,j+1-k]];
          b[i+k,j+1-k]:=b[i,j];
          t[b[i,j]]:=false;
          try(i,j);
          t[b[i,j]]:=true;
          b[i+k,j+1-k]:=0;
          b[i,j]:=0;
        end;
      End;
    { *-----* }
  Procedure WriteNo;
    Var   f : Text;
    i,j : byte;
    Begin
      Assign(f,fo); Rewrite(f);
      Writeln(f,'-1');
      Close(f);
    End;
  { *-----* }
  BEGIN
    readf;
    Init;
    Try(1,1);
    WriteNo;
  END.

```