Chuyên đề. QUY HOẠCH ĐỘNG TRẠNG THÁI

Đa số các bài quy hoạch động đều dựa vào trạng thái. Tuy nhiên có trạng thái dễ phát hiện, có trạng thái khó phát hiện, có bài toán số lượng trạng thái ít, có bài toán số lượng trạng thái rất nhiều. Vì vậy, khi nói đến quy hoạch động trạng thái ta thường nghĩ ngay rằng đây là các bài toán tương đối phức tạp, có không gian trạng thái lớn mà ta cần phải sử dụng kỹ thuật mã hóa bằng dãy bit nhị phân. Sau đây là một số bài toán có thể giải quyết triệt để bằng phương pháp quy hoạch động trạng thái.

1. Chọn ô - Select

Cho ma trận vuông a kích thước n×n (1≤n≤20). Các hàng được đánh số từ trên xuống dưới bắt đầu từ 1, các cột được đánh số từ phải sang trái bắt đầu từ 1. Ô nằm giao của hàng i và cột j có tọa độ [i, j]. Trên mỗi ô a[i, j] có chứa một số nguyên.

Yêu cầu: Hãy chọn trên ma trận n ô sao cho

- Mỗi hàng có nhiều nhất một ô được chọn;
- Mỗi cột có nhiều nhất một ô được chọn;
- Tổng giá trị của các ô được chọn là lớn nhất.

Input: cho trong tệp văn bản SELECT.INP:

- Dòng 1: ghi số nguyên dương n;
- N dòng tiếp theo, mỗi dòng ghi n số nguyên dương không vượt quá 10^9 thể hiện dòng thứ i của ma trận.

Output: ghi ra tệp văn bản SELECT.INP trên một dòng là tổng lớn nhất tìm được.

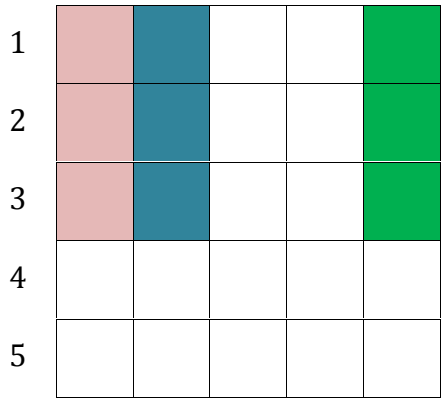
Ví dụ:

Input	Ouput
3 3 1 2 1 1 2 1 4 2	9

Giải: Quy hoạch động trạng thái

- Gọi S là trạng thái chọn cột, như vậy S là một dãy n bit, mỗi bit có giá trị 0 hoặc 1. Các bit được đánh số từ phải sang bắt đầu từ 0 (đánh số từ bit thấp đến bit cao). Ý nghĩa như sau:
  + bit thứ i của trạng thái S = 0: cột i+1 chưa được chọn;
  + bit thứ i của trạng thái S = 1: cột i+1 đã được chọn;
  (0≤i<n)
- Ví dụ với bảng gồm 5 dòng, 5 cột thì trạng thái S là một dãy gồm 5 bit. Trạng thái S = 10011=19 là một ô nào đó của các cột 1, 2, 5 đã được chọn.

1 2 3 4 5



- Gọi T[S] là giá trị tốt nhất khi chọn các ô ở k dòng đầu tiên với trạng thái S (k là số bit 1: số cột được chọn của trạng thái S).
- Ví dụ, khi ta tính T[{1, 2, 5} = 10011 = 19] thì có thể:
  + T[{1, 2, 5}] = T[{1, 2} = 00011 = 3] + a[3, 5]
  + T[{1, 2, 5}] = T[{1, 5} = 10001 = 17] + a[3, 2]
  + T[{1, 2, 5}] = T[{2, 5} = 10010 = 18] + a[3, 1]

  Có nghĩa là T[{10011}=19] là giá trị tốt nhất khi chọn 3 ô ở 3 dòng, 3 cột đầu tiên sẽ bằng max(tổng cách chọn 2 ô ở hai dòng, 2 cột đầu tiên với một ô ở dòng thứ 3). Vì là giá trị lớn nhất nên ta phải lấy max.

- Tổng quát ta có công thức quy hoạch động như sau:

  $$T[S] = \max (T[P] + a[k, j])$$

  Trong đó:
  + S là trạng thái gồm có k cột được chọn
  + P là trạng thái bản sao của S nhưng chỉ khuyết ở cột j (gồm có k – 1 cột được chọn, riêng cột thứ j ở trạng thái S có giá trị 1, trạng thái P có giá trị 0)
- Khi lập trình, ta phải thực hiện hai thao tác xử lý bit:
  + Tắt bit thứ j của trạng thái S, ta sẽ được trạng thái liền trước preS của S

```
function TurnOff(state:long; j:byte):long;
begin
    TurnOff:=state and not (1 shl j);
end;
```

  + Lấy bit thứ j của trạng thái S, thao tác này được sử dụng để giải mã trạng thái (hiển thị các bit 1 của trạng thái)

```
function getBit(state:long; j:byte):byte;
begin
    getBit:= (state shr j) and 1;
end;
```

- Thao tác quy hoạch động được thực hiện bình thường: tính giá trị của trạng thái sau thông qua các trạng thái kề trước nó:

```
function getMax(state:long):long;
var j,k : byte; preState,max : long;
    b : array[1..nm] of byte;
begin
    {giai ma trang thai luu vao mang b}
    fillchar(b,sizeof(b),0); k:=0;
    for j:=1 to n do
        if getBit(state,j-1)=1 then
        begin
            inc(k);
            b[k]:=j;
        end;
    {Tim max cua cac trang thai lien truoc state + a[k,j] voi j
     la bit khac duy nhat giua preState voi state}
    max:=0;
    for j:=1 to k do
        begin
            preState:=TurnOff(state,b[j]-1);
            if max<T[preState]+a[k,b[j]] then max:=T[preState]+a[k,b[j]]
        end;
    getMax:=max;
end;
procedure DPBitmask; {Dynamic Programming with Bitmask}
var state,first,last:long;
begin
    T[0]:=0;
    first:=1; last:=1 shl n - 1; {2^n - 1}
    for state:=first to last do T[state]:=getMax(state);
    sumMax:=T[last];
end;
```

2. Tour du lịch của Sherry - Mã bài: LEM3

Trong kì nghỉ hè năm nay sherry được bố thưởng cho 1 tour du lịch quanh N đất nước tươi đẹp với nhiều thắng cảnh nổi tiếng (vì sherry rất ngoan). Tất nhiên sherry sẽ đi bằng máy bay.

Giá vé máy bay từ đất nước i đến đất nước j là C_{ij} (dĩ nhiên C_{ij} có thể khác C_{ji}). Tuy được bố thưởng cho nhiều tiền để đi du lịch nhưng sherry cũng muốn tìm cho mình 1 hành trình với chi phí rẻ nhất có thể để dành tiền mua quà về tặng mọi người (Các chuyến bay của sherry đều được đảm bảo an toàn tuyệt đối).

Bạn hãy giúp sherry tìm 1 hành trình đi qua tất cả các nước, mỗi nước đúng 1 lần sao cho chi phí là bé nhất nhé.

Input

- Dòng 1: N ($5 < N < 16$)
- Dòng thứ i trong N dòng tiếp theo: Gồm N số nguyên, số thứ j là C_{ij} ($0 < C_{ij} < 10001$)

Output

- Gồm 1 dòng duy nhất ghi chi phí bé nhất tìm được

Example

Input	Output
6 0 1 2 1 3 4 5 0 3 2 3 4 4 1 0 2 1 2 4 2 5 0 4 3 2 5 3 5 0 2 5 4 3 3 1 0	8

Giải: Quy hoạch động trạng thái

- Gọi S là trạng thái thể hiện hành trình du lịch của Sherry, như vậy S là dãy gồm n bit với ý nghĩa:
  + S[i] = 0: thành phố i+1 chưa được thăm;
  + S[i] = 1: thành phố i+1 đã được thăm.

  Các bit được đánh số từ 0..n-1, từ phải sang trái (bắt đầu từ bit thấp nhất).
- Ví dụ, trạng thái S = 10101 = 21 thể hiện Sherry đã đi qua 3 thành phố 1, 3, 5 (ta chưa cần quan tâm đến thứ tự thăm lúc này)
- Gọi T[i, S] là chi phí nhỏ nhất để đến thành phố i với trạng thái S, ta có công thức quy hoạch động như sau:

$$T[i, S] = \min(T[j,P] + C[j, i])$$

- Với P là trạng thái liền trước của trạng thái S, là bản sao của S, chỉ khác ở bit i – 1 (thành phố i chưa được thăm); j là vị trí của các bit 1 trong trạng thái P (các thành phố đã được thăm ở trạng thái P).
- Khi cài đặt, ta phải thực hiện hai thao tác xử lý bit

```
function getBit(state:long; i:byte):byte;
begin
    getBit := (state shr i) and 1;
end;
function TurnOff(state:long; j:byte):long;
begin
    TurnOff := state and not (1 shl j);
end;
```

- Thao tác quy hoạch động được thực hiện bình thường

```
procedure DPBitmask;
var
    u,i,j,k,state,preState,first,last : long;
```


```
    b : array[1..nm] of byte;
begin
    first:=1; last:= 1 shl n -1;
    for u:=1 to n do
    for state:=0 to last do T[u,state]:=oo;
    for u:=1 to n do T[u,1 shl (u-1)]:=0;
    for state:=first to last do
    begin
        fillchar(b,sizeof(b),0);
        k:=0;
        {giai ma trang thai state}
        for i:=1 to n do
            if getBit(state,i-1)=1 then
            begin
                inc(k);
                b[k]:=i;
            end;
        for i:=1 to k do
        begin
             u:=b[i];
             preState:=TurnOff(state,b[i]-1);
             for j:=1 to k do
                if (i<>j) and
(T[b[j],preState]+c[b[j],u]<T[u,state]) then
T[u,state]:=T[b[j],preState]+c[b[j],u];
        end;
    end;
    KetQua:=oo;
    for u:=1 to n do
        if T[u,last] < KetQua then KetQua:=T[u,last];
end;
```

3. VOI06 Chọn ô - Mã bài: QBSELECT

Cho một bảng hình chữ nhật kích thước 4×n ô vuông. Các dòng được đánh số từ 1 đến 4, từ trên xuống dưới, các cột được đánh số từ 1 đến n từ trái qua phải.

Ô nằm trên giao của dòng i và cột j được gọi là ô (i,j). Trên mỗi ô (i,j) có ghi một số nguyên aij , i =1, 2, 3, 4; j =1, 2, ..., n. Một cách chọn ô là việc xác định một tập con khác rỗng S của tập tất cả các ô của bảng sao cho không có hai ô nào trong S có chung cạnh. Các ô trong tập S được gọi là ô được chọn, tổng các số trong các ô được chọn được gọi là trọng lượng của cách chọn. Tìm cách chọn sao cho trọng lượng là lớn nhất.

Ví dụ: Xét bảng với n=3 trong hình vẽ dưới đây:

	1	2	3
1	-1	9	3
2	-4	5	-6
3	7	8	9
4	9	7	2

Cách chọn cần tìm là tập các ô S = {(3,1), (1,2), (4,2), (3,3)} với trọng lượng 32.

Input

- Dòng đầu tiên chứa số nguyên dương n là số cột của bảng.
- Cột thứ j trong số n cột tiếp theo chứa 4 số nguyên a_{1j}, a_{2j}, a_{3j}, a_{4j}, hai số liên tiếp cách nhau ít nhất một dấu cách, là 4 số trên cột j của bảng.

Output

- Gồm 1 dòng duy nhất là trọng lượng của c ách chọn tìm được.

Example

Input	Output
3 -1 9 3 -4 5 -6 7 8 9 9 7 2	32

Hạn chế: Trong tất cả các test: $n \le 10000$, $|a_{ij}| \le 30000$. Có 50% số lượng test với $n \le 1000$.

Giải:

- Theo đề bài thì bảng có 4 dòng và n cột;
- Gọi S là trạng thái chọn các ô ở cột thứ j, ta có thể biểu diễn S bằng 4 bit (các bit được đánh số từ phải sang bắt đầu bằng 0) với ý nghĩa:
  + S[i-1] = 0: dòng thứ i của cột j không được chọn;
  + S[i-1] =1: dòng thứ i của cột j được chọn.
- Với 4 bit, S có thể biểu diễn 16 trạng thái từ {0000} đến {1111} (từ 0 đến 15), tuy nhiên ta nhận thấy chỉ có 8 trạng thái sau là thỏa yêu cầu của bài toán: {0000}, {0001}, {0010}, {0100}, {1000}, {1001}, {0101}, {1010} (tương ứng với các giá trị 0, 1, 2, 4, 5, 8, 9, 10).

- Gọi T[S, j] là trọng lượng lớn nhất khi chọn các ô đến cột thứ j với trạng thái chọn là S, ta có công thức quy hoạch động như sau:

$$T[S, j] = \max(T[P, j-1] + value(S))$$

với P là trạng thái của cột liền trước của S sao cho P và S không có 2 bit 1 đồng thời ở cùng vị trí, còn value (S) là giá trị cách chọn cột j với trạng thái S.
- Khi cài đặt, với bài toán này, ta chỉ cần xây dựng hàm getBit để giải mã trạng thái S:
- Còn thao tác quy hoạch động được thực hiện bình thường

```
procedure DPBitmask;
var i,j,k:int; x:long; state:byte;
begin
    for i:=1 to 8 do T[i,0]:=0;
    for j:=1 to n do
    for i:=1 to 8 do
    begin
        state:=S[i];
        x:=value(state,j);
        T[i,j]:=x;
        for k:=1 to 8 do
            if (state and S[k] = 0) and (T[k,j-1]+x>T[i,j]) then
T[i,j]:=T[k,j-1]+x;
    end;
    KetQua:=T[1,n];
    for i:=2 to 8 do
        if T[i,n]>KetQua then KetQua:=T[i,n];
end;
```

LUYỆN TẬP

1. Trò chơi trên ma trận - Mã bài: QBGAME

Ngày nay các nhà khoa học đã nghĩ ra 1 trò chơi trên ma trận rất thú vị. Thông qua đó có thể đo IQ một cách khá hiệu quả. Trò chơi được mô tả như sau:

Bạn có 1 ma trận A kích thước 8 x N trên đó gồm các số nguyên là điểm của các ô đó. Người ta sẽ yêu cầu bạn chọn 1 tập khác rỗng các ô trên ma trận này sau đó tính tổng điểm trên những ô này. Trong những ô được chọn không có hai ô nào kề cạnh. IQ của người chơi sẽ tỉ lệ thuận với số điểm nhận được. Sherry tham gia trò chơi và đạt kết quả khá tốt.Và bây giờ Sherry muốn biết tổng điểm lớn nhất nhận được trong trò chơi này là bao nhiêu. Bạn hãy giúp sherry nhé !!!

Input

- Dòng 1 là số nguyên N (1 <= N <= 10000)
- 8 dòng tiếp theo: Mỗi dòng gồm n số nguyên. Số nguyên ở hàng i, cột j là A_{ij} ($|A_{ij}| <= 10^8$)

Output

- Gồm 1 dòng duy nhất là số điểm lớn nhất tìm được

Example

Input	Output
2 -22 2 -33 45 56 -60 -8 -38 79 66 -10 -23 99 46 1 -55	279

Giải thích: Chọn các ô (3,1) (5,1) (7,1) (2,2)

2. Cô gái chăn bò - Mã bài: COWGIRL

Trên một thảo nguyên nhỏ bé có 1 gia đình gồm 3 anh em: 2 người anh trai là Nvutri và Andorea còn người em gái là Lola. Cuộc sống gia đình khá giả nhưng gia đình có truyền thống chăn nuôi và muốn để các con tự lập nên cha mẹ 3 người quyết định để các con hằng ngày sẽ đi chăn 1 số bò nào đó (tùy ý 3 người con).

Thảo nguyên là 1 cánh đồng chia làm M*N ô vuông, mỗi con bò chỉ đứng trong 1 ô và mỗi ô chỉ chứa 1 con bò.Chỉ có 1 quy tắc duy nhất là không bao giờ được để 4 con bò tạo thành 1 hình vuông 2*2 hoặc để trống 1 khu đất 2*2.

Hai người anh mải chơi nên đã hối lộ kem để Lola chăn bò 1 mình. Lola muốn biết tất cả có bao nhiêu cách xếp bò thỏa mãn quy tắc trên để đề phòng mọi trường hợp. Vì con số này rất lớn nên hãy giúp Lola tính toán con số này.

Input

- Dòng đầu gồm 1 số T duy nhất là số test (T ≤ 111)
- T dòng tiếp gồm 2 số M, N cho biết kích thước của thảo nguyên (M*N ≤ 30)

Output

- Gồm T dòng, mỗi dòng ứng với 1 test là số cách xếp bò của test đó.

Example

Input	Output
1 1 1	2

3. Đàn bò hỗn loạn – Mã bài: MIXUP2

Mỗi trong N cô bò (4 <= N <= 16) của bác John có một số seri phân biệt S_i (1 <= S_i <= 25,000). Các cô bò tự hào đến nỗi mỗi cô đều đeo một chiếc vòng vàng có khắc số seri của mình trên cổ theo kiểu các băng đảng giang hồ.

Các cô bò giang hồ này thích nổi loạn nên đứng xếp hàng chờ vắt sữa theo một thứ tự gọi được gọi là 'hỗn loạn'.

Một thứ tự bò là 'hỗn loạn' nếu trong dãy số seri tạo bởi hàng bò, hai số liên tiếp khác biệt nhau nhiều hơn K (1 <= K <= 3400). Ví dụ, nếu N = 6 và K = 1 thì 1, 3, 5, 2, 6, 4 là một thứ tự 'hỗn loạn' nhưng 1, 3, 6, 5, 2, 4 thì không (vì hai số liên tiếp 5 và 6 chỉ chênh lệch 1).

Hỏi có bao nhiêu cách khác nhau để N cô bò sắp thành thứ tự 'hỗn loạn'?

Dữ liệu

- Dòng 1: Hai số N và K cách nhau bởi khoảng trắng.
- Dòng 2..N+1: Dòng i+1 chứa một số nguyên duy nhất là số seri của cô bò thứ i: S_i

Kết quả

- Dòng 1: Một số nguyên duy nhất là số cách để N cô bò sắp thành thứ tự 'hỗn loạn'. Kết quả đảm bảo nằm trong phạm vi kiểu số nguyên 64-bit.

Ví dụ

Input	Output
4 1 3 4 2 1	2

4. Tổng trên ma trận! - Mã bài: VMMTFIVE

Cho một bảng số 5x5. Nhiệm vụ của bạn là sẽ phải điền vào ma trận sao cho tổng của các phần tử trên mỗi hàng và mỗi cột bằng một số nguyên cho trước. Mỗi phần tử trong bảng số từ 1 đến 25 và không có hai phần tử bất kì nào giống nhau.

Input

- Dòng thứ nhất gồm 5 số là tổng của các số từ dòng thứ 1 đến dòng thứ 5 của bảng số.
- Dòng thứ hai gồm 5 số là tổng của các số từ cột thứ 1 đến cột thứ 5 của bảng số.

Output

- Gồm 5 dòng, mỗi dòng 5 số thể hiện bảng 5x5 là kết quả của bạn. Nếu có nhiều đáp án, hãy in ra một đáp án bất kì. Dữ liệu đầu vào luôn luôn có kết quả.

Example

Input	Output
60 86 59 38 82 61 59 57 89 59	15 5 9 25 6 17 10 23 20 16 12 19 3 18 7 13 14 1 2 8 4 11 21 24 22

5. Kỷ lục đổ DOMINO (Nguồn bài: Thầy Đỗ Đức Đông)

Một kỷ lục thế giới về xếp domino đổ đã được ghi nhận vào hôm 17/11/2006. Kỷ lục này thuộc về Hà Lan khi 4.079.381 quân domino đã lần lượt đổ xuống theo phản ứng dây chuyền trong tiếng vỗ tay reo hò của các cổ động viên. Những người tổ chức sự kiện Ngày Domino ở Hà Lan cho biết, 4.079.381 quân domino đã lần lượt đổ xuống trong vòng 2 giờ đồng hồ.

Những quân domino đã di động uyển chuyển trên nền những điệu nhạc cổ điển và đương đại là nét đặc biệt nhất của màn trình diễn domino. Tác giả Robin Paul Weijers nói: "Hơn 4 triệu quân domino, điều này chưa bao giờ xảy ra. Chúng tôi còn thành công trong việc khiến cho những quân bài domino nhảy múa trong tiếng nhạc. Tôi rất hạnh phúc vì đã thành công."

Với màn trình diễn tuyệt vời này, những kỷ lục gia domino Hà Lan đã phá vỡ kỷ lục của chính họ lập được năm 2005 với 4.002.136 quân bài domino.

Sắp tới, Bờm dự định xây dựng một công trình lớn hơn để phá kỷ lục của người Hà Lan. Công trình sẽ bao gồm 2 công đoạn chính:

- Công đoạn 1: Xếp $M \times N - T$ quân domino vào các ô còn trống trên hình chữ nhật kích thước $M \times N$ ($M, N \leq 16$), trong hình chữ nhật đó có T ô đã được đặt trước T vật trang trí.
- Công đoạn 2: Xếp $R \times L$ quân domino thành một dãy độ dài L ($L \leq 10^6$), mỗi hàng có đúng R ($R \leq 8$) quân (có thể được hiểu như xếp vào hình chữ nhật kích thước $R \times L$).

Điểm độc đáo trong công trình này là sự phối màu giữa các quân domino lân cận chung cạnh. Các quân domino được xếp bằng hai loại domino, loại 1 có màu xanh nhạt và loại 2 có màu xanh đậm. Quân domino ở vị trí ô (i, j) sẽ phải thỏa mãn điều kiện: nếu $i + j$ lẻ thì màu quân domino này sẽ phải có màu không nhạt hơn các quân ở các ô chung cạnh (nếu có), nếu $i + j$ chẵn thì màu quân domino này sẽ phải có màu không đậm hơn các quân ở các ô chung cạnh (nếu có).

Để có những thông tin thú vị khi giới thiệu về công trình, Bờm muốn biết số lượng cách xếp khác nhau của công đoạn 1 và công đoạn 2. Hai cách xếp được gọi là khác nhau nếu khi chồng khít 2 cách lên nhau (không xoay hoặc lật) có ít nhất một quân khác màu.

Dữ liệu vào trong file "DOMINO.INP" có dạng:

- Dòng 1: gồm 1 số nguyên dương K ($K \leq 10^9$), các kết quả tìm được sẽ mod cho K;
- Dòng 2: bắt đầu là 3 số nguyên dương M, N, T ($M, N \leq 16; T < M \times N$), trong đó M, N là kích thước hình chữ nhật trong công đoạn 1, T là số lượng ô trong hình chữ nhật đã đặt vật trang trí, tiếp theo là T cặp số, cặp số i, j là tọa độ ô đã đặt vật trang trí;
- Dòng 3: gồm 2 số nguyên dương R, L ($R \leq 8$; $L \leq 10^6$) là kích thước hình của công đoạn 2.

Kết quả ra file "DOMINO.OUT" có dạng:

- Dòng 1: số cách xếp công đoạn 1 khác nhau mod K;
- Dòng 2: số cách xếp công đoạn 2 khác nhau mod K.

DOMINO.INP	DOMINO.OUT
1000 5 5 1 3 3 3 10000	240 593