Chương V : TOÁN HỌC TRONG TIN HỌC

I. Phương Trình Gaus :

Chúng ta biết rằng giải hệ phương trình Gaus N ẩn trong toán học . Thế nhưng ứng dụng của nó vào tin cũng là rất quan trọng . Để hiểu rõ hệ phương trình Gaus chúng ta di từ phương pháp Gaus trong toán học . (Các bạn có thể tham khảo phần này ở quyển Cẩm nang thuật toán) .

Tóm tắt Phương pháp :

Tổng quát cho hệ N phương trình N ẩn :

a11x1+a12x2+...+a1nxn=b1
a21x1+a22x2+...+a2nxn=b2
.....
an1x1+an2x2+...+annxn=bn

Hệ phương trình này được viết dưới dạng ma trận như một phương trình duy nhất :

a11 a12 ... a1n		x1		b1
a21 a22 ... a2n		x2		b2
.....	...	.....		
an1 an2 ... ann		xn		bn

Hay viết đơn giản Ax=b , trong đó A là đại diện cho ma trận , x oaji diện cho các biến và b đại diện vế phải của chương trình . Vì các dòng của A được thao tác dọc theo các phần tử của b , để tiện lợi xem b như là cột thứ N + 1 của mảng và dùng mảng N*(N+1) để chứa chúng .

Pha khử -tiến được tóm tắt như sau : trước hết khử biến đầu tiên trong mọi phương trình trừ phương trình thứ nhất bằng cách cộng phương trình thứ nhất (đã nhân với một hằng số thích hợp) với từng phương trình còn lại , kế đến khử biến thứ hai trong mọi phương trình trừ hai phương trình đầu tiên bằng cách cộng phương trình thứ hai (đã nhân với một hằng số thích hợp) với từng phương trình kể từ phương trình thứ ba đến phương trình thứ N , kế đến khử biến thứ ba trong mọi phương trình trừ ba cái đầu tiên ,v,v...Để khử biến thứ i trong phương trình thứ j (với j nằm giữa i+1 và N) ta nhân phương trình thứ i với aji/aii và trừ nó ra khỏi phương trình thứ j . Tiến trình này được mô tả ngắn gọn hơn qua các dòng lệnh nhưsau :

```
For i:=1 to N do
    For j:=i+1 to N do
        For k:=N+1 Downto i do
            A[i,j]:=A[j,k]-A[i,k]*A[j,i]/A[i,i] ;
```

Đoạn mã này có ba vòng lặp ,thời gian thực hiện tỉ lệ với N3 . Vòng lặp thứ ba truy ngược lên để tránh phá huỷ nội dung của a[j,i] trước khi dùng nó để điều chỉnh giá trị của các phần tử khác trong cùng một dòng . Đoạn mã trên quá đơn giản để mà có thể đúng hoàn toàn : a[i,i] có thể là 0 , vì vậy có thể xảy ra trường hợp chia cho 0 . Tuy nhiên điều này dễ sửa vì có thể đổi chỗ bất kỳ dòng nào (từ i+1 đến N) với dòng thứ i để a [i,i] khác 0 ở vòng lặp ngoài cùng . Nếu không có dòng nào như vậy , thì ma trận này là kì dị : không có nghiệm duy nhất . (Chương trình nên thông báo tường minh trường hợp này , hoặc cứ để lỗi chia cho 0 xảy ra) . Cần viết thêm một đoạn mã để tìm dòng thấp hơn có phần tử ở cột i khác 0 và đổi chỗ dòng này với dòng i . Phần tử a[i,i] ,cuối cùng được dùng để khử các phần tử khác 0 dưới đường chéo trong cột thứ i , được gọi là *phần tử trụ* .

Thật sự , tốt nhất nên dùng dòng (từ i + 1 đến N) mà phần tử ở cột i có giá trị tuyệt đối lớn nhất . Lý do là có thể xảy ra lỗi sai nếu giá trị pivot dùng để chia quá nhỏ . Nếu a[i,i] quá nhỏ thì kết quả của phép chia a[j,i]/a[i,i] được dùng để khử biến i ra khỏi phương trình j (với j từ i+1 đến N) sẽ rất lớn . Thật sự , nó có thể lớn như vậy là để kéo các hệ số a[j,k] về một điểm mà tại đó giá trị a[j,k] trở nên méo mó do lỗi sai .

Nói cách khác , các số khác biệt nhiều về độ lớn không thể được cộng hay trừ một cách chính xác theo số dấu chấm động , hệ thống thường dùng để biểu diễn các só thực , và việc dùng một phần tử trụ nhỏ làm tăng đáng kể khả năng những phép toán được thực hiện . Dùng giá trị lớn nhất trong cột i từ dòng i+1 đến N sẽ chắc chắn rằng kết quả của phép chia trên luông luôn nhỏ hơn 1 và sẽ tránh được lỗi sai này . Có thể nhắm đến việc nhìn ở trước cột i để tìm phần tử lớn nhất . Ngời ta đã chứng minh rằng có thể đạt được câu trả lời đúng đắn mà không cần dùng bện pháp phức tạp .

Đoạn mã sau minh hoạ ph khử -tiến. Với mỗi i từ 1 đến N , rà xuống cột để tìm phần tử lớn nhất (trong những dòng thứ i +1 trở lên) . Dòng có chứa phần tử này được đổi chỗ với dòng i , và biến thứ i bị khử khỏi trong các phương trình từ i+1 đến N :

```
Procedure    eliminate ;
var
        i , j , k       :       integer ;
        t       :       real ;
        begin
                for i := 1 to n do
                begin
                        max:=i ;
                        for j:=i=1 to n do
                                if abs(a[[j,i])>abs(a[max,i]) then max:=j ;
```

```
                    for k:=i to n+1 do
                    begin
                        t := a [ i,k]; a[i,k]:=a[max,k]; a[max,k]:=j
;
                    end ;
                    for j:=i+1 to n do
                        for k:=n+1 downto i do
                            if a[j,k]:=a[j,k]-a[i,k]*a[j,i]/a[i,i];
                end ;
            end ;
```

Một số thuật giải có yêu cầu phần tử trụ a[i,i] phải được dùng để khử biến thứ i ra khỏi mọi phương trình ngoại trừ phương trình thứ i (không chỉ là thứ i+1 đến thứ N)

Sau khi thực hiện xong pha khử-tiến ,mảng a có những phần tử nằm dới đường chéo là 0 , kế đến thực hiện *pha thay-ngược* . Sau đây là đoạn mã của pha này :

```
procedure   substitute ;
var
    j , k   :       integer ;
    t       :       real ;
    begin
        for j:=n downto 1 do
        begin
            t:=0.0 ;
            for k:=j+1 to n do t := t + a[j,k]*x[k];
            x[j]:=(a[j,n+1]-t)/a[j,j]
        end ;
    end ;
```

Chúng ta có tính chất sau :

Một hệ N phương trình đồng thời có N biến phải dùng N3/3 phép nhân và cộng để giải nghiệm .

Vấn đề chính là chúng ta phải giải quyết tốt về tính toán của phương pháp này . Sau đây Chúng ta sẽ đi ứng dụng của nó trong các bài toán trong tin học .

Bài toán 1 : Biến Đổi Mảng

Đề Bài :

Cho ma trận nhị phân N*N phần tử (tức là các phần tử trong ma trận chỉ có thể là số 0 hoặc số 1). Vì một mục đích nào đó trong các phép quản lí dữ liệu nên có phép tác động lên hàng nào đó thì trên hàng đó sẽ đảo Bit và một phép tác động lên một cột nào đó thì trên cột đó của

ma trận sẽ đổi Bit. (Đảo Bit là từ Bit 0 thì thành Bit 1). Trong các quá trình tác động đó , sau một số hữu hạn nào đó thì chúng ta sẽ có một ma trận mới.

Yêu cầu đặt ra ở đây là : Khi chúng ta biết được một ma trận mới nào đó, hãy tìm các phép tác động ít nhất đã tác động lên ma trận ban đầu để biến đổi thành ma trận đích đó.

Dữ liệu : Vào từ file : BIENDOI.INP nhưsau:

-Dòng đầu ghi số N là kích thước của ma trận.

-N Dòng tiếp theo , mỗi dòng ghi N số biểu diễn ma trận ban đầu.

-Tiếp đó là một hàng cách.

-N dòng cuối ghi ma trận đích đã được tạo thành.

Kết quả : Ghi ra file : BIENDOI.OUT nhưsau :

-Dòng đầu nếu không thể biến đổi được thì ghi số 0 , ngược lại thi ghi số phép tác động ít nhất để biến đổi được.

-Dòng thứ hai ghi N số ,Số thứ i biểu diễn số phép tác động lên hàng i

-Dòng thứ ba cũng ghi N số , số thứ i biểu diễn số phép tác động lên cột j.

Yêu cầu Kỹ thuật:

Chương trình không được chạy quá 2 giây

N<=100.

Ví Dụ :

BIENDOI.INP BIENDOI.OUT

4 0 1 0 1 1 0 0 1 0 0 1 1 1 1 0 0 1 1 1 0 1 1 0 1 0 1 1 1 1 0 0 0 2 1 0 0 0 0 1 0 0

Hướng Dẫn :

Chúng ta nhận thấy một cách tự nhiên như sau : nếu một hàng / cột nào đó được biến đổi thì nó chỉ biến đổi một lần thì mới đảm bảo số lần biến đổi ít nhất . Chúng ta gọi X[i] và Y[i] là số lần tác động lên hàng i và cột i . Cho nên X[i] hoặc Y[j] chỉ có thể là 0 hoặc 1 . Chúng ta thấy rằng :

(A[i,j]+X[i]+Y[j])Mod 2 = B[i,j] .cho nên ta sẽ có : X[i]+Y[j]:=(B[i,j]+2-A[i,j])mod 2. Chúng ta xây dựng mảng C như sau :

C[i,j]:=(B[i,j]+2-A[i,j])Mod 2 . Thì vậy X[i]+Y[j] =C[i,j]. mà C[i,j] là hoàn toàn xác định . Chúng ta sẽ xây dựng được hệ phương trình 2*N ẩn nhưng có tới N*N phương trình :

```
X[1]+Y[1]=C[1,1];
X[1]+Y[2]=C[1,2];
.....
X[1]+Y[n]=C[1,n];
.....
```

X[n]+Y[n]=C[n,n] ;

Nhưng chúng ta có điều kiện để hệ có nghiệm trong bài toán này hết sức đơn giản : là với 2 ô (i,j)và (u,v) thì chúng ta luôn có :

(C[i,j]+C[u,v])Mod 2 = (C[i,v]+C[u,j])Mod 2 = (X[i]+X[u]+Y[j]+Y[v])Mod2 .

Chính vì thế Nếu có C[i,j] và C[u,v] không thoả mãn điều kiện trên thì không tồn tại X[i] và Y[j] . Nhưng không mất tổng quát chúng ta giữ nguyên (i,j) là (1,1) thì chỉ cần xét với mọi(u,v) thoả mãn là được .

Có một điều đặc biệt là bài toán này giải hệ Gaus tương đối đơn giản . Chương trình Pascal nhusau :

```
(* sau khi đã xây dựng được mảng C  *)

Function        Kiemtra:boolean ;
Var
        i , j :  Integer ;
begin
        kiemtra := false ;
        for i :=1 to n do
         for j :=1 to n do
          If (c[i,j]+c[1,1])mod 2<>(c[i,1]+c[1,j])mod 2 then exit ;
        kiemtra := true ;
end ;

Procedure    Giai_He ;
Var
        i , T : Integer ;
        Begin
                For X[1]:=0 to 1 do
                 Begin
                        For i := 1 to n do
                         Y[i]:=(C[1,i]+2-X[1])Mod 2 ;
                        For i := 2 to n do
                         X[i]:=(C[i,1]+2-Y[1])Mod 2 ;
                        For i:=1 to n do  T:=T+X[i]+Y[i];
                        If T>Min then
                        Begin
                                Min:=T ;
                                Lu_Gi;
                        End ;
                End ;
        End ;
```

Bài toán này tương đối dễ , nhưng nó là bài toán cơ sở để chúng ta làm quen với một thuật giải hết sức mới này . Tuy nó chưa có nhiều bài toán , nhưng lại có ứng dụng trong việc giải quyết các bài toán thực tế rất nhiều .Chúng ta có thể xét các bài toán mở rộng khác :

Bài toán 2 : Biến Đổi (2)

Đề Bài :

Cho hai ma trận A và B (N*M ; 2<=N,M<=100) ghi các số từ 0 đến K (K cho trước).

Một phép biến đổi trên hàng i là : nếu các số trên hàng đó là X thì thành: (X+H) Mod K.

Một phép biến đổi trên cột j là : nếu các số trên cột đó là X thì thành : (X+H)Mod K

Yêu Cầu : Hãy tìm một số phép biến đổi ít nhất để biến đổi từ Ma trận A thành ma trận B (Nếu có thể).

Dữ liệu: Vào từ file: BIENDOI2.INP nhưsau:

Dòng đầu tiên ghi bốn số : N,M,H,K.

2*N +1 dòng tiếp biểu diễn ma trận A và ma trận B , giữa chúng cách bởi một hàng trống.

Kết quả: Ghi ra file: BIENDOI2.OUT nhưsau:

- Dòng đầu tiên ghi số phép biến đổi ít nhất (nếu không thể thì ghi số 0)

Dòng thứ hai ghi M số ,số thứ i ghi các phép biến đổi lên cột i .

Dòng thứ ba ghi N số , số thứ i ghi các phép biến đổi lên hàng i.

Yêu cầu kỹ thuật:

Chương trình phải luôn đa ra kết quả tối ưu.

Chạy không quá 2 giây.

ở OUTPUT : nếu dòng đầu tiên ghi số 0 thì hai dòng sau không cần phải viết.

Ví Dụ:

BIENDOI2.INP

4 4 1 2 0 1 0 1 1 0 0 1 1 0 0 1 0 1 1 0 1 1 1 0 1 1 0 1 1 1 0 1 0 0 1 0 2 0 1 0 0 1 0 0 0

Hướng Dẫn :

Hoàn toàn tương tự bài toán trên về thuật giải , đó là chúng ta đa nó về hệ phương trình Gaus , sau đó giải nghiệm có thể có của hệ đó ta sẽ cho được kết quả của bài toán . Đối với bài toán này đó là chúng ta xây dựng mảng C nhưsau :

C[i,j]:=A[i,j]-B[i,j]

Gọi X[i] và Y[j] là số phép biến đổi của hàng và cột . Ta sẽ có :

(A[i,j]+X[i]*H+Y[j]*H) Mod K = B[i,j] .
Nh vậy hệ Gaus sẽ là : (X[i] + Y[j])*H Mod K = C[i,j] .

Bài toán 3 : Nổ bom

Đề Bài :

Cho mảng A[NxN] , gồm các ký tự 0 và 1 . Mỗi một ô trên đó là một bit . Biết rằng khi chúng ta tác động lên một bít nào đó , thì tất hẳn bốn bít bên cạnh (có chung cạnh với nó)sẽ bị đổi bit (tức là 0 thành 1 và 1 thành 0) và cả nó nữa . Ngời ta muốn điều khiển bảng bit đó thành một bảng nào đó , thế nhưng cần biến đổi sao cho số lần tác động bít là ít nhất .

Dữ liệu : Vào từ file BatBit.Inp :
Dòng đầu tiên ghi số N (N<=50)
N dòng sau , mỗi dòng ghi N số biểu diễn bảng bít ban đầu
N dòng sau ghi bảng bit cần biến đổi tới .

Kết quả : Ghi ra file BatBit.Out :
Dòng đầu tiên ghi 'YES' Hoặc 'NO' nếu nhưcó thể và không thể bật bit về nhau
Nếu có thì làm các yêu cầu sau :
Dòng kế tiếp ghi số lần bấm
Các dòng còn lại ghi toạ độ các ô cần bật bit

Ví dụ :

BATBIT.INP BATBIT.OUT
4 0 1 0 1 1 0 0 0 0 0 0 1 1 1 1 0 0 0 0 1 0 1 0 0 0 0 1 0 1 1 0 0 YES 2 2

Hướng Dẫn :

Ta có bảng A biến đổi về mảng B thì nếu có thì mỗi ô cao lắm thì chỉ bật bit một lần (hoặc không) . Ta sẽ đa bài toán về hệ Gaus nhưsau :
Gọi C[i,j] là số lần bật bit [i,j] tức là C[i,j]=0 hoặc 1 . Ta sẽ có :
(A[i,j]+C[i,j]+C[i-1,j]+C[i+1,j]+C[i,j-1]+C[i,j+1]) Mod 2=B[i,j]
Với i , j = 1 , .. N . Nếu i = 1 hoặc N thì không có C[i-1,j] , C[i+1,j] .
Hoặc Nếu J=1 hoặc N thì không có C[i,j-1] , C[i,j+1] .

Trở lại bài toán này , ta có một kết quả sau :
" Nếu ta biết được hàng thứ 1 của mảng C thì ta sẽ biết được cả bảng đó " .

Chứng Minh :
C11 , C12 , C13 , ... C1n
C21 , C22 , C23 , ... C2n
.....
Cn1 , Cn2 , Cn3 , ...Cnn
Nếu đã có C11 , C12 thì sẽ có được C21 , vì :

(A[1,1]+C[1,1]+C[1,2]+C[2,1]) Mod 2 = B[1,1] (theo hệ Gaus)
A[1,1] , C[1,1], C[1,2] , B[1,1] đã biết . Cho nên C[2,2] cũng sẽ tính được

Mà khi có C11,C12,C13 thì sẽ biết C22 :
(A[1,2]+C[1,1]+C[1,2]+C[1,3] + C[2,2]) Mod 2 = B[1,2] (Theo hệ Gaus)

Vậy khi có C1i,C1i+1 , C1i-1 thì chúng ta biết được C2i .

Tương tự như vậy khi xác định được cả hàng 2 thì xác định được hàng thứ 3 , cứ như vậy ta xác định được cả bảng C

Một cách tổng quát ta có thể chứng minh hoàn toàn tương tự như trên :

" Khi biết được Hàng hoặc cột biên nào của bảng C thì ta sẽ xác định được cả bảng C " .

Khi đó chúng ta chỉ việc dựa vào công việc duyệt các phép biến đổi có thể có của hàng (hoặc cột) biên nào đó . Chính vì thế công việc duyệt chỉ là 2 50 (chưa kể nhánh cận) .

Nhưng hãy chú ý rằng : Kết quả bài toán trên thật sự có lợi khi chúng ta xác định được rằng một giá trị nào đó của C[i,j] (ở biên thì chỉ phụ thuộc vào hàng (cột) gần biên mà thôi) . thì sẽ không phải duyệt nó nữa . Tức là công việc đầu tiên của bài toán sẽ là :

Tính số ô nằm trên biên chắc chắn không bị biến đổi .

Biên nào có số ô chắc chắn không phải biến đổi nhiều nhất sẽ dùng trong việc duyệt và sẽ dựa vào đó sẽ tìm bảng C . Trong tất cả các bảng C có thể có ta sẽ lấy bảng có tổng số hệ số ít nhất làm kết quả .

Bài toán 4 : Turn Around

Đề bài :

Để sắp xếp đống củi vừa thu được , cu tí muốn sắp xếp nó một cách hiệu quả . Bằng cách cậu xếp các đống củi (mẩu gỗ tròn) theo trật tự giảm dần từ đáy và tuân theo quy tắc vật lý .

Ví Dụ : cậu có 9 mẩu gỗ thì cu cậu sẽ sắp xếp nó thành một khối như sau :

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

Mỗi mẩu gỗ có một hướng trong 4 hướng (đông , tây , nam , bắc) . Sau khi cu tí đã xếp xong thì cu tí không ng ý với cách xếp của nó . Tuy nhiên, khi xoay một mẩu gỗ thì các mẩu gỗ kề nó sẽ quay theo . Ví dụ : nếu cu tí quay mẩu gỗ thứ 6 thì các mẩu gồ 2 ,3,7,9,8,5 sẽ quay theo chiều ngược lại .

Yêu cầu : Hãy tìm cách xoay mẫu gỗ để từ một trạng thái ban đầu , đa về trạng thái mong muốn (là tất cả cùng quay về hướng bắc)

Dữ liệu : Vào từ file Turn.Inp :

Dòng đầu tiên ghi số N (N<50)

Dòng kế tiếp ghi N số thể hiện cho hướng của từng mẩu gỗ

Kết quả : Ghi ra file Turn.Out :

Dòng đầu tiên ghi số bớc cần quay

Dòng kế tiếp , mỗi dòng là cặp số (x,t) thể hiện mẩu x quay đi 90*t .

Ví dụ :

Hướng dẫn :

Thuật toán : Duyệt (hạn chế những phần không duyệt) , nhưng sử dụng Gaus để hạn chế bớc duyệt

Đầu tiên chúng ta điền các vị trí vào mảng như sau :

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

Hoàn toàn tương tự cho các ví dụ khác về cách điền . Ta thấy một nhận xét rằng :

Khi chúng ta biết được số lần xoay của hàng dưới cùng . Thì chắc chắn hàng trên nó cũng biết được.Thật vậy Khi chúng ta biết được số lần biến đổi của ô 1 ,ô 2 thì ô 5 cũng biết : vì Số lần biến đổi ô 1 bằng số lần biến đổi của ô 5 và ô 2 và cả ô 1 . Nhưng chúng ta xác định được ô 1 , 2 thì 5 xác định được . Hoàn toàn tương tự : khi biết ô 1 , 5 , 2 , 3 thì ô 6 xác định được . Cứa như vậy ta biết hết hàng 2 tương tự cho hàng trên nữa . Cứ như vậy cho đến hết bảng .

Vì vậy chúng ta chỉ cần duyệt hàng dưới (về cách biến đổi) còn các hàng trên chỉ phụ thuộc theo mà thôi . Nhưng số hạng nhiều nhất ở hàng dưới là 10 . (vì số khúc gỗ không quá 50) . Tức là Chúng ta chỉ cần duyệt 410 là cùng .

Bài toán 5 : Clock

Đề Bài :

Có 9 đồng hồ với các tên AI sắp xếp như hình vẽ dưới . Kim của mỗi đồng hồ chỉ ở một trong 4 vị trí 12 h , 3 h , 6h và 9h . Vị trí của các kim trên đồng hồ được biểu thị bởi một mảng A[1..3,1..3] of 0..3 trong đó A[i,j] =0/1/2/3 biểu thị việc kim đồng hồ tương ứng chỉ 12h , 3h , 6h , 9h . Ví dụ : Tình trạng các kim của 9 đồng hồ được thể hiện bởi mảng bên dưới :

A B C

3 3 0

2 2 2

D E F

2 1 2

G H I

Ta được dùng 9 thao tác sau để quay kim của các đồng hồ :

Thao tác	Các Đồng Hồ	Thao tác	Các Đồng hồ
1	ABDE	2	ABC
3	BCEF	4	ADG
5	BDEFH	6	CFI
7	DEGH	8	GHI
9	EFHI		

Trong đó các thao tác có tên từ 1 đến 9 , cột các đồng hồ ghi nhóm các đồng hồ chịu ảnh hưởng của thao tác tương ứng , thao tác thực hiện việc quay mọi kim đồng hồ trong nhóm một góc 900 theo chiều kim đồng hồ . Với Ví Dụ trên , nếu thực hiện dãy thao tác 5 , 8 , 4 , 9 ta có tình trạng của 9 đồng hồ thay đổi như sau :

3 3 0 3 0 0 3 0 0 0 0 0 0 0 0 0

2 2 2 *5* 3 3 3 *8* 3 3 3 *4* 0 3 3 *9* 0 0 0

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

Dữ liệu : Nhập từ file Input.TXT một mảng 3x3 các số nguyên tố từ 0 đến 3 biểu thị tình trạng ban đầu của 9 đồng hồ , mỗi dòng của file ghi một dòng của mảng

Kết quả : Ghi ra file Output.TXT dãy các thao tác cần tiến hành để đa mọi kim đồng hồ về vị trí 12h . Số thao tác cần ít nhất có thể được .

Ví Dụ :

INPUT.TXT	OUTPUT.TXT
3 3 0	5 8 4 9
2 2 2	
2 1 2	

Hướng Dẫn :

Gọi mảng B[1..9] là mảng cho biết số lần thực hiện thao tác từ 1 đến 9 . Ta có thể đa nó về hệ gaus thông qua cách biến đổi . Và nghiệm của nó như sau :

B[1]:=(8+A[1]+A[2]*2+A[3]+A[4]*2+A[5]*2-A[6]+A[7]-A[8])MOd 4;

B[2]:=(A[1]+A[2]+A[3]+A[4]+A[5]+A[6]+2*A[7]+2*A[9]) Mod 4;
B[3]:=(8+A[1]+2*A[2]+A[3]-A[4]+2*A[5]+2*A[6]-A[8]+A[9])Mod 4;
B[4]:=(A[1]+A[2]+A[3]*2+A[4]+A[5]+A[7]+A[8]+2*A[9])Mod 4;
B[5]:=(4+A[1]+2*A[2]+A[3]+2*A[4]-
A[5]+2*A[6]+A[7]+2*A[8]+A[9])Mod 4;
B[6]:=(2*A[1]+A[2]+A[3]+A[5]+A[6]+2*A[7]+A[8]+A[9])Mod 4;
B[7]:=(8+A[1]-A[2]+2*A[4]+2*A[5]-A[6]+A[7]+A[8]*2+A[9]) Mod 4;
B[8]:=(2*A[1]+2*A[3]+A[4]+A[5]+A[6]+A[7]+A[8]+A[9])Mod 4;
B[9]:=(8-A[2]-A[4]+A[3]+2*A[5]+2*A[6]+A[7]+A[8]*2+A[9]) Mod 4;
Trong đó A[i] là giá trị của mảng đó khi trải ra thành một hàng .

Bài toán 6 : Con Kì Nhông Đổi Màu

Đề Bài :

Chúng ta biết trong toán học có bài toán con kì nhông đổi màu . Bài toán có thể được phát biểu như sau :

“ Cho m con kì nhông , trong đó có i con màu nâu , j con màu xanh , và m-i-j con màu trắng . Khi hai con khác nhau gặp nhau thì cả hai con đó cùng chuyển thành màu của màu mà không phải hai con đó có . Hỏi có thể sau một số lần nào đó mà tất cả các con kì nhông đó cùng màu được không ? “

áp dụng tin vào toán học , các bạn hãy giải quyết vấn đề này khi chúng ta biết được số lợng mỗi con mỗi loại .

Dữ liệu : Cho từ file KINHONG.INP : Gồm nhiều bộ số :

Dòng đầu tiên là N : số bộ số cần kiểm tra

N dòng tiếp theo , mỗi dòng ghi ba số biểu diễn số con kì nhông mỗi màu của từng bộ số tương ứng

Kết quả : Ghi ra file KINHONG.OUT :

N dòng , mỗi dòng hoặc “NO” hoặc “YES” Nếu như không thể và có thể thoả mãn điều kiện bài toán của từng bộ số .

Ví dụ :

KINHONG.INP	KINHONG.OUT
1	NO
15 17 40	

Hướng Dẫn :

Giả sử số con các màu đầu tiên là : X , T , N . Ba trường hợp đơn giản nhất khi các con kì nhông này gặp nhau :

Kiểu a : 1 con xanh gặp 1 con nâu thành hai con trắng :

a : (N,T,X) (N-1,T+2,X-1)

Tương tự cho kiểu b : (N,T,X)(N+2,X-1,T-1) , c: (N,T,C)(N-1,X+2,T-1)

Chúng ta thấy rằng thứ tự các kiểu không quan trọng : ab=ba , nghĩa là kết quả của hai kiểu gặp a và b , và b rồi a là như nhau :

(X,N,T)(a) (X-1,N-1,T+2)(b)(X-2,N+1,T+1)
(X,N,T)(b)(X-1,N+2,T-1)(b)(X-2,N+1,T+1)

Giả sử chúng ta có kiểu gặp a , kiểu gặp b , và kiểu gặp c . Lúc đó theo tính chất trên ta có thể không mất tính tổng quát , giả sử có kiểu a rồi đến kiểu b và cuối cùng là kiểu c . Tức là hiện trạng màu của các con kì nhông qua các lần trên là :

(X,T,N) x a (X- ,T+2*,N-) x b (X--,T+2*-,N-+2*) x c (X--+2*,T+2*--,N-+2*-) .(*)

Tức là chúng ta cần xem xét với N ,T,X thì phương trình (*) phải có 2 nghiệm bằng 0 . Chúng ta dễ thấy điều kiện để có nghiệm là :

Ta gọi số con các loại cuối cùng là X',T',N' thì ta sẽ có :

N'-X'=N-X+3*(-) ; N'-T'=N-T+3*(-) ; T'-N'=T-N+3*(-)

Giả sử chúng có màu trắng cả thì : N'=X'=0 thì N-X Mod 3 = 0 , tức là nếuN-X mà không chia hết cho 3 thì không có thể có nghiệm nào thoả mãn

Tương tự cho các màu Nâu , Xanh .

Để có thể biết được số lần chuyển màu như thế nào , chúng ta có thể dùng phương pháp tìm kiếm . Bằng cách hoàn toàn tương tự bài toán rót nớc vào các bình .

Bài toán 7 : Party

Đề Bài :

Có N đèn màu đánh số từ 1 đến N và 4 nút thay đổi trạng thái của chúng . ấn nút 1 thay đổi trạng thái tất cả các đèn , ấn nút 2 thay đổi trạng thái các đèn có số hiệu lẻ , nút 3 thay đổi trạng thái các đèn có số hiệu chẵn , ấn nút 4 thay đổi trạng thái các đèn có số hiệu dạng 3k+1 (k>=0) .Có một máy đếm c để đếm tổng số các lần ấn các nút trên . ban đầu , tất cae các đèn đều sáng và c = 0

Yêu cầu : Cho giá trị của c và trạng thái cuối cùng của một số đèn . Hãy lập chương trình xác định tất cả các trạng thái có thể có cuối cùng của N đèn tương ứng với các thông tin đã cho

Dữ liệu : Vào từ file text : Party.inp chứa 4 dòng

dòng 1 cho số N

dòng 2 cho giá trị của c

dòng 3 và dòng 4 cho danh sách các đèn có trạng thái cuối cùng tương ứng là sáng tắt . Số hiệu các đèn trong mỗi danh sách cách nhau bởi dấu cách và kết thúc bởi số -1

Kết quả : Ghi ra file text Party.Out chứa tất cả các trạng thái cuối cùng có thể có của các đèn . Mỗi trạng thái ghi trên 1 dòng gồm N kí tự 0 và 1 , trong đó 0 tương ứng với tắt và 1 là sáng .

Hạn chế : 0<=N<=100 , 1<=c<=10000

Số lợng các đèn sáng và tắt ở trạng thái cuối cùng <=2 . Dữ liệu vào đảm bảo có ít nhất 1 trạng thái của N đèn thoả mãn .

Ví Dụ :

Party.Inp	Party.Out
10	0000000000
1	0110110110
-1	0101010101
7 -1	

Hướng Dẫn :

Chúng ta sẽ phân tập hợp các đèn ra như sau :

Tập 1 : Gồm các đèn có số hiệu chia 3 d 1 và chắn

Tập 2 : Gồm các đèn có số hiệu chia 3 d1 và lẻ

Tập 3 : Gồm các đèn có số hiệu chia 3 d khác 1 và chẵn

Tập 4 : Gồm các đèn có số hiệu chia 3 d khác1 và lẻ

Ta thấy ngay khi bấm nút 1 thì tất cả các đèn ở 4 tập sẽ thay đổi . Khi bấm nút 2 thì tất cả các đèn ở tập 2 và tập 4 sẽ thay đổi . Khi bấm nút 3 thì các đèn ở tập 1 và tập 3 thay đổi . Và khi bấm nút 4 thì các đèn ở tập 1 và tập 2 sẽ thay đổi . Như vậy 4 bóng đèn có số hiệu 1 , 2 ,3 ,4 là bốn bóng đại diện cho 4 tập . Chúng ta sẽ chỉ quan tâm đến các trạng thái cuối cùng của bốn bóng này . Vì các bóng khác có hiện trạng giống nố. Như vậy có tất cả : 24 trạng thái có thể có tất cả (nếu có) . Hay là chúng ta tìm tất cả cá khả năng trong 16 khả năng đó . Khả năng nào thoả mãn thì thoả mãn bài ra .

Bài toán 8 : BIẾN ĐỔI MẢNG

Đề bài :

Cho một số nguyên dương P và một mảng M+1 dòng , N cột .Mỗi phần tử của mảng là mốt số nguyên trong phạm vi từ 0 đến P-1 .Các biến đổi mảng cho phép là như sau : Cộng các phần tử của dòng thứ i, (1<=i<=M) với những phần tử tương ứng của dòng thứ M+1 , nếu kết quả của phép cộng lớn hơn P-1 thì trừ đi P (phép cộng mod P) .Một mảng được gọi là tốt nếu sau một số phép biến đổi nêu trên ta nhận được mảng với dòng cuối cùng (dòng thứ M+1) chỉ gồm toàn số 0.

Yêu cầu : cho một mảng , hãy xét xem mảng đó có tốt không ?

Dữ liệu: được cho bởi file : VIRT.INP

Dòng thứ nhất ghi ba số P,N,M (1<=N,M<=100,2<=P<255) .

Trong M+1 dòng tiếp theo , dòng thứ i ghi N số của dòng thứ i của mảng.

Kết qủa: ghi ra file : VIRT.OUT như sau :

Nếu mảng không tốt :

+ dòng thứ nhất ghi số 0

Nếu mảng tốt :

+ dòng thứ nhất ghi số 1 và ghi tiếp trên cùng 1 dòng đó M số mà số thứ i là số lần cộng dòng thứ i của mảng vào dòng thứ M+1.

Ví dụ:

VIRT.INP	VIRT.OUT	VIRT.INP	VIRT.OUT
4 2 2 2 2 2 2 3 3	0	3 2 4 1 0 2 0 0 0 0 1 2 1	1 1 0 0 2

Hướng Dẫn :

Thực chất bài toán đa về giải hệ phương trình :

Gọi X[i] là số lần sử dụng hàng i .

Ta sẽ có :

(X[1]*A[1,1]+X[2]*A[2,1]+...X[m]*A[m,1] + A[m+1,1])Mod p = 0 ;

(X[1]*A[1,2]+X[2]*A[2,2]+...X[m]*A[m,2] + A[m+1,2])Mod p = 0 ;

.....

.....

(X[1]*A[1,n]+X[2]*A[2,n]+...X[m]*A[m,n] + A[m+1,n])Mod p = 0 ;

Bài toán chỉ đòi hỏi chúng ta là hệ này có nghiệm hãy không mà thôi . Cho nên chúng ta chỉ cần tìm nghiệm của chúng (nếu có) là được .

Bài toán 9 : Game 21

Đề bài :

Xét một lới gồm 21 ô vuông được sắp xếp như hình vẽ :

Trong đó ký tự X được thay thế cho số 0 hoặc 1 . Ta gọi phép đảo ngược bit là việc thay thế bởi 1 hoặc thay thế bởi 0 . Cho phép thực hiện 3 phép biến đổi sau đây :

Đảo ngược bit trong một hình vuông kích thước 3x3 của bảng

Đảo ngược bit trong 5 ô của một chữ thập trong bảng

Đảo ngược tất cả bit ở tất cả các ô của bảng .

Hình vẽ :

```
    x  x  x
x   x  x  x  x
x   x  x  x  x
x   x  x  x  x
    x  x  x
```

```
    0  0  0
```

```
0     0  0  0  0
0     0  0  0  0
0     0  0  0  0
      0  0  0

      1  0  0
1     1  0  1  1
0     1  1  1  1
0     0  1  1  1
      0  0  0
```

Yêu cầu : Cho trạng thái của lới xuất phát và lới đích hãy xác định số phép chuyển lới từ trạng thái xuất phát về trạng thái đích hoặc thông báo không thể thực hiện được

Dữ liệu : Vào từ file Game21.Inp :

Dòng đầu tiên chứa 21 số trên lới xuất phát được liệt kê theo thứ tự từ trên xuống dưới , từ trái qua phải .

Dòng thứ hai chứa 21 số trên lới đích được liệt kê theo thứ tự từ trên xuống dưới , từ trái sang phải . Các số trên cùng một dòng được ghi cách nhau bởi dấu cách .

Kết quả : Ghi ra file Game21.Out : số lợng phép biến đổi ít nhấtcần thực hiện đối với bảgn xuất phát để thu được bảng đích hạoc ghi số -1 nếu không thể biến đổi được .

Ví Dụ :

```
GAME21.INP                                  GAME21.OUT
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0  1 0 0 1 1 0 1 1 0 1 1 1 1 0 0 1 1 1
0 0 0                                       2
```

Hướng Dẫn :

Gải hệ phương trình Gaus cho các hệ nghiệm đã cho . Xây dựng hệ nghiệm (hệ phương trình này có 21 phương trình) .

II. Số và Dãy số :

Các bài toán dạng này thường gặp đó là :

Dựa vào tính chất của một loại số nào đó , thông thường đó là tính chất đặc trng của loại số đó

Các bài toán về dãy thì thông thường lợi dụng tính chất của từng số trong dãy đó

Ta xét các bài toán sau :

Bài toán 10 : Rotation

Đề Bài :

Cho một số X(số chữ số của X <=14). Chúng ta gọi số đó là một số xoay khi chúng ta xoay số X 180 thì ta vẫn được số X. Ví Dụ : 11,69,96 là những số xoay . Yêu cầu đặt ra là : Khi cho một số K , Hãy tìm xem với những số có K Chữ số thì có bao nhiêu số K và đó là những số nào ?.

Dữ liệu : Vào từ File Input: Rotation.Inp Chỉ ghi duy nhất một số nguyên dương K(1<=K<=14) .

Kết quả : Xuất ra File rotation.Out : chỉ một dòng ghi tất cả các số thoả mãn ,chúng được ghi cách nhau bởi một dấu cách.

Ví Dụ :

ROTATION.INP	ROTATION.OUT
2	11 69 88 96

Hướng dẫn :

Chúng ta có thể thấy ngay trong hệ thập phân thì các số từ 0 đến 9 thì có những số :0,1,6,8,9 là những số khi xoay 180 thì tạo ra một số có nghĩa .Mặt khác khi xoay như thế để toạ thành một số xoay thì số đó phải có một trục đối xứng (tức là các chữ số từ 1K Div 2 thì là kết quả của phép xoay của các số từ K Div 2+1 K). Chính vì thế chúng ta có thể duyệt các chữ số trong K div 2 chữ số để tìm ra một số mà khi xoay 180 thì vẫn tạo ra số đó .Công việc giải quyết vấn đề trên có thể được mô tả cụ thể như sau :

Đầu tiên chúng ta dùng hàm xoay một số (09) tạo một số mới có nghĩa :

00 , 11, 69, 96, 88.

Xây dựng hàm Try(I:Integer) để duyệt các khả năng của số thứ I .

Nếu K Chẵn thì chúng ta chỉ xét các chữ số từ 1 đến K Div 2 rồi sau đó dùng hàm xoay để xác định các số ngược lại .

Nếu K Lẻ thì số thứ (K+1) div 2 chỉ có thể là các số 0,1,8. còn các số từ 1K div 2 thì ta dùng thủ tục try để tìm các số đó rồi sau đó dùng hàm ngược để xác định các giá trị ngược .

Tôi có thể viết rõ các thủ tục trên như sau :

```
Function  Xoay(I:Byte):Byte;
Begin
        Case I Of
        0: Xoay:=0;
        1: Xoay:=1;
        6: Xoay:=9;
        8: Xoay:=8;
        9: Xoay:=6;
```

```
        End;
End;

Procedure Try(I:Integer);
Var J:integer;
Begin
        For J:=1 To 5 Do
        Begin
                If (I=0)And(J=1) Then Break;
                B[I]:=A[J]To K Do
                If I=K Div 2 Then Xuat
                Else Try(I+1);
        End;
End;
```

Mảng A[1..5]=(0,1,6,8,9)
Mảng C[1..3]=(0,1,8);
Trong đó thủ tục Procedure Xuat Như sau :

```
Procedure Xuat;
Var I,J:Integer;
Begin
        For I:=K Div 2+1 To K Do
        B[I]:=Xoay(B[K -I+1]);
        If Odd(K) Then
        Inc(Dem,3);
        For J:=1 To 3 Do
        Begin
                B[(K+1) Div 2]:=C[J];
                For I:=1 To K Do Write(F,B[I]);
                Write(F,' ');
        End;
End;
```

Bài toán 11 : Số Queen

Đề Bài :

Cho bảng (N*N) trong đó có N*N ô vuông ,mỗi ô vuông ghi một số nằm trong khoảng từ 1 đến N*N . Biết rằng hai ô khác nhau thì ghi hai số khác nhau . ngời ta đặt N con hậu vào bảng trên sao cho không có hai con nào ăn con nào . Tổng số các số ghi trên tất cả n ô vuông đặt hậu đó gọi là số "queen" .

Hỏi có bao nhiêu cách điền các số vào bảng trên sao cho mọi cách đặt hậu vào thi các số "queen" không đổi .

Dữ liệu Vào từ file : numqueen.inp chỉ ghi một số là N(4<=N<=10).

Kết quả Ghi ra file: numqueen.out ghi số cách điền vào bảng đó
Yêu cầu Chương trình của bạn chạy không quá 5 giây
Ví dụ :

N=4 Numqueen = 561 ;

Hướng Dẫn :

Trong ma trận A(N N), khi ta điền các số từ 1 đến N x N theo tuần tự từ trên xuống dưới (gọi là ma trận khởi đầu). Ta sẽ thu được một ma trận có tính chất : tổng các số trong N ô vuông chọn ra , với không có một ô vuông nào cùng hàng hoặc cùng cột thì bằng một giá trị không đổi . (*)

Ta có thể chứng minh như sau : với một ô vuông (i,j) thì A[i,j]=N*(i-1)+j

vì vậy khi ta chọn ra N ô vuông khác nhau và không có hai ô nào cùng hàng hoặc cùng cột thì mọi chỉ số hàng , chỉ số cột đều tham trong đó . chính vì vậy thì tổng của N ô vuông (thoả mãn điều kiện trên) thì có giá trị bằng :

T= (i-1)*N + j với (i= 1 --> N , j=1-->N) . suy ra T= (N*N+1)*N / 2 .

Sau đây là một hệ quả của kết quả trên :

Hệ quả :

Khi đổi chỗ hai hàng hoặc hai cột của ma trận khởi đầu , hoặc ma trận sau khi tạo bởi biến đổi đó , thỉ kết quả trên vẫn không đổi .

Thật vậy ta gọi i,j là hai cột đổi chỗ cho nhau , thì hai ô (i,t1) , và ô (j ,t2) là hai ô ban đầu mà chúng ta chọn ra . Vậy khi đổi chỗ hai cột đó thì giá trị ô (i,t1) bây giờ là giá trị của ô (j,t1) còn ô (j,t2) có giá trị là ô (i,t2) . Như vậy tổng giá trị của hai ô này không đổi

vì A[i,t1]+A[j,t2]=(i-1)*N+t1+(j-1)*N+t2 = (j-1)*N +t1 + (i-1)*N +t2 = A[j,t1]+A[i,t2].

Tương tự cho phép đổi hàng cũng như vậy .

Từ hệ quả trên ta thấy như vậy khi đổi chỗ chỗ các cột cho nhau , hoặc các hàng cho nhau thì ma trận tạo thành vẫn thoả mãn tính chất (*). Đến đây ta có khi một hàng cố định

Ta dùng phép biến đổi hàng thì có (N-1)! ma trận mới tạo thành thoả mãn (*) và đều khác nhau . Trong một hàng khi một ô giữ nguyên không đổi chỗ thì có (N-1)! ma trận mới tạo thành mà khác nhau và thoả mãn (*) . Như vậy cứ một ô giữ nguyên không được đổi chỗ thì có (N-1)!*(N-1)! ma trận mới tạo thành , và thoả mãn (*) . Như vậy ma trận khởi đầu có N*N ô thì có (N-1)!*(N-1)!*N*N = N!*N! mà trừ đi N*N-1 ma trận trùng với ma trận ban đầu , thì như vậy có :N!*N!-N*N+1 ma trận mới tạo thành sau các phép biến đổi các hàng hoặc các cột cho nhau , và các ma trận này đều thoả mãn (*) .

Vậy với mỗi N ta có số ma trận có thể có là : N!2-N2+1 .

Bảng Test : tương ứng giữa N và Numqueen .

NUMQUEEN.INP	NUMQUEEN.OUT
4	561
5	14376
6	518365
7	25401552
8	1625702337
9	131681894320
10	13168189439901

Bài toán 12 : Số Chính

Đề Bài :

Cho một dãy số gồm N phần tử (N<=106). Các số của nó nằm trong khoảng lognint . Ngời ta định nghĩa rằng một dãy có số chính khi số đó lặp đi lặp lại quá N/2 . Hỏi cho một dãy cho trước , dãy đó có số chính không . Nếu có thì hãy tìm số đó .

Dữ liệu : Vào từ file văn bản Sochinh.Inp :

Là một dãy các dòng ghi các số của dãy (các số viết trên từng dòng)

Kết quả : Ghi ra file văn bản Sochinh.Out :

Nếu không có số chính thì viết NO ngược lại ghi YES và dòng tiếp ghi số chính đó .

Ví dụ :

SoChinh.Inp	SoChinh.Out
2	YES
3	2
4	
2	
2	
3	
2	

Hướng Dẫn :

Chúng ta thấy rằng , khi một số nào đó lặp lại quá N/2 lần thì khi chúng ta lấy số số đó có trong dãy trừ đi số số còn lại mà lớn hơn 0 thì chúng ta có số đó . Chính vì thế chúng ta có tính chất ấy nên , áp dụng Diricle chúng ta có thuật toán sau :

```
Procedure   sochinh ;
var
f       :       text ;
i , count , j : longint ;
begin
```

```
                assign ( f,'sochinh.inp');reset ( f);
                n:=1 ;
                Count:=1;
                readln ( f,i);
                while not eof(f) do
                begin
                        inc ( n ) ;
                        readln ( f , j ) ;
                        if j=i then inc(count) else
                        begin
                                count:=1 ; i:=j;
                        end ;
                end ;
                close(f) ;
                if count>1 then
                begin assign(f,'sochinh.inp');reset(f) ;
                        for j:=1 to n do begin readln ( f , t ) ; if t=i then inc (
count ) ;end ;
                        if count>N/2 then sochinh:=i else sochinh:=không có
;
                end ;
        end ;
```

Bài toán 13 : Dòng tiêu đề

Đề bài :

Mỗi chương của một cuốn sách được lu giữ dưới dạng một dãy các dòng văn ban dạng Text khác dấu cách. Các dòng văn bản đó được chia làm hai loại :

Văn bản thông thường thể hiện nội dung của chương đó . Các dòng văn bản thông thường thì không lặp lại . Hay nói cách khác là hai dòng văn bản thông thường thì đôi một khác nhau .

Dòng tiêu đề ,mỗi dòng tiêu đề của chương do xuất hiện trong quyển sách thì hoàn toàn giống nhau . Có ít nhất hai dòng trong dãy là dòng văn bản và không dưới 10% số dòng là dòng tiêu đề có thể cuất hiện tại một vị trí bất kì trong dãy .

Tất cả mỗi dòng trong dãy đều có độ dài nhưnhau . Số dòng trong dãy cũng không quá 1000000 dòng và độ dài của dòng cũng không vợt quá 80 kí tự và không nhỏ hơn 1.

Yêu Cầu : Nhiệm vụ đặt ra là chúng ta hãy tìm dòng tiêu đề của quyển sách đó .

Dữ liệu : Vào từ File TieuDe.Inp Gồm nhiều dòng , mỗi dòng ghi các dòng hoặc các dòng tiêu đề của chương . Chúng ta hãy tìm xem dòng

tiêu đề trong quyển sách đó là dòng nào , và hãy tính xem trong cuốn sách đó có bao nhiêu chương .

Kết quả : Ghi Ra File TieuDe.Out nhưsau :

Dòng đầu tiên ghi dòng tiêu đề của cuốn sách nếu có tiêu đề ,ngược ghi 0 khi không có tiêu đề.

Dòng Thứ hai ghi số chương của cuốn sách (nếu dòng 1 ghi 0 thì không phải ghi)

Ví Dụ:

TIEUDE.INP TIEUDE.OUT
SACH chuong1tinhoclagi SACH chuong SACH taisaolaihoctin SACH tinhoccoungdunggi SACH mucluc SACH 5

Hướng Dẫn :

Chúng ta sẽ dùng Diricle . Thì theo đề bài chỉ có một xâu là được lặp lại .Cho nên chúng ta sẽ tìm 20 dòng một , trong 20 dòng đó nếu tồn tại hai dòng nào đó giống nhau thì đó chính là dòng cần tìm .

Bài toán 14 : Dãy Chia Hết

Đề bài :

Biểu thức chia là biểu thức số học có dạng sau đây :

x1/x2/x3/.../xn

Trong đó xi là số nguyên dương với mọi i (1<=i<=k) . Biểu thức chia được tính giá trị theo thứ tự từ trái sang phải . Chẳng hạn giá trị của biểu thức :

1/2/1/2

Là 1/4 , ngời ta có thể đặt các dấu ngoặc vào biểu thức để thay đổi giá trị của nó . Ví dụ giá trị của biểu thức :

(1/2)/(1/2)

Là 1 .

Yêu cầu : Cho biểu thức chia E , hỏi có thể đặt các dấu ngoặc vào nó để thu được biểu thức E’ có giá trị là một số nguyên hay không ?

Dữ liệu : Vào từ file Div.Inp :

Dòng đầu tiên chứa số nguyên dương d (d<=5) là số bộ số dữ liệu trong file

Tiếp đến là các bộ dữ liệu được ghi theo quy cách sau L

+ Dòng đầu tiên của một bộ dữ liệu chứa số nguyên N (2<=N<10000) là số lợng số nguyên trong biểu thức

+ Mỗi dòng trong số N dòng tiếp theo chứa một số nguyên dương không vợt quá 109 , số ở dòng thứ i tương ứng với số nguyên thứ i trong biểu thức .

Kết quả : Ghi ra file Div.Out :

Dòng thứ i (1<=i<=d) chứa chữ YES nếu biểu thức thứ i trong file dữ liệu có thể biến đổi thành biểu thức có giá trị nguyên hoặc chứa chữ NO nếu trái lại .

Ví dụ :

DIV.INP	DIV.OUT
2 4 1 2 1 2 3 1 2 3	YES NO

Hướng Dẫn :

Chúng ta thấy X2 luôn phải ở mẫu số (cho dù chúng ta bỏ các dấu ngoặc như thế nào đi nữa) .Như vậy Khi tồn tại một cách viết biểu thức nào đó mà kết quả nguyên thì
ít ra tích của các tử số cũng phải chia hết cho X2 . Như vậy lúc đó ta hoàn toàn viết được một biểu thức mà chỉ có X2 là mẫu số :

(X1/X2)/X3/X4/...Xn .

Vậy nếu tồn tại lời giải thì Ta phải có Tích của dãy đó (trừ số thứ hai) là phải chia hết cho số thứ hai . Nếu không thoả mãn tức là không tồn tại cách viết nào để thoả mãn điều kiện kết quả nguyên .

Bài toán 15 : Danh Sách Vòng

Đề Bài :

Để làm việc với một danh sách gồm N số nguyên cần phải có 2 thao tác . Thao tác Top chuyển phần tử đầu tiên của danh sách xuống vị trí cuối cùng của danh sách , còn thao tác Bottom chuyển phần tử cuối cùng của danh sách lên vị trí đầu tiên của danh sách . Ta gọi một phép biến đổi danh sách đã cho là việc thực hiện đầu tiên là K thao tác Top , tiếp đến là L thao tác Bottom . Do đó lần thực hiện các thao tác với danh sách là rất lớn nên đòi ỏi phải có những thủ tục thực hiện hiệu quả hai thao tác nói trên để thực hiện liên tiếp X phép biến đổi để chuyển danh sách về trạng thái cuối cùng .

Yêu cầu : Viết chương trình cho phép : đối với một danh sách và hai số K , L cho trước , xác định trạng thái của danh sách sau X lần thực hiện phép biến đổi .

Dữ liệu : Vào từ file Clist.Inp :

Dòng đầu tiên chứa 3 số nguyên dương N , K , L (1<=N,K,L<=100)

Dòng thứ hai chứa N số nguyên , mỗi số có trị tuyệt đối không vợt quá 10000, được sắp xếp theo thứ tự tương ứng với trạng thái khởi đầu của danh sách .

Dòng thứ ba chứa số nguyên X (0<=X<=2000000000)

Kết quả : Ghi ra trên một dòng của file văn bản : Clist.Out danh sách các phần tử của danh sách được xếp lại sau X phép biến đổi .

Ví dụ :

Clist.Inp	Clist.Out
5 2 1 1 2 3 4 5 10	5 1 2 3 4

Hướng dẫn :

Chúng ta sẽ có một danh sách vòng . Khi chúng ta chuyển lên trước hoặc về sau thì chúng ta chỉ cần di chuyển thanh chỉ của danh sách vòng . Mặt khác , khi chúng ta sử dụng một số lần liên tiếp các phép Bottom hoặc Top xen kẽ nhau thì chúng ta chỉ cần làm liên tiếp với số hiệu c= K-L . Mặt khác ta chỉ cần chuyển một số lần là phép d của C*X Mod N . Lúc đó thanh chỉ của danh sách sẽ chỉ cho chúng ta biết được vị trí của số hạng đầu tiên .

Bài toán 16 : Số thống kê

Đề Bài :

Xét số 5553141.Nếu thống kê số lần xuất hiện các chữ số, ta có "*Hai 1, một 3, một 4, , ba 5"* . Nếu viết thống kê nay toàn bằng số, ta có số mới là 21131435.Số này được gọi là số thống kê của số ban đầu. Ngời ta phát hiến ra một số tự sinh ra nó, tức là số bằng số thống kê của nó. Ví dụ số 31123314.

Một số được gọi là tự sinh sau j bớc (j 1) . Nếu j là số nhỏ nhất, sao cho sau j lần thống kê, ta được số tự sinh.

Ví dụ số 21221314 là số tự sinh sau 2 buớc vì:

21221314 31321314 31123314

Số cuối cùng trong dãy là số tự sinh

Một số được gọi là thuộc chu trình thống kê k (k 2) . Nếu k là số nhỏ nhất, sao cho tồn tại j nguyên (j 0) giá trị thống kê thứ j cho kết quả giống giá trị thống kê thứ j4k.

Ví dụ : số 314213241519 thuộc chu trình thống kê k = 2. vì :

314213241519 412223241519 314213241519

(Trong trường hợp này j = 0).

Yêu cầu : Hãy viết chương trình đọc dãy số nguyên không âm . Với mỗi số nguyên hãy xác định xem nó thuộc loại tự sinh sau j bớc hay thuốc chu trình thống kê k, hoặc không thuộc loại nào sau 20 bớc thống kê.

Dữ liệu : Vào từ file văn bản STATJCJNP mỗi dòng chứa môt só nguyên không quá 30 chữ số và không có 0 ở đầu.

Kết quả : Đa ra file văn bảnTATIC.OUT mỗi dòng nhắc loại số đó và kết luận loại số.

Ví dụ:

STATIC.INP	STATIC.OUT
22	22 TU SINH
314213241519	314213241519 THUOC CHU TRINH 2
21221314	21221314 TU SINH SAU 2 BUOC

Hướng Dẫn :

Chúng ta sẽ làm một thủ tục thực hiện kiểm tra sau một lần đếm thì số mới sẽ là một số như thế nào . Công việc này quá đơn giản . Rồi sau đó cho một vòng lặp để kiểm tra (nếu quá 20 bớc thì dừng) nếu số nào thoả mãn số thống kê thì đúng

Bài toán 17 : DÃY 0/1

Đề Bài :

Xuất phát từ số A1 = 1, ngời ta xây dựng dãy bít A2, A3, ... An... theo cách sau: Dãy Ai+1 nhận được từ dãy Ai bằng cách viết tiếp vào sau nó các chữ số nghịch đảo của Ai.

A1 = 1
A2 = 10
A3 = 1001
A4 = 10010110
. . . . . .

Cho số nguyên K (trong phạm vi Longint). Hãy xác định giá trị bít thứ K của AN (với N đủ lớn).

Dữ liệu: Vào từ file BIT.INP, mỗi dòng một số nguyên K.

Kết quả: Đa ra file BIT.OUT các bít tìm được, mỗi bít trên một dòng.

Ví dụ:

BIT.INP	BIT.OUT
3	0
6	1
14	0

Hướng Dẫn :

Chúng ta đặt mảng Mu2[i] = 2i . Ta xét một số x nào đó thì ta sẽ tìm đoạn mà chứa nó , và tìm đoạn đối xứng với đoạn đó trong cách viết . Cứ như thế cho đến khi xuống tận cùng thì hoặc là nó giống với 0 =A[2] , hoặc 1 = a[1] .

Bài toán 18 : CHIA HẾT:

Đề Bài :

Xét một dãy gồm các số nguyên tuỳ ý. Ta có thể đặt các dấu cộng hoặc trừ vào giữa hai số hạng của dãy để thu được các biểu thức số học

khác nhau. Chẳng hạn xét dãy số: 17, 5, 21, 15. Có 8 biểu thức khác nhau:

17 + 5 + -21 + 15 = 16
17 + 5 + -21 - 15 = - 14
17 + 5 - -21 + 15 = 58
17 + 5 - -21 - 15 = 28
17 - 5 + -21 + 15 = 6
17 - 5 + -21 - 15 = - 24
17 - 5 - - 21 + 15 = 48
17 - 5 - -21 - 15 = 18

Ta nói dãy số là chia hết cho K nếu có thể đặt các dấu cộng hoặc trừ vào giữa các số hạng của nó để thu được biểu thức số học có giá trị là một số chia hết cho K. Trong ví dụ trên dãy số đã cho là chia hết cho 7 (17 + 5 - 21 - 15 = -14) nhưng nó không chia hết cho 5.

Hãy viết chương trình xác định tính chia hết của một dãy số đã cho.

Dữ liệu: Vào từ file văn bản có tên DIV.IN:
Dòng đầu tiên chứa hai số nguyên N và K (1 N 10000, 2 K 100) ghi cách nhau dấu trắng.
Các dòng tiếp theo ghi các số hạng của dãy số đã cho, mỗi số hạng của dãy số có giá trị tuyệt đối không quá 10000 được ghi cách nhau bởi dấu trắng hoặc dấu xuống dòng.

Kết qủa: Ghi ra file DIV.OUT số 1 nếu dãy đã cho chia hết cho K và số 0 nếu ngược lại.

Ghi chú: Bạn chỉ được điểm nếu như trả lời đúng cả câu khẳng định lẫn phủ định.

Ví dụ:

DIV.IN	DIV.OUT	DIV.IN	DIV.OUT
4 7	1	4 5	0
17 5 -21 15		17 5 -21 15	

Hướng Dẫn :

Chúng ta sẽ dùng một tập hợp lu lại các giá trị của số d của tổng và hiệu của các số mới đọc với các số đã có trong tập hợp đó . Cứ một lần đọc một số nào đó thì ta sẽ cập nhật lại số d đó . Nếu cuối cùng mà 0 có trong tập hợp đó thì bài toán có nghiệm . Nhưng nếu không thì sẽ không tồn tại cách điền các dấu + , - nào thoả mãn chia hết .

Các bạn có thể xem chương trình ở lời giải mẫu (trong phần test) để hiểu thêm bài toán .

Mở rộng :

Hãy điền các dấu cộng và trừ trước các số của một dãy nào đó (số chữ số có thể lên đến 10000) để kết quả là một số nguyên cho trước không . Nếu có thì hãy tìm cách điền .

Chúng ta có thể giải bài toán đó như sau :

Giả sử cách đặt dấu - , + tương ứng với các hệ số -1 và 1 ta gọi hệ số của số thứ i là Xi thì ta sẽ có :

A1 * X1 + A2*X2 + ...+An*Xn=M

Tức là giả sử tập các số có hệ số -1 là : T1 , T2,..Tk . thì ta có :

M := S - 2 (T1+T2+..Tk) . Trong đó S:=A1+A2+..An .

Tức là ta sẽ đa về bài toán tìm các số trong n số đã cho có tổng là : (S-M)/2 . Bài toán này có thể được giải bằng phương pháp quy hoạch động . Các bạn có thể giải quyết một cách đơn giản . (Giống như bài toán chia kẹo mà thôi) .

Bài toán 19 : Cân Đòng

Đề bài :

Có N quả cân với các trọng lợng tương ứng là 1kg, 3kg, . . ., 3N-1kg và một cân bàn. Các quả này được đánh theo số hiệu : 1 , 2, 3, .. N . Ta muốn chỉ dùng cân bàn và N quả cân này để cân túi đường có trọng lợng M kg trong một lần cân. Liệu ta có thể cân được không.

Dữ liệu : Cho trong file văn bản Duong.INP :

Ghi trên một dòng duy nhất hai số N <=15 , và M<=100000000

Kết quả : Ghi ra file văn bản Duong.Out :

Nếu không thể cân được thì dòng đầu tiên viết : NO còn nếu cân được thì ghi dòng đầu tiên ghi YES và tiếp tục bớc sau

Dòng đầu tiên ghi số hiệu các quả cân nếu có ở phía chứa vật (nếu không có quả cân nào thì để trống)

Dòng tiếp ghi số hiệu các quả cân ở phía không có vật .

Ví dụ :

Canduong.Inp	CanDuong.Out
7 255	YES 6 3 2

Thuật toán :

Ta quy định bên chứa vật là (I) còn bên kia là (II) .

Chúng ta thấy đây chỉ là việc chuyển một số nào đó sang hệ cơ số ba mà thôi . Nhưng mỗi một mũ ba chỉ có thể có hệ số 1 hoặc 0 :

giả sử số X khi phân tích ra cách cân thì nó có hai bên :

X+3i1+3i2+..3in=3j1+3j2+..+3jm .

Tức là : X:= (3j1+3j2+..+3jm)-(3i1+3i2+..3in) . (*)

Giả sử số X trong hệ cơ số 3 là :

X:=A1*31+A2*32+...An*3n . (**)

với Ai=0,1,2.

Để chuyển biểu thức (**) sang biểu thức (*) thì chúng ta thấy :

Nếu Ai=0 thì quả cân thứ i không tham gia trong quá trình cân

Nếu Ai=1 thì để nguyên tức là quả cân thứ i sẽ nằm phía (II .)

Nếu Ai=2 thì ta có : 2*3i=3i+1-3i . tức là chúng ta sẽ sử dụng quả thứ i về bên thứ (I) còn đối với quả cân thứ i+1 thì chúng ta sẽ tăng hệ số Ai+1 :=Ai+1+1 ; rồi tiếp tục xét với hệ số của i+1

Nếu Ai =3 (do trường hợp vừa rồi tạo nên) thì chúng ta có :

3*3i=3i+1 tức là quả cân thứ i không tham gia trong quá trình cân , mặt khác chúng ta tăng hệ số của Ai+1 lên một đơn vị .

Cứ như vậy chúng ta sẽ biết được quả cân sẽ nằm bên nào hoặc không tham gia cân . Nhưng chúng ta nên nhớ rằng nếu số X>31+32+..+3n = (3n+1-1)/2 thì không tồn tại cách cân .

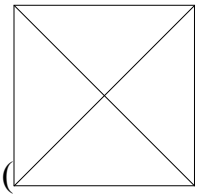
Bài toán 20 : DÃY CON LỚN NHẤT

Đề Bài :

Trên mỗi vòng tròn ngời ta đánh dấu N vị trí. Các vị trí được đánh số thứ tự từ 1 đến N theo chiều kim đồng hồ. Tại vị trí thứ i ngời ta ghi số nguyên ai, i = 1, 2, ... N. Cần tìm cách chọn ra dãy số con gồm một số số liên tiếp nhau trên vòng tròn có tổng các số hạng là lớn nhất.

Dữ liệu: Vào từ file văn bản DAY.INP.

Dòng đầu tiên ghi số nguyên dương N (N 10);



Dòng thứ i trong số N dòng tiếp theo ghi số ai (10000), i = 1, 2, ... N.

Kết quả: Ghi ra file văn bản DAY.OUP.

Dòng đầu tiên ghi K là số lợng phần tử của dãy con được chọn.

Dòng thứ j trong số K dòng tiếp theo ghi số thứ tự trên vòng tròn của số hạng thứ j của dãy con được chọn.

Ví dụ:

DAY.INP	DAY.OUT	DAY.INP	DAY.OUT
7	6	11	3
2	3	- 10	8
- 4	4	- 1	9
1	5	- 3	10
7	6	5	
4	7	6	
6	1	- 4	
- 1		- 9	
		20	

- 5
6
- 3

Hướng Dẫn :

Bài to án này có một đặc biệt đó là các số nằm trên một vòng tròn .

Gọi S1 là tổng các số trong dãy . S2 là tổng các số nguyên liên tiếp trong dãy đó (không kể vòng) là nhỏ nhất , S3 là tổng các số nguyên liên tiếp trong dãy (không kể vòng) là lớn nhất . Ta sẽ có dãy số vòng thoả mãn sẽ có tổng bằng :

S:=Max S3 , S1-S3

Sau đó dựa vào đó ta sẽ lu được dãy số thoả mãn tổng lớn nhất . Các bạn có thể xem thêm ở lời giải trong bộ Test kèm theo .

Bài toán 21 : Hệ cơ số -2

Đề bài :

Chúng ta biết rằng trong hệ cơ số 2 được ứng dụng rất nhiều . Các nhà khoa học cũng đã tìm rất nhiều hệ cơ số để làm cho ngôn ngữ máy tính trở nên thuân lợi hơn . Vì chỉ toàn các hệ số 0 , 1 nên hệ nhị phân đã được ứng dụng vào máy tính . Thế nhưng chúng ta cũng thấy rằng hệ cơ số -2 cũng chỉ có hệ số là 0 , 1 . Nhưng công việc giải quyết cộng trừ trong đó rất khó khăn . Chính vì thế các bạn hãy giải quyết giúp chúng tôi bài toán cộng các số trong hệ cơ số -2 .

Biết rằng một số X phân tích trong hệ cơ số -2 :

X=An*(-2)n+An-1*(-2)n-1 +...A1*(-2)1+A0*(-2)0 . Thì bộ số :

An,An-1,...A0 được gọi là số dạng cơ số -2 của X .

Yêu cầu : Cho hai số ở dạng cơ số -2 . Hãy tính tổng của nó trong hệ cơ số -2 (tức là hiệu và tổng cũng phải biểu diễn trong hệ cơ số -2)

Dữ liệu : Cho từ file : Cosotru2.Inp :

Hai dòng , mỗi dòng gồm các số 0 hoặc 1 (biểu diễn hệ cơ số trừ hai của hai số cần thực hiện) . Biết rằng số chữ số của nó không vợt quá 200 .

Kết quả : Cho ra file Cosotru2.Out :

Một dòng là tổng (đều biểu diễn dưới dạng -2)

Ví Dụ :

COSOTRU2.INP	COSOTRU2.OUT
110 10110	10100

Hướng Dẫn :

Chúng ta gọi công thức nhị phân của hai số đó là :

An-1....A1A0 và Bn-1...B1B0 .

Công thức cộng :

Gọi mảng C[i] là : C[i]=A[i]+B[i] .

Lúc đó ta có với mỗi C[i] :

* Nếu C[i] = 0 hoặc 1 thì để nguyên
* Nếu C[i] = 2 thì : giá trị tại đó =2 *(-2)i = (-2)i+2 +(-2)i+1 lúc đó ta sẽ làm như sau : C[i+1]:=C[i+1]+1 ; C[i+2]:=C[i+2]+1 ; và C[i]:=0 ;
* Nếu C[i]= 3 thì : giá trị tại đó = 3*(-2)i =(-2)i+2+(-2)i+1+(-2)i .lúc đó ta sẽ làm như sau : c[i+1]:=C[i+1] ; C[i]:=1 ;C[i+2]:=C[i+2]+1 ;
* Nếu c[i]=4 thì : giá trị tại đó = 4*(-2)i=(-2)i+2 , lúc đó thì C[i]:=0 ; C[i+2]:=c[i+2]+1 ;

Cứ thực hiện cho đến hết thì mảng C cuối cùng chính là hệ cơ số -2 của tổng chúng .

Mở rộng : *Các bạn hãy giải phép trừ trong hệ cơ số -2*

Bài toán 22 : Match

Đề Bài :

Một nhà máy sản xuất động cơ xe máy sản xuất được N pittông và N xilanh . Các pittông được đánh số từ 1 đến N , các xilanh cũng được đánh số từ 1 đến N . Mỗi pittông và mỗi xilanh đều được gán với một chỉ số chất lợng là một số nguyên . Gọi chỉ số chất lợng của pittông i là ai , i=1,2,..N, còn chỉ số chất lợng của xilanh j là bj , j=1,2..N . Nếu lắp ráp pittông i với xilanh j thì ta được một bộ pittông-xilanh hoàn chỉnh có đánh giá chất lợng là ai x bj.

Yêu cầu :

Hãy giúp nhà máy lắp rắp K bộ pittông-xilanh hoàn chỉnh với giá chất lợng là lớn nhất .

Dữ liệu : Vào từ file văn bản Match.Inp :

Dòng đầu tiên ghi hai số nguyên dương N và K (N<=107 , K<=N và K<=103)

Dòng thứ i trong số n dòng tiếp theo chứa hai số nguyên ai , bj được ghi cách nhau bởi dấu cách , ai<=32767 , bj<=32767 , i =1,2,..N

Kết quả : Ghi ra file văn bản Match.Out :

Dòng đầu tiên chứa tổng đánh giá chất lợng của K bộ pittông-xilanh hoàn chỉnh tìm được

Mỗi dòng trong số K dòng tiếp theo chứa hai chỉ số p , q trong đó p là chỉ số của pittông còn q là chỉ s của xilanh của một bộ pittông-xilanh hoàn chỉnh trong số K bộ pittông-xilanh cần lắp rắp

Ví dụ :

MATCH.INP	MATCH.OUT
6 2 1 5 3 3 4 6 1 9 -2 4 2 -3	54 3 4 2 3

Hướng Dẫn :

Chúng ta cần tìm K cặp sao cho chúng có tích lớn nhất . Nhưng chúng ta phải tính đến trường hợp một số âm nhân với một số âm thì cho kết quả là một số dương .

Thuật toán :

Bớc 1 : Sắp xếp dãy Ai tăng dần , gọi là dãy A1i

Bớc 2 : Sắp xếp dãy Ai giảm dần , gọi là dãy A2i

Bớc 3 : Sắp xếp dãy Bi tăng dần , gọi là dãy B1i

Bớc 4 : Sắp xếp dãy Bi giảm dần , gọi là dãy B2i

Bớc 5 : Với mỗi cặp trên cùng của tích Ai1[top1]*Bi1[top2] với Ai2[last1]*Bi2[last2] thì cặp nào lớn hơn thì lấy nó ra tạo thành một cặp , và loại các số này ra khỏi các bảng chứa chúng .

Cứ tiếp tục thực hiện bớc 5 cho đến khi lấy được K cặp .

Bài toán 23 : Dãy Catalan

Đề bài :

Cho số nguyên dương N (N <= 15) ,dãy Catalan là dãy C1 , C2 … C2n+1 gồm các số nguyên không âm thoả mãn : C(1) = C(2n+1) = 0 với i bất kì 1 <= i <= 2n thì C(i) , C(i+1) hớn kém nhau 1 đơn vị .

Với mỗi n ta sắp xếp các dãy Catalan theo thú tự từ điển , dánh số từ 1 trở đi . Yêu cầu :

1Cho một dãy Catalan , hãy tìm thứ tự của dãy.

2.Cho só nguyên dương k hãy tìm dãy có thứ tự k

Dữ liệu : Vào từ file CATALAN.INP

Dòng đầu ghi n .

Dòng hai ghi một dãy Catalan cấp n

Dòng 3 ghi một số nguyên dương k (k có thể rất lớn nhưng đảm bảo luôn có nghiệm)

Kết Quả : Ghi ra file CATALAN.OUT

Dòng 1 ghi số thứ tự dãy ở dòng 2 Input

Dòng 2 ghi dãy ứng với số thứ tự

Hướng dẫn :

Ta xây dựng bảng số n+1 * n+1 đánh số (0..n , 0..n), ứng với mỗi dáy Catalan , bắt đầu ở vị trí 0 , 0 của bảng ta lớt qua các cặp ci , ci+1 cho i chạy từ 2n về 1 của dãy C , nếu ở ci - 1 = ci+1 tăng 1 thì ở bảng ta đi xuống , ngược lại thì ta sang phải . Cối cùng bao giờ ta cùng đếnvị trí n,n của bảng .

Ví dụ dãy 0 1 2 3 2 1 2 1 0 (n = 4) ứng với đường đi :

Xuống 1 0
Xuống 2 0
Ngang 2 1 Xuống 3 1
Xuống 4 1
Ngang 4 2 Ngang 4 3 Ngang 4 4 Tới được 4 , 4

Với mỗi vị trí i , j của bảng , ta gọi T(i,j) là số các dãy đến được từ 0 , 0 đến (i,j) , và ta tính bằng truy hỗi : T(i,j) = T(i-1,j) + T(i,j-1) .Ta được bảng như sau :

Hàng Cột	0	1	2	3	4	5
0	1	0	0	0	0	0
1	1	1	0	0	0	0
2	1	2	2	0	0	0
3	1	3	5	5	0	0
4	1	4	9	14	14	0
5	1	5	14	28	42	42

Dãy C tương tứng một dãy gồm xuống và ngang. Ta tính từ n , n về 0 , 0 , dùng biến s lu tổng số dãy nhỏ hơn nó khởi tạo = 0. Ta xét các điểm trên đường đi :
Tại i,j nếu đường đi tương ứng qua i,j về 0,0 nếu là sang phải thì ta cộng thêm biến vào T(i,j-1).
Cuối cùng s là tổng só các dãy nhỏ hớn dãy C
Ví dụ : dãy 0 1 2 3 2 1 2 1 0 đường dãy đường đi về 0 , 0 là
(4,4)
(4,3) cộng T(3,4) = 0
(4,2) cộng T(3,3) = 5
(4,1) cộng T(3,2) = 5
(3,1)
(3,0) cộng T(2,1) = 2
(2,0)
(1,0)
(0,0)

vậy s = 5 + 5 + 2 = 12 là số các dãy nhỏ hơn . Số thứ tự là s+1 = 13.
Tương tự , áp dụng cách tìm thứ tự ta có thuật toán ngược lại tìm dãy từ số thứ tự

III. Xử lí Bit

A. Các kiến thức về Bit :

Mỗi Bit bao gồm 8 bit được mã số từ phải sang trái còn gọi là từ bit thấp đến bit cao . Bit nằm ở bên phải được xem là thấp hơn bit nằm ở bên trái . Các bít có mã số như sau:

7,6,5,4,3,2,1,0

Mỗi Bit có thể nhận 2 giá trị 0, 1 .

Các phép toán sau đây thực hiện trên các giá trị nguyên và cho kết quả nguyên :

1. Phép đảo bit Not :

Đổi giá trị của mọi bit t 0 thành 1 và ngược lại .

Ví Dụ:

```
var x,y:Integer;
Begin
    x:=19; ( x=00010011)
    y:= not x ;( 11101100=26)
End;
```

2. Phép cộng logic trên các Bit (Or) :

Thực hiện trên từng cặp Bit Tương ứng của các toán hạng theo bảng cộng :

Bit a	Bit b	a Or b
0	0	0
1	1	1
0	1	1
1	0	1

Quy Tắc: tổng hai Bit bằng 0 khi và chỉ khi cả hai Bit bằng 0 ,ngoài ra tổng nhận giá trị 1.

3. Phép nhân logic trên các Bit (And) :

Thực hiện trên từng cặp Bit tương ứng của các toán hạng theo bảng sau đây :

Bit a	Bit b	a And b
0	0	0
0	1	0
1	0	0
1	1	1

Quy Tắc : Tích hai Bit bằng 1 khi và chỉ khi cả hia Bit bằng 1 ,ngoài ra tích nhận giá trị 0.

4. Phép cộng loại trừ trên các Bit (Xor):

Thực hiện trên từng cặp Bit tương ứng của các toán hạng theo bảng sau:

Bit a	Bit b	a Xor b

0	0	0
0	1	1
1	0	1
1	1	0

Quy Tắc: tổng loại trừ của hai Bit bằng 1 khi và chỉ khi hai Bit đó chứa các giá trị khác nhau ,ngoài ra tổng loại trừ nhận giá trị 0

5. Phép dịch sang phải (Shr) x

Shr i cho giá trị nhận được từ số nguyên x sau khi dịch chuyển số đó qua phải i Bit

Quy Tắc: Muốn tìm d của phép chia nguyên một số nguyên cho 2 ta dịch số đó qua phải 1 Bit .Tổng quát ,muốn chia một số nguyên cho 2 (Mũ i) ta dịch số đó qua phải i Bit.

x Shr i=x Div n , Với N=2 Mũ i.

Quy Tắc : Muốn tìm d của phép chia nguyên một số nguyên cho số n =2 mũ i ta nhân logic số đó với n-1.

x Mod n =x And(n-1) Với n=(2 mũ i)

6. Phép dịch sang trái (Sh l) x Shl i co giá trị nhận được từ số nguyên x sau khi dịch số đó qua trái i Bit

Quy Tắc : Muốn nhân một số nguyên với (2 mũ i) ta dịch số đó qua trái I Bit :

x Shl i=x*n (n= 2 mũ i)

7. Các Hàm và thủ tục tác động lên Bit :

```
Function GetBit(X,I:Word ):Byte;
Begin
        GetBit:=(X Shr i) And i;
End;
```

Hàm Sizeof(x) Cho ta số Byte trong x .

```
Procedure Bprint(x:word);
var i:Byte;
Begin
        For I:=(Sizeof(x) shl 3) -1  Downto 0 do
        write(GetBit(x,i));
End;

Procedure BatBit(Var X:Byte;i:Byte);
Begin
        X:=X Or ( 1 Shl I);
```

```
End;

Procedure TatBit(Var X:Byte;I:Byte);
Begin
        X:=Not(1 Shl I) And X;
End;
```

B. Bài toán :

Bài toán 24 : Firstrow

Đề Bài :

Cho một văn bản gồm N dòng (1<=N<=50000) mỗi dòng gồm không quá 255 ký tự . Trong file văn bản đã cho có một dòng đặc biệt chỉ xuất hiện trong văn bản đúng một lần , còn các dòng khác đều bị lặp lại và hơn thế nữa lại là một số lần chẵn . Bạn cần tìm dòng thông tin đặc biệt này

Dữ liệu : Vào từ file Firstrow.Inp chứa các dòng của văn bản đã cho , trong đó dòng cuối cùng của file có dấu # đánh dấu kết thúc của văn bản và dòng này không được tính vào văn bản

Kết quả : Ghi trên một dòng của file văn bản Firstrow.Out dòng đặc biệt tìm được

Ví dụ :

FIRSTROW.INP FIRSTROW.OUT

I Love You Hoa_Tra_My I Love You I Hate You Love forever I Hate You Love forever # Hoa_Tra_My

Hướng dẫn :

Chúng ta sử dụng phép Xor trong bit , tức là phép toán này sẽ khử đi những giá trị mà nó lặp lại những lần chẵn . Còn những giá trị lặp lại một số lần lẻ thì nó sẽ tồn tại cho đến cùng . Chúng ta quy đổi một xâu làm từng kí tự một , và giá trị từng kí tự được tính bởi vị trí nó trong mã Acsii .

Chương trình :

```
Program Firstrow;
Uses crt;
Const
    fi='firstrow.inp';
    Fo='firstrow.out';
    Maxn=255;
Var
  a:array[1..maxn] of byte;
```

```
  i,j:integer;
  s:string;
  f1,f2:text;

Procedure Init;
Begin
    Assign(f2,fo);
    Rewrite(f2);

    Assign(f1,fi);
    Reset(f1);
End;

Procedure Main;
Begin
    Fillchar(a,sizeof(a),0);

    While not eof(f1) do
    Begin
        readln(f1,s);
        If s='#' then break;

        For i:=1 to length(s) do
            a[i]:=a[i] xor ord(s[i]);
    End;
End;

Procedure Done;
Begin
    For i:=1 to maxn do
       if a[i]<>0 then
        write(f2,chr(a[i]));

    Close(f1);
    Close(f2);
End;

BEGIN
    Clrscr;
    Init;
    Main;
    Done;
```

END.

Bài toán 25 : Numberself

Đề Bài :

Cho một số N , ngời ta gọi D(n) là N+ tổng các chữ số của n . Ví dụ d(33)=33+3+3=39 . N được gọi là Generator của D(n) .

Một số mà không có Generator được gọi là Self-Number . Nghĩa là : m là Self-Number nếu không tồn tại số n để D(n)=m . Ví dụ số 97 là một Self-Number .

Yêu Cầu : hãy đa ra tất cả các số self-number nằm trong khoảng [a,b] .

Dữ liệu : Cho từ file Numself.Inp :

Gồm 1 dòng ghi hai số a và b

Kết quả : Ghi ra file Numself.Out :

Dòng đầu tiên ghi số các số number-self trong khoảng [a,b]

Tiếp theo ghi các số đó .

Ví dụ :

Numself.Inp	Numself.Out
1 100	13 1 3 5 7 9 20 31
42 53 64 75 86 97	

Hướng Dẫn :

Ta có D(X) sẽ nằm trong khoảng [X,X+45] vì số chữ số cao lắm là 5 . Cho nên D(X) <X+9*5 ;

Một cách đơn giản là chúng ta sẽ cho giá trị i chạy từ a -45 đến b . với mỗi lần chạy thì chúng ta tìm giá trị D(N) của chúng và đánh dẫu mảng này . Sau đó làm ngược lại một lần nữa thì chúng ta sẽ tìm những D(n)chưa được đánh dấu thì sẽ lấy ra .

Sau đây là một cách dùng mảng đánh dấu hiệu quả với dữ liệu lớn . Đó là chúng ta dựa vào việc trạng thái của bit (còn thông thường thì ta dùng trạng thái byte : True /False) . Mà một byte lại có 8 bit , tức là dữ liệu tối đa là : 64000*8 =512000 .

DD:Array[0..64000] of byte ;

fillchar (dd , sizeof (dd) , 0) ; (* bật bit tất cả về trạng tháichưa có *)

```
procedure Bat_bit(t:longint);
var  i,j,tg,k:longint;
begin
  k:=t; tg:=t;
  repeat
     k :=k + tg mod 10; tg := tg div 10;
```

```
  until tg=0;
  i:=k div 8;
  j:=k mod 8;
  dd[i]:=dd[i] or (1 shl j);
end;
```

Còn công việc nhận dạng cuối cùng thì chỉ cần dùng hàm lấy bit thì sẽ cho biết được bit đó đã được bật haychưa mà thôi .

Bài toán 26 : Dãy nhị phân

Đề bài :

Một tập hợp S gồm các dãy N bit 0, 1 trong đó không có hai bit 1 nào kề nhau. Ví dụ với N = 5 thì S gồm các dãy 00000, 00001, 000101, …. Tập S được săp xếp theo chiều tăng dần của số nguyên tương ứng mà dãy bit bieu dien. Cho số N và một số nguyên M hãy cho biết dãy bit thứ M trong S.

Dữ Liệu :

N, M (N <= 40, M đảm bảo có nghiệm)

Kết Quả :

Dãy N số 0, 1 ghi liền nhau mô tả dãy nhị phân tìm được.

Hướng Dẫn :

Gọi C(i) là số các phần tử của tập S ứng với N = i.

Công thức tính truy hồi C(i) = C(i-1) + C(i-2)

C(0) = 1; C(1) = 1;

Thuật toán xây dng dãy M nhưsau:

```
For i := 1 to N do
 Begin
   If  M  > C(N-i) then
     Begin
       Bit[i] = 1;
        M = M - C(N-i);
     End
    Else Bit[i] = 0;
 End;
```

Bài toán sau đây là bài toán tiếp theo của hướng phát triển bài toán :

Bài toán 27 : Liên lạc vũ trụ

Đề bài :

Để liên lạc với tàu thăm dò tự động ngời ta chuẩn bị danh sách các thông báo , đánh số từ 1 trở đi và cài vào trong bộ nhớ của máy tính trên trạm thăm dò . Số lợng thông báo là 1000000. Trạm điều khiển mặt đất

hoặc tàu thăm dò chỉ cần phát đi số tứ tự thay vì cho chuyển bằng hệ thống phát xung Laser định hướng . Nhưng việc phát xung (tức là các tín hiệu 1) . Vì vậy các nhà khoa học quyết định phát mỗi số ứng với một dãy bit có không quá 3 số 1 . Các dãy bit có cùng độ dài là 200 , được sắp xếp lại theo thứ tự tăng dần của giá trị nhị phân tương ứng . Số thứ tự của dãy bit trong danh sách sẽ chính là số nguyên cần gửi .

Hãy lập trình cài vào máy phát chuyển đổi từ giá trị số sang xâu bit cần phát .

Dữ liệu : Vào từ file Impulse.Inp :

Dòng đầu tiên là số lợng thông báo cần phát R (R<=10000)

Các dòng sau chứa các số nguyên dương (số thứ tự thông báo) , các số nếu ở trên 1 dòng - cách nhau ít nhất 1 dấu cách .

Kết quả : Đa ra file Impulse.Out : R dòng , mỗi dòng chứa một xâu bit ứng với số cần phát . Bỏ qua các số 0 trước 1 đầu tiên trong xâu , trừ trường hợp số cần phát là 1 thì kết quả ra được ghi là 0

Ví Dụ :

IMPULSE.INP	IMPULSE.OUT
6 1 2 3 4 5 100	0 1 10 11 100 100000110

Hướng Dẫn :

Chương trình :

```
{$A+,B-,D+,E+,F-,G-,I+,L+,N-,O-,P-,Q+,R+,S+,T-,V+,X+,Y+}
{$M 65520,0,655360}
PROGRAM Space_Communication;
 Const
  InpName      = 'IMPULSE.INP';
  OutName      = 'IMPULSE.OUT';
  MaxLength    =         1000;
  MaxBits      =         3;
 Var
  F, G         : Text;
  R, j, L,
  N, i, k, k1  : LongInt;
  Mess         : Array [1..MaxLength] Of Byte;
  C, Px        : Array [0..MaxLength, 0..MaxBits] Of LongInt;

 Procedure PreCalc;
  Begin
   Fillchar(C, SizeOf(C), 0);
   For i := 1 to MaxLength do
    C[i][1] := i;
   For i := 1 to MaxLength do
    Begin
```

```
    For k := 2 to MaxBits do
     C[i][k] := C[i-1][k] + C[i-1][k-1];
   End;
  For i := 1 to MaxLength do
   C[i][0] := 1;

  For k := 1 to MaxBits do Px[1][k] := 2;
  For i := 2 to MaxLength do
   For k := 1 to MaxBits do
    For j := 0 to k-1 do Inc(Px[i][k], C[i-1][j]);
 End;

Procedure BuildMessage;
 Begin
  Fillchar(Mess, SizeOf(Mess), 0);
  L := 1;
  k := MaxBits;
  Repeat
   If N = 2 Then
    Begin
     Mess[1] := 1;
     Break;
    End Else
   If N = 1 Then Break;
   i := 1;
   While N > Px[i][k] do
    Begin
     Dec(N, Px[i][k]);
     Inc(i);
    End;
   Mess[i] := 1;
   Dec(k);
   If L < i Then L := i;
  Until FALSE;
 End;

Procedure SendMessage;
 Begin
  For i := L downto 1 do Write(G, Mess[i]);
  Writeln(G);
 End;
```

```
BEGIN
 PreCalc;
 Assign(F, InpName); Reset(F);
 Assign(G, OutName); ReWrite(G);
 Readln(F, R);
 For j := 1 to R do
  Begin
   Readln(F, N);
   BuildMessage;
   SendMessage;
  End;
 Close(F); Close(G);
END.
```

Bài toán 28 : Mã Vạch

Đề Bài :

Một mã vạch (thường dùng để ghi giá trị hàng hoá)kà một dãy các vạch đen trắng xen kẽ nhau bắt đầu bằng vạch đen bên trái , mỗi vạchcó một độ rộng tính bằng số nguyên dương . Hình sau cho ví dụ về một mã 4 vạch trải trên 1+2+3+1 = 7 đơn vị rộng

1 2 3 4 5 6 7

1 2 3 4

Tổng quát , một tập mã vạch Mv(n,k,m) là tập mọi mã vạch gồm k vạch trải trên đúng n đơn vị rộng và mỗi vạch không rộng quá m đơn vị . Mã vạch trong ví dụ trên thuộc tập Mv(7,4,3) nhưng không thuộc tập Mv(7,5,2) . ta có thể biểu diễn mỗi mã vạch bởi một dãy nhị phân bằng cách dùng số 1biểu thị một đơn vị rộng màu đen và số 0 biểu thị một đơn vị rộng mầu trắng . Ví dụ mã vạch trong hình trên sẽ được biểu thị bởi dãy 1001110

Với biểu diễn nhị phân như vậy mỗi tập Mv(n,k,m) với các giá trị cụ thể của m , k , n là một tập các xâu độ dài n chỉ gồm các ký tự 0 hay 1 . Ta có thể liệtt kê tập đó theo thứ tự từ điển với quy ớc ký tự 0 trước ký tự 1 và theo thứ tự liệt kê đó ta cho mỗi mã vạch một số hiệu .

Ví dụ tập Mv(7,4,3) gồm 16 mã vạch sẽ được biểu diễn bởi 16 xâu nhị phân độ dài 7 và theo cách sắp xếp từ điển của các biểu diễn nhị phân ta có các mã vạch với các số hiệu từ 01 đến 16 là :

01: 1 0 0 0 1 0 0	09: 1 1 0 0 1 0 0
02: 1 0 0 0 1 1 0	10: 1 1 0 0 1 1 0
03: 1 0 0 1 0 0 0	11: 1 1 0 1 0 0 0
04: 1 0 0 1 1 0 0	12: 1 1 0 1 1 0 0
05: 1 0 0 1 1 1 0	13: 1 1 0 1 1 1 0
06: 1 0 1 1 0 0 0	14: 1 1 1 0 0 1 0
07: 1 0 1 1 1 0 0	15: 1 1 1 0 1 0 0
08: 1 1 0 0 0 1 0	16: 1 1 1 0 1 1 0

Dữ liệu : Nhập dữ liệu từ file BARCODE.INP với cấu trúc như sau :

Dòng thứ nhất ghi 3 số n , k , m (1<=n,k,m<=33) là ba tham số của tập mã vạch

Dòng thứ hai ghi số s (s<=50) trong s dòng tiếp theo ghi mỗi dòng một xâu gồm n ký tự 0 hay 1 là biểu nhị phân của một mã vạch thuộc tập Mv(n,k,m) .

Kết quả : xuất ra file BARCODE.OUT : S dòng , mỗi dòng ghi số hiệu của mã vạch đã cho

Ví Dụ :

BARCODE.INP	BARCODE.OUT
7 4 3	16
5	5
1001110	16
1110110	4
1001100	5
1001110	1
1000100	

Hướng Dẫn :

Gọi Mv(n,k,m) là số mã vạch có thể có của bộ (n,k,m) thì ta có công thức truy hồi như sau :

Mv(n,k,m)=(Mv(N-i,K-i,M) , với i = 1 ,..M

Trong đó Mv(0,0,m)=1 ; Mv(i,0,m)=0 , i=1,..n .

Dựa vào đó ta có thể tính số hiệu mã vạch như sau :

gọi mã vạch đó là : (x1,x2,..xk) . ta xét với t =1,..k thì gọi F là số hiệu của mã vạch đó thì (đầu tiên F=0) ta có với mỗi t :

T chẵn :

F:=F+Mv(N-(Xv)-i,K-t,M) với i = Xt+1,..M và v=1..t-1 .

T lẻ

F :=F+Mv(N-(Xv)-i,K-t,M) với i = 1 ..Vt-1 , v = 1..t-1 .

Bài toán 9 : Có thể được giả bằng Bit .

Các ứng dụng của các phép toán bit là rất lớn . Nó không đứng riêng biệt trong các bài toán . mà nó là một bộ phận khiến cho chương trình của bài toán trở nên đơn giản hơn , ngắn gọn hơn và đặc biệt nhanh hơn rất nhiều .Phần này chỉ nhằm nêu ra các ứng dụng phép toán bit chứ không phải là giải các bài toán có liên quan đến bit .

IV. Bài toán Bảng .

Bài toán 29 : Ma Phương Bậc Lẻ

Đề Bài :

Cho một bộ số gồm N*N số (N lẻ), trong đó các số đều nguyên dương . Một ma trận được gọi là một ma phương , nếu như tổng các số trên các hàng và các cột , đường cháo đều bằng nhau . Với một bộ số cho trước , biết rằng có thể điền được vào ma trận N*N để thành một ma phương . Bạn hãy điền vào ma trận N*N để ma trận đó thành một ma phương .

Dữ liệu Vàotừ file : MAPHUONG.INP gồm

dòng đầu ghi số N là kích thước ma trận và có nghĩa là bộ số sẽ có N*N số

dòng hai ghi N*N số biểu diễn bộ số đó . (N<=100)

Dữ liệu: Ra file MAPHUONG.OUT

gồm N dòng , mỗi dòng gồm N số , biểu diễn ma trận ấy .

biết rằng mọi bộ dữ liệu cho luôn thoả mãn điền được thành một ma phương . Và chương trình của bạn phải chạy không quá 5 giây , Kết quả ma phương chỉ cần đa ra một nếu có nhiều cách điền .

Các số ghi cách nhau một dấu cách .

Ví dụ :

MAPHUONG.INP	MAPHUONG.OUT
7	1 5 6 7 2 3 4
10 23 22 11 12 13 14	15 20 16 19 25 28 24
29 30 32 31 37 40 42	43 47 60 58 61 52 55
57 59 56 49 41 48 46	50 51 34 38 39 33 21
	28 59 20 51 12 43 4
	5 29 60 21 52 13 37
	38 6 30 61 22 46 14
	15 39 7 31 55 23 47
	48 16 40 1 32 56 24
	25 49 10 41 2 33 57
	58 19 50 11 42 3 34

Hướng Dẫn :

Trước tiên chúng ta giải quyết bài toán ma phương như sau :

Với một ma phương bậc chẵn thì hiện naychưa có một cách giải hữu hiệu . Nhưng với một ma phương bậc lẻ thì chúng ta có rất nhiều

thuật toán . Sau đây là một thuật toán điền mà tôi xuất phát từ một kết quả có trong báo TH&TT (số 260) . Thuật toán đó là cách kẻ các ô phụ :

Với một ma trận N*N thì chúng ta kẻ thêm các ô phụ bên ngoài . Tạo thành một ma trận mới mà có kích thước (N+N div 2)*(N+N div 2) . trong đó các ô của ma phương là phần từ N div 2+1 đến N+N div 2 . xuất phát từ ô (1,N) ta điền các số bắt đầu từ 1 đến N*N và điền theo chiều hình chéo (hay cạnh của một hình vuông có kích thước N*N nhưng là hình vuông xiên có một đỉnh là ô (1,n)) . Sau đó ta đa các số ở ngoài ma trận cần xây dựng vào , còn các ô trong ma trận cần xây dựng thì vẫn dữ nguyên . Ta đa vào theo nguyên tắc sau : giả sử ô (i,j) ngoài ma trận thì ta đa nó đến ô theo quy tắc sau đây

nếu i<= N div 2 thì đến ô (i+N div 2,j)
nếu i>=N div 2+ N thì đến ô (i-N div 2,j)
nếu j<=N div 2 thì đến ô (i,j+N div 2)
nếu j>=N+ N div 2 thì đến ô (i,j-N div 2);

Như vậy ma trận ta cần điền chính là các số của ma trận mà ta vừa xây dựng nhưng nó là hình vuông có bốn đỉnh là : (N div 2 +1, N div 2+1) , (N div 2+1,N div 2+ N) , (N div 2+ N , N div 2+ 1) , (N div 2+N , N div 2 + N) .

Chúng ta có thể ví dụ : Với N=3 thì : lần lợt theo các giai đoạn :

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

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

Hay với N=5 ta có :

```
              1
          6       2
      11      7       3
  16      12      8       4
21      17      13      9       5
  22      18      14      10
```

23 19 15
24 20
25

11	24	7	20	3
4	12	25	8	16
5	17	9	13	21
10	18	1	14	22
23	6	19	2	15

Chắc bạn có thể tự viết được chương trình xây dựng ma phương trên , vì thuật toán trên khá đơn giản và dễ thực hiện .

Nhưng cũng từ cách điền vào ma trận này chúng ta hình thành một cách điền một bộ dữ liệu cho trước vào một ma trận (bậc lẻ và bộ dữ liệu trên phải đảm bảo tồn tại một cách điền được) thoả mãn tính chất của một ma phương . Thuật toán trên cũng giả quyết được cách điền đó . chúng ta làm như sau : đầu tiên phải sắp xếp bộ dữ liệu đó tăng dần . Sau đó điền theo tần tự như thuật toán đã nêu ở trên . Đó chính là thuật giải của bài toán trên .

Bài Toán 30 : Điền vào ma trận

Đề Bài :

1	1	3	5	1
3	3	2	0	3
3	0	3	2	3
1	4	0	3	3
3	3	3	1	1

Cho hình vẽ trên là một hình vuông 5x5 ô vuông nhỏ . Trên mỗi ô ghi một chữ số tự nhiên như hình vẽ . Biết rằng các chữ số điền vào phải thoả mãn các điều kiện sau :

12 số gồm : 5 số năm chữ số theo từng dòng dọc từ trái sang phải , 5 số năm chữ số theo từng cột đọc từ trên xuống dưới , 2 số năm chữ số theo hai đường chéo đọc từ trái sang phải đều là các số nguyên tố với 5 chữ số có nghĩa , 12 số này không nhất thiết khác nhau .

Tổng các chữ số của 12 số trên đều bằng nhau đều bằng một số cho trước .

Trong ví dụ trên thì tổng của chúng đều bằng 11 .

Yêu cầu : Cho biết tổng các chữ số đó , hãy tìm tất cả các cách điền có thể thoả mãn tính chất bài toán

Dữ liệu : Cho từ file Input.TXT trong đó :

Dòng thứ nhất ghi tổng chữ số của những số cần xây dựng

Dòng thứ hai ghi số ở góc trái trên cùng của bảng .

Kết quả : Ghi ra file Output.TXT tất cả các cách điền , với một bộ số cách nhau một dấu cách

Ví Dụ :

INPUT.TXT

```
11
1
```

OUTPUT.TXT

```
1 1 3 5 1
1 4 0 3 3
3 0 3 2 3
5 3 2 0 1
1 3 3 1 3

1 1 3 5 1
3 3 2 0 3
  3 0 3 2 3
    1 4 0 3 3
      3 3 3 1 1

1 3 3 1 3
  1 3 0 4 3
    3 2 3 0 3
      5 0 2 3 1
        1 3 3 3
```

Hướng Dẫn :

Bớc 1 : Xây dựng các số nguyên tố có 5 chữ số với tổng định trước . Chúng ta sẽ ghi chúng vào file Prime.Dat mỗi dòng biểu diễn một số với tổng được sắp xếp từ nhỏ đến lớn (mỗi dòng hai số , một số là số nguyên tố , một số là tổng của các chữ số của nó)

Có một thủ tục kiểm tra nguyên tố rất hữu hiệu , đối với một loạt số thì nó sẽ tăng tốc độ chương trình rất nhiều .

```
Function    Nguyento ( X : longint ) : Boolean ;
Var
    i , j   :     integer ;
    Begin
        Nguyento := False ;
```

```
                If ( X mod 2 = 0 ) Or ( X mod 3 = 0 ) then Exit ;
                i := 1 ; j := trunc ( Sqrt ( x )) ;
                Repeat
                        inc ( i , 6 ) ;
                Until ( i > j ) or ( x mod i = 0 ) or ( x mod ( i + 2
) = 0 ) ;
                Nguyento:=(i>j) ;
            End ;
        Bớc 2 :
                        C1    C2   C3   C4    D5
```

Chúng ta sẽ điền theo tuần tự sau : H1,C1,D2,H2,C2,H3,C3,H4,C4,H5,C5,D1. Nếu thoả mãn tất cả thì bảng số cuối cùng của cách điền là kết quả cần tìm .

Bài toán 31 : Lới tam giác

Đề bài :

Lới tam giác là một tam giác đều được chia thành các tam giác nhỏ bằng cách vẽ các đường thẳng song song với các cạnh và cách đều nhau . Các tam giác con trong lới được đánh số từ trên xuống dưới , từ trái qua phải bắt đầu từ 1 (xem hình vẽ) . Từ một tam giác con bất kỳ chỉ được quyền di chuyển sang tam giác con có chung cạnh với nó . Ta gọi việc di chuyển từ một tam giác con sang tam giác con chung cạnh với nó là một bới di chuyển .

Yêu Cầu : Tìm cách di chuyển bắt đầu từ tam giác con với chỉ số N sang tam giác con với chỉ số M sao cho số bớc chuyển cần thực hiện là ít nhất

Dữ liệu : Vào từ file văn bản Trenet.Inp chứa hai số nguyên dương N , M ghi cách nhau bởi dấu cách (1<=N<=M<=100000)

Kết quả : Ghi ra file văn bản Trenet.Out :

Dòng đầu tiên ghi số nguyên K là số lợng bớc di chuyển ít nhất cần thực hiện

K dòng tiếp theo mỗi dòng chứa một chỉ số của tam giác con theo thứ tự trên đuờng di chuyển tìm được bắt đầu từ chỉ số của tam giác xuất phát và kết thúc bởi chỉ số của tam giác cần đến .

Ví dụ :

1

2 3 4

5 6 7 8 9

10 11 12 13 14 15 16

TRENET.INP	TRENET.OUT
14 3	5 14 13 7 6 2 3

Hướng Dẫn :

Chúng ta xây dựng mảng như sau :

			1			
		2	3	4		
	5	6	7	8	9	
10	11	12	13	14	15	16

(Hình trên biểu diễn cho 4 dòng .)

Ta có công thức toạ độ của số thứ s là :

dong(s) := trunc (sqrt(s))+1 ;

cot (s) := s - sqr (dong (s)) ;

kí hiệu dong (s) , cot (s) là toạ độ dòng và cột của ô thứ s .

Từ ô xuất phát s đến ô kết thúc t chúng ta có cách đi như sau :

Dòng :

Nếu dòng (t) >dòng (s) nếu đi xuống được thì ta đi xuống (trên mảng đã xây dựng)

Nếu Dòng (t)<dòng(s) nếu đi lên được thì ta đi lên (trên mảng đã xây dựng)

Cột :

Nếu Cot(t)>Cot(s) thì ta đi sang phải (của bảng đã xây dựng)

Nếu Cot(t)<Cot(s) thì ta đi sang trái (của bảng đã xây dựng)

Chúng ta có thể có một số thủ tục quan trọng của bài toán như sau :

```
Function   Dong ( s : Longint ) : Longint ;
Begin
      Dong := Trunc ( Sqrt ( S ) ) + 1 ;
End ;
```

```
Function   Cot (  s : Longint ) : Longint ;
Begin
      Cot := s - Dong ( s ) * dong ( s ) ;
End ;

Procedure  didong ;
Begin
      if D1>D2 then If odd(D1-D2) then
      begin
            Dec ( C1 ) ; D1:=D1-1 ; S := S-4;
      End
      else if D1<D2 then If odd(d1-c1) then
      begin
            inc ( c1 ) ; inc ( d1 ) ; s :=s+4;
      end ;
End ;

Procedure  dicot ;
Begin
      If c1>C2+D1-D2 then
      begin
            dec ( s )  ; dec ( c1 ) ;
      end
      else
      begin
            if c1<c2+d1-d2 then
            begin
                  inc ( c1 ) ; inc ( s ) ;
            end
            else
            begin
                  if c1=2*d1+1 then
                  begin
                        dec ( c1) ; dec ( s ) ;
                  end
                  else
                  begin
                        inc ( c1 ) ; inc ( s ) ;
                  end ;
            end ;
      end ;
end ;
```

```
Procedure    Main ;
Begin
        d1:=dong(s) ; c1:=cot(s) ;
        d2:=dong(t) ; c2:=cot(t);
        While s<>t do
        begin
                didong ; dicot ;
        end ;
end ;
```

Bài toán 32 : Lighnet

Đề bài :

Cho một lới ô vuông kích thớc NxM . Mỗi ô của lới có gần một bóng đèn màu có hai trạng thái màu xanh và đỏ . Lới đèn được điều khiển bởi các nút . Mỗi dòng của lới có một nút đen . Khi ta ấn vào nút đen đó thì tất cả các bóng đèn trên dòng tương ứng với nút này sẽ chuyển trạng thái màu (tất cả bóng đang ở trạng thái màu xanh sẽ chuyển sang màu đỏ , và tất cả các bóng có trạng thái màu đỏ thì sẽ chuyển sang màu xanh) .

Mỗi cột của lới có một nút màu trắng . Khi ta ấn đồng thời vào đúng hai nút trắng thì các bóng đèn ở cột thứ nhất sẽ chuyển sang trạng thái các đèn của cột thứ hai tương ứng , và các bóng đèn ở cột thứ hai cũng chuyển thành các bóng đèn có trạng thái như cột thứ nhất .

Yêu cầu : Biết trạng thái của lới đèn ở trạng thái xuất phát . Hỏi có cách ấn các nút điều khiển để đa lới đèn về trạng thái đích hay không ?

Dữ liệu : Vào từ file Lnet.Inp :

Dòng đầu tiên của filr chứa số K là số bộ dữ liệu

Các bộ dòng tiếp theo , mối bộ dòng một dữ liệu , có cấu trúc như sau :

+ Dòng đầu tiên của một bộ dữ liệu là số N , M (1<=N,M<=100)

+ Tiếp đến là N dòng mô tả trạng thái xuất phát của lới đèn . Mỗi dòng chứa M từ , mỗi từ hoặc là RED hoặc là BLUE được ghi cách nhau bởi đúng một dấu cách .

+ Cuối cùng là N dòng mô tả trạng thái đích của lới đnè được ghi theo khuôn dạng giống nhau đã dùng để mô trả trang thái đích

Kết quả : Ghi ra file LNET.OUT :

Dòng thứ i trong số K dòng , mỗi dòng tương ứng với câu trả lời của một bộ dữ liệu là NO hoặc YES để hiểu là không và có thể chuyển được về trạng thái đích

Ví Dụ :

LNET.INP LNET.OUT
2 3 4 BLUE RED BLUE RED RED BLUE BLUE RED BLUE BLUE BLUE BLUE BLUE RED BLUE RED RED RED BLUE BLUE BLUE BLUE BLUE BLUE 2 2 BLUE BLUE BLUE RED RED RED RED RED YES NO

Hướng Dẫn :

Chúng ta thấy rằng bản chất của hai bảng này mà giống nhau thì khi chúng ta sắp theo một cách quy củ (cùng giống nhau về cách sắp xếp) thì bảng cuối cùng là bảng giống nhau nếu như chúng có thể biến đổi đến nhau , và ngược lại thì không thể đến nhau .Chính vì thế chúng ta sẽ làm như sau :

Quy định màu xanh là = 0 , màu đỏ =1

Đầu tiên chúng ta dùng các phép biến đổi dòng , bién hai bảng sao cho cột 1 của chúng mang giá trị 0

Sắp xếp các cột của bảng 1 và bảng 2 theo hai dãy với thứ tự từ điển

So sánh hai bảng , nếu có giá trị khác nhau thì ghi NO ngược lại ghi YES .

Bài toán 33 : Maxcub

Đề bài :

Cho một khối lập phương kích thước N được chia ra làm N3 khối lập phương đơn vị (có cạnh độ dài 1) bởi các mặt phẳng song song với các mặt của khối lập phương . Trong mỗi khối lập phương con ngời ta viết một số nguyên . Bạn cần tìm khối lập phương con của khối lập phương đã cho có tổng các số trong các khối lập phương đơn vị trong nó là lớn nhất . Khối lập phương như vậy ta sẽ gọi là khối lập phương lớn nhất và tổng các số trong các hình lập phương đơn vị được gọi là trọng lợng của nó .

Ví Dụ : Khối lập phương với độ dài cạnh 3 với các số viết tổng các khối lập phương đơn vị của nó được cho bởi các bảng số sau :

0 -1 3	-1 -3 -1	3 1 -1
-5 7 4	2 -1 5	1 3 2
-8 9 1	0 -1 3	1 -2 1
Lớp thứ 1	Lớp thứ 2	Lớp th 3

Khi đó khối lập phương lớn nhất có trọng lợng là 31 cạnh độ dài 2 được xác định bởi các bảng số sau :

7 4	-1 5	3 2
9 1	-1 3	-2 1
Lớp 1	Lớp 2	Lớp 3

Dữ liệu : Vào từ file văn bản có tên : Maxcub.Inp :

Dòng đầu tiên ghi số N (N <=20)

Tiếp theo là N nhóm dòng mỗi nhóm mô tả một các số trên lớp cắt của khối lập phương nhìn từ mặt trước theo thứ tự gần đến xa , mỗi nhóm gồm n dòng , mỗi dòng chứa n số nguyên liệt kê các số trên một lớp cắt theo thứ tự từ trái qua phải , từ trên xuống dưới .

Kết quả : Ghi ra file văn bản Maxcub.Out trọng lợng của khối lập phương lớn nhất tìm được .

Ví Dụ :

MAXCUB.INP	MAXCUB.OUT
3 0 -1 3 -5 7 4 -8 9 1 -1 -3 -1 2 -1 5 0 -1 3 3 1 -1 1 3 2 1 -2 1	31

Hướng dẫn :

Chúng ta tìm với mỗi lớp , tại giá trị (i,j) thì ta sẽ lấy nó làm gốc (dưới cùng) và thử tất cả các khối lập phương có thể có . Trong các khối lập phương này chúng ta lấy khối lập phương có tổng lớn nhất .

V. Các Bài toán tập hợp

A. Lý Thuyết :

Cho các số từ 1 đến N ta biểu diễn một hoán vị của các số từ 1 đến N là (A1,A2,..An) thì trong A chỉ có một giá trị bằng 1 giá trị nào đó [1,N] .

Như vậy một tập sẽ có N! hoán vị . Và sẽ có số tổ hợp chập K là : = N!/(K!*(N-K)!)

số chỉnh hợp chập K là : = N!/()

B. Bài toán :

Bài toán 34 : Đánh số các hoán vị

Đề Bài :

Giả sử A là tập N số nguyên dương đầu tiên, 3<=N<=12, ta biết rằng có tất cả N! hoán vị của tập A. Một cách liệt kê lần lợt mọi hoán vị có thể có là liệt kê theo thứ tự từ điển: với hai hoán vị cho trước

B = (b1, b2, . . ., bN)

B' = (b'1, b'2, . . ., b'N)

ta nói B nhỏ hơn B' và ký hiệu B<B' nếu có chỉ số I<=N sao cho bk=b'k với k<I và bI<b'I.

Ta xếp thứ tự các hoán vị của A theo quy ớc trên. Theo thứ tự đó, ta đánh số các hoán vị của A từ 1 đến N!, số thứ tự của của hoán vị được gọi là số hiệu của hoán vị

Ví dụ với n = 3, các hoán vị với các số hiệu tương ứng là:

1	2	3	1
1	3	2	2
2	1	3	3
2	3	1	4
3	1	2	5
3	2	1	6

Cần giải quyết hai vấn đề:

1. Cho số N và một số hiệu S, tìm hoán vị có số hiệu S.
2. Cho số N và hoán vị B, tìm số hiệu của B.

Dữ liệu : Vào được cho bởi file HV.INP gồm một số dòng, mỗi dòng có một trong hai dạng:

N S

b1 b2 . . . bN

Với mỗi dòng đọc từ file HV.INP, ghi ra file HV.OUT một dòng tương ứng: nếu dòng thuộc loại 1, ghi hoán vị có số hiệu S, nếu dòng đọc được thuộc loại 2, ghi số hiệu của hoán vị (b1, b2, . . ., bN).

ví dụ :

HV.INP			HV.OUT		
3	5		3	1	2
3	2	1	6		

Hướng Dẫn :

Chúng ta xét dãy hoán vị (A1,A2,.An) của (1,2,..N) . Chúng ta chỉ cần xét vị trí của nó so với vị trí của hoán vị đầu tiên hơn kém nhau bao nhiêu mà thôi . Ta thấy rằng tại vị trí i thì ngoại trừ các số trước đã có mặt thì sau đó thì kể đến nó sẽ có n-i sốchưa được dùng , tức là sẽ có (n-i)! * dem , trong đó dem bằng số các số nhỏ hơn a[i] màchưa được dùng . Ta sẽ có số hiệu của nó trong dãy các hoán vị là :

```
Function so_hieu ( A : mang ; N : integer ) : longint ;
var
        dem,i,j:Integer;
        tong    :longint;
begin
        tong:=0;
        fillchar(c,sizeof(c),true);
        for i:=1 to n-1 do
         begin
        dem:=0;
        for j:=a[i]-1 downto 0 do
                if c[j] then inc(dem);
        tong:=tong+gt[n-i]*dem;
        c[a[i]]:=false;
      end;
```

```
            so_hieu := tong + 1 ;
        end;
```

Khi chúng ta biết được số hiệu của một hoán vị ,thì ta sẽ có cách lật ngược xây dựng hoán vị đó bằng cách ngược lại:
Sau đây là thủ tục lật ngược đó :

```
procedure nguoc(so__hieu:longint;n:Integer);
var i,j,dem,ll:Integer;
   sohieu:longint;
begin
    fillchar(b,sizeof(b),0);
    fillchar(a,sizeof(a),0);
    fillchar(c,sizeof(c),true);
    ll:=0;
    sohieu:=so__hieu;
    for i:=n downto 2 do
    begin
        if sohieu mod gt[i-1]=0 then
        begin
            b[i]:=sohieu div gt[i-1]-1;
            sohieu:=gt[i-1];
        end
        else
        begin
            b[i]:=sohieu div gt[i-1];
            sohieu:=sohieu mod gt[i-1];
        end;
    end;
    for i:=n downto 2 do
    begin
        dem:=0;
        j:=0;
        while dem<>b[i]+1 do
        begin
            inc(j);
            if c[j] then inc(dem);
        end;
        inc(ll);
        a[ll]:=j;
        c[j]:=false;
    end;
    for j:=1 to n do if c[j] then a[n]:=j;
```

```
    for j:=1 to n do  write(g,a[j],' ');writeln(g);
end;
```

Bài toán 35 : Số Sát Sau

Đề Bài :

Trong bài toán trên , chúng ta đã biết được vị trí của một hoán vị trong dãy và ngược lại . Thế nhưng trong bài toán này chúng ta xét một trường hợp khác của hoán vị :

Khi chúng ta cho một hoán vị (a1,a2,..an) . Bài toán đặt ra là hãy tìm hoán vị liền sau của nó . Ví dụ : ta có hoàn vị (1,2,3) thì hoán vị sau nó sẽ là (1,3,2) .

Dữ liệu : Vào từ file Sosatsau.Inp :

Dòng đầu tiên ghi số N (N <=20000)

Dòng sau ghi N số biểu diễn hoán vị đó .

Kết quả : Ghi ra file Sosatsau.Out :

Ghi trên một dòng dãy hoán vị sát sau của hoán vị đã cho từ file input

Ví Dụ :

SOSATSAU.INP	SOSATSAU.OUT
3 1 2 3	1 3 2

Hướng Dẫn :

Nếu đã làm bài toán trên thì chúng ta sẽ có một cách sau : Ta sẽ đổi dãy hoán vị đó ra số hiệu vị trí hoán vị của nó . Rồi sau đó tìm dãy hoán vị có số hiệu là số hiệu đó cộng thêm 1 . Nhưng cách giải bài toán này mà như thế thì không tối u , sau đây là thủ tục tìm số đó :

```
Procedure    Sosatsat ;
(* đó là hoán vị sau của (a1,a2,..an) *)
begin
        j:=n-1 ;
        while Aj>Aj+1 do j:=j-1 ;
        k:=n ;
        WHile Aj>Ak do k:=k-1 ;
        Swap (Aj,Ak) ;
        R:=N ;
        S:=j+1 ;
        While r>s do
        begin
                swap(Ar,As) ; r:=r-1; s:=s+1;
        end ;
end ;
```

Mảng A chính là hoán vị sau ta đã tìm được .

Bài toán 36 : N!

Đề Bài :

Cho số tự nhiên N (1 <=N <1 tỉ), hãy tìm chữ số cuối cùng khác 0 của N!

Dữ Liệu : Từ file TanCung.Inp :

Số N được cho từ bàn phím.

Kết Quả : Cho ra file TanCung.Out :

Cho chữ số cuối cùng khác 0 cần tìm.

Hướng Dẫn :

Bằng công thức La grăng, ta có thể dễ đàng tính được bậc của luỹ thừa 2, 5 trong n!

Như vậy bài toán tng đng với tìm chữ số cuối cùng của số M bằng bỏ tất c luỹ thừa của 10 trong n!. Rõ ràng trong M chỉ còn lại luỹ thừa của 2, không còn luỹ thừa của 5 .

M = f(1)*f(2)*f(3)*…f(n)*luỹ thừa còn lại của 2 .

Trong đó f(k) là số tự nhiên nhận được từ k bằng loại bỏ hết các luỹ thừa của 2, 5.

Mặt khác vì chỉ quan tâm đến chữ số cuối cùng của M nên trong các thừa số tạo nên M ta chỉ quan tâm đến chữ số cuối cùng của các thừa số, tức là số lần các chữ số 1, 3, 7, 9

là số cuối cùng của một số nào đó . Việc đếm số lần xuất hiện của 1, 3, 7, 9 có thể đếm được dễ dàng bằng phân lớp các số từ 1 đến n thành các lớp có cùng luỹ thừa 2, 5. Trong mỗi lớp k = 2^x*5^y, ta chia mỗi số p cho k, tức là sẽ nhận được f(p). dãy f(p) nhận được là dãy liên tục từ 1 đến n div k, bỏ đi các số có tận cùng 2, 4, 5, 6, 8 (Tức là 1, 3, 7, 9, 11,1 3, 17, 19 ….) . Từ đó ta có thể dễ dàng tìm được chữ số cuối cùng của M bằng lấy tích của các luỹ thừa tìm 1, 3, 7, 9, 2 chia d cho 10 .

Chú ý : ta không tính trực tiếp luỹ thừa của 3, 7, 9, mà chỉ lấy với số mũ là số mũ tìm được chia d cho 4 . (vì 3^4, 7^4, 9^4 có tận cùng là 1).

(Có thể đọc thêm bài viết 2 x 5 = 10 của thầy Nguyễn Xuân Huy trên ISM)

Mở rộng :

Bài toán có thể mở rồng tìm hai chữ số cuối cùng trong đó chữ số sau khác 0 của n!

Bài toán 37 : Đánh số các tập con

Đề Bài :

Giả sử A là tập N số nguyên dương đầu tiên, N<=30, với mỗi tập con B của A, ta luôn viết các phần tử của B theo giá trị tăng dần:
B = {b1, b2, . . ., bK} với b1< b2< . . .< bK

Ta xếp thứ tự các tập con của A theo nguyên tắc sau: giả sử B và C là hai tập con của A: B = {b1, b2, . . ., bK}, C = {c1, c2, . . ., cM}, B<C nếu có i<=min(K,M) sao cho b1=c1, . . .,bi-1=ci-1, và bi<ci hoặc bi=ci và K=min(K,M). Trên cơ sở xếp thứ tự các tập con, ta đánh số các tập con của A từ 1 đến 2N, tập trống được đánh số 1.

Ví dụ với n = 3, thứ tự các tập hợp là:

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

Cần giải quyết hai vấn đề:

1. Cho số N và một số hiệu S>1, tìm tập B có số hiệu S.
2. Cho số N và tập B không trống, tìm số hiệu của B.

Dữ liệu: Vào được cho bởi file SUBSET.INP gồm một số dòng, mỗi dòng có một trong hai dạng:

1 N S
2 N b1 b2 . . . bK

Với mỗi dòng đọc từ file SUBSET.INP, ghi ra file SUBSET.OUT một dòng tương ứng mà nếu đầu dòng đọc được ghi số 1, ghi tập con có số hiệu S, nếu đầu dòng đọc được ghi số 2, ghi số hiệu của tập con {b1, b2, . . ., bK}.

Ví dụ

SUBSET.INP				SUBSET.OUT
1	3	8		3
2	3	2	3	7

Hướng Dẫn :

Chúng ta thấy nếu tập hợp có N phần tử thì sẽ có 2N tập con . Tức là trong bài toán chúng ta tính thì với mỗi N thì nó sẽ phân ra N loại : loại có 1 phần tử , loại có 2 phần từ , ...loại có N phần tử . Và mặt khác nó tính theo hệ từ điển , nên ta sẽ phân ra N loại rõ ràng hơn : có 1 đầu tiên , có 2đầu tiên ,...Có N đầu tiên . Nhờ vào việc này chúng ta có thể tính được vị trí cũng như khi tìm tập con đó . Các bạn có thể xem chương trình sau để tham khảo :

```
const
    fi='subset.inp';
    fo='subset.out';
    lt:array[0..30]                                                    of
longint=(1,2,4,8,16,32,64,128,256,512,1024,2048,4096,8192,16384,327
68,65536,131072,262144,524288,

1048576,2097152,4194304,8388608,16777216,33554432,67108864,134
217728,268435456,536870912,1073741824);
type
   tc=array[0..30] of longint;
var
  f,g:text;
  n,s,yc,k:longint;
  a:tc;
function sh(a:tc):longint;
var
  dem:longint;
  i,j,m:longint;
begin
dem:=1;a[0]:=0;
for i:=1 to k do
   begin
   j:=a[i]-a[i-1]-1;
   if j=0 then inc(dem);
   if j>0 then
     begin
     for m:=a[i-1]+1 to a[i]-1 do inc(dem,lt[n-m]);
     inc(dem);
     end;
   end;
sh:=dem;
end;
procedure timsh;
begin
read(f,n);
k:=0;
while not(seekeoln(f)) do
     begin
        inc(k);read(f,a[k]);
     end;
```

```
readln(f);

writeln(g,sh(a));
end;
procedure tim(s:longint);
var
  dem,j:longint;
begin
        if s=0 then exit;
        inc(k);j:=a[k-1]+1;
        if (k=1) and(s=2) then
        begin
                a[k]:=j;
                exit;
        end;
        if s=1 then
        begin
                a[k]:=j;
                exit;
        end;
        if k=1 then dem:=2 else dem:=1;
        repeat
                inc(dem,lt[n-j]);
                inc(j);
        until (j>n)or(dem>=s);
        if j>n then
        begin
                a[k]:=n;
                exit;
        end;
        if dem=s then
        begin
                a[k]:=j;
                exit;
        end;
        dec(dem,lt[n-j+1]);
        dec(s,dem);
        a[k]:=j-1;tim(s);
end;

procedure timtc;
var
```

```
  i:longint;
begin
        readln(f,n,s);
        k:=0;a[k]:=0;tim(s);
        for i:=1 to k do write(g,a[i]:3);writeln(g);
end;

procedure lam;
begin
        assign(f,fi);reset(f);
        assign(g,fo);rewrite(g);
        while not(seekeof(f)) do
        begin
                read(f,yc);
                if yc=2 then timsh;
                if yc=1 then timtc;
        end;
        close(f);
        close(g);
end;
Begin
BEGIN
        lam;
END.
```

VI. Một số bài toán khác

Bài toán 38 : Cây Xăng

Đề Bài :

Một đại lí 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 ên ứng xăng cho các cây xăng . Trên đường cao tốc ngời 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 di (d1<d2<..Dn)

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

Dữ liệu : Vào từ file Vitri.Inp :

Dòng đầu tiên ghi 2 số nguyên dương N ,K (N<=200,K<=30,K<=N)

Dòng thứ hai ghi các số D1,D2,..Dn . (với Di không vợt quá 32000)

Kết quả : Ghi vào file Vitri.Out :

Gồm K dòng , mỗi dòng ghi chỉ số cây xăng đặt bể chứa cây xăng

Ví Dụ :

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

Hướng Dẫn :

Bài toán này không phải là một bài toán dễ . Nhưng nó lại rất dễ ở thuật giải nếu như chúng ta không nên nhận định sai lệch bài toán . Không nên quan trọng vấn đề, có rất nhiều bạn sẽ dùng rất nhiều thuật toán : quy hoạch động , duyệt .. .Nhưng các cách đó chỉ làm chúng ta rối thêm mà thôi .

Chúng ta thấy rằng theo nguyên tắc diricle thì tồn tại một đoạn giữa các bể xăng sẽ có hơn n/k cây xăng . Tức là chúng ta sẽ tìm các khoảng cách có thể có giữa hai cây xăng mà trong đó khi chúng ta rải chúng trên một đoạn đường sẽ vẫn phủ hết cả đoạn đường . Tức là chúng ta sẽ tìm các khoảng cách có thể có , lu vào một mảng kiểm tra . Thực hiện tìm kiếm nhị phân (nhưng phải có hàm kiểm tra có thoả mãn phủ cả đoạn câo tốc) . để tìm ra đoạn ngắn nhất trong các đoạn dài nhất .

Để hiểu thêm các bạn nên tìm hiểu lời giải của bài toán có kèm theo . Sau đây là một bài toán hoàn toàn giống nhưng nó có đến hai điều kiện . Các bạn hãy thử nghĩ lời giải của nó xem ?

Bài toán 39 : Post office

Bài toán:

Có một số làng nằm dọc theo đường cao tốc . đường cao tốc được biểu diễn bằng một trục số nguyên và vị trí mỗi hàng được xác định bởi một số nguyên duy nhất . Không có hai làng nào ở cùng vị trí . Khoảng cách giữa hai vị trí bằng trị tuyệt đối của hiệu giữa các toạ độ nguyên của chúng.

Một số bu cục được xây dựng ở một số làng nhưng không nhất thiết tại mọi làng . mỗi làng và bu cục thuộc nó có cùng vị trí . để xây dựng các bu cục , vị trí của chúng cần chọn sao cho tổng các khoảng cách từ mỗi làng đến bu cục gần nhất đối với làng đó là nhỏ nhất.

Bạn cần viết chương trình sao cho khi biết vị trí của các làng và số lợng các bu cục , hãy tìm tổng nhỏ nhất có thể được của các khoảng cách từ mỗi làng đến bu cục gần nhất đối với nó và các vị trí tương ứng của các bu cục .

Dữ liệu vào: tên file : post.in

dòng thứ nhất chứa hai số nguyên , số thứ nhất là số làng V,1<=V<=300. số thứ hai là số lợng bu cục P,1<=p<=30, p<=v.

dòng thứ hai chứa V só nguyên theo thứ tự tăng dần .V số nguyên này là các vị trí của V làng ,với mỗi vị trí X ,1<=X<=10000.

Dũ liệu ra: tên file : post.out

dòng thứ nhất chứa một số nguyên S là tổng nhỏ nhất có thể được của các khoảng cách từ mỗi làng đến bu cục gần nhất đối với nó theo thông báo trong dòng thứ hai .dòng thứ hai chứa P số nguyên theo thứ tự tăng dần .Các số nguyên này là các vị trí của các làng khác nhau tại đó đặt bu cục . có thể có một số lời giải , chương trình của bạn chỉ cần đa ra một.

Ví dụ:

POST.IN	POST.OUT
10 5 1 2 3 6 7 9 11 22 44 50	9 2 7 22 44 50

Cách cho điểm :

nếu ouput không thoả mãn điều kiện bài ra ,điểm của bạn bằng 0. trong các trường hợp khác , điểm của bạn sẽ được tính theo bảng 1 cho dưới đây . Nếu chương trình của bạn cho tổng S, tổng nhỏ nhất có thể được là Smin thì tính q=S/Smin và tìm điểm C của bạn theo dòng dưới :

Q=S/Smin	Q=1,0	1,0<Q<=1,1	1,1<Q<=1,1	1,15<Q<=1,2	1,2<Q<=1,25
C	10	5	4	3	2

Bài toán 40 : Chu kỳ các bộ số

Đề Bài :

Cho hàm F xác định trên tập các số nguyên dương từ 1 đến m . Giá trị của hàm cũng là số nguyên dương từ 1 đến M . Cho một bộ có thứ tự gồm n (1<=n<=1000) số nguyên dương x1 ,x2,..xn (1<=xi<=M) . Từ bộ số này chúng ta xây dựng một dỹa các bộ số mới theo quy tắc sau :

1. V0 = x1 ,x2 , .. Xn
2. V1= F(x1) , F(X2) ,...F(xn)
3. V2= F(F(x1)) , F(F(x2)) , ...F(F(xn))
4. V3=F(F(F(x1))) , F(F(F(x2))) , ...F(F(F(xn)))
5. .....

Do tập hợp các giá trị của hàm F là hữu hạn , nên dãy Vi sẽ bị lặp lại . Bạn cần xác định độ dài của phần không lặp lại và độ dài của chu kỳ . Độ

dài của phần không bị lặp lại cũng như độ dài của chu kỳ phải là nhỏ nhất .
Chú ý : Theo định nghĩa , chu kỳ phải là số dương
Dữ liệu : Vào từ file Period.Inp :
Dòng đầu tiên chứa số M
Dòng thứ 2 chứa các giá trị F(1) , F(2) , .. .F(M)
Dòng thứ 3 chứa số N
Dòng thứ 4 chứa các số : X1 ,X2 ,...Xn
Kết quả : Ghi ra file văn bản Period.Out độ dài của phần không bị lặp lại và độ dài của chu kỳ
Các số trên một dòng được ghi cách nhau bởi dấu cách hoặc dấu xuống dòng
Ví Dụ :

PERIOD.INP	PERIOD.OUT
10 5 6 4 3 2 5 1 1 5 4 4 1 10 8 1	2 6

Hướng Dẫn :
Tìm chu kì riêng và phần đầu không lặp của mỗi số
Chu kì của dãy là BCNN của các chu kì
Phần đầu không lặp là phần Max của phần đầu
Chú ý :
BCNN có thể vợt qua giới hạn số học nên chúng ta phải xử lí số lớn
THủ tục tìm BCNN có thể cải tiến theo dạng :
a=2a13a2...Pnan
b=2b13b2...Pmm
C:=BCNN(a,b)=pi ; pi=max(ai,bi) ;

Bài toán 41 : Ghép Nguyên Tố

Đề Bài :
Giả sử C là dãy các số nguyên tố tăng dần :
(C) : 2 , 3 , 5 , 7 , 11 , 13 , 17 , 19 , ...
Từ dãy C , ta lập dãy D gồm các số là ghép của những cặp số gồm số hạng có thứ tự lẻ với số hạng ngay sau nó của dãy C
(D) : 23 , 57 , 1113 , 1719 , ...
Bằng cách xoá đi khỏi dãy D các số hạng không là nguyên tố ta thu nhận được dãy E .
Yêu cầu : Cho trước một số nguyên dương N không lớn 500 , hãy tìm số hạng thứ N của dãy E
Dữ liệu : Đọc cho từ file GNT.INP gồm một số dòng , mỗi dòng ghi một số N

Kết quả : Với mỗi N đọc từ file GNT.INP, hãy ghi ra file GNT.OUT các dòng tương ứng có số nguyên tố đó .
Ví Dụ :

GNT.INP	GNT.OUT
1	23

Hướng Dẫn :

Hoàn toàn đơn giản , chúng ta sẽ xây dựng một thủ tục kiểm tra nguyên tố thật tốt (như bài toán 27 đã nêu) . Sau đó ta ghép số đó rồi sử dụng một file trung gian để lu lại các giá trị đó . Khi đọc đến giá trị nào thì ta sẽ ghi ra giá trị đó .

Bài toán 42 : phủ nhỏ nhất

Đề Bài :

Cho tập gồm n đoạn thẳng (đánh số từ 1 đến n) với các đầu mút có toạ độ nguyên [LI, RI], i = 1, 2, ... n và một đoạn thẳng [P, Q] (P, Q là các số nguyên).

Yêu cầu: Cần tìm một số ít nhất đoạn thẳng trong tập đã cho để phủ kín đoạn thẳng [P, Q] (nghĩa là mỗi điểm x [P, Q] phải thuộc vào ít nhất một trong số các đoạn thẳng được chọn).

Dữ liệu: Vào từ file văn bản PHU.INP:
Dòng đầu tiên ghi 3 số n, P, Q phân cách nhau bởi dấu trắng;
Dòng thứ i trong số n dòng tiếp theo chứa hai số Li, Ri phân cách nhau bởi dấu trắng (i = 1, 2, ..., n);.
1 n 100000; Q - P 5000 ; [Li] 50000; [Ri] 50000, i = 1, 2, ... n.

Kết quả: Ghi ra file văn bản PHU.OUT:
Dòng đầu tiên ghi số k là số lợng đoạn cần chọn (quy ớc ghi số 0 nếu không tìm được lời giải);
Nếu k > 0 thì mỗi dòng trong số k dòng tiếp theo ghi chỉ số của một đoạn thẳng được chọn.

Ví dụ:

PHU.INP			PHU.OUT
2	0	1	1
-1	0		2
0	1		

PHU.INP	PHU.OUT
2 0 1	0
-1 0	

- 5 - 3
1 5
Hướng dẫn :

Trước hết ta thấy độ dài đoạn thẳng [P,Q] không lớn hơn 5000 . Còn số đoạn thẳng trong tập đã cho có thể lên tới 100000. Ta không thể lu tất cả các đoạn thẳng này lại được bằng cách lu từng đoạn một . Vậy chúng ta có thể lu các đoạn thẳng này theo cách : Dùng một mảng có độ dài chính bằng độ dài của đoạn [P,Q] , kích thớc của mnảg này là 5000x2 .

A : Array [0..5000,1..2] Of Longint ;

Bây giờ ta coi đoạn thẳng [P,Q] là mảng A trên . Ta lần lợt đọc các đoạn thẳng của tập N các đoạn thẳng , Với mỗi đoạn thẳng vừa đọc được , ta xét xem chúng phủ đoạng [P,Q] bắt đầu từ đâu và độ dài mà đoạn thẳng đó phủ được . Ta sẽ lu lại trên mảng A ở phần tử mà đoạn thẳng chúng ta vừa đọc bắt đầu phủ đoạn [P,Q] chỉ số của đoạn thẳng đó và độ dài của chúng phủ được .

A[phần tử bắt đầu phủ , 1]=chỉ số của đoạn vừa đọc

A[phần tử bắt đầu phủ , 2] =độ dài mà đoạn đó phủ được

Nếu có nhiều đoạn thẳng bắt đầu phủ đoạn [P,Q] tại cùng một vị trí thì chọn đoanj nào có độ dài phủ được là lớn nhất .

Sau khi làm xong công việc trên ta sẽ được một mảng gồm các đoạn thẳng phủ lên đoạn [P,Q] . Bây giờ ta sẽ tiến hành tìm xem số đoạn phủ ít nhất sẽ là bao nhiêu bằng cáhc : ta sẽ bắt đầu từ vị trí 0 (nếu tại vị trí này mà không có đoạn nào phủ thì sẽ xem không có lời giải) ta tìm xem có đoạn thẳng nào giao với đoạn thẳng có phủ dài nhất thì ta lu đoạn này mà tạo thành một đoạn thẳng có phủ dài nhất thì ta lu đoạn này lại (nếu không có đoạn thẳng nào giao với đoạn thẳng này để tạo được một đoạn thẳng có độ che phủ lớn hơn độ che phủ đoạn trên thì sẽ không có lời giải) . Tiếp tục làm tương tự với đoạn thẳng vừa tìm được cho đến khi các đoạn thẳng được chọn sẽ phủ kín đoạn thẳng [P,Q] ta sẽ được số đoạn thẳng ít nhất để phủ đoạn [P,Q] .

Bài toán 43 : Quan Hệ

Đề Bài :

Xét một tập N đối tợng có thể so sánh được (N<100).Giữa 2 đối tợng a và b có thể tồn tại 1 trong 3 quan hệ phân loại :

a=b , a<b , a>b

Quan hệ '=' có tính chất đối xứng nên không được nêu ở lại trên

Như vậy , với 3 đối tợng (a,b,c) có thể có 13 quan hệ như sau :

a=b=c , a=b<c , c<a=b , a<b=c , b=c<a ,a=c<b , b<a=c , a<b<c , a<c<b , b<a<c ,b<c<a ,c<a<b , c<b<a

Cho số N , hãy xác định số lợng quan hệ phân loại khác nhau

Dữ liệu : vào từ file COND.INP , gồm nhiều số nguyên N (trong phạm vi từ 2 đến 99) ,mỗi số trên 1 dòng

Kết quả : Đa ra file COND.OUT các số lợng quan hệ phân loại tìm được , mỗi số trên một dòng

Ví Dụ :

COND.INP	COND.OUT
2 3	3 13

Hướng Dẫn :

Chúng ta sẽ có với 1 bộ N số : x1,x2,..Xn . Như vậy khi chúng ta điền các dấu >,<,= để tạo ra một mối quan hệ đúng tức là sẽ không có trường hợp :

....< xi > ....

Thì như vậy khi chúng ta để nguyên thứ tự của một dãy đó , và chỉ thay đổi dấu thì có hai cách điền :

Một là toàn > và =

hai là toàn < và =

Và chúng ta lại đổi vị trí hai số nào đó , rồi lại thực hiện điền như vậy . Cứ như vậy cho đến hết N! cách đổi chỗ có thể có .

Chung quy lại , kể cả n! cách đổi vị trí , ta sẽ có số mối quan hệ là= n!* 2n-2 +1(kể cả toàn dấu =) .

Chú ý : Vì N có thể lên đến 99 , nên chúng ta phải tính toán các số trong phạm vi các số lớn (tức là phải viết các thủ tục cộng và nhân các số lớn) .

Bài toán 44 : Map

Đề Bài :

Ngời du lịch thường mang theo bản đồ được gấp gọn . Có bốn thao tác gấp khác nhau :

Kiểu A : gấp mép trên trùng với mép dưới và đè lên trên mép dưới

Kiểu B : Gấp mép trên trùng với mép dưới và nằm dưới mép dưới

Kiểu C : Gấp mép phải trùng với mép trái và đè lên trên mép trái

Kiểu D : Gấp mép phải trùng với mép trái và nằm dưới mép trái .

Trước khi gấp hoặc sau khi mở bản đồ , nó được đặt hướng bác chỉ lên trên . Khi mở bản đồ , các đường gấp để lại trên bản đồ có hai loại : loại 1 - vết gấp hẳn lên trên , loại 0 - vết gấp hẳn xuống dưới . Các đường gấp theo hàng được đánh số từ 1 trở đi từ trên xuống dưới , các đường gấp theo cột được đánh số từ trái sang phải từ 1 trở đi . Các mép trên và trái được coi là hàng 0 và cột 0

Một cách gấp bản đồ sẽ tương ứng với một xâu các ký tự A,B,C,D .Ví Dụ : , với cách gấp BAADC , sau khi mở bản đò ta được bức tranh nêu ở hình dưới gồm một lới các ô hình chữ nhật . Cạnh của các ô trên các vết gấp được gọi là một đoạn và cũng chia thành 2 loại 0 và 1 tuỳ theo vết gấp là thuộc loại 0 hay 1 . Các đoạn loại1 được thể hiện bằng nét thẳng liền . Các đoạn loại 0 được thể hiện bằng đoạn thẳng nét đứt .

Yêu cầu : Biết cách gấp , hãy xác định số đoạn loại 0 và 1 .

Dữ liệu : Vào từ file văn bản MAP.INP gồm một dòng chứa xâu S xác định cách gấp , S không quá 50 ký tự

Kết quả : Ghi ra file MAP.OUT hai dòng :

Dòng 1 ghi số lợng đoạn 0

Dòng 2 ghi số lợng đoạn 1

Ví Dụ :

Map.Inp	Map.Out
BAADC	24
	28

Hướng Dẫn :

Chúng ta thấy rằng các nếp gấp chỉ phụ thuộc vào cách gấp lần đầu tiên , còn các cách gấp của các bớc sau không quan trọng về vị trí

của nó . Thật vậy , khi chúng ta gấp nếp đầu tiên , thì nó tạo ra một đường gấp nào đó , thì sau đó , các đường gấp khác mà vuông góc với nó thì sẽ tạo ra gấp đôi đường gấp trên đó .

Ta có gấp C,D thì sẽ tạo ra nếp dọc , nếu gấp A,B thì tạo ra nếp ngang . Mà cứ một lần gấp thì số nếp ngang , hoặc nếp dọc sẽ tăng gấp đôi . Vậy có T1 hàng và T2 cột . Trong đó DemCD , DemAB là số lần gấp cả Cvà D và Số lần gấp cả Avà B . Thì ta có :

T1:=2DemAB ; T2:=2DemCD ;

Vậy ta sẽ có tổng số nếp 0 và 1 là : = T1*T2-T1-T2 . Mặt khác ta có :

Quy định cách gấp A và B là cách gấp hàng , cách gấp C và D là cách gấp cột

Cách gấp đầu tiên , thì trên đường gấp đầu tiên này , (Gọi là đường cơ bản)sau K lần gấp của cách gấp khác loại với cách gấp đầu tiên - là cách gấp hàng hoặc cột - thì tăng số nếp ở đường này lên 2K nếp . Còn các cách gấp sau thì , khi chúng ta gấp bất kỳ theo cách nào thì tổng số nếp 0 và 1 ở các đường không phải là đường cơ bản sẽ tăng lên bằng nhau . Tức là hiệu của số nếp gấp 0 và 1 chính là bằng số nếp ở đường cơ bản .

Khi tìm hai số nếu chúng ta biết tổng và hiệu thì ta sẽ tìm được hai số đó .

Thủ tục cơ bản của bài toán như sau :

```
Tính DemAB và DemCD
T1:= 2DemAB ; T2:= 2DemCD ;
Tong:= T1*T2-T1-T2 ;
Case  S[1] Of
'A' : Hieu := -T2 ;
'B' :  Hieu:=T2 ;
'C' : Hieu :=-T1 ;
'D' : Hieu:=T1 ;
So nep 1:=(tong+hieu)/2;
so nep 0:=(tong-hieu)/2;
```

Bài toán 45 : Đục Lỗ

Đề Bài :

Cho tờ giấy kẻ caro , kích thước 2Nx2Nô mỗi chiều (3<=N<=500). Ngời ta gấp tờ giấy này n-3 lần , mỗi lần gấp như sau : gấp mép dưới lên mép trên để mặt trước đè lên nhau , sau đó gấp mép phải đè lên trên mép trái . Như vậy , sau mỗi lần gấp kích thước mỗi chiều giảm đi một nửa . Kết quả cuối cùng ta có xếp giấy kích thước 8*8 . Bằng máy dập , ngời ta đục một số ô của xếp giấy đồng thời ở tất cả các lớp . hãy xác định , sau khi mở lại tờ giấy , ta có bao nhiêu phần rời nhau , biết rằng 2 ô dính với nhau khi chúng có chung 1 cạnh

Dữ liệu : vào từ file list.inp
dòng đầu tiên ghi số N
8 dòng sau , mỗi dòng chứa 8 số 0 hoặc 1 : 1 để biểu thị ô bị đục , các số cách nhau 1 dấu cách
Ví Dụ :

LIST.INP LIST.OUT
4 0 1 0 0 0 0 1 0 1 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 0
0 0 0 0 0 1 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 11
Hướng dẫn :

Chúng ta sẽ lần ngược trở lên , Với mỗi lần như thế , ta chỉ quan tâm tới các miền ô liên kết nằm ở biên (dưới và phải) . Nếu nó gần nhau , tức là khi mở ra thì nó vẫn tạo thành 1 miền , còn các miền khác (không phải là miền biên) thì được tăng gấp đôi . Chính vì thế ta cứ thực hiện cho đến hết .

Bài toán 46 : Phân Vùng

Đề Bài :

Cho một hình vuông được chia thành 2Nx2N ô vuông bằng nhau bằng các đường song song với các cạnh của hình vuông, ta nói hình vuông đó được chia ở mức N. Mỗi ô vuông được đặt tên như sau:
- Nếu mức N=1 tên các ô như trong hình vẽ sau:

1 2
3 4

- Nếu mức N=k, tên của một ô X được đặt như sau: *nếu tên của ô vuông Y ở mức k-1 chứa ô vuông X là a1a2. . . ak-1 thì tên của ô X = a1a2. . . ak-1ak trong đó ak bằng 1/2/3/4 tuỳ theo vị trí của X trong Y như hình trên*.
Ví dụ với N=3, ta có tên một số ô như sau:

. . . . 121. 222
. . . . 213. . . . .
. . . . . 142. . . . . .
. . . .
. . . . 412. . . .
. . . .
. . . .
. . . .

Viết chương trình làm các việc sau:

1. Đọc từ file text với tên INP1.BL1 trong đó mỗi dòng ghi một xâu ký tự độ dài N chỉ gồm các ký tự 1, 2, 3, 4 thể hiện tên các ô vuông của hình được chia ở mức N. Hãy in ra file OUT1.BL1 trên mỗi dòng số dòng và số cột của ô có tên tương ứng. Ví dụ:

INP1.BL1	OUT1.BL1
121	1 3
142	3 4
213	2 5
222	1 8
412	5 6

2. Đọc từ file dữ liệu INP2.BL1 trong đó dòng thứ nhất ghi hai số i, j, trong các dòng tiếp theo, mỗi dòng ghi một xâu ký tự chỉ gồm các ký tự U, D, R,L. Ý nghĩa của các dữ liệu này là như sau: i, j là số dòng và số cột của ô trong hình vuông được chia ở mức N với N là số nguyên dương nhỏ nhất mà 2N>= Max(i,j); mỗi xâu thể hiện một cách đi từ ô [i,j] gồm một dãy các di chuyển qua các ô kề cạnh, có bốn cách di chuyển: U - lên trên, D - Xuống dưới, R - sang phải, L - sang trái. Ghi ra file OUT2.BL1 mỗi dòng thông tin về mỗi cách đi từ ô [i,j] như sau: nếu trong quá trình đi có di chuyển ra ngoài hình vuông thì ghi OUT, nếu trong suốt quá trình đi không có di chuyển ra ngoài hình vuông thì ghi tên của ô đi đến theo cách đặt tên ở Câu 1.

Hướng Dẫn :

1 2
3 4

Chúng ta sẽ dùng các thủ tục xác định toạ độ của một ô nào đó . Thông qua việc tìm kiếm miền . Công việc này hết sức dễ dàng , ta xét số nào đó , thì đi trở ngược vào , cứ nếu gặp số 1 thì nằm ở 1 / 4 trái trên của bảng còn lại , gặp 2 tức là ở 1 / 4 phải trên , gặp 3 tức là ở 1 / 4 trái dưới và 4 là 1 / 4 phải dưới của bảng còn lại . Mặt khác , ta có thể quy đổi số nào đó về toạ độ và ngược lại .

Bài toán 47 : Cấp Số Cộng

Đề Bài :

Cho dãy gồm N số nguyên a1,a2,..an . Ta gọi dãy con của dãy đã cho là dãy thu được bởi việc xoá khỏi dãy đã cho một số số hạng của nó và giữ nguyên thứ tự của các số hạng còn lại . Bản thân dãy đã cho cũng được coi là dãy con của chính nó .

Yêu cầu : Trong số các dãy con của day đã cho lập thành một cấp số cộng với công sai d thoả mãn : 1<=d<=100 , hãy tìm dãy con gồm nhiều phần tử nhất hoặc thông báo là dãy đã cho không chứa dãy con như vậy .
Dữ liệu : Vào từ file BL1.INP :
Dòng đầu tiên chứa số N
Dòng thứ i trong số N dòng tiếp théo chứa số hạng thứ i của dãy số đã cho .
Kết quả : Ghi ra file BL1.OUT :
Dòng đầu tiên ghi số -1 nếu không tìm được dãy thoả mãn điều kiện đầu bài . Nếu trái lại , dòng đầu tiên ghi hai số k và d , trong đó k là số lợng phần tử con , d là công sai của dãy con tìm được
Nếu có thì ghi tiếp k dòng sau , mỗi dòng một chỉ số trong dãy đã cho của một số hạng của dãy con tìm được theo thứ tự xuất hiện của chúng trong dãy con
Hạn chế : 1<=N<=100000 ; -30000<=ai<=30000
Ví Dụ :

BL1.INP	BL1.OUT
13 -100 -91 -90 -80 -70 -69 -60 -58 -47 -36 0 1 2	6 11 2 4 6 8

Hướng Dẫn :

Với yêu cầu dữ liệu như bài toán ra thì gần như không có thuật giả tốt . Ngay cả trong đáp án và mẫu test của HSGQG2001-2002 thì : N<=10000 , |Ai|<=8000 . Nhưng ngay cả với dữ liệu như vậy thì các bạn làm chương trình cũng quá khó khăn rồi . Ví tốc độ cả lần độ tối u . Chính vì thế ta có cách giải sau :

Dùng mảng A: Array[-8000..8000] of link ;
Trong đó danh sách

```
Link = record
        x : integer ;
        next : link ;
end ;
```

Mảng A[i] nhiệm vụ giữ lại những vị trí có thể có của i trong dãy đó . Bởi vì công sai chỉ là 1 đến 100 . Nên với mỗi công sai D có thể có thì chúng ta làm như sau :

Xuất phát từ số nhỏ nhất trong dãy , tìm dãy con dài nhất của dãy lớn có công sai d mà chứa nó (nếu chứa nó thì đánh dấu nó đã tham gia) . Sau đó lại tiếp tục tìm các số khácchưa có trong dãy vừa có rồi lại tìm kiếm các số tiếp theo trong dãy . (vì lu trữ dưới dạng danh sách trên nên công việc tìm kiếm đơn giản và rất nhanh . (chỉ cần truy nhập tời gía trị A[d+A[i]] rồi lại tiếp tục tìm kiếm) .

Như vậy chương trình chạy của chúng ta sẽ không thể vợt quá: 100*N . Cho nên chương trình sẽ nhanh hơn là tìm từng đoạn một như cách thông thường của chúng ta .

Bài toán 48 : Lát Nền

Đề bài :

Cho một hình vuông kích thớc 2N x 2N bị khuyết một ô, trong đó N là một số nguyên dương >= 4 và không chia hết cho 3. Hãy tìm cách lát hình vuông bị khuyết 1 ô đó bằng các hình thớc thợ như hình vẽ sau :

Dữ liệu : Vào từ file LatNen.Inp

Dòng đầu tiên ghi số N

Dòng 2 ghi u,v là hàng và cột của ô bị khuyết

Kết quả : Ghi ra file LatNen.Out

Ma trận 2Nx2N trong đó mỗi ô ghi một số nguyên s :

s = 0 nếu đó là ô khuyết, ngược lại là thứ tự của viên gạch lát nền .

Hướng Dẫn :

Trước hết chúng ta giải quyết bài toán sau khó hơn :

" Ngời ta cần lát kín một nền nhà hình vuông cạnh dài n=2k , (k là một số tự nhiên trong khoảng 1..6) khuyết một phần tử phía trên bằng những viên gạch màu kích thớc thợ tạo bởi như hình trên . Hai viên kề cạnh phải có hai màu khác nhau .Hãy cho biết một cách lát với số màu ít nhất ."

Hướng Dẫn :

Đầu tiên ta gọi thủ tục Init để khởi trị với hình vuông cạnh v=2 nằm ở góc dưới trái của nền nhà được biểu diễn dưới dạng một mảng hai chiều A : ba ô trong hình vuông 2x2 sẽ được điền giá trị 1 , ô nằm ở góc trên phải được điền giá trị 2 . Gọi hình được khởi tạo là A . Môi bớc tiếp theo ta thực hiện ba thao tác biến hình sau đây :

3 3 1 1 D 3 2 2 1 1 2 3 3 1 1 3 2 3 3 1 2 B 3 2 1 1 1 2 2 3 1 1 3 3

3 3 1 2 A 3 2 1 1 1 2 2 3 1 1 3 3 2 3 1 1 C 3 3 2 1 1 2 2 3 1 1 3 3

Tịnh tiến A theo đường chéo để thu được B

Lật A sang phải để thu được hình C

Lật A lên trên để thu được hình D

Mỗi lần lặp như vậy ta sẽ thu được hình vuông cạnh tăng gấp đôi hình trước . Làm như thế cho đến khi kích thước bằng 2N . Vfa sẽ lấy 3 mảnh : A,D,C làm kết quả .
Nhưng từ giải thuật trên ta sẽ thu được hình 2Nx2N . Trong đó có một hình bị thiếu ở góc trên phải (ở hình B có số 2) . Như vậy ta có thể giải quyết bài toán trên bằng cách di chuyển viên gạchchưa lát đó đến vị trí cần bỏ trống . (Nhưng ta đã mở rộng bài toán trên đi rất nhiều - đó là đã tô màu chúng , và quy định số màu ít nhất) .

Hoặc là chúng ta dùng thuật toán đệ quy theo bài toán giống như bài toán lát gạch (nhưng đây không phải là đệ quy có nhớ) .Nên tốc độ chương trình không nhiều .

Bài toán 49: Dãy đỉnh các đa giác

Đề bài :

Xét đa giác đều N đỉnh với cạnh là đơn vị (3<=N<=1000) . Lấy 1 đỉnh nào đó làm tâm đồng dạng phối cảnh , ngời ta phóng đa giác lần lợt lên 2 , 3 ,4 ...lần . Bắt đầu từ tâm đồng dạng phối cảnh , trên chu vi của đa giác ngời ta đánh dấu các điểm cách nhau 1 đơn vị . Nếu coi tâm đồng dạng là hình phối cảnh với hệ số đồng dạng là 0 , thì tổng số điểm được đánh dấu khi lần lợt phóng đại đa giác tạo thành 1 dãy số nguyên dương , gọi là dãy số bậc đa giác .

Ví Dụ trên N = 3 ta có dãy số bậc đa giác là : 1 , 3 , 6 ,10 ,...
Yêu cầu : Cho M loại đa giác đều khác nhau (3<=M<=50) , mỗi loại có số đỉnh N và một số nguyên dương S (0<S<=10000) , hãy chỉ ra 5 số nhỏ nhất lớn nhất lớn hơn hoặc bằng S , mối số thuộc ít nhất 2 dãy số bậc đa giác .
Dữ liệu : Cho từ file POLIGON.INP
Dòng đầu tiên là số nguyên M
Dòng thứ hai chứa M số nguyên dương khác nhau , theo thứ tự tăng dần , xác định loại đa giác cần xét
Kết quả : đa ra file POLIGON.OUT :
5 dòng , mỗi dòng tương ứng với một số tìm được (theo thứ tự tăng dần của giá trị) và có quy cách :
Số nguyên : N1 , N2 , ...
trong đó N1 , N2 , N3 ,... (theo thứ tự tăng dần) là loại đa giác trong danh sách đã cho có dãy số chứa số nguyên .

Ví Dụ :

POLIGON.INP	POLIGON.OUT
5 3 4 13 36 124 1 : 3 4 13 36 105 :3 36 171 : 3 13 1225 : 3 4 124	1 : 3 4 13 36 124 36

Hướng Dẫn :

Xây dựng các mảng :

a: 1 3 6 10 15 21 .....

b: 2 3 4 5 6 .....

c: 1 1 1 1 .....

Chúng ta sẽ xây dựng các mảng có bậc đỉnh theo tuần tự từ c ->b->a ->....

Như vậy sẽ ra được dãy đỉnh đó .

Bài toán 50 : VAT

Đề Bài :

Để tăng ngân sách qua việc thu thuế và đảm bảo độ tin cậy cao trong việc xử lí tự động các bản khai , chính phủ một quần đảo Thái Bình Dương quyết định áp dụng biểu thuế mới như sau : Nếu X là giá trị thuế mức cũ và Y là giá trị thuế mới , thì Y phải thoả mãn các điều kiện sau :

Y>=X , Y - nguyên

Trong dạng biểu diễn hệ 10 của Y không có các chữ số d1,d2,..dn (1<=n<9)

Không tồn tại số nguyên Z<Y và thoả mãn 2 điều kiện trên

Yêu cầu : Hãy lập trình tính thuế mới theo mức thuế cũ .

Dữ liệu : Vào từ file VAT.INP :

Dòng đầu tiên ghi số nguyên N

Dòng thứ 2 ghi N số nguyên D1 ,D2 ,..Dn . Các số trên 1 dòng cách nhau ít nhất một dấu cách

Dòng thứ 3 ghi số X (X>0) (số chữ số <=20) .

Kết quả : Đa ra file VAT.OUT giá trị Y .

Ví Dụ :

VAT.INP	VAT.OUT
3 1 3 8 17	20

Hướng Dẫn :

Xây dựng mảng A [i] là mảng số đó . Nên khi xây dựng mảng số sẽ loại trừ những số đó ra khỏi danh sách có thể có của các chữ số . Khi

chúng ta duyệt các giá trị có thể của các hàng thì chúng ta nên duyệt các số từ hàng lớn nhất trở vào (lợi dụng tính chất nhỏ hơn của hai số để giảm bớc duyệt) .

Bài toán 51 : Lật hình xếp

Đề bài :

Xét một khối lập phương được lắp ghép từ các hình lập phương đơn vị . Kích thớc mỗi chiều là N (0<N<12) . Từ khối lập phương lớn này ngời ta tạo ra một hình khối bằng cách lấy đi một số khối lập phương đơn vị , sao cho :

Nếu lật hình đó để mặt trái thành mặt đáy hay mặt phải thành mặt đáy thì hình vẫn giữ nguyên , không có khối vuông nào bị rơi hoặc bị xê dịch

Trong trường hợp này có thể mô tả bằng mô hình số gồm N dòng , mỗi dòng mô tả các độ cao thuộc một lớp của hình , song song với mặt phải , tính từ mặt trái ra .

Ví Dụ :

3 3 1
3 1
2

Hình có thể được lật và xoay , để mặt trái trở thành mặt đáy và mặt phải trở thành mặt trái (phép xoay trái) hoặc mặt phải thành mặt đáy và mặt trái thành mặt phải (phép xoy phải)

Ví Dụ :

3 2 1	3 3 2
2 1 1	2 1 1
2 1	1

Yêu cầu : Hãy lập trình xác định mô hình số nhận được bởi phép xoay trái và xoay phải .

Dữ liệu : Vào từ file Solid.Inp :

Dòng đầu tiên là số nguyên N

N dòng sau , mỗi dòng gồm 1 số số nguyên dương tương ứng với 1 dòng của mô hình số và kết thúc bằng 1 số 0 , các số cách nhau 1 dấu cách

Kết quả : Ghi ra file Solid.Out :

N dòng đầu tiên tương ứng với mô hình nhận được khi xoay trái

Dòng thứ N+1 là dòng trắng

N dòng tiếp theo ứng với mô hình nhận được khi xoay .

Ví Dụ :

SOLID.INP
3 3 3 1 0 3 1 0 2 0
2 2 1 1 1

SOLID.OUT
3 2 1 2 1 1 2 1 3 3

Hướng Dẫn :

Lu vào mảng ba chiều : A[1..N,1..N,1..N] sau đó khi chúng ta lu một khối nào đó thì bằng cách chúng ta lu các khối đơn vị là : A[i,j,z]=1 nếu tại vị trí khối [i,j,z] có khối . Ngược lại thì bằng 0 . Rồi sau đó thay cho việc lạt ma trận thì chúng ta chỉ cần đổi vai trò của hàng cột ban đầu mà thôi . Rồi đếm các chiều cao của các khối trong hình lật đó .

Bài toán 52 : Dãy con khiêm tốn

Đề bài :

Cho một dãy số :

A1,A2,...An (10000>=N>=4)

Chúng ta cần tìm một dãy con của dãy trên . Nhưng với điều kiện : Tổng của ba số bất kỳ trong dãy đó không được lớn hơn tổng các số còn lại .

Yêu cầu : Hãy tìm dãy con dài nhất

Dữ liệu : Vào từ file DCKT.Inp :

Dòng đầu tiên là số N

Dòng sau ghi N số biểu diễn dãy số A

Kết quả : Ghi vào file DCKT.OUT :

Dòng đầu tiên là số phần tử của dãy con đó

Dòng tiếp theo ghi các chỉ số thuộc dãy ban đầu là thuộc dãy con đó

Hướng Dẫn :

Chúng ta nhận xét răng nếu dãy thoả mãn tính chất vậy thì : X1,X2,X3 là ba số lớn nhất của dãy đó cũng thoả mãn : X1+X2+X3<tổng các số còn lại . Mặt khác ta thấy răng dãy con nào có tính chất X1+X2+X3<tổnh các số còn lại thì dãy đó cũng là dãy khiêm tốn . Chính vì thế bài toán này ta giải quyết nhusau :

Sắp xếp dãy theo thứ tự giảm dần

Tìm từ vị trí đầu đến cuối nếu có vị trí nào thoả mãn : A[i]+A[i+1]+A[i+2]<A[i+3]+..A[n] thì sẽ thoát và các số từ i đến N sẽ thuộc dãy con đó .

Bài toán 53 : Dãy hình sóng

Đề bài :

Cho dãy gồm N số nguyên khác nhau X1, X2,....., XN. Ta nói dãy đó là có dạng hình sin bậc M nếu dãy gồm đúng M+1 đoạn liên tiếp luân phiên đơn điệu tăng - giảm hoặc giảm - tăng.

Ví dụ: Dãy số
1 5 9 13 12 8 4 2 3 7 11 15 14 10 6
là dãy hình sin bậc M=3, dãy đó được phân chia thành 4 đoạn:

Đoạn 1: 1 5 9 13	Đơn điệu tăng
Đoạn 2: 13 12 8 4 2	Đơn điệu giảm
Đoạn 3: 2 3 7 11 15	Đơn điệu tăng
Đoạn 4: 15 14 10 6	Đơn điệu giảm

Cho dãy X gồm N số nguyên khác nhau X1, X2,....., XN (2<N<=100) và cho M<=N-2

Kiểm tra xem dãy X có phải là dãy hình sin bậc M hay không? Nếu không thì sắp xếp lại dãy đó để nhận được dãy hình sin bậc M.
Dữ liệu vào được cho bởi file văn bản với tên BAI1.INP có cấu trúc nhưsau:
- dòng đầu gồm hai số nguyên N và M,
- dòng sau ghi N số nguyên X1, X2,....., XN

Các số trên một dòng cách nhau ít nhất một dấu cách.
Kết quả ghi ra file văn bản với tên BAI1.OUT theo cấu trúc sau:

Dòng đầu ghi số 1 nếu dãy đã cho là dãy hình sin bậc M và ghi số 0 nếu ngược lại ;

Dòng thứ i trong số M+1 dòng tiếp theo ghi các phần tử của đoạn đơn điệu thứ i (i=1,2,.....M+1). (Nếu dòng thứ nhất ghi số 0 thì trong M+1 dòng sau, ghi các đoạn đơn điệu tương ứng với dãy đã được sắp xếp lại).

Hướng Dẫn :

Có rất nhiều cách giải quyết bài toán này . Nhưng cách đơn giản nhất :

Đầu tiên chúng ta sắp xếp các số đó theo trật tự tăng dần

Đếm số sau đó cứ một lần ta lại đổi chỗ hai số liền nhau thì tạo ra một dãy sóng . Cứ nhưvậy chúng ta sẽ đổi chỗ cho đủ M sóng .

Bài toán 54 : Xâu nhị phân tương đơng

Đề bài :

Cho 2 xâu A và B có cùng độ dài N (1<=N<=100) chỉ chứa các ký tự 0 và 1 . Ngời ts có thể biến đổi xâu A hoặc B theo quya tắc sau : Nếu một xâu con nào đó chứa một số lợng chẵn số 1 thì ngời ta viết xâu con đó theo trình tự ngược lại . Ví Dụ : A='11010100' ,xâu con từ vị trí 3 đến vị trí 6 có thể được viết ngược lại và ta sẽ nhận được xâu : '11101000' .

Hai xâu A và B được gọi là tương đơng nếu từ A có thể nhận được B bằng các phép biến đổi nêu trên .Hãy lập trình xác định 2 xâu A và B

có tương đơng hay không . Nếu có , hãy chỉ ra một cách biến đổi đa A về B .

Dữ liệu : Vào từ file Convert.Inp :

Dòng đầu tiên chứa xâu A

Dòng thứ hai ghi xâu B

Kết quả : Đa ra file Convert.Out :

Dòng đầu tiên chứa thông báo YES cho trường hợp 2 xâu tương đơng hoặc NO cho trường hợp ngược lại .

Nếu 2 xâu tương đơng , thì các dòng sau xác định các xâu con tham gia biến đổi , mỗi dòng tương ứng với 1 xâu con gồm 2 số nguyên chỉ điểm đầu và điểm cuối của xâu con , xâu con được nêu theo trình tự biến đổi .

Ví Dụ :

Convert.Inp	Convert.Out
100011100	YES
001011001	6 9
	3 8
	1 5

Hướng dẫn :

Bài toán có một đặc điểm rất hay : Nếu như hai xâu vừa cho dài bằng nhau , có cùng số số 1 một thì chúng ta có thể biến đổi từ S1 sang S2 . Chúng ta có thể thực hiện bài toán sau :

Kiểm tra xem chúng có thể biến đổi được tói nhau không

Nêu có thì :

+ Nếu S1[i]<>S2[i] thì

```
dem:=0;
for j:=i to n do
begin
  if s1[j]='1' then inc(dem);
  if (s1[j]=s2[i])and(dem mod 2=0)then
  begin
    str:='';
    for u:=j downto i do
     str[u]:=s1[i+(j-u)];
    for u:=i to j do s1[u]:=str[u];
     writeln(f,i,' ',j);
    break;
  end;
end;
```

Nếu nhưcuối cùng S1vẫn khác S2 thì tức là chúng không thể biến đổi được đến nhau .

Bài toán 55 : sort3

Đề bài :

Sắp xếp là một trong những công việc tính toán hay phải làm nhất. Xét bài toán sắp xếp cụ thể sau đây: Cần sắp xếp các bản ghi theo giá trị của khoá, trong đó các bản ghi cần sắp xếp có nhiều nhất ba giá trị khoá khác nhau. Chẳng hạn, bài toán nhưvậy sẽ phải giải quyết khi ta muốn sắp xếp những ngời được huy chương trong một cuộc thi theo giá trị của huy chương mà họ nhận được, có nghĩa là những ngời đạt huy chương vàng xếp trước, tiếp theo là huy chương bạc và cuối cùng là những ngời đạt huy chương đồng.

Trong bài này ba giá trị có thể có của khoá là các số nguyên 1, 2 và 3. Cần tiến hành sắp xếp theo thứ tự không giảm của giá trị khoá. Việc sắp xếp cần được thực hiện bằng một dãy các thao tác đổi chỗ. Một thao tác đổi chỗ được xác định bởi hai số p và q chỉ rõ cần đổi chỗ hai phần tử ở vị trí p và q cho nhau.

Bạn được cho một dãy các giá trị khoá. Hãy viết chương trình tính số ít nhất các thao tác đổi chỗ cần thực hiện để sắp xếp dãy giá trị khoá đã cho thành một dãy không giảm. (Câu A). Đồng thời, hãy xây dựng dãy các thao tác đổi chỗ tương ứng với cách sắp xếp tìm được. (Câu B).

Dữ liệu vào

Dòng đầu tiên của file INPUT.TXT chứa N là số lợng bản ghi (1<=N<=1000). Mỗi một trong số N dòng tiếp theo chứa một giá trị khoá.

Dữ liệu ra

Ghi ra dòng đầu tiên của file OUTPUT.TXT số lợng nhỏ nhất L các thao tác đổi chỗ cần thực hiện để sắp xếp dãy đã cho thành dãy không giảm (Câu A). N dòng tiếp theo ghi dẫy các thao tác đổi chỗ tương ứng theo trình tự thực hiện. Mỗi dòng chứa một thao tác đổi chỗ được mô tả bởi hai số p và q là các vị trí của hai phần tử được đổi chỗ cho nhau (Câu B). Các vị trí được đánh số từ 1 đến N.

Ví dụ dữ liệu vào và ra

INPUT.TXT	OUTPUT.TXT
9	4
2	1 3
2	4 7
1	9 2
3	5 9
3	
3	
2	

3
1

Hướng Dẫn :

Đếm số lợng các Số 1 , Số 2 , số 3 ,các số lợng này xác định vùng cần đặt từng loại viên bi sau khi sắp xếp theo yêu cầu. Đi từ trái sang phải dãy bi ta tiến hành các xem xét sau:

Nếu có số 2 nằm ở vùng 1 và số 1 nằm ở vùng 2 thì đổi chỗ hai số này

Nếu có số 2 nằm ở vùng 3 , và sô 3 nằm ở vùng 2 thì đổi chỗ hai số này

Nếu có số 3 nằm ở vùng 1 và số 1 nằm ở vùng 3 thì đổi chỗ hai số này

Nếu có số 1 nằm ở vùng 2 , có số 2 nằm ở vùng 3 và có số 3 nằm ở vùng 1 thì đổi chỗ vòng ba số này

Nếu có số 2 nằm ở vùng 1 , có số 3 nằm ở vùng 2 , và số 1 nằm ở vùng 3 thì ta đổi chỗ ba số này sao cho chúng trở về vị trí cần thiết của chúng .

Bài toán 56 : Số mũ cao nhất

Đề bài :

Cho hai số nguyên dương N và P, N, P<=30000. Hãy tìm số M lớn nhất sao cho PM là ớc của N!.

Dữ liệu : Vào từ file Mu.Inp :

Một dòng duy nhất ghi hai số N,P

Kết quả : Ghi ra file Mu.Out :

Một dòng duy nhất ghi số M

Hướng Dẫn :

Trước tiên ta thấy rằng , p là một số nguyên tố , thì gọi Rk là số mũ cao nhất của p thoả mãn K! Chia hết pRk . ta có :

Rk = [K/P]+[K/P2]+.....+[K/Pt]

Trong đó t là số thoả mãn :

pt<=k<=pt

Dựa vào bài toán phụ trên ta sẽ thực hiện bài toán bằng cách : Phân tích N thành các số nguyên tố , với mỗi số nguyên tố ta tính Rk sau đó chọn Rk min là số thoả mãn .

Bài toán 57 : Chia đôi bảng

Đề bài :

Ngời ta cho một ma trận NxN (N<=100) . Trên đó ghi các số tự nhiên . Ngời ta muốn chia bảng đó làm hai phần sao cho tổng số các số tự nhiên ở hai phía phải chênh lệch ít nhất . Chính vì thế ngời ta cho một

robot (robot giả nào đó - natural) đi trên bảng đó . Đòng đi đó sẽ cắt đôi bảng làm hai phần . Nhưng các số trên đường đi đó không được tính là phần nào cả .
Yêu cầu : Tìm cho con robot đó chia bảng bằng cách đi nhưthế nào
Dữ liệu : Vào từ file Chiabang.Inp :
Dòng đầu tiên là số N
N dòng sau biểu diễn ma trận đó
Kết quả : Ghi ra file Chiabang.Out :
Dòng đầu tiên ghi số chênh lệch giữa hai phần
Các dòng tiếp theo , mỗi dòng ghi hai số lần lợt biểu diễn các toạ độ đi qua của robot đó .
Hướng Dẫn :

Bài toán 58 : Tìm cách đặt nớc

Đề bài :

Cho một mảnh vờn hình vuông có cạnh bằng N . Ngời ta chia vờn này thành các mảnh ô vuông nhỏ . Vì diện tích của vờn là rất lớn , mặt khác chúng ta phải cung cấp nớc tới cho vờn . Chính vì thế cần phải bố trí các vùng nớc sao cho chỗ nào trong vờn cũng được tới nớc . Biết rằng chi phí đào đất , cũng nhưnếu nhưmất nhiều đất thì sẽ phải tốn rất nhiều tiền (và không đủ đất để trồng cây) . Mặt khác , công suất của máy tới chỉ bằng một mảnh vờn , tức là nó chỉ tới được các ô cùng chung cạnh với nó .
Yêu cầu : Hãy tìm cách bố trí các vũng nớc sao cho ít vũng nhất .
Dữ liệu : Vào từ file Daoao.Inp :
Dòng đầu tiên chứa số N
Kết quả : Ghi ra file Daoao.Out :
N dòng , mỗi dòng N số biểu diễn mảnh vờn đó . Trong đó ô ghi số 1 sẽ là ao , còn ghi 0 sẽ là vờn
Ví Dụ :

DAOAO.INP	DAOAO.OUT
4 0 0 0 0 1 0	0 1 0 0 0 0 0 1 1 0

Hướng Dẫn :

Duyệt các vị trí có thể đặt . Nhưng sau khi đặt một ô nào đó thì các ô đã được tới sẽ không phải duyệt đến nữa . Sử dụng thêm nhánh cận về cách đặt tham lam nào đó (bất kỳ - giả sử nhưcứ lấy ô (1,2) làm ao , rồi tiếp tục lấy các ô theo tuần tự trên xuống dưới ,trái sang phải màchưa được tới sẽ làm tới) .

Bài toán 59 : Gỡ bài

Đề Bài :

Có N cột bài xếp thành một hàng ngang được đánh số từ 1 tới N , cột thứ i có Ai quân . Mỗi quân bài có một mầu nào đó trong k màu , các mầu được đánh số từ 1 tới k nào đó . Ngời ta cho phép lấy các quân bài ra theo quy tắc sau : Mỗi bớc được lấy ra các quân bài cùng màu nằm ở phía trên ở một số cọc nằm liên tiếp nhau . Với một trạng thái N cọc bài cho trước , hãy tìm số bớc ít nhất lấy bài để lấy hết tất cả các quân bài ra khỏi hàng .

Dữ liệu : Vào từ file Bai.Inp trong đó :

Dòng đầu tiên chứa số N (N<=100)

Dòng thứ i trong số N dòng tiếp theo chứa : Đầu dòng là số ai (ai<=100): số bài trên cọc thứ i , tiếp theo là ai số lần lợt cho biết màu của các quân bài trên cọc i được liệt kê từ dưới lên trên .

Kết quả : Ghi ra file Bai.Out trong đó ghi một số duy nhất là số bớc tìm được

Ví Dụ :

BAI.INP	BAI.OUT
2 2 1 2 3 3 1 2	3

Hướng Dẫn :

Bài toán 60 : Số đặc biệt

Đề bài :

Xét tập các số nguyên dương ,mỗi số trong tập ở dạng thập phân không có chữ số giống nhau . Các số trong tập được sắp xếp theo thứ tự tăng dần và được đánh số bắt đầu từ 1 trở đi . Hãy lập trình để từ số thứ tự tìm được số trong tập và ngược lại .

Dữ liệu : Vào file NUMBER.INP :

Mỗi dòng trừ dòng cuối cùng có 1 trong 2 dạng :

+ 1 N - Yêu cầu tìm số thứ N trong tập , N không quá 10 chữ số .
+ 2 số nguyên trong tập - yêu cầu tìm thứ tự của số nguyên này

File kết thúc bằng dòng chứ 1 số 0 .

Kết quả : Đa ra file NUMBER.OUT :

Bắt đầu bằng số 1 hoặc 2 (tuỳ theo dòng tương ứng của dữu liệu vào) , 1 dấu cách và sau đó là kết quả tương ứng .

Ví Dụ :

NUMBER.INP	NUMBER.OUT
1 14 2 102 0	1 15 2 91

Hướng Dẫn :

Bài toán 61 : Tính toán

Đề bài :

Cho hai phép toán *2 (nhân với 2) và /3(chia nguyên cho 3) . Cho trước số 1 . Bằng cách sử dụng hai phép toán trên ta xây dựng được các biểu thức cho giá trị là một số tự nhiên .

Ví Dụ :

1*2*2*2*2*2/3/3*2=6

(các phép toán được thực hiện từ trái sang phải)

Dữ liệu : Cho từ file Num.Inp : Chứa M dòng (M<=15) , mỗi dòng là một số tự nhiên có không quá 70 chữ số .

Kết quả : Hãy xác định các xâu tính toán nhưtrên (viết trên mỗi dòng) mà cho kết quả tương ứng nhưfile input .

Bài toán 62 : UGLY NUMBERS

Đề bài :

Chúng ta nói số X là một số " Ugly Numbers " là khi số đó phân tích ra thừa số nguyên tố thì thừa số đầu tiên là số 2,3 hoặc 5. Các số này khi được viết theo một thứ tăng dần thì ta có dãy các số "Ulyg Numbers" nhưsau:

2,3,4,5,6,8,9,10,12,14,15,16.....

Chúng ta có hai nhiệm vụ cần đặt ra là:

Nhệm vụ 1: Cho trước số i ,chúng ta cần tìm số nào là số có vị trí thứ i trong dãy 'Ugly Numbers "

Nhiệm vị 2: Cho trước số i , chúng ta cần tìm vị trí của nó trong dãy 'Ugly Number Dữ liệu Vào từ file : UGLY.INP Gồm nhiều bộ dữ liệu . Mỗi dòng ghi hai số x,y.

Nếu x=0 thì chúng ta cần thực hiện nhiệm vụ 1 đối với y.

Nếu x=1 thì Chúng ta cần thực hiện nhiệm vụ 2 đối với y.

Kết quả ghi ra file : UGLY.OUT nhưsau:

Mỗi dòng ghi ứng với một bộ số : X,Y . Trong đó x=0 hoặc 1 ứng với dòng tương ứng trong dữ liệu vào . Còn y là kết quả của nhiệm vụ đặt ra với dữ liệu của dòng tương ứng trong tệp vào.

Ví Dụ:

UGLY.INP	UGLY.OUT
0 6 0 10 1 5 1 20	0 8 0 14 1 4 1 14

Bài toán 63: Tráo bài

Đề Bài :

Giả thiết có 2n lá bài, đánh số từ 1 đến 2n. Ban đầu các lá bài được sắp theo thứ tự từ 1 đến 2n. Quy tắc tráo bài là nhưsau: Sau một lần tráo bài từ trật tự ban đầu ta có trật tự ban đầu ta có trật tự các bài là n+1, 1, n+2, 2, ... 2n, n. nhưvậy, n lá bài đầu tiên sẽ nằm ở các vị trí 2, 4, 6 ... 2n. Các lá bài còn lại sẽ ở các vị trí lẻ: 1, 3, 5, ... 2n-1, Với số nguyên n cho trước, sau một số lần tráo, ta lại nhận được trình tự ban đầu của bộ bài. Hãy lập trình xác định số lần tráo để có lại trình tự ban đầu.
Dữ liệu: Vào từ file SHUFFLE.INP, mỗi dòng 1 số nguyên n (n < 10000).
Kết quả: Đa ra file SHUFFLE.OUT, mỗi dòng một số nguyên - kết quả ứng với dòng tương ứng của file dữ liệu vào.
Ví dụ:

SHUFFLE.INP	SHUFFLE.OUT
10	6
20	20

Hướng Dẫn :

Thực chất ta cần tìm hàm F(x) là số lần tráo bài để quân bài X trở về vị trí X . Ta chỉ cần tìm hàm F(1) .

Bài toán 64 : (P,Q)-số

Đề bài :

Cho hai số nguyên dương P và Q. Một số M được gọi là một (P,Q)-số nếu M thoả mãn *một trong các điều kiện* sau:

1. M chia hết cho P hoặc M chia hết cho Q.
2. Trong biểu diễn thập phân của M có P hoặc Q xuất hiện.
3. Trong biểu diễn thập phân của M có một chữ số xuất hiện ít nhất P hoặc Q lần liên tiếp.

Ví dụ nếu P = 3 và Q = 7 thì các số sau là các (3,7)-số: 14, 21, 13, 75, 24111145, 100000000.

Nếu P = 13 và Q = 21 thì các số sau là (13,21)-số: 42, 31388, 82199, 69, 15555555555555556.

Với mỗi (P,Q)-số M, ta có thể biến đổi thành số khác nhờ một trong các biến đổi sau:

1. Nếu M chia hết cho P/Q thì chia M cho P/Q, ta ký hiệu phép biến đổi này là C P/Q.
2. Nếu P/Q xuất hiện trong M tại vị trí thứ I tính từ trái sang phải, xoá số P/Q đó và nếu có J số 0 vô nghĩa xuất hiện ở đầu số mới nhận được, cũng xoá luôn, ta ký hiệu biến đổi này là B P/Q I J (J >=0).
3. Nếu trong biểu diễn thập phân của M có một chữ số xuất hiện P/Q lần liên tiếp bắt đầu từ vị trí I tính từ trái sang phải thì xoá P/Q chữ số đó, ta ký hiệu biến đổi này là X P/Q I.

Cho trước một (P,Q)-số M, cần giải quyết hai bài toán sau:

BT1. Hãy tìm một dãy biến đổi ít nhất để biến M thành một số nguyên dương M1 không là (P,Q)-số.

BT2. Hãy tìm một dãy các biến đổi để biến M thành một số nguyên dương M2 không là (P,Q)-số sao cho M2 lớn nhất.

Dữ liệu vào được cho bởi file PQ.INP gồm một dòng ghi ba số M, P, Q, M, P, Q<=230. Kết quả ghi ra file PQ.OUT nhưsau. Một nhóm dòng ghi kết quả BT1 gồm dòng thứ nhất ghi số M1, dòng thứ hai ghi số S1 là số biến đổi cần dùng, sau đó là S1 dòng ghi mỗi dòng ký hiệu một biến đổi mà theo trình tự đó, M biến thành M1. Tiếp theo là một nhóm dòng ghi kết quả BT2 gồm dòng thứ nhất ghi số M2, dòng thứ hai ghi số S2 là số biến đổi cần dùng, sau đó là S2 dòng ghi mỗi dòng ký hiệu một biến đổi mà theo trình tự đó, M biến thành M2.

Ví dụ

PQ.INP	PQ.OUT	(Tiếp tục PQ.OUT)
999 3 7	999	11
	11	3
	3	C 3
	C 3	C 3
	C 3	B 3 1
	B 3 1	C 3
	C 3	

Bài toán 65 : Kim Cơng hay Rubi

Đề bài :

Hai ngời thợ đào đá quý P và Q sàn được N viên đá quý G1,G2,...Gn thuộc hai loại kim cơng và rubi . Với mỗi viên đá quý Gi biết :

Vi là giá trị tính bằng USD

Ti loại đá , ti=1 nếu nó là kim cơng , ti=0 nếu nó là rubi

P nhận trách nhiệm chia phần . Anh ta muốn chia đá quý thành hai phần P , Q sao cho :

Số lợng viên Rubi trong phần P không ít hơn số lợng rubi trong Q

Số lợng viên kim cơng trong phần P không nhiều hơn số lợng kim cơng trong Q

chênh lệch :

= (tiền của P) - (tiền của Q) là nhỏ nhất

Dữ liệu : Vào từ file GEMS.INP :

Dòng đầu tiên ghi số lợng đá quý N (N<=100)

Dòng thứ i trong số N dòng tiếp theo chứa hai số Vi và Ti

Kết quả : Ghi ra file GEMS.OUT :

Dòng đầu tiên chứa là chênh lệch theo cách chia phần tìm được

Dòng thứ i trong số N dòng tiếp theo ghi ký tự P nếu viên đá Gi nằm trong phần P , ghi ký tự Q nếu nó nằm trong phần Q.

Ví dụ :

GEMS.INP	GEMS.OUT
5	5
13 1	P
25 1	Q
20 1	Q
15 0	P
12 0	P

Bài toán 66 : ĐỔI CHỖ

Đề bài :

Ngời ta định một quan hệ hai ngôi đối xứng trên tập hợp các chữ số {0, 1, ..., 9} gọi là quan hệ có thể đổi chỗ được.

Một số nguyên bất kỳ được biến đổi theo cách sau: đổi chỗ chữ số thứ i và chữ số thứ i+1 nếu hai chữ số này thuộc quan hệ đang xét. Phép biến đổi nhưthế được ký hiệu là i (các chữ số trong một số nguyên được xếp thứ tự 1, 2, ... theo chiều từ trái sang phải).

Hãy tìm dãy biến đổi ngắn nhất trên một số nguyên cho trước sao cho thu được một số nguyên lớn nhất.

Dữ liệu: đọc vào từ file văn bản DC.INP gồm:

dòng đầu ghi số cặp chữ số có thể đổi chỗ được,

các dòng tiếp theo, mỗi dòng ghi một cặp chữ số có thể đổi chỗ được, các chữ số ghi cách nhau ít nhất một dấu trắng,

dòng cuối ghi số nguyên cần biến đổi, các chữ số ghi sát nhau.

Kết quả: ghi ra file văn bản DC.OUT gồm:

dòng đầu ghi số phép biến đổi (có thể bằng 0),

dòng tiếp ghi số hiệu các phép biến đổi (cách nhau ít nhất một dấu trắng) theo đúng thứ tự để được số nguyên lớn nhất (nếu số phép biến đổi bằng 0 thì dòng này bỏ qua).

Giới hạn: số nguyên cho trước không quá 1000 chữ số, có thể có những chữ số 0 ở đầu.

Thí dụ:

DC.INP	DC.OUT
7 1 7 4 3 6 4 5 6 5 4 9 2 0 8 3546	3 2 1 3

Hướng Dẫn :

Gọi số đó là : A1...An

Chúng ta sẽ u tiên các số từ trái qua phải về sự lớn nhất . Tức là với mỗi Ai thì chúng ta tìm các giá trị Aj , j=i+1,..n thoả mãn Aj có thể đổi được với Ai và Aj là lớn nhất , số phép chuyển là ít nhất .