Chương I: QUY HOẠCH ĐỘNG

Các Bài toán quy hoạch động chiếm một vị trí khá quan trọng trong việc tổ chức hoạt động và sản xuất (Nhất là việc giải quyết các bài toán tối ưu). Chính vì lẽ đó mà trong các kỳ thi học sinh giỏi Quốc Gia và Quốc Tế chúng ta thường gặp loại toán này. Tư tưởng chủ đạo của phương pháp này dựa trên nguyên lí tối ưu của BellMan phát biểu như sau :

"Nếu một dãy các lựa chọn là tối ưu thì mọi dãy con của nó cũng tối ưu "

Ngoài ra khi thiết kế các thuật toán quy hoạch động ta thường dùng kỹ thuật "Phân vùng để xử lí", Nghĩa là để giải quyết một bài toán lớn ta chia nó thành nhiều bài toán con có thể giải quyết độc lập. Trong phương pháp quy hoạch động, việc thể hiện nguyên lí này được đẩy đến cực độ. Để giải quyết các bài toán quy hoạch động ta có thể theo sơ đồ sau :

a.) Lập hệ thức : Lập hệ thức biểu diễn tương quan quyết định của bước đang xử lí với các bước đã xử lí trước đó. Hệ thức này thường là các biểu thức đệ quy do đó dễ thấy hiện tượng tràn bộ nhớ.

b.) Tổ chức dữ liệu chương trình : Tổ chức giữ liệu tính toán dần theo từng bước. Nên tìm cách khử đệ quy. Thông thường, trong các bài toán tin chúng ta hay gặp đòi hỏi một vài mảng lớn.

c.) Làm tốt : Làm tốt thuật toán bằng cách thu gọn hệ thức quy hoạch động và giảm kích thước miền nhớ.

Các thao tác tổng quát của quy hoạch động :

1. Xây dựng hàm quy hoạch động
2. Lập bảng lưu lại giá trị của hàm
3. Tính các giá trị ban đầu của bảng
4. Tính các giá trị còn lại theo kích thước tăng dần của bảng cho đến khi đạt được giá trị tối ưu cần tìm
5. Dùng bảng lưu để truy xuất lời giải tối ưu .

Trong các lời hướng dẫn các bài toán, chúng tôi sẽ đưa các bạn đi theo từng phần như sơ đồ giải quyết trên. Chúng ta có thể phân loại các bài toán quy hoạch động theo nhiều cách. Để các bạn tiện theo dõi, tôi xin phân loại theo cách lưu (tức là tổ chức chương trình) là các mảng một chiều hay nhiều chiều.

I. Dạng Một:

Đa Phần dạng bài toán thường gặp trong loại này đó là loại có công thức truy hồi như sau :

Mind[I]:=Min Mind[J] +Giá Trị Để JI ;J=0..I Hoặc là :

Maxd[I]:=MaxMaxd[J]+Giá Trị Để JI ;J=0..I .

Chúng ta có thể thấy rõ ràng đối với các bài toán mà chúng ta sẽ xét sau đây :

Bài Toán 1: Bài Đổi tiền

Đề bài :

"Một ngân hàng có N loại tiền mệnh giá A[1],A[2],...A[N] với số lượng tiền mỗi loại không giới hạn. Cần chi trả cho khách hàng một số tiền M đồng. Hãy cho biết cần bao nhiêu tiền mỗi loại để chi trả sao cho số lượng tờ là ít nhất.

Dữ liệu vào từ File : Tien.Inp như sau :

Dòng đầu tiên ghi 2 số N,M . (N<=100,M<=10000)

Dòng thứ hai ghi N Số : A[1], A[2],...A[N]

Kết quả: ghi ra File : Tien.Out như Sau :

Dòng Đầu tiên ghi số tờ cần dùng, Nếu không thể đổi được thì ghi số 0 và không cần thực hiện tiếp.

Dòng tiếp theo ghi số n số biểu hiện cho số tờ cần dùng cho mỗi loại.

"

Hướng Dẫn :

Chúng ta gọi Mind[I] là số lượng tờ ít nhất để trả số tiền I, như vậy bài toán yêu cầu chúng ta xác định Mind[M]. Ta nhận thấy rằng để được số tiền là I thì chúng ta sẽ có các cách để tạo thành số tiền đó khi chúng ta dùng thêm một tờ Là:
I-A[K1],I-A[K2],...I-A[KJ] ,Trong đó KJ Là số Thoả mãn mà A[KJ]<I. Vậy để số tiền tối ưu nhất là chúng ta cần tìm thấy trong các Mind[I-A[K1]]+1, Mind[I-A[K2]]+1,...Mind[I-A[KJ]]+1 . Có Công thức quy hoạch động như sau:

Mind[I]:=MinMind[I-A[J]]+1, J Thoả Mãn : A[J]<I

Từ đó chúng ta có thủ tục quy hoạch động như sau :

```
Procedure Quy_Hoach_Dong;
Begin
        Mind[0]:=0;
        For I:=1 To M Do
         Begin
                Min:=Maxint;
                For J:=1 To N Do
                 If (Mind[I-A[J]]+1<Min)And (A[J]<I) Then
                 Begin
                        Min:=Mind[I-A[J]]+1;
                        Luu[I]:=J;
                 End;
                Mind[I]:=Min;
        End;
End;
```

Trong đó mảng luu là mảng chứa đựng là loại tiền nào cần dùng cuối cùng để đến số tiền I . như vậy chúng ta có cách để tìm lại các loại tiền cần dùng bằng mảng Luu như sau :

```
J:=Luu[M];
I:=M;
While J<>0 Do
Begin
        Write(A[J]);
```

```
        I:=I-J;
        J:=Luu[I];
    End;
```

như vậy chúng ta sẽ giải quyết bài toán trên một cách ngắn gọn và đơn giản. Để tăng tính tự sáng tạo của các bạn, kể từ bài toán sau chúng tôi chỉ nêu qua các thủ tục và công thức quy hoạch động. Nếu các bạn không giải quyết được vấn đề nhỏ đó thì có thể tham khảo phần lời giải của chúng tôi. Tiếp sau đây là một loạt bài toán tương tự bài toán 1 mà thực chất chúng chỉ là một dạng bài cố định, nó chỉ biến dạng đi về lời lẽ nhưng đều giống nhau về bản chất .

Bài Toán 2: Bài toán nối điểm (Wires)

Đề :

" Trên hai đường thẳng song song L1 và L2, Người ta đánh dấu trên mỗi đường N Điểm, Các điểm trên đường thẳng L1 Được đánh số từ 1 đến N, từ trái qua phải, còn các điểm trên đường thẳng L2 được đánh số bởi P[1],P[2],...P[N] cũng từ trái qua phải, trong đó P[1],P[2],..P[N] là một hoán vị của các số 1,2,...N

Ta gọi các số gán cho các điểm là số hiệu của chúng. Cho phép nối hai điểm trên 2 đường thẳng có cùng số hiệu.

Yêu Cầu : Tìm cách nối được nhiều cặp điểm nhất với điều kiện các đoạn nối không được cắt nhau.

Dữ Liệu : Vào từ File BaiToan2.Inp:

Dòng đầu tiên chứa số nguyên dương N(N<=1000)

Dòng thứ hai chứa các số P[1],P[2],....P[N]

Kết Quả Ghi Ra File : BaiToan2.Out

Dòng Đầu tiên chứa K là số lượng đoạn nối tìm được

Dòng tiếp theo chứa K số hiệu của các đầu mút của các đoạn nối được ghi theo thứ tự tăng dần.

Ví Dụ :

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

"

Hướng Dẫn :

Gọi Maxd[I] là số đoạn thẳng tối đa của các cặp nối của các điểm t 1I. Chúng ta sẽ có công thức quy hoạch động như sau :

Maxd[I]:=MaxMaxd[P[J]]+1;J:=0..ViTri(I) Trong đó ViTri (I) Là hàm cho biết Vị trí Của I Trong Dãy P[1],P[2],..P[N] (Tức là I=P[X] Thì ViTri(I)=X);

Bằng cách phân tích hoàn toàn tương tự như bài toán 1 mà ta có công thức truy hồi như trên . Các bước giải bài toán có thể được nói gọn trong hai thủ tục và hàm :

```
Funtion ViTri(I:Integer):Integer;
Var J:Integer;
Begin
```

```
                For J:=1 To N Do
                 If P[J]=I Then
                 Begin
                        ViTri:=J;
                        Exit;
                End;
        End;
```

Thủ Tục Quy Hoạch động như sau :

```
        Procedure Quy_Hoach_Dong;
        Begin
                Maxd[0]:=0;
                For I:=1 To N Do
                 Begin
                        Max:=0;
                        For J:=0 To ViTri(I) Do
                         If Maxd[P[J]]>Max Then
                         Begin
                                Luu[I]:=p[J];
                                Max := Maxd[P[J]];
                         End;
                        Maxd[I]:=Max;
                 End;
        End;
```

Mảng Luu là mảng luu[I] lưu lại điểm trước I mà tiếp đó sẽ nối I. Chính vì điều đo chúng ta có thể ghi ra ngược lại một cách dễ dàng. Hoàn tương tự, chúng ta có thể giải quyết tương tự cho các bài toán sau :

Bài Toán 3: Dạo Chơi Bằng Xe Buýt

Đề :

" Trên một tuyến đường ở thành phố du lịch nổi tiếng X có ô tô Buýt công cộng phục vụ việc đi lại của du khách. Bến xe buýt có ở từng Km của tuyến đường. Mỗi lần đi qua bến xe đều đỗ cho du khách lên xuống. Mỗi bến đều có xe xuất phát từ nó, nhưng mỗi xe chỉ chạy không quá B Km kể từ bến xuất phát của nó. Hành khách khi đi xe sẽ phải trả tiền cho độ dài đoạn đường mà họ ngồi trên xe. Cước phí cần trả để đi đoạn đường độ dài i là Ci (I=1,2,..B). Một du khách xuất phát từ một bến nào đó muốn đi dạo L Km trên tuyến đường nói trên. Hỏi ông ta phải lên xuống xe như thế nào để tổng số tiền phải trả cho chuyến dạo chơi bằng xe buýt là nhỏ nhất.

Dữ Liệu : Vào Từ File : Bus.Inp

Dòng đầu tiên chứa 2 số nguyên dương B,L (B<=100,L<=10000)

Dòng thứ hai chứa B số Nguyên dương C1,C2,..Cn , được ghi cách nhau bởi dấu trắng.

Kết Quả : Ghi ra File Văn bản : Bus.Out

Dòng đầu tien ghi chi phí tìm được, và số lần xuống xe K .

Dòng tiếp theo ghi K số là độ dài của các đoạn đường của K lần ngồi xe.
Ví Dụ:

BUS.INP	BUS.OUT
10 15	147 3
12 21 31 40 49 58 69 79 90 101	3 6 6

"
Hướng Dẫn :
Gọi Mind[I] Là số tiền ít nhất cần trả khi người đó cần đi I Km . Chúng ta sẽ có công thức quy hoạch động :
Mind[I]:=MinMind[I-J]+C[J]; J: J<=I ;
Giá trị Mind[L] là gía trị cần tính .

Bài Toán 4: Dãy Con Tăng Cực Đại

Đề :
" Cho một dãy số nguyên dương A1,A2,.. An. Hãy tỉa bớt một số ít nhất các phần tử của dãy số nguyên đó vẫ giữ nguyên thứ tự các phân tử còn lại sao cho dãy số còn lại là một dãy tăng dần. Ta gọi dãy số nguyên tăng dần còn lại sau khi đã tỉa bớt một số phần tử là dãy con của dãy đã cho.
Dữ Liệu : Vào từ File BaiToan4.Inp :
Dòng đầu tiên ghi số N là số phần tử (N<=10000)
Dòng tiếp theo ghi N số là các số nguyên của dãy
Kết Qủa : Ghi Ra FIle : BaiToan4.Out
Dòng đầu tiên ghi số phần tử của dãy con lớn nhất đó
Dòng thứ hai ghi các số của dãy cần tìm .
Hướng Dẫn :
Gọi Maxd[I] là số phần tử lớn nhất của dãy con dài nhất của các phần tử từ 1I . Chúng ta sẽ có công thức quy hoạch động :
Maxd[I]:=MaxMaxd[J]+1; Với J=1..I-1 , và A[J]<A[I]

Bài Toán 5: Bố Trí Phòng Họp

Đề bài :
Có N cuộc họp đánh số từ 1 đến N đăng ký làm việc tại một phòng hội thảo. Cuộc họp i cần được bắt đầu tại thời điểm Ai và kết thúc tại thời điểm Bi (i=1,2,...N). Hai cuộc họp bất kỳ chỉ được nhận phục vụ nếu các khoảng thời gian làm việc tương ứng chỉ có thể được giao nhau tại đầu mút. Hãy tìm một lịch cho phòng hội thảo để có thể phục vụ được nhiều cuộc họp nhất .
Dữ Liệu: Vào được cho trong file Activity.Inp gồm :
Dòng đầu tiên ghi giá trị N .
Dòng thứ i trong số N dòng tiếp ghi 2 số nguyên Ai và Bi cách nhau ít nhất một dấu trắng .

Kết Quả : Cần ghi ra file Activity.Out như sau :

Dòng đầu tiên ghi giá trị K là số cuộc họp tối đa có thể bố trí được

K dòng tiếp theo, mỗi dòng ghi số hiệu của cuộc họp được phục vụ theo trình tự lịch bố trí.

Giới Hạn kích thước :

N không quá 10000

Các giá trị Ai, Bi (i=1,2,..N) không quá 32000.

Ví Dụ:

Activity.Inp	Activity.Out
5	3
1 3	1
2 4	4
1 6	5
3 5	
7 9	

Hướng Dẫn :

Chúng ta gọi Maxd[i] là số cuộc họp nhiều nhất có thể bố trí nếu có cuộc họp i ở trong đó. Ta sẽ có :

Maxd[i]:=MaxMaxd[j]+1; j=0..i-1 ;

Sau đó số cuộc họp nhiều nhất có thể bố trí là giá trị lớn nhất trong số các Maxd[i].

Chú ý : bài toán trên có thể thay đổi cách ra đề : coi một cuộc họp là một lần ghi âm chẳng hạn. Chính vì thế thực chất bài CDWrite và bài này là một (nếu các bạn đã đọc bài CDWrite, nếu cha thì các bạn chỉ cần làm bài này thôi).

Bài toán 6: Vòng Quanh Thế Giới

(Đề Thi Học Sinh Giỏi Quốc Gia 2000-2001 - Bảng A)

-Đề :

Trên tuyến đường của xe chở khách du lịch vòng quanh thế giới xuất phát từ bến X có N khách sạn đánh số từ 1 đến N theo thứ tự xuất hiện trên tuyến đường, trong đó khách sạn N là địa điểm cuối cùng của tuyến đường mà tại đó xê bắt buộc phải dừng. Khách sạn I cách địa điểm xuất phát Ai Km (I=1,2,..N);A1<A2<...<AN

Để đảm bảo sức khoẻ cho khách hàng, theo tính toán của các nhà chuyên môn, sau khi đã chạy được P (Km) xe nên dừng lại cho khách nghỉ ở khách sạn. Vì thế, nếu xe dừng lại cho khách nghỉ ở khách sạn sau khi đã đi được Q(Km) thì lái xe phải trả một lượng phạt là : (Q-P)^2.

Ví Dụ :

Với N=4 ,P=300,A1=250, A2=310, A3=550 ,A4=590 . Xe bắt buộc phải dừng lại ở khách sạn 4 là địa điểm cuối cùng của hành trình. Nếu trên đường đi lái xe chỉ dừng lại tại khách sạn thứ 2 thì lượng phạt phải trả là :

(310-300)^2+((590-310)-300)^2=500

Yêu Cầu : Hãy xác định xem trên tuyến đường đến khách sạn N, xe cần dừng lại nghỉ ở những khách sạn nào để tổng lượng phạt mà lái xe phải trả là nhỏ nhất.

Dữ Liệu : Vào từ File văn bản có tên Bai5.Inp :

Dòng đầu tiên chứa số nguyên dương N(N<=10000);

Dòng thứ hai chứa số nguyên dương P (P<=500);

Dòng thứ ba chứa các số nguyên dương A1,A2,A3,..An (hai số liên tiếp cách nhau ít nhất bởi 1 dấu cách) (Ai<=2000000 ,i=1,2,..N)

Kết Quả : Ghi ra File Văn Bản Bai5.Out:

Dòng đầu tiên ghi Z là lượng phạt mà lái xe phải trả ;

Dòng thứ hai ghi K là tổng sô khách sạn mà lái xe cần dừng lại cho khách nghỉ;

Dòng thứ ba chỉ chứa chỉ số của K khách sạn mà xe dừng lại cho khách nghỉ (Trong đó nhất thiết phải có khách sạn thứ N)

Ví Dụ:

BAI5.INP	BAI5.OUT
4	50
300	2
250 310 550 590	2 4

Hướng dẫn :

Gọi Mind[i] là lượng phạt ít nhất nếu người lái xe dừng lại địa điểm i. Chúng ta sẽ có công thức truy hồi :

Mind[i]:=MinMind[j]+sqr(a[i]-a[j]-p); j=1,..i-1

Tuy nhiên bài toán này chưa phải là đã được giải quyết. Bởi vì nó còn quá nhiều vấn đề cần giải quyết khác :

Lượng phạt có thể rất lớn, vượt quá longint mà nếu chứa trong real thì sẽ không thể lưu được mảng có 10000 phần tử. Chính vì thế chúng ta cần giải quyết tốt dữ liệu bài toán này.

Nếu N=10000 thì chương trình sẽ phải chạy : (9999+1)*9999/2 . Tức là rất lâu.

Chính vì thế các bạn cần phải hoàn thành một cách đúng đắn các điều kiện trên .

Hoàn toàn biến dạng về ngôn ngữ diễn tả bài toán, nhưng có rất nhiều bài toán đã ẩn rõ hơn về thuật toán này, chúng ta xét các bài toán sau :

Bài toán 7 : Car

Đề bài :

Cho một đoàn xe hộ tống có n chiếc đi trên một đường một chiều đã được bố trí theo thứ tự từ 1 đến n. Mỗi một xe trong đoàn tren thì có một vận tốc là V và trọng lượng là W.

Khi đi qua một chiếc cầu có trọng tải không quá là P thì phải chia đoàn xe trên thành các nhóm sao cho tổng trọng lượng của mỗi một nhóm là không quá P. Thêm vào đó nữa là các nhóm phải đi tuần tự. Nghĩa là nhóm thứ i chỉ đi được khi mà toàn bộ xe của nhóm thứ i-1 đã qua cầu.

Vận tốc đí của mỗi một nhóm là hoàn toàn khác nhau và phụ thuộc vào xe có tốc độ chậm nhất có thể được.

Dữ Liệu: vào từ file Car.Inp gồm :

Dòng đầu tiên ghi 3 số n(n 1000) và P,L thể hiện cho số xe, trọng lượng tối đa của cầu và L là độ dài của cầu.

N dòng kế tiếp , mỗi dòng gồm 2 số W và V thể hiện cho trọng lượng và vận tốc của xe

Kết Quả : ra file Car.Out như sau:

Dòng đầu tiên là tổng thời gian nhỏ nhất để đoàn xe qua cầu

Dòng kế tiếp gồm các số X1,X2,...Xk thể hiện: Nhóm 1là từ 1..X1,nhóm 2 là từ X1+1 .. X2,...

Ví Dụ :

car.Inp	car.Out
10 100 100 40 25 50 20 70 10 12 50 9 70 49 30 38 25 27 50 19 70	25 25 1 3 6 8 10

Hướng Dẫn :

Gọi Mind[i] là tổng thời gian nhỏ nhất để cho đoàn xe có số hiệu từ 1 đến i qua cầu . Ta có công thức truy hồi :

Mind[i]:=MinMind[j]+Time(j,i) ; j = 1,..i-1

Với time(j,i) là thời gian để cho đoàn xe gồm từ chiếc thứ j cho tới chiếc thứ i qua cầu cùng một lúc (có nghĩa là nếu vượt quá trọng tải thì Time(j,i)= .)

Bài toán 8 : Khuyến Mại

Đề bài :

Vào ngày No-en, N cửa hàng trong thành phố tặng quà cho các khách hàng. Các cửa hàng có tên 1 .. N , N<=100. Một lần tặng quà được thể hiện bằng ba số nguyên dương X , Y , Z với ý nghĩa cửa hàng X tại thời điểm Y tặng món quà giá trị Z . Với mỗi lần tặng quà, người muốn nhận phải có mặt không muộn hơn thời điểm phát quà và các vủă hàng đều rất chu đáo đến mức thời gian nhận quà xem như bằng 0. Không có hai lần tặng quà nào diễn ra tại một thời điểm .

An muốn tận dụng ngày đó để hy vọng có nhiều món quà hấp dẫn. An xuất phát từ nhà tại thời điểm 0 và phải quay về đến nhà không muộn hơn thời điểm M. Hãy lập cho An kế hoạch đi nhận quà sao cho tổng số giá trị thu được là lớn nhất.

An biết được thời gian cần đi từ nhà đến từng cửa hàng và giữa các cửa hàng cũng như chi phí trên từng chặng đường tương ứng. Thời gian và chi phí trên mỗi chặng đường là tối ưu, không nhất thiết như nhau theo hai chiều. Trong khi đi từ một cửa hàng này đến mộ cửa hàng khác, An không ghé thăm qua cửa hàng nào khác. Tổng giá trị An thu được bằng tổng giá trị quà tặng được trừ đi tổng chi phí mà An phải trả trên các chặng đường từ nhà đến cửa hàng đầu tiên, từ đó lần lượt đến các cửa hàng khác và quay về đến nhà .

Dữ liệu : vào được cho bởi file văn bản : KM.INP trong đó dòng thứ nhất ghi ba số N , M , K mà K là số lần phát quà, N<=100 , M<=60000, K <= 5000. Tiếp theo là K dòng, dòng thứ i trong K dòng này ghi ba số X , Y , Z thể hiện một lần phát quà và ta quy ước gọi lần phát quà thứ i. Sau đó là N + 1 dòng, dòng thứ i ghi N + 1 số mà số thứ j là thời gian đi từ cửa hàng i đến cửa hàng j. Cuối cùng là N + 1 dòng, dòng thứ i trong n + 1 dòng ghi N + 1 số, mà số thứ j là chi phí để đi từ cửa hàng thứ i đến cửa hàng thứ j. Nhà của An xem như cửa hàng rhứ N + 1. Với mọi cửa hàng i, thời gian và chi phí từ i đến i là bằng 0. Tổng giá trị lớn nhất AN thu được không quá 2 tỷ.
Kết Quả : ghi ra file : KM.OUT như sau :

Dòng thứ nhất ghi số S là tổng giá trị An thu được

Tiếp theo là một số dòng, mỗi dòng ghi số hiệu một lần nhận qùa mà theo trình tự đó An lần lượt đến nhận.

Ví Dụ :

KM.INP KM.OUT

2 13 5 1 2 10 2 3 20 1 4 25 1 5 10 2 10 10 0 2 2 2 0 3 1 1 0 0 1 1 1 0 1 1 1 0 52 1 3 4 5

Hướng Dẫn :

Các cửa hàng này sắp xếp trên một con đường thẳng theo trật tự tăng dần của thời gian khuyến mại. Nếu cửa hàng nào có nhiều lần khuyến mại thì chúng ta coi nó như một cửa hàng khác mới hơn . Bài toán sẽ trở thành một bài toán mới :

Trên đường đi trên con đường đó thì chúng ta cần ghé thăm những cửa hàng nào để số tiền khuyến mại là lớn nhất.

Gọi Maxd[i] là số tiền khuyến mại lớn nhất nếu ta đến nhận khuyến mại tại thời điểm i (bởi vì tại một thời điểm thì chỉ có một cửa hàng khuyến mại , nên coi nó như một ánh xạ duy nhất). Nếu không nhận khuyến mại tại thời điểm i thì số tiền sẽ là 0. và ta sẽ tìm theo công thức :

Maxd[i]:=Maxmaxd[j]+tiền nhận được từ i tới j - tiền mất đi; j=1,..i-1 với điều kiện tại thời điểm j thì phải có một khuyến mại nào đó;

maxMaxd[i] ;i=1,..k chính là số tiền cần lấy .

Bài Toán 9: XÂY THÁP

Có N khối đá hình hộp chữ nhật. Người ta muốn xây một cái tháp bằng cách chồng các khối đá này lên nhau. Để đảm bảo an toàn, các khối đá được đặt theo nguyên tắc:

+ chiều cao của mỗi khối là kích thước nhỏ nhất trong ba kích thước,

+ các mép của các khối được đặt song song với nhau sao cho không có phần nào của khối nằm trên bị chìa ra ngoài so với khối nằm dưới.

Hãy tìm phương án xây dựng để tháp đạt được độ cao nhất.

Dữ liệu vào được cho trong file Tower.INP gồm:

+ dòng đầu là số N,

+ N dòng sau, mỗi dòng ghi 3 số nguyên dương là kích thước một khối đá. Các khối đá được đánh số từ 1 theo trình tự xuất hiện trong file.

Kết quả ghi ra file Tower.OUT theo quy cách:

+ dòng thứ nhất ghi số M là số lượng khối đá dùng xây tháp,

+ M dòng tiếp theo ghi các khối xếp từ đáy tháp lên đỉnh, mỗi dòng gồm 4 số theo thứ tự: K A B C, trong đó K là số hiệu khối đá, A là kích thước chọn làm đáy nhỏ, B là kích thước chọn làm đáy lớn, C là kích thước chọn làm chiều cao.

Các số trên cùng một dòng trong các file được ghi cách nhau ít nhất một dấu trắng. Giới hạn số khối đá không quá 5000 và các kích thước của các khối đá không quá 255.

Thí dụ:

Tower.INP	Tower.OUT
9 7 5 5 4 4 8 1 1 5 4 2 2 5 1 5 4 2 7 2 9 2 1 3 3 5 5 5 9 5 5 5 5 5 1 4 2 4 2	4 1 5 7 5

Hướng Dẫn:

Chúng ta thấy rằng một hình nếu ở dưới một hình khác thì các kích thước ngang và dọc đều lớn hơn hoặc bằng kích thước trên. Chúng ta không mất tổng quát, quy định chiều của các tháp theo một chiều nhất định (ví dụ : dài là độ dài có kích thước lớn nhất tròn ba kích thước, rộng là lớn thứ hai và cuối cùng là cao).

Sắp xếp các tháp theo chiều giảm dần của diện tích. Sau đó thì Maxd[i] là độ cao nhất nếu xếp tháp thứ i trên cùng (trong dãy sau khi đã sắp xếp).

Maxd[i]:=maxmaxd[j]+cao[i] ; với j =1,..i-1 và rộng [j]>=rộng[i],dài[j]>=dài[i]; Sau đó ta lấy giá trị lớn nhất của các Maxd[i] chính là độ cao của tháp cần xếp.

Bài toán 10 : RenTing

Đề Bài:

Tại một thời điểm 0, ông chủ một máy tính năng suất cao nhận được đơn đặt hàng thuê sử dụng máy của N khách hàng. Các khách hàng I cần sử dụng máy từ thời điểm Di đến thời điểm Ci (Di,Ci là các số nguyên và 0<Di<Ci<109) và sẽ trả tiền sử dụng máy là Pi (Pi nguyên , 0<=Pi<=107) Bạn cần xác định xem ông chủ cần nhận phục vụ những khách hàng nào sao cho khoảng thời gian sỉ dụng máy của 2 khách hàng được nhận phục vụ bất kỳ không đuợc giao nhau, đồng thời tổng số tiền thu được là nhiều nhất.

Dữ Liệu : Vào Từ File vân bản RENTING.INP :

Dòng đầu tiên ghi số N (0<N<=1000)

Dòng thứ I+1 trong số n dòng tiếp theo ghi 3 số Di,Ci,Pi cách nhau bởi dấu cách(I=1,..N)

Kết quả : Ghi Ra file Văn Bản RENTING.OUT :

Dòng đầu tiên ghi hai số nguyên dương theo thứ tự là số lượng khách hàng nhận phục vụ và tổng tiền thu được từ việc phục vụ họ

Dòng tiếp theo ghi chỉ số của các khách hàng được nhận phục vụ .

Ví Dụ:

RENTING.INP	RENTING.OUT	RENTING.INP	RENTING.OUT

3 150 500 150 1 200 100 400 800 80 2 180 2 3 4 400 821 800 200
513 500 100 325 200 600 900 600 2 1100 2 4

Hướng Dẫn :

Sắp xếp theo thứ tự tăng dần của Di. Sau đó gọi Maxd[i]laf tổng số tiền nhận được khi phục vụ người thứ i (trong dãy sau khi đã sắp) . Lúc đó ta sẽ có :

Maxd[i]:=MaxMaxd[j]+Pi; j=1..i-1; và Di>=Cj;

Vậy số tiền lớn nhất là = Max maxd[i];i=1..n.

II. Dạng Hai :

Các Bài toán dạng này thường có chương trình giống thuật toán Ford-Bellman.

```
Repeat
    Ok:=True ;
    IF tìm được Mind[i] nào thoả mãn Mind[i]>Mind[j]+giá trị từ j đến i
    Then
    Begin
        Ok:=False ;
        Mind[i] :=Mind[j]+giá trị từ j đến i
    End ;
Until Ok ;
```

Tương tự cho Maxd [i].Vơí quá trình j đến i là một quá trình thoả mãn điều kiện của bài toán .

Bài Toán 11: Apower

Đề Bài :

Chúng ta định nghĩa hàm AR(m,n), với m,n là những số tự nhiên (0<n<10, 0<m<1000) là một số tự nhiên sao cho khi chúng ta biểu diễn n bằng các phép toán : +,- ,*, / và các phép toán ghép số, thì số nhỏ nhất cần thiết các chữ số n để được kết quả là số m là giá trị của hàm AR(m,n) .

Để cho các bạn dễ hiểu chúng ta xét hàm AR(42,2)=4 vì ta có cách biểu diễn :

2*22-2=42 hay chúng ta có AR(22,1)= 4 vì : 11*(1+1)=22.

Bài toán đặt ra cho chúng ta là hãy tìm AR(m,n) ,với m,n biết trước .

Dữ liệu : Vào từ file văn bản Apower.Inp gồm nhiều bộ số m,n. Mỗi dòng viết một bộ số.

Kết Quả: Ghi ra file văn bản Apower.Out gồm nhiều dòng, mỗi dòng ứng với kết qủa của mỗi dòng của file input.

Ví Dụ:

APOWER.INP	APOWER.OUT
42 2 22 1 6 3	4 4 3

Hướng Dẫn :

Chúng ta sẽ gọi Ar[m,n] là giá trị hàm Ar(m,n). Ta có thủ tục quy hoạch động :

```
Fillchar(Ar,Sizeof(Ar),Maxint);
```

```
Ar[0]:=2 ;
Ar[n]:=1;
RePeat
        Ok := True ;
        For i:=1 to (m+1)*n do
         if Ar[m,i]<>maxint then
         For j:=1 to  (m+1)*n do
        Begin
                If (Ar[i]+Ar[j]<A[i+j])then
                Begin
                        Ok:=false ;
                        A[i+j]:=a[i]+a[j];
                End ;
                If (Ar[i]+Ar[j]<Ar[i-j])and(i>j)then
                Begin
                        Ar[i-j]:=Ar[i]+Ar[j];
                        Ok:=False ;
                End ;
                If (Ar[i]+Ar[j]<A[i*j])then
                Begin
                        Ok:=False ;
                        Ar[i*j]:=Ar[i]+Ar[j];
                End ;
                If (I mod j = 0 ) and (Ar[i]+A[j]<A[i div j]) then
                Begin
                        Ok:=False ;
                        Ar[i div j]:=Ar[i]+Ar[j];
                End ;
        End ;
Until OK ;
```

Nhưng để tránh những trường hợp đặc biệt như :333 với 3 thì Ar(333,3)=3 . Cho nên trước hết chúng ta tính Ar[3..3,3]:=số số 3 có trong nó. Rồi sau đó thì ta sẽ dùng thủ tục trên .

Bài Toán 12: Giá Trị Biểu Thức

Đề Bài :

Giả thiết X,Y là hai số nguyên dương. Kí hiệu Sx là tổng các chữ số trong dạng biểu diễn cơ số 10 của X , Dmax_y và Dmin_y là chữ số lớn nhất và nhỏ nhất trong dạng biểu điễn cơ số 10 của Y. Phép tính hai ngôi # với các toán hạng nguyên dương X,Y được định nghĩa như sau:

(X#Y)=Sx*Smax_y+Dmin_y

Ví Dụ : (30#9)=3*9+9=36 hay (9#30)=9*3+0=27

Với X cho trước ,một số biểu thức hợp lệ là:

(X#X) và ((X#X)#X) và (X#(X#X)#(X#X)#X) ...

Ký hiệu kết quả biểu thức là K. Cho X và K(0<X,K<109-1) cần xác định số ít nhất m các phép # để từ đó có thể xây dựng biểu thức thuộc dạng đang xét với X cho kết quả K và biểu thức biểu diễn của biểu thức.

Dữ Liệu : vào từ file văn bản Bai16.Inp dòng thứ nhất chứa số X , dòng thứ hai chứa K

Kết Quả : Ghi ra file văn bản Bai16.Out : dòng thứ nhất chứa số m, dòng thứ hai chứa biểu thức .

Ví Dụ:

BAI16.INP	BAI16.OUT
718 81	3 ((718#(718#718))#718)

Hướng Dẫn :

Ta thấy 0<X,K<109 nên ta có : 1<SX<=9*9 ; 1<=Dmax_x<=9 ; 0<=Dmin_x<=9 ;

Nên 1<=X*X<=9*9*9+9=738 ; Vậy giá trị của một biểu thức hợp lệ bất kỳ phải nằm trong đoạn [1,738], đây chính là cốt lõi lời giải cho bài toán. Nếu K nằm ngoài khoảng này thì chắc chắn vô nghiệm. Xét biểu thức X#X có 1 dấu #, dễ thấy cí 3 cách mở rộng biểu thức 1 dấu # này là :

- X#(X#X) ; (X#X)#X ; (X#X)#(X#X) ;

Giả sử B là một biểu thức tạo bởi X và n dấu # thế thì có 3 cách mở rộng biểu thức này :

X#B (n+1 dấu #) ; B#X (n +1 dấu #) ; B#B (2 * n+1 dấu #) .

Ta lập mảng một chiều Mind[1..738] trong đó Mind[i] cho biết số phép # ít nhất để tạo từ X. Ta có các bước giải quyết bài toán :

- Bước 1 : Khởi tạo mảng A[1..738]:=0 đánh dấu các giá trị đã tạo được từ biểu thức ó X và #, Khởi tạo mảng Mind nhận các giá trị Maxint .

Bước 2 : Tìm T=X#X ; A[T]:=1 ; Mind[T]:=1 ;

Bước 3 : Thực hiện cho đến khi mảng Mind không bị thay đổi :

```
For i:=1 to 738 do
 If A[i]=1 then
Begin
        T:=X#i;
        If Mind[T]>Mind[i]+1 then begin Mind[T]:=Mind[i]+1 ;A[T]:=1 ; end ;
        T:=i#X ;
        If Mind[T]>Mind[i]+1 then begin Mind[T]:=Mind[i]+1 ;A[T]:=1 ; end ;
        T:=i#i ;
        If Mind[T]>2*Mind[i]+1 then
        begin
                Mind[T]:=2*Mind[i]+1 ;A[T]:=1 ;
        end ;
End ;
```

Bài Toán 13 : Đọc Đĩa

(Đề Thi Học Sinh Giỏi Quốc Gia 2001-2002 - Bảng B)

Đề bài :

Các kĩ sư của một công ti tin học đang thử nghiệm chế tạo đĩa từ có dung lượng thông tin cực lớn . Đĩa có nhiều đường ghi và khoảng cách giữa 2 đường ghi liên tiếp nhau là rất nhỏ. Các đường ghi được đánh số từ 0 đến N, từ ngoài vào trong. Đối với loại đĩa này, việc dịch chuyển đầu đọc từ một đường ghi sang một đường ghi kế tiếp là rất khó đảm bảo độ chính xác cao cho các chuyển động cơ học trên khoảng cách quá bé do không có đủ thời gian để khởi động và phanh đầu đọc.

Người ta thiết kế mạch điều khiển với 2 lệnh : Lệnh T và lệnh L .

Lệnh T- đưa đầu đọc tiến lên phía trước P đường ghi (P>0). Ví dụ đầu đọc đang ở đường ghi K. Sau khi thực hiện lệnh T thì nó chuyển tới đường ghi số K +P. Lệnh T không áp dụng được khi K+P>N.

Lệnh L đưa đầu đọc lùi Q đường ghi (Q>0). Nếu đầu đọc đang ở đường ghi K, sau khi thực hiện lệnh L thì đầu đọc sẽ chuyển tới đường ghi K-Q. Lệnh L không áp dụng khi K-Q<0. Để di chuyển đầu đọc từ đường ghi ưu tới đường ghi V có thể phải áp dụng một dãy các lệnh T,L. Dãy m lệnh T (L) liên tiếp nhau được viết gọn dạng Tm(Lm) , trong đó m - số nguyên dương , m>=1 .

Yêu Cầu : Với N,P,Q cho trước (),N<=20000,0<P,Q<N) hãy chỉ ra dãy ít nhất câu lệnh L , T đưa đầu đọc từ đường ghi U tới đường ghi V (0<=U,V<=N) hoặc cho biết không tồn tại dãy câu lệnh như vậy .

Dữ Liệu : Vào từ file văn bản DISK.INP gồm L dòng 5 số nguyên N , P , Q , U , V , các số trên một dòng cách nhau ít nhất một dấu cách .

Kết Quả : Đa ra file văn bản DISK.OUT :

Dòng đầu tiên là số nguyên K - số câu lệnh cần thực hiện , K=-1 nếu không tồn tại cách đưa đầu đọc về đường ghi V .

Dòng thứ 2 chứa dãy câu lệnh cần thực hiện , trước tên lệnh T(L) phải có một dấu cách .

Ví Dụ :

DISK.INP	DISK.OUT
10 5 3 7 6	3 L2 T1

Hướng dẫn :

Chúng ta có Mind[i] là số lần thực hiện các lệnh chuyển đĩa ít nhất khi chuyển từ U đến i . ta thực hiện như sau :

```
Khởi tạo toàn bộ Mind[i] = maxint ;
Mind[u]:=0 ;
Repeat
    Ok:=True ;
    For i:=0 to n do
     if mind[i]<>maxint then
```

```
        Begin
            If (i-q>0)and(mind[i]+1<mind[i-q] then
            begin
                mind[i-q]:=mind[i]+1;
                luu[i-q]:=-q ;
                ok:=false ;
            end ;
            If  (i+p<=n)and(mind[i]+1<mind[i+p]) then
            Begin
                Mind[i+p]:=mind[i]+1 ;
                Ok:=False ;
                Luu[i+p]:=p;
            End ;
        End ;
Until Ok ;
```

Ta dùng mảng luu[i] để khi lần ngược trở nên dễ dàng hơn .

Bài Toán 14 : Busways

Đề bài :

Một hệ thống các xe buýt có nhiệm vụ chuyên chở hành khách đi lại giữa một số ga sao cho đảm bảo tính liên thông hai chiều giữa các ga này. Hệ thống bao gồm một số tuyến đường, mỗi tuyến đường bao gồm một số ga khác nhau theo thứ tự mà xe buýt đi qua. Xe buýt thuộc tuyến đường nào chỉ chạy trên tuyến đường đó, lần lượt qua các ga thuộc tuyến cho đến hết, sau đó lại quay lại chạy theo hướng ngược lại. Có thể có một số ga chung cho một số tuyến đường. Một hành khách muốn đi từ ga đầu đến ga cuố , có thể đi trên một tuyến hoặc phải chuyển tuyến một số ga cuối sao cho số lần phải chuyển tuyến là ít nhất. Nếu tồn tại nhiều phương án như vậy, hãy tìm phương án đi qua ít nhất .

Dữ liệu : vào trong file : Busways.inp gồm :

Dòng đầu tiên là số tuyến đường

Các dòng tiếp theo, mỗi dòng mô tả một tuyến đường, gồm một chuỗi ký tự viết liền nhau, mỗi kí tự mô tả một tên ga theo đúng thứ tự của các ga trên tuyến (chú ý các ga trên cùng một tuyến là khác nhau, nhưng các ga trên các tuyến khác nhau có thể trùng nhau, tên ga là một ký tự bất kì hiển thị được trong bảng mã ASCII)

Dòng tiếp theo là số hành trình cần tìm

Các dòng tiếp theo , mỗi dòng môt tả một hành trình cần tìm, gồm một cặp ký tự viết liền nhau, xác định tên ga đầu và tên ga cuối .

Giả thiết rằng dữ liệu cho là hợp lệ, không cần kiểm tra. Giới hạn kích thước 100 cho số các tuyến đường, và 50 cho số các ga trên một tuyến đường.

Kết quả : Ghi ra file BusWays.Out : Trong đó hành trình được viết trên một dòng, gồm các ký tự biểu diễn tên ga viết theo thứ tự được đi . Các tên ga này được viết thành từng nhóm theo tuyến đường : nếu thuộc cùng một tuyến đường thì viết liền nhau,

nếu sàn tuyến đường khác thì viết cách nhau một dấu trắng, tên ga chung được viết lặp lại .
Ví Dụ :

BUSWAYS.INP	BUSWAYS.OUT
3 ABC DBE GAEH 2 HC GB	HEA ABC GA AB

Hướng dẫn :

Đầu tiên chúng ta coi một ga là một đỉnh có số hiệu là vị trí nó trong bảng mã Ascii. Sau đó ta tạo một đồ thị có các cung mà nếu các ga trong một tuyến thì có độ dài 1 nếu trái tuyến thì độ dài 1000. Sau đó dùng thuật giải giả Ford-Bellman (giống các bài toán trên) . để tìm đường đi ngăn nhất .

Hoàn toàn tương tự các bạn có thể giải quyết bài toán sau :

(bài 5 - Phần nâng cao , bài tập lập trình pascal - Nguyễn Xuân My)

Bài toán phụ :

Bài toán 15 :

" Có M tuyến xe buýt, M<=20. Mỗi tuyến xe được cho bởi dãy tên các bến liên tiếp từ đầu đến cuối của tuyến đó, mọi tuyến xe đều đi được hai chiều. Tên bến là các số nguyên dương. Các tuyến xe có thể có các bến chung. Nếu đi từ một bến đến một bến tiếp theo trên cùng tuyến thì mất 1 đồng còn nếu đang đi trên một tuyến mà chuyển sang tuyến khác tại cùng một bến để đi đến bến trên tuyến khác đó thì mất thêm 3 đồng. Cho tên hai bến I và J. Hãy tìm một hành trình đi từ I đến J sao cho:

1. Chi phí là ít nhất
2. Số lần chuyển tuyến ít nhất.

Dữ liệu vào được cho bởi file INP.B5 trong đó dòng thứ nhất ghi hai số nguyên dương I, J; trong các dòng tiếp theo (không quá 20 dòng), mỗi dòng ghi không quá 20 số nguyên dương khác nhau từng đôi thể hiện một tuyến xe. Các tuyến xe nhận số hiệu từ 1, 2, 3, . . . kể từ trên xuống dưới.

Kết quả ghi ra file OUT.B5 như sau:

Câu 1: dòng thứ nhất ghi chi phí, dòng thứ hai ghi hành trình từ I đến J bằng cách viết các bến liên tiếp trên hành trình, mỗi bến ghi như sau: tên bến/số hiệu tuyến. Ví dụ bến 25 trên tuyến 6 sẽ ghi 25/6.

Câu 2: dòng thứ ba ghi số lượng tuyến xe, dòng thứ t ghi hành trình từ I đến J tương tự như câu 1. "

III. Dạng ba : lưu dữ dữ liệu và cách tiết kiệm biến :

Bài toán 16 : Palindrome

(Đề thi Quốc Tế năm 2000)

Đề Bài :

Palindrome là một xâu đối xứng tức là một xâu mà đọc từ trái sang phải cũng như đọc từ phải sang trái . Bạn cần viết một chương trình với một xâu cho trước , xác định số ít nhất các kí tự cần chèn vào xâu để nhận được một *palindrome* .

Ví dụ : Bằng cách chèn 2 kí tự vào xâu ‘ Ab3d ‘ ta nhận được một *palindrome* . Tuy nhiên nếu chèn its hơn 2 kí tự thì không thể tạo được một palindrome .

Dữ liệu: Vào file input : PALIN .IN

Dòng thứ nhất gồm một số nguyên là độ dài N của xâu , 3<=N<=5000.

Dòng thứ hai gồm một xâu có độ dài N. xâu gồm các kí tự là các chữ cái hoa A..Z, các chữ cái thường : a.. z và các chữ số thập phân 0..9 ,các chữ cái hoa và thường xem

nh khác nhau .

D liệu : Ra file output : PALIN.OUT

Dòng một là số lượng các kí tự cần chèn vào

Ví dụ:

PALIN.IN	PALIN.OUT
5 Ab3bd	2

Hướng Dẫn :

Gọi xâu dữ liệu vào là s . Ta sẽ tìm chiều dài của dãy con đối xứng cực đại trích từ s là s1 . Khi đó số ký tự cần thêm sẽ là =Length(s)-Length(s1) .Dãy con ở đây được hiểu là dãy thu được từ s bằng cách xoá đi một số phần tử trong s .

Gọi Maxd[i,j] là chiều dài của dãy con dài nhất thu được từ đoạn s[i..j] . Ta sẽ có :

Nếu S[i]=S[j] thì Maxd[i,j]=Maxd[i+1,j-1]+2 ;

Nếu S[i]<>S[j] thì Maxd[i,j]:=Max Maxd[i,j-1] , Maxd[i+1,j]

Tức là chúng ta cần tính Maxd[1,n]. Nhn với dữ liệu N<=5000 thì điều này trở thành hoang tởng . Ta có thể nhìn nhận rõ hơn thì thấy rằng chúng ta có thể hoàn toàn tiết kiệm được rất nhiều bộ nhớ :

Gọi Luu[0..N+1] là mảng cập nhật các bước thực hiện . Tại bước cập nhật thứ j ta có Luu[j]:=Maxd[i,j].

Ta sẽ có lại bảng truy hồi như sau :

Nếu S[i]=S[j] thì Luu[i]:=Luu[i+1] cũ + 2

Nếu S[i]<>S[j] thì Luu[i]:=Max Luu[i] cũ , Luu[i+1] cũ

Ta tính từ dưới lên , tức là tính Luu[i] với i:=n ..1 thì D[i+1] cũ sẽ bị ghi đè . Và lúc đó dùng tg để lưu lại .

Thủ tục quy hoạch động như sau :

```
procedure qhd;
begin
   for j:=1 to n do
   begin
      luu[j]:=1;       tg:=0;
      for i:=j-1 downto 1 do
       begin
          t:=luu[i];
                if s[i]=s[j] then luu[i]:=tg+2 else
```

```
                Luu[i]:=max(Luu[i],Luu[i+1]);
            tg:=t;
        end;
    end;
end;
```

Nh vậy số ký tự cần thêm vào : N-Luu[1].

Bài toán 17 : Sign

Đề Bài :

Giám đốc một công ti trách nhiệm hữu hạn muốn xin chữ kí của ông kiến trúc s trởng thành phố phê duyệt dự án xây dựng trụ sở làm việc của công ty . ông kiến trúc s trởng chỉ ký vào giấy phép khi bà th ký của ông ta đã ký duyệt vào giấy phép . Bà th kí làm việc tại tầng thứ M của một toà nhà được đánh số từ 1 đến M , từ thấp lên cao . Mỗi tầng của toà nhà có N phòng được đánh số từ 1 đến N , từ trái sang phải . Trong mỗi phòng chỉ có 1 nhân viên làm việc . Giấy phép của bà th kí ký duyệt khi có ít nhất một nhân viên ở mỗi tầng của toà nhà đã kí xác nhận . Một nhân viên bất kỳ có thể chỉ kí xác nhận vào giấy phép khi có ít nhất một trong các điều kiện sau được thoả mãn :

Nhân viên đó làm việc ở tầng 1

Giấy phép đã được kí xác nhận bởi một nhân viên làm việc ở phòng liền kề (hai phòng được gọi là liền kề khi chỉ số phòng sai khác nhau một đơn vị)

Giấy phép được ký xác nhận bởi nhân viên làm việc ở phòng cùng số phòng ở tầng dưới .

Mỗi nhân viên khi đã kí xác nhận đều phải có một chi phí nhất định . hãy chỉ ra cách xin chữ kí sao cho xin được chữ kí của ông kiến trúc s trởng mà chi phí bỏ ra là ít nhất .

Dữ liệu : Vào từ file Sign.Inp như sau :

Dòng đầu tiên ghi M , N (1 <= M <=100 ; 1<=N<=500) ;

Dòng thứ i trong số M dòng ghi N số biểu diễn chi phí khi phải kí ở phòng đó . (Cij<=109)

(Tổng chi phí cần trả là ít hơn 109)

Kết quả : Ghi ra file : Sign.Out như sau :

Dòng đầu tiên ghi hai số F , K theo thứ tự là chi phí cần trả và số lượng phòng cần đi qua

Các dòng tiếp theo chỉ ghi số của các phòng theo thứ tự cần đi qua , mỗi chỉ số ghi trên một dòng .

Ví Dụ :

Sign.Inp	Sign.Out
3 4 10 10 1 10 2 2 2 10 1 10 10 10	8 5 3 3 2 1

Hướng Dẫn :

Bài Toán thực chất là Bài Toán con gián đi từ một ô nào đó trên ô trên cùng để đến một ô ở hàng cuối với chi phí đi là nhỏ nhất .

Gọi Mind[i,j] là số tiền ít nhất để đi đến ô (i,j) .

Ta có :

Mind[i,j]:=MinMind[i-1,j];Mind[i,j-1];Mind[i,j+1] +C[i,j];

Nhưng ta thấy rằng trong chương trình này thì ta có phải lưu dưới dạng một mảng [100*500] Of longint , thêm vào đó để lưu lại đường đi thì ta phải dùng các biến lưu toạ độ , mà như thế thì dữ liệu sẽ không đáp ứng được .

Sau đây là cách làm chương trình đáp ứng được điều ấy :

```
Type    tang          = Array [0..maxN] of Longint ;
      tru             = Array [0..maxN] of Integer ;
      tr              = Array [0..maxM] of ^tru ;

Var     C ,Mind             : Array [0..maxM] of ^tang ;
      try ,trx          : tr ;
Procedure Quy Hoạch Động ;
Begin
        new(d[1]) ;  new(trx[1]) ;  new(try[1]) ;
        Fillchar(trx[1]^ ,sizeof(trx[1]^) ,0) ;
        Fillchar(try[1]^ ,sizeof(try[1]^) ,0) ;
        For i := 1 to N do Mind[1]^[i] := c[1]^[i] ;
        For u := 2 to M do
     Begin
        new(Mind[u]) ; new(trx[u]); new(try[u]) ;
        Fillchar(trx[u]^ ,sizeof(trx[u]^) ,0) ;
        Fillchar(try[u]^ ,sizeof(try[u]^) ,0) ;
        For i := 1 to N do
        Begin
                Mind[u]^[i] := Mind[u-1]^[i]+c[u]^[i];
                trx[u]^[i] := u-1 ;try[u]^[i] := i;
        End ;
        For i := 1 to N-1 do
        If Mind[u]^[i]+c[u]^[i+1] < Mind[u]^[i+1] then
         Begin
                Mind[u]^[i+1] := Mind[u]^[i] + c[u]^[i+1];
                trx[u]^[i+1] := u ;try[u]^[i+1] := i;
         End ;
        For i := N downto 2 do
        If Mind[u]^[i]+c[u]^[i-1] < Mind[u]^[i-1] then
         Begin
                Mind[u]^[i-1] := Mind[u]^[i] + c[u]^[i-1] ;
                trx[u]^[i-1] := u ;try[u]^[i-1] := i;
         End ;
        dispose(c[u-1]) ;dispose(Mind[u-1]) ;
```

End ;

Chúng ta thấy dữ liệu được đóng mở một cách khéo léo mà không tốn bộ nhớ . Còn ta chỉ quan tâm đến kết quả cuối cùng là Mind[m]^ . Còn đường đi thì đã được lưu bởi hai bảng Trx và Try . Chính vì thế chúng ta có thể đóng đi những Mind[i] mà không dùng đến trong quá trình quy hoạch về sau (Cũng đóng luôn mảng C [i] không cần thiết) . Với cách giải quyết thông minh như trên , các bạn hoàn toàn giải quyết rất nhiều bài toán quy hoạch động đòi hỏi nhiều bộ nhớ khác .

Bài toán 18 : RTicket

Đề Bài :

Trên tuyến đường sắt từ thành phố A đến thành phố B đi qua một số ga .Tuyến đường có thể biểu diễn bởi một đoạn thẳng ,các nhà ga là các điểm ở trên nó . Tuyến đường bắt đầu từ A (có số hiệu là 1) và B là ga cuối cùng .

Giá vé đi lại giữa hai nhà ga chỉ phụ thuộc vào các khoảng cách giữa chúng . Cách tính giá vé được cho trong bảng sau:

Khoảng cách giữa hai nhà ga -X	Giá vé
0<X<=L1	C1
L1<X<=L2	C2
L2<X<=L3	C3

Vé để đi thẳng từ nhà ga này đến nhà ga khác chỉ có thể đặt mua nếu khoảng cách giữa chúng không vượt quá L3 .Vì thế nhiều khi để đi từ ga này đến ga khác ta phải đặt mua một số vé.

Ví dụ , trên tuyến đường sắt cho bởi đoạn thẳng sau :

Ga: 1 2 3 4 5 6 7

L1

L2

L3

Để đi từ ga 2 đến ga 6 không thể mua vé đi thẳng . Có nhiều cách đặt mua vé để đi từ ga 2 đến ga 6 .Chẳng hạn, đặt mua vé từ ga 2 đến ga 3 mất chi phí C2 và sau đó mua vé đi từ ga 3 đến ga 6 mất chi phí C3 và chi phí tổng cộng để đặt mua vé đi từ ga 2 đến ga 6 theo cách này là C2+C3 .Lu ý rằng mặc dù khoảng cách từ ga 2 đến ga 6 là 2*L2 nhưng không thể đặt mua vé với giá vé để đi từ ga 2 đến ga 6 là 2*C2vì mỗi vé chỉ có giá trị đi lại giữa hai ga nào đó .

Yêu Cầu: Tìm cách đặt mua vé để đi lại giữa hai nhà ga cho trước với chi phí mua vé là nhỏ nhất .

Dữ Liệu: Vào từ file văn bản Rticket.Inp :

Dòng đầu tiên ghi các số nguyên L1,L2,L,C1,C2,C3 (1 L1< L2<L3 109,1 C1<C2<C3 109) theo đúng thứ tự vừa liệt kê.

Dòng thứ hai chứa số lượng nhà ga N (2 N 10000).
Dòng thứ ba ghi hai số nguyên S,T là các chỉ số của hai nhà ga cần tìm đặt mua vé với chi phí nhỏ nhất để đi lại giữa chúng
Dòng thứ i trong số N-1 dòng còn lại ghi số nguyên là khoảng cách từ nhà ga A (ga 1) đến nhà ga thứ i+1 (i=1,2,..N-1) . Chi phí tù nhà ga A đến nhà ga cuối cùng B không vượt quá 109 .

Kết Quả : Ghi ra file Rticket.Out là chi phí nhỏ nhất tìm được.
Ví Dụ:

RTICKET.INP	RTICKET.OUT
3 6 8 20 30 40 7 2 6 3 7 8 13 15 23	70

Hướng Dẫn :

Tuy công thức truy hồi của bài toán này hết sức đơn giản , như các công thức truy hồi của các bài toán trước , nhưng dữ liệu bài toán là rất lớn . Chính vì thế vòng lặp của quy hoạch động trở nên phải ít hơn . áp dụng trong bài toán này chúng ta chỉ việc so sách với các ga gần nhất của nó có khoảng cách nhỏ hơn L1,L2,L3 . các công đoạn chính của bài toán có thể được mô ta như sau :

```
Procedure ReadFile;
Begin
   readln(f1, l1, l2, l3, c1, c2, c3);
   readln(f1, n);
   readln(f1, x1, x2);
   if (x1<x2) then
   begin
      i:=x1;
      j:=x2;
   end
   else
   begin
      i:=x2;
      j:=x1;
   end;
   x1 := i;
   x2 := j;
   A^[1] := 0;
   for i:=2 to n do
   begin
       readln(f1,t);
       a^[i]:=t;
   end;
```

```
End;

Procedure Xuli;
Begin
   i1 := x1;
   i2 := x1;
   i3 := x1;
   for i:=x1+1 to x2 do
   begin
      while (A^[i]-A^[i1])>l1 do inc(i1);
      while (A^[i]-A^[i2])>l2 do inc(i2);
      while (A^[i]-A^[i3])>l3 do inc(i3);
      B^[i] := B^[i3] + c3;
      if (i<>i2) then
         if (B^[i2]+c2)<B^[i] then   B^[i] := B^[i2] + c2;
      if (i<>i1) then
         if (B^[i1]+c1)<B^[i] then   B^[i] := B^[i1] + c1;
   end;
```

IV. Các Bài toán Khác :

Bài toán 19 : RoBot1

Đề Bài :

Để thám hiểm , khảo sát các vùng đất nguy hiểm ngoài trái đất , người ta chế tạo các RoBốt đơn giản , hoat động theo chương trình cài sẵn hoặc theo lệnh điều khiển phát đi từ Trái Đất . Các lệnh điều khiển là : L : rẽ trái (so với hướng đang chuyển động) ,R: rẽ phải (so với hướng đang chuyển động), U: tiến thẳng ,D:quay lui. Với mỗi lệnh rôbốt di chuyển một đơn vị khảng cách .Vùng cần khảo sát được kẻ thành lới ô vuông và các nút lới có toạ độ nguyên . Ban đầu rôbốt được đưa tới điểm có toạ độ (X0,Y0) và hướng theo chiều song song với một trục toạ độ . Nhiệm vụ là phải đưa rôbốt tới điểm có toạ độ (X1,Y1) bằng đúng K lệnh di chuyển (0<=| X0-X1| ,|Y0-Y1|<=16,0<K<=16). Hãy xác định xem tồn tại bao nhiêu chương trình khác nhau có thể cài đặt vào bộ nhớ của rôbốt.

Dữ Liệu : Vào từ file văn bản : Bai18.Inp gồm 5 số nguyên K,X0,Y0,X1,Y1 các số thuộc phạm vi Integer , cách nhau ít nhất một dấu cách .

Kết Quả : Ghi ra file văn bản : Bai18.Out một số nguyên xác định số lượng chương trình tìm được.

Ví Dụ:

BAI18.INP	BAI18.OUT
3 0 0 1 0	9

Hướng Dẫn :

Chúng ta thấy rằng để đến ô (i,j) thì có 4 ô từ đó có thể đến nó là : (i,j+1) , (i,j-1) , (i-1,j) , (i+1,j) .

Gọi Count[i,j,k] là số cách đi có thể để đến ô (i,j) từ ô xuất phát (x0,y0) sau k bước .

Ta sẽ có :

Count [i,j,k]:= Count[i , j-1 , k-1] + Count[i , j+1 , k-1] + Count[i-1 , j , k-1] + Count [i +1 , j , k - 1];

Nhưng để tránh trùng lặp nên chúng ta phải xuất phát đặc biệt : cho i , j chạy từ (xo,yo) về sau k bước , rồi sau đó lại cho (i,j) chạy từ (xo,yo) về trước .

Bài toán 20 : RoBot2

Đề Bài :

Một xởng sản xuất có mặt là một hình chữ nhật kích thước M x N , M , N là các số nguyên dương không lớn hơn 100 . Mặt bằng được chia thành các ô vuông đơn vị gồm các dòng đánh số từ 1 đến M từ trên xuống dưới và các cột được đánh số từ 1 đến N đánh số từ trái sáng phải . Cuối ca làm việc , để thu gom các sản phẩm , người ta dùng một số con robot phải xuất phát từ ô [1,1] . robot chỉ có thể chuyển động từ trái sang phải , trên xuống . Khi robot đến nơi đặt máy nào , nó có thể thu hết mọi sản phẩm của máy đó . Robot kết thúc hành trình tại ô [M,N] .

Yêu Cầu : Hãy bố trí một số ít nhất robot để thu gom sản phẩm .

Dữ Liệu : Nhập vào từ file Robot.inp trong đó đòng thứ nhất ghi hai số M , N . Tiếp theo là M dòng mô tả mặt bằng của xởng ,mỗi dòng ghi N số chỉ gồm các số 0 / 1 mà 0 có nghĩa là ô tương ứng không có máy , ngược lại ghi số 1 .

Kết Quả : Xuất ra file robot.out trong đó : dòng thứ nhất ghi số r là số ro bot cần dùng . R dòng tiếp theo ghi hành trình của các con robot đó .chỉ gồm các kí tự liên tiếp : D , R : D có nghĩa là đi xuống , R có nghĩa là sang phải .

Ví dụ :

Robot.Inp

```
10 12 0 1 1 1 0 0 1 0 1 0 0 0  0 1 0 0 0 1 0 0 0 0 0 0  0 0 0 1 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 1 1 1 1 0 0 0  1 1 1 1 1 1
0 0 0 1 0 0  1 0 0 0 0 0 0 0 0 1 0 0  1 0 0 0 0 0 0 0 0 1 0 0                    5
```

Robot.Out

```
DDDDDDDDDRRRRRRRRRR                    RDDDDDDDRRRRRRRRDDRR
RRRRRDDDDDDRRRRRRDDD                   RRRDDDDRRRRRRRRRDDDD
RRRRRRRDDDDDDDDDDRRR
```

Hướng Dẫn :

Gọi cc là số Robot ít nhất cần dùng . Dùng mảng p[i,j] để lưu lại các giá trị . Ta sẽ có xét với các hàng i , thì ta có các bước sau :

FillChar(P, SizeOf(P), 0);

cc:=0 ;

Với hàng i :

+ bước i: Ta tìm vị trí cuối cùng của hàng này có chứa máy cần thu gom (kí hiệu mm
) . Nếu không có thì quy định : Ok:=False nếu có thì Ok:=True ;
+ bước ii : Xét các vị trí trong hàng theo thứ tự giảm dần của cột (j)
Nếu P[i-1,j]>0 thì :
Nếu j <= mm thì : P[i,mm]:=P[i-1,j]; Ok:=False ;thực hiện tiếp bước i
Nếu j>mm thì : P[i,j]:=P[i-1,j]
Nếu cuối cùng mà Ok:=True thì cc:=cc+1 ; tức là tăng số robot lên một con .
Cứ như vậy thực hiện chúng ta sẽ ra được số con robot ít nhất cần dùng .

Bài toán 21 : Truyền Tin

(Đề Thi học sinh giỏi quốc gia 2000-2001-bảng B)

Đề Bài :

Người ta cần truyền n gói tin được đánh số từ 1 đến n từ một điểm phát đến một điểm thu . Để thực hiện việc truyền tin có thể sử dụng m đường truyền được đánh số từ 1 đến m .Biết rằng nếu truyền j gói tin theo đường truyền tin j thì phải trả là Sij (Sij là một số nguyên dương , Sij 32767 , i=1,2...m , j=1,2...n)

Yêu Cầu : Hãy xác định số lượng gói tin cần truyền theo mỗi đường truyền tin để việc truyền tin n gói tin được thực hiện với tổng chi phí phải trả là nhỏ nhất .

Dữ Liệu : Vào từ file Bl3.Inp như sau :

Dòng đầu tiên chứa hai số nguyên dương n và m (n,m 100)

Dòng thứ i trong số m dòng tiếp theo chứa n số nguyên dương Si1,Si2,...Sin ,i=1,2..m

Kết Qủa : Đa ra file văn bản Bl3.Out như sau:

Dòng đầu tiên chứa số S là tổng chi phí phải trả theo cách truyền tin tìm được

Dòng thứ hai chứa m số nguyên không âm Q1,Q2...Qm , trong đó Qi là số gói tin cần truyền theo đường truyền i .

Ví Dụ:

Bl3.Inp	Bl3.Out
3 3 20 20 20 4 3 10 1 3 20	4 0 2 1

Hướng Dẫn :

Gọi Mind[i,j] là tổng chi phí nhỏ nhất cần trả cho việc truyền i gói tin mà sử dụng j đường tin (1,,j), Chúng ta có công thức truy hồi như sau :

Mind[i,j]:=MinMind[i-1,k]+S[i,j-k]; K=0,..j ;

Bài toán 22 : Cửa Hàng Bán Hoa

Đề Bài :

Tại 1 cửa hàng người ta muốn cắm một số loài hoa vào chậu hoa nhỏ , tất cả có F loài hoa và V chậu hoa (F<=V) . Các chậu hoa được đánh số từ 1 đến V và xếp theo thứ tự từ trái sang phải . Mỗi loài hoa cũng được đánh số tuân theo điều kiện : với i<j ,

loại hoa i phải ở phía trái của loại hoa j , hay nói cách khác hoa i được cắm ở chậu Vi và hoa j được cắm ở chậu Vj thì ta phải có Vi<Vj .
Ta có bảng hệ số thẩm mỹ của việc cắm hoa :

		Chậu Hoa				
		1	2	3	4	5
HOA	1 (đỗ quyên)	7	23	-5	-24	16
	2 (hải đường)	5	21	-4	10	23
	3(cẩm chướng)	-21	5	-4	-20	20

bảng Ai,j với 1<=i<=F , 1<=j<=V .có ý nghĩa : nếu hoa i được cắm ở chậu j thì đạt điểm thẩm mỹ ại . ví dụ ta có bảng hệ số trên .
Yêu cầu bài toán là tìm một phương án camứ hoa sao cho đạt tổng số điểm lớn nhất .
Hạn chế kỹ thuật :

1<= F<=100 với F là số các loài hoa .
F<=V <=100 với V là số các chậu hoa .
-50<= Ai,j <=50 với Ai,j là hệ số thẩm mỹ thu được khi loài hoa i cắm vào chậu hoa j .

5Dữ Liệu : Cho từ file Bai22.Inp như sau :

Dòng đầu tiên ghi 2 số F, V .
F dòng tiếp theo : mỗi dòng ghi V số nguyên , như vậy số Ai,j là ghi ở vị trí j tại dòng i+1.

Kết Quả : Ghi ra file Bai22.Out như sau :

Dòng đầu tiên ghi tổng số điểm thẩm mỹ của cách xếp .
Dòng thứ 2 ghi lần lượt F số , số thứ K ghi số chậu hoa của loài hoa thứ k đã xếp .

Yêu cầu chạy không quá 2 giây.
Ví dụ :

Bai22.Inp	Bai22.Out
3 5 7 23 -5 -24 16 5 21 -4 10 23 -21 5 -4 -20 20	5 3 2 4 5

Hướng Dẫn :

Chúng ta có Maxd[i,j] là giá trị thẩm mĩ lớn nhất khi cắm các tranh từ 1 đến i vào các vị trí 1..j . Ta sẽ có công thức truy hồi như sau :

i<= j then Maxd[i,j]:=MaxMaxd[i-1,j-1]+A[i,j] , Maxd[i,j-1]
i>j then Maxd[i,j]:=Maxd[i-1,j] ;

Bài toán 23 : Treo Tranh

Đề Bài :

Cho n bức tranh mã số từ 1 đến n , không vượt quá 50 . Người ta cần chọn ra 1 bức tranh để đặt ở cửa ra vào phòng tranh , số còn lại được treo thẳng hàng trong phòng treo theo trật tự nghiêm ngặt sau đây : tranh có số hiệu nhỏ phải treo ở trên trái tranh có số hiệu lớn . Biết các thông tin sau về mỗi tranh :

Tranh thứ i tro tại của sẽ đạt giá trị thẩm mỹ C[i]

Tranh thứ i treo tại vị trí thứ j sẽ đạt giá trị thẩm mỹ V[i,j] .
Hãy xác định một phương án treo tranh để có giá trị thẩm mỹ là lớn nhất .
Dữ Liệu : Vào từ file văn bản : TRANH.INP
- Dòng thứ nhất : hai trị số : N , M
  Dòng tiếp theo là n giá trị C .
  Tiếp theo là N dòng , dòng thứ i gồm m số V[i,1], V[i,2],.....V[i,m]

Kết Quả Ra tệp văn bản : TRANH.OUT
- Dòng thứ nhất : giá trị thẩm mỹ lớn nhất tìm được
  Dòng thứ hai : mã hiệu bức tranh treo ở cửa phòng tranh
  Từ dòng thứ ba : N - 1 số tự nhiên là số hiệu vị trí được chọn để treo tranh trong phòng

Ví Dụ :

TRANH.INP	TRANH.OUT	
	40	1 20 1
		1 10
1 3		2 1
2 2	1 3 0 10	40 2 2 4

Hướng Dẫn :

Trước tiên chúng ta xét bài toán : Cho N tranh , treo vào M vị trí , biết các giá trị A[i,j] là giá trị thẩm mĩ của cách treo tranh thứ i vào vị trí thứ j . Biết quy định treo như quy định bài toán . Chúng ta có Maxd[i,j] là giá trị thẩm mĩ lớn nhất khi cắm các tranh từ 1 đến i vào các vị trí 1..j . Ta sẽ có công thức truy hồi như sau :

i<= j then Maxd[i,j]:=MaxMaxd[i-1,j-1]+A[i,j] , Maxd[i,j-1]

i>j then Maxd[i,j]:=Maxd[i-1,j] ;

áp dụng vào bài toán : chúng ta sẽ thử từng tranh một khi để ở cửa , sau đó chúng ta giải quyết cho bài toán N-1 tranh để vào M vị trí để có giá trị thẩm mĩ lớn nhất . Trong các trường hợp như vậy chúng ta lấy trường hợp có tổng giá trị thẩm mĩ lớn nhất . Chính là cách treo tranh cần đặt .

Bài toán 24 : Project

Đề Bài :

Giám đốc điều hành của một công ti tin học cần xác định số lượng nhân công cần sử dụng trong mỗi tháng để thực hiện một dự án phát triển tin học . Ông giám đốc nắm được số lượng nhân công tối thiểu cho mỗi tháng . Mỗi lần thuê hoặc sa thải một công nhân luôn mất thêm một khoản chi phí . Mỗi khi một thợ nào đó được thuê , anh ta luôn nhận được tiền lơng ngay cả khi anh ta không phải làm việc . Giám đốc nắm được chi phí thuê một công nhân mới , chi phí sa thải một nhân công , lơng một tháng của một công nhân . vấn đề đặt ra cho giám đốc là phải xác định số lượng công nhân cần thuê hoặc sa thải trong mỗi tháng để cho chi phí thực hiện dự án là tối thiểu .

Dữ Liệu : Vào từ file văn bản Project.Inp :

Dòng thứ nhất ghi thời gian thực hiện dự án n (đơn vị thời gian : số tháng , n <=12)

Dòng thứ hai ghi ba số nguyên dương theo thứ tự là chi phí thuê một công nhân mới , lơng tháng của một công nhân , chi phí để sa thải một nhân công .

Dòng cuối cùng ghi n số nguyên dương D1 , D2 ,....Dn trong đó Di là số lượng nhân công tối thiểu cần cho tháng i .

Kết Quả : Ghi ra file : Project.Out :

Dòng thứ nhất ghi chi phí tối thiểu cần cho công việc

Mỗi dòng trong số N dòng tiếp theo ghi số Si , trong đó nếu Si > 0 thì nó là số lượng nhân công thuê thêm ở tháng thứ i , còn nếu Si <0 thì |Si| là số lượng nhân công cần sa thải ở tháng thứ i của dự án , Si = 0 thì khong có biến động về sa thải hay thuê thêm .

Ví Dụ :

PROJECT.INP	PROJECT.OUT
3	199
4 5 6	10
10 9 11	0
	1

Hướng Dẫn :

Gọi T_max là số công nhân của tháng nhiều nhất công nhân . mảng Scn[1..T] cho biết số công nhân tối thiểu cần cho tháng ấy . C[T,T_max] ,trong đó C[i,j]cho biết chi phí tối thiểu của i tháng đầu tiên của dự án nếu tại tháng thứ i có j công nhân trong biên chế .

C[i,j]:=minc[i-1,k]+chi phí để từ k người thành j người , i =1,..T,j=Scn[i]...T_max , k = scn[i-1]..T_max

Bài toán 25 : Triangle

Đề Bài :

```
          7
        3   8
      8   1   0
    2   7   4   4
  4   5   2   6   5
```

Hình trên là mộ bảng tam giác các số nguyên không âm .Hãy viết chương trình để tính tổng lớn nhất các số trên đường đi từ đỉnh tam giác và kết thúc tại một điểm nào đó ở trên đáy tam giác .

+ Mỗi nớc đi ta được quyền đi thẳng xuống bên trái hay bên phải của số đó
+ Số hàng trong tam giác lớn hơn 1 và 100.
+ Các số trong tam giác đều là các số nguyên không âm và nhỏ hơn 100.

Dữ Liệu: cho trong file triangle.Inp như sau:

Dòng đầu tiên ghi số lượng các dòng trong tam giác (N)

Dòng i+1 (1 i N) ghi i số

Kết Quả xuất ra file triangle.Out là tổng lớn nhất tìm được

Ví Dụ:

Triangle.Inp	triangle.Out	Triangle.Inp	Triangle.Out
5 7 3 8 8 1 0 2 7 4 4 4 5 2 6 5	30	10 7 3 8 8 1 0 2 7 4 4 4 5 2 6 5 3 4 8 6 3 5 3 5 1 5 3 7 8 3 5 7 1 7 8 3 7 8 6 5 3 1 4 5 7 8 3 5 1 6 7 4 3 2 1 3	58

Hướng Dẫn :

Maxd[i,j] là độ dài lớn nhất khi đi từ ô[1,1] đến ô[i,j] . Ta có công thức truy hồi sau :

Maxd[i,j]:=MaxMaxd[i-1,j]+a[i,j] , maxd[i-1,j-1]+a[i,j] ;

Trong đó maxd[1,1]:=a[1,1] ;

Bài toán 26 : Mua hàng

Đề bài :

N hàng ở nớc ngoài (một nớc có thể được ghé nhiều lần) người ta quay về nớc 1 , bán tất cả các hàng và thu được nhiều tiền nhất . (Làn mua cuối cùng tại nớc 1 xem như lần K+1).

Dữ Liệu : Cho trong file văn bản Muahang.Inp gồm M+1 dòng :

Dòng 1 gồm các số M,N,S,K

Dòng thứ i+1 trong các hàng còng lại ghi N số C[i,1],C[i,2],...C[i,n.

Kết Quả: ghi ra file văn bản Muahang.Out gồm :

Dòng 1 : tổng số tiền thu được theo phương án tối ưu

Các dòng còn lại biểu diễn cách mua tối ưu :

+ Dòng đầu tiên ghi tên đơn vị hàng mua ở nớc 1, và số lượng của nó .

+ k Dòng tiếp , mỗi dòng ghi 3 số : tên nớc bán cần bán hàng trước và cần tiêu thụ và mua tiếp , tên hàng mua ở nớc đó ,số lượng mua ở nớc đó .

Ví Dụ:

Muahang.Inp	Muahang.Out
5 10 100 4 1 3 5 5 5 5 5 5 5 4 5 1 1 1 5 5 5 5 4 5 4 5 5 5 1 5 5 4 5 5 5 5 5 5 5 1 5 5 5 5 5 5 5 5 5 5 1 5 5 5 5 4 5 5 5 5	187500 1 100 3 500 1 1 2500 1 3 12500 1 5 62500 1

Hướng Dẫn :

Gọi Maxd[i,j,k] là số hàng i tối đa mua tại nớc j sau k lần mua bán . (S0,K0 là số tiền ban đầu , và số lần bán mua) .

Maxd[i,j,k]:= maxMaxd[i , t ,k-1]

Số tiền tối đa các bạn có thể kiếm được sau K0 lần mua bán sẽ là MaxMaxd[i,j,k0] ; i=1,..M , j =1,..N

Nhưng trong công thứ truy hồi , bài toán sẽ đòi hỏi một bộ nhớ khá lớn . Chúng ta sẽ thực hiện như sau để giảm tối thiểu bộ nhớ :

Gọi S là mảng sao cho S[i,j] là số hàng i tối đa mua tại nớc j

Gán toàn bộ S = 0 ;

```
    S[1,i]:=S0 div C[1,i] ;
    For k := 1 to K0 do
    Begin
        + Gán toàn bộ S2=0
        + For i :=1 to m do
                for j:=1 to n do
                        for i1:=1 to m do
                                if (i1<>i) then
                                for j1:=1 to n do
                                        if ( j1<>j) then
                                        nếu S[i1,j1]*c[i,j1]div c[i,j] >s2[i,j]
                                        thì S2[i,j]:=S[i1,j1]*c[i,j1] div c[i,j] ;
        + S := S2 ;
End ;
```

Bài Toán 27 : Mua Ximăng

Đề Bài :

Một tổng công ti xây dựng nhận thầu một công trình .Người ta dự kiến chia công trình thành N công đoạn ,mỗi công đoạn kéo dài trong 1 tháng . Lượng xi măng cần thiết cho mỗi công đoạn thứ i là Mi tấn (Mi nguyên dương) .Vì xi măng tồn kho cần phải trả một chi phí tổn thất nhất định nên người ta không mua đủ lượng xi măng cho cả công trình ngay từ đầu mà chia thành nhiều đợt ,mua vào các đầu tháng. Tuy vậy ,do giá xi măng thay đổi theo từng tháng nên không nhất thiết tháng nào cũng phải mua đúng số lượng xi măng cho tháng đó .

Yêu Cầu : Hãy cho biết số lượng xi măng cần mua cho từng tháng là bao nhiêu để phí tổn tổng cộng (mua và bảo quản) là ít nhất . Biết rằng gí xi măng cho tháng thứ i là Ci (trăm ngàn đồng) mỗi tấn . Ngoài ra , nếu số lượng xi măng tại một tháng nào đó còn thừa trong kho là d tấn thì chi phí bảo quản trong tháng đó là qd2 (trăm ngàn đồng) .Để đơn giản ,ta quy ớc đơn vị mua xi măng nhỏ nhất là tấn và mọi tham số ở đây đều là nguyên .

Dữ Liệu : vào từ file văn bản ximang.Inp với :

Dòng đầu tiên chứa n,q(n 10) .

Dòng thứ hai chứa n giá trị : m1,m2,..Mn (30 Mi , i);

Dòng thứ ba chứa n gía trị : C1,C2,..Cn.

Kết Quả : ghi ra file văn bản Ximang.Out với :

Dòng đầu tiên chứa S là phí tổn tối ưu .

Dòng thứ hai ghi n gía trị X1,X2,..Xn là lượng xi măng cần mua của từng tháng một.

Ví Dụ:

XimangInp	Ximang.Out
4 2 5 3 10 6 14 20 10 30	356 6 2 15 1

Hướng Dẫn :

Gọi A[i,j] là lượng chi phí bỏ ra để mua và bảo quản từ tháng thứ i cho hết N tháng , với lượng xi măng có hiện tại trong kho là j tấn . Ta sẽ có công thức truy hồi như sau (đối với A[i,j]) :

```
For i := n downto 1 do
begin
        T := M[i]+M[i+1]+...+M[n] .
        For j:=0 to T do
        begin
                if i = n then
                begin  a[i,j]:=c[n]*(m[n]-j) ; k := m[n]-j ; end
                else if j>=m[i] then
                begin   a[i,j]:=Sqr(j-m[i])+a[i+1,j-m[i]] ; k := 0 ; end
                else
                begin  a[i,j]:=(m[i]-j)*c[i]+a[i+1,0] ; k:= m[i]-j ; end ;
                A[i,j]:=Minx*c[i]+sqr(j+x-m[i])*q+a[i+1,j+x-m[i]] ;
                                                        x := k + 1 , ...T
        end ;
end ;
```

Bài Toán 28: Cung Cấp Vật Liệu

Đề Bài :

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

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

Dữ Liệu: Cho trong file CungCap.Inp gồm 4 dòng :

Dòng thứ nhất ghi N và R (N 100)
Dòng thứ hai ghi N số biểu diễn D[1],D[2],....D[n]
Dòng thứ ba ghi N số : A[1],A[2],..A[n].
Dòng cuối cùng ghi N số : B[1],B[2],...B[n]

Kết Qủa : xuất ra file Cungcap.Out như sau :

Dòng 1: ghi một số nguyên dương là tổng chi phí vận chuyển ít nhất
Dòng 2: Ghi N số nguyên dương tương ứng số đơn vị hàng mà khi A cung cấp cho từng công trình theo thứ tự
Dòng 3:Ghi N số nguyên dương tương ứng số đơn vị hàng mà khi B cung cấp cho từng công trường theo thứ tự.

Ví Dụ:

Cungcap.Inp Cungcap.Out

5 100 30 80 80 50 40 3 5 10 4 23 4 6 2 7 4 1070 30 20 0 50 0 0 60 80 0 40

Hướng Dẫn :

Gọi Mind[i,j] là cước phí nhỏ nhất mà kho 1 phải cung cấp j đơn vị hàng cho các công trường 1..i

Ta sẽ có công thức truy hồi :

Mind [i,j]:=MinA[i]*K+B[i]*(D[i]-K)+Mind[i-1](j-K)

Với K=0..D[i] ; và K<=j ;

Bài toán 29 : Mua hàng giảm giá

Đề Bài :

Trong một cửa hàng ,mỗi loại hàng có một giá .Ví dụ gía một bông hoa là 2 đồng và giá một cái bình là 5 đồng .Để thu hút nhiều khách hàng ,cửa hàng đề ra một số cách bán đặc biệt

Một cách bán giá đặc biệt liên quan đến việc bán một hay một số hàng với giá chung được giảm.

Ví dụ : 3 bông hoa bán với giá 5 đồng thay vì 6 đồng .

2 cái bình và 1 bông hoa bán với giá 10 đồng thay vì 12 đồng .

Viết chương trình tính giá mà một khách hàng phải trả cho một nhu cầu mua hàng để tận dụng một cách tối ưu các cách bán đặc biệt ,nghĩa là phải trả ít nhất .Ví dụ : với các giá và các cách bán đặc biệt , nêu trên , giá thấp nhất để mua 3 bông hoa và 2 cái bình là 14 đồng ; 2 bình và 1 hoa là 10 đồng ; 2 hoa với giá bình thường là 4 đồng.

Dữ Liệu: cho trong 2 file offer.In1 và Bai24.In2 như sau:

File thứ nhất Bai24.Inp mô tả nhu cầu mua :

Dòng đầu tiên chứa số B là số loại hàng cần mua (0 B 5)

Mỗi dòng trong B dòng tiếp theo ghi 3 số c,k,p. Giá trị c là mã của loại hàng (1 c 999) . Giá trị k là số đơn vị hàng cần mua với mã c (1k 5). Giá trị p là giá bình thường của một đơn vị hàng với mã c (1 p 999) .

File thứ hai offer.In2 mô tả các cách mua bán đặc biệt:

Dòng đầu tiên chứa số s là số cách bán đặc biệt (0 s 99)

Mỗi dòng trong s tiếp theo mô tả một cách bán đặc biệt . Số đầu tiên n của mỗi dòng như vậy là số loại hàng trong cách bán đặc biệt tương ứng với dòng đó (1 n 5); n cặp số tiếp theo (c,k) trong đó c là mã loại hàng, k là số đơn vị hàng đó (1 k 5;1 c 999) . Số p cuối cùng trong dòng là giá đã được giảm trong lô hàng nay . Giá đã được giảm nhỏ hơn tổng các giá trị bình thường

Kết Qủa :xuất ra file offer.Out là số tiền nhỏ nhất để mua hàng

Biết rằng hàng và tiền phải nguyên .

Ví Dụ:

offer.In1	offer.In2	offer.Out
3 2 5 10 1 4 6 5 3 4	4 2 2 3 1 8 20 3 8 2 9 3 10 2 9 2 7 5 9 8 6 3 2 1 1 1 5 1 10 56	

Hướng Dẫn :

Dùng phương pháp quy hoạch động để giải. Giả sử ta có một yêu cầu mua 5 loại mặt hàng với số lượng tương ứng là SL[1], SL[2], SL[3], SL[4], SL[5]. Ta phải sử dụng các cách mua đặc biệt sao cho tổng giá mua vừa đúng yêu cầu mua thấp nhất. Quá trình tổ chức việc mua được hình dung như một quá trình quy hoạch nhiều giai đoạn. Nếu ký hiệu A[i1, i2, i3, i4, i5] là tổng giá mua thấp nhất để mua các loại hàng với các số lượng tương ứng là i1, i2, i3, i4, i5 thì theo quy hoạch động, nó phải là nhỏ nhất trong mọi giá trị có thể có của A[j1, j2, j3, j4, j5]+giá mua theo giá gốc/giá mua theo cách mua đặc biệt của phần hàng hoá thêm với mọi bộ j1, j2, j3, j4, j5 <= i1, i2, i3, i4, i5. Mỗi cách tính toán thể hiện bởi một thủ tục TIM1/TIM2.

Sau đây là Chương trình và giai thích chương trình bằng tiếng việt :

```
Uses crt;
Const
        fn='input.txt';
        fn1='offer.txt';
        gn='output.txt';
Type
        mang=array[1..5] of record  vat,sl:integer; end;
{Kiểu dữ liệu này để ghi nhận về loại hàng và số lượng trong một cách bán đặc biệt}
Var
        ten,sl,gt:array[1..5] of integer;
{Ghi nhận một yêu cầu mua}
        num:array[1..100] of integer;
        c:array[1..100] of mang;
        cost:array[1..100] of integer;
{Ba mảng NUM, C, COST dùng để ghi nhận các cách bán đặc biệt}
        b:array[1..100] of boolean;
{Dùng để ghi nhận các cách bán đặc biệt có thể dùng được, một cách bán đặc biệt không thể dùng được nếu nó có mặt hàng không xuất hiện trong yêu cầu mua, khi đó ta cho giá trị b[i] tương ứng bằng FALSE}
        a:array[0..5,0..5,0..5,0..5,0..5] of longint;
{A[i1, i2, i3, i4, i5] ghi nhận giá thấp nhất (qua từng bước) để thực hiện một yêu cầu mua hàng với số lượng các mặt hàng tương ứng bằng i1, i2, i3, i4, i5}
        cs,d:array[1..5] of byte;
{CS và D dùng để ghi nhận các mức trong quá trình tìm cách mua thấp nhất; quá trình chọn cách mua xem như một quá trình nhiều giai đoạn, ý tởng dùng quy hoạch động được thể hiện qua các thủ tục TINH1 và TINH2 sau này}
        m,n:byte;
{M: số cách bán đặc biệt; N: số mặt hàng cần mua}

procedure nhap;
var
```

```
        f:text;
        i,j:integer;
 begin
   assign(f,fn);
   reset(f);
   readln(f,n);
   for i:=1 to n do
        readln(f,ten[i],sl[i],gt[i]);
   close(f);
   assign(f,fn1);
   reset(f);
   readln(f,m);
   for i:=1 to m do
     begin
       read(f,num[i]);
       for j:=1 to num[i] do read(f,c[i,j].vat,c[i,j].sl);
       readln(f,cost[i]);
     end;
   close(f);
 end;
function tim(x:integer):byte;
{Hàm TIM(X) bằng 0 nếu mặt hàng X không xuất hiện trong yêu cầu mua hàng, bằng i
nếu mặt hàng X là mặt hàng thứ i trong yêu cầu mua hàng, hàm này dùng để khởi tạo
mảng B nói trên trong thủ tục KHOITAO tiếp theo}
 var
   i:integer;
 begin
   for i:=1 to n do
     if ten[i]=x then
       begin
         tim:=i;
         exit;
       end;
   tim:=0;
 end;

procedure khoitao;
{Thủ tục này gồm hai phần, phần thứ nhất nhằm làm cho mọi yêu cầu mua có đúng 5
mặt hàng, thủ pháp này làm cho việc viết trình đơn giản; phần thứ hai nhằm ghi nhận
những cách bán đặc biệt có thể sử dụng được để thực hiện yêu cầu mua đã cho}
 var
   i,j,k:integer;
   ab:mang;
```

```
 begin
  if n<5 then
   begin
    for i:=n+1 to 5 do
     begin
      ten[i]:=0; sl[i]:=0; gt[i]:=0;
     end;
    n:=5;
   end;
  for i:=1 to m do
   begin
    b[i]:=true;
    ab:=c[i];
    for j:=1 to 5 do
     begin
      c[i,j].vat:=0; c[i,j].sl:=0;
     end;
    for j:=1 to num[i] do
     if b[i] then
      begin
       k:=tim(ab[j].vat);
       if k=0 then b[i]:=false
       else begin
            c[i,k].vat:=ab[j].vat;
            c[i,k].sl:=ab[j].sl;
           end;
      end;
   end;
 end;

procedure tinh1;
{So sánh việc mua theo giá gốc với việc mua theo các phương án mua thấp nhất đã có
với các bộ số lượng mặt hàng "ít hơn" và việc mua thêm hàng dùng giá gốc}
 var
  s,i,min,j:longint;
 begin
  s:=0;
  for i:=1 to 5 do s:=s+gt[i]*cs[i];
  min:=s;
  for i:=1 to 5 do
   begin
    for j:=1 to 5 do d[j]:=0;
    for j:=1 to cs[i] do
```

```
        begin
          d[i]:=j;
          s:=a[cs[1]-d[1],cs[2]-d[2],cs[3]-d[3],
                 cs[4]-d[4],cs[5]-d[5]]+gt[i]*j;
          if s<min then min:=s;
        end;
     end;
   a[cs[1],cs[2],cs[3],cs[4],cs[5]]:=min;
  end;

procedure tinh2;
{So sánh việc mua theo giá sau bước TINH1 với việc mua theo các phương án mua thấp nhất đã có với các bộ số lượng mặt hàng "ít hơn" và việc mua thêm hàng dùng các cách bán đặc biệt có thể dùng được}
  var
    min,s,h   °^dìD
]];
    for i:=1 to m do
      if b[i] and (cs[1]>=c[i,1].sl) and (cs[2]>=c[i,2].sl) and
      (cs[3]>=c[i,3].sl) and (cs[4]>=c[i,4].sl) and (cs[5]>=c[i,5].sl) then
        begin
          s:=a[cs[1]-c[i,1].sl,cs[2]-c[i,2].sl,cs[3]-c[i,3].sl,
                 cs[4]-c[i,4].sl,cs[5]-c[i,5].sl]+ cost[i];
          if s<min then min:=s;
        end;
    a[cs[1],cs[2],cs[3],cs[4],cs[5]]:=min;
  end;

procedure xuly;
  var
    g:text;
  begin
    khoitao;
    a[0,0,0,0,0]:=0;
    for cs[1]:=0 to sl[1] do
      for cs[2]:=0 to sl[2] do
        for cs[3]:=0 to sl[3] do
          for cs[4]:=0 to sl[4] do
            for cs[5]:=0 to sl[5] do
              begin
                tinh1;
                tinh2;
              end;
```

```
   assign(g,gn);
   rewrite(g);
   writeln(g,a[sl[1],sl[2],sl[3],sl[4],sl[5]]);
   close(g);
 end;

Begin
  nhap;
  xuly;
End.
```

Bài Toán 30: Con Dế

Đề Bài :

Giả sử có một khối đất sét hình lập phương cạnh n (n 30) đơn vị được chia thành các khối lập phương đơn vị ở các mặt có các đường hầm vuông góc với mỗi mặt với chiều sâu là số nguyên dương . Hình lập phương này đặt trong hệ trục toạ độ vuông góc Oxyz sao cho một đỉnh là gốc toạ độ O ,các trục toạ độ trùng với ba cạnh của hình xuất phát từ O .Hình lập phương nằm trong góc phần tám (X0, Y 0, Z 0)

Câu1 : Biết toạ độ của một điểm trên một mặt khối đất ,từ đó một con dế chui vào , bò theo các đường hầm ,rồi chui ra tại một điểm nào đó trên mặt đối diện .Dế đã đi theo đường ngắn nhất .Hãy chỉ ra đường đi của dế và độ dài đường đi đó .

Câu 2: Trường Hợp không tìm được đường đi nói trong câu 1 , thì dế sẽ khoan bổ sung một số đoạn đường hầm. Mỗi lần khoan bổ sung được bắt đầu từ đỉnh của khối lập phương đơn vị nào đó dọc theo cạnh của nó. Biết rằng tổng độ dài đường dế đã khoan là ngắn nhất . Hãy cho biết đường mà dế đã đi (kể cả các đoạn khoan bổ sung)

Dữ Liệu : vào từ file văn bản Bai25.Inp có cấu trúc như sau:

Dòng đầu tiên ghi số nguyên dương n,

Dòng thứ hai ghi ba số nguyên không âm là toạ độ điểm xuất phát

Trong các dòng tiếp theo ,mỗi dòng có bốn số nguyên , theo thứ tự là toạ độ điểm của các đường hầm và độ sâu của nó .

Kết Quả : ghi ra file Bai25.Out theo cấu trúc như sau :

Dòng đầu tiên ghi chữ CAU1 Hoặc CAU2 tuỳ theo việc con dế không cần hoặc cần khoan thêm đường hầm.

Dòng thứ hai ghi độ dài đường đi ngắn nhất (nếu là kết quả câu 1) hoặc tổng độ dài các đoạn cần khoan (nếu là câu 2);

Tiếp theo là một nhóm dòng ghi đường đi của con dế theo quy cách như sau: dòng thứ nhất ghi toạ độ điểm xuất phát ,tiếp theo, trình tự đi trên đường đi ghi trên mỗi dòng toạ độ điểm tại đó hành trình đổi hướng , dòng cuối cùng ghi toạ độ điểm kết thúc.

Hướng Dẫn :

Ta dùng thuật toán Dijstra tìm đường đi ngắn nhất của đồ thị từ đỉnh xuất phát đến một đỉnh thuộc mặt đối diện. Đồ thị này có tập đỉnh là các đỉnh của các khối lập phương đơn vị, như vậy có tất cả (N+1)3 đỉnh. Tại mỗi đỉnh, con dế chỉ có thể chọn

không quá sáu di chuyển sang đỉnh kề nó theo cạnh nối hai đỉnh đó, nếu hai đỉnh đó cùng thuộc một lỗ khoan, ta cho độ dài cạnh tương ứng bằng 1, nếu không (tức là dế phải khoan), ta cho độ dài cạnh tương ứng bằng 1000. Khi đó, nếu độ dài đường đi ngắn nhất nhỏ hơn 1000, ta kết luận dế chui được sang mặt đối diện mà không cần khoan thêm đường hầm. Nếu ngược lại, ta kết luận dế cần khoan thêm đường hầm và tổng độ dài đường hầm ngắn nhất cần khoan chính là thơng (nguyên) của phép chia độ dài đường đi ngắn nhất cho 1000. Lời giải xin xem ở phần giải mẫu để biết thêm .

Bài toán 31 : Chiếu Kho Xăng

Đề Bài :

Khu vực đặt các bể xăng của một tổng đại lý Xăng dầu có dạng một hình chữ nhật được chia thành m n ô vuông .Các ô vuông đánh toạ đọ từ trên xuống dưới , từ trái qua phải theo thứ tự hàng ,cột . Tại k ô của lới có đặt các bể xăng .Người ta cần xây dựng một hệ thống đèn pha chiếu dọc theo hàng hoặc là cột của lới ô vuông sao cho mỗi bể chứa phải được chiếu sáng bởi ít nhất một đèn pha chiếu dọc theo hàng hoặc theo cột chứa nó. Biết:

Ai là chi phí xây dựng đèn chiếu sáng dọc theo hàng i (i=1,2...m)

Bj là chi phí xây dựng đèn chiếu sáng dọc theo cột j (j=1,2,..n)

Yêu Cầu : Tìm cách xây dựng hệ thống đèn chiếu với tổng chi phí xây dựng là nhỏ nhất

Dữ Liệu: Vào từ file Khoxang.Inp :

Dòng đầu tiên chứa ba số nguyên dương m,n,k (m,n< 100) ;

Dòng thứ hai chứa m số nguyên dương A1,A2,...Am

Dòng thứ ba chứa n số nguyên dương : B1,B2,..Bn

Dòng thứ i trong số k dòng còn lại chứa toạ độ của bể xăng thứ i (i=1,2,..k)

Kết Quả : ghi ra file Khoxang.Out như sau :

Dòng đầu tiên ghi tổng chi phí theo cách xây dựng tìm được

Dòng thứ hai ghi hai số P và Q theo thứ tự là số lượng đèn chiếu dọc theo hàng và cột (ghi số 0 nếu không có)

P+Q dòng tiếp theo lần lượt các toạ độ của các hàng rồi đến các cột có đặt đèn chiếu ,mỗi dòng ghi 1 số .

Ví Dụ:

Khoxang.Inp	Khoxang.Out	Khoxang.Inp	Khoxang.Out
2 3 4 10 5 12 4 2 1 2 1 3 2 1 2 3		11 1 2 2 2 3	2 3 4 15 17 2 4 6 1 1
2 2 2 3 2 1	12 0 3 1 2 3		

Hướng Dẫn :

Gọi Mind[i,j] là tổng số chi phí cần dùng khi chiếu một hình có kích thước : 1..i và 1..j . Ta sẽ có công thức truy hồi :

Nếu C[i,j]=0 thì :

+ Nếu Cột j và hàng i cha được chiếu thì : Mind[i,j]:=Mind[i-1,j-1]

+ Nếu Cột j đã được chiếu , còn hàng i cha được chiếu thì : Mind[i,j]:=Mind[i-1,j]

+ Nếu Cột j đã được chiếu , hàng i đã được chiếu thì : Mind[i,j]:=MinMind[i-1,j]+a[i],Mind[i,j-1]+b[j]
+ Nếu Cột j cha được chiếu , hàng i đã được chiếu thì : Mind[i,j]:=Mind[i,j-1];
Nếu C[i,j]=1 thì
+ Nếu Cột j và hàng i cha được chiếu thì : Mind[i,j]:=Mind[i-1,j-1]+MinA[i],B[j]
+ Nếu Không thì Mind[i,j]:=MinMind[i-1,j]+a[i],Mind[i,j-1]+B[j]

Đây là một bài toán khá hay , ngoài cách giải bằng quy hoạch động , bài toán còn có một cách giải khác mà còn lí thú hơn nhiều . Để tiện theo dõi , các bạn nên xem thêm bài toán này ở chương II , phần Luồng Cực Đại để tìm hiểu thêm .

Bài toán 32 : Fish

Đề Bài :

Cu tí có ý định đi câu cá trong vùng và anh ta có một thời gian là H (tính theo đơn vị giờ) Có n cái hồ (lakes) mà cu tí có ý định câu được đánh số từ 1 đến n theo thứ tự nằm trên đường một chiều và cu tí chỉ có thể đi từ hồ thứ i đến hồ thứ i+1 và thời gian đi từ hồ i đến hồ i+1 là Ti (Ti chia hết cho 5) . Tại mỗi một hồ thì cu tí không nhất thiết phải dừng lại để câu cá . Nếu hết thời gian cu tí đang ở hồ nào đó đi chăng nữa thì cu tí cũng phải quay về nhà .

Để có được một chuyến đi câu hiệu quả cu tí đã thu nhập thông tin về các hồ . Cho mỗi hồ i ,số cá mà cu tí có thể câu được trong 5 phút đầu là Fi (Fi>0) và cứ sau 5 phút số cá đói giảm một lượng Di . Nếu như số cá mà câu được ở 5 phút trước không lớn hơn Di thì số các câu được ở 5 phút sau sẽ là 0

Hãy viết một chương trình giúp cu tí câu được nhiều cá nhất . Thời gian để cu tí câu cá tại mỗi hồ hoặc bằng 0 hoặc chia hết cho 5 .

Dữ Liệu : vào từ file Fish.Inp như sau :

Dòng đầu tiên là số n và h (n<=20,h<=50).
Dòng kế tiếp là n-1 số thể hiện cho T1,T2,...Tn-1
Dòng kế tiếp là n số thể hiện cho F1,F2,...Fn
Dòng cuối cùng là n số thể hiện cho D1, ...Dn

Kết Quả : ghi ra file Fish.Out

Dòng đầu tiên là số cá thu được
Dòng kế tiếp gồm n số thể hiện cho thời gian cu tí sẽ dừng lại câu cá ở các hồ 1 đến n.

Hướng Dẫn :

Ta thấy rằng tổng thời gian chạy qua các ao của cu tí sẽ là không đổi . như vậy số lượng các chỉ phụ thuộc vào thời gian vào từng ao của cu tí mà thôi .Gọi thời gian mà cu tí
ghé vào các ao . Ta coi mỗi đoạn thời gian là khoảng mà cu tí dừng tại một ao (5 phút) . như vậy tại khoảng 5 phút thứ h1 nào đó thì ta dùng hàm Find (i : Integer) là hàm cho biết khoảng thời gian mà cu tí dừng lại tại ao thứ i (nếu sử dụng khaỏng thời gian để câu cá) . Các bạn có thể xem ở chương trình bài toán sau :

```
Uses Crt;
Const Fi='Fish.Inp';
      Fo='';
Var
   Save,A,C,D,M,Sum,T:Array[1..20]Of LongInt;
   f:Text;
   n,h,i:LongInt;
Procedure ReadF;
Begin
 Assign(f,Fi);
 Reset(f);
 Readln(f,n,h);
 For i:=1 to n-1 do
  Begin Read(f,T[i]);T[i]:=T[i] Div 5;End;
 Sum[1]:=0;
 For i:=2 to n do Sum[i]:=Sum[i-1]+T[i-1];
 For i:=1 to n do Read(f,M[i]);
 For i:=1 to n do Read(f,D[i]);
 Close(f);
End;
Function Find(i:LongInt):LongInt;
Var k:LongInt;
    Max,l:LongInt;
Begin
 Max:=0;
 Find:=0;
 For k:=1 to i do
  If M[k]-C[k]*D[k]>Max Then Begin Max:=M[k]-C[k]*D[k];l:=k;End;
 Find:=l;
End;
Procedure Solution;
Var h1,Max,K,Ca:LongInt;
Begin
 h:=h*12;Max:=0;
 For i:=1 to n do
  Begin
   FillChar(C,SizeOf(C),0);
   Ca:=0;
   h1:=h-Sum[i];
   Repeat
    k:=Find(i);
    If k=0 Then Break;
    If M[k]-C[k]*D[k]>0 Then Inc(Ca,M[k]-C[k]*D[k]);
```

```
    Inc(C[k]);
    h1:=h1-1;
   Until h1=0;
   If Ca>Max Then
    Begin
     Max:=Ca;
     Save:=C;
    End;
  End;
 Writeln(Max);
 For i:=1 to n do
  Write(Save[i]*5:6);
End;
BEGIN
 Clrscr;
 ReadF;
 Solution;
END.
```

Bài toán 33 : Xếp hàng mua vé xem biểu diễn - Ticket

Đề Bài :

Có N người sắp hàng mua vé dự buổi hoà nhạc . Ta đánh số họ từ 1 đến N theo thứ tự đứng trong hàng . Mỗi người cần mua vé , song người bán vé được phép bán cho mỗi người tối đa hai vé . Vì thế ,một sô người có thể rời hàng và nhờ người đứng trước mình mua vé hộ. Biết Ti là thời gian cần thiết để người i mua xong vé cho mình .Nếu người i+1 rời khỏi hàng và nhờ người i mua vé hộ thì thời gian để người thứ i mua được vé cho cả hai người là Ri.

Yêu Cầu : Xách định xem những người nào cần rời khỏi hàng và nhờ người đứng trước mua hộ vé để tổng thời gian phục vụ bán vé là nhỏ nhất .

Dữ Liệu : vào từ file Ticket.Inp :

Dòng đầu tiên chứa số N (1<N<2000)

Dòng thứ hai ghi n số nguyên dương T1,T2,..Tn

Dòng thứ ba ghi n-1 số nguyên dương R1,R2..Rn .

Kết Quả : ghi ra file văn bản Ticket.Out

Dòng đầu tiên ghi tổng thời gian phục vụ

Dòng tiếp theo ghi chỉ số của các khách hàng cần rời khỏi hàng (Nếu không có ai cần rời khỏi hàng thì quy ớc ghi một số 0)

Ví Dụ:

Ticket.INP	Ticket.OUT	Ticket.INP	Ticket.OUT
5 2 5 7 8 4 4 9 10 10	18 2 4	4 5 7 8 4 50 50 50 5024 0	

Hướng Dẫn :

Gọi Mind[i] là thời gian cần dùng ít nhất để mua vé cho số người từ 1 đến i (theo lần lượt) . Ta có công thức truy hồi :

Mind[i]:=MinMind[i-1]+T[i] ; Mind[i-2]+T[i-1] ;

Bài toán 34 : Monet

Đề Bài :

Trong cao ốc N tầng là trụ sở của một nhà băng (N <= 150) xảy ra hoả hoạn . Hoả hoạn lan truyền với tốc độ 1 tầng / phút . Trong cao ốc có thang máy chuyển động với vận tốc 10 tầng / phút . Nếu thang máy di chuyển qua tần đang có lử bốc cháy thì nó cũng bốc cháy theo . Tại thời điểm ban đầu bắt đầu hoả hoạn thang máy ở tầng thứ 1 . Với mục đích cứu những tài sản quý giá (các hòm với các đông tiền vàng) được cất giữ trên một số tầng của cao ốc tất cả nhân viên của nhà băng phải dời cao ốc bằng cầu thang còn thang máy được dành cho nhiệm vụ cứu của . Biết vị trí tâng phát sinh hoả hoạn , các tầng có chứa hòm của cùng số lượng đồng tiền vàng trong chúng và thời gian để chuyển chúng vào thang máy là 1,5 phút , bạn cần tìm cách cứu được lượng đồng tiền vàn lớn nhất .

Dữ Liệu : Vào từ file văn bản : MONET.INP

Dòng đầu tiên chứa các số nguyên dương N , H là số lượng tần của cao ốc và chỉ số của tầng cháy .

Dòng thứ i trong số N - 1 dòng tiếp theo chứa số lượng đông tiền vàng có trong hòm của tầng thứ i + 1 (tầng1 không có của) . Số lượng đồng tiền vàng ở mỗi tầng giả thiết là số nguyên .

Kết Quả : ghi ra một dòng của file văn bản : MONET.OUT số lượng tiền có thể cứu được .

Ví Dụ :

MONET.INP	MONET.OUT
5 5 100 0 300 1000	100

Hướng Dẫn :

Hoàn toàn tương tự nội dung như bài toán Fish . Thực ra bài toán chỉ đổi số liệu mà thôi .

Bài toán 35 : Hanoitower

Đề Bài :

Bài toán tháp Hà Nội trở thành nổi tiếng vào năm 1883 , sau bài báo của Lucas là một nhà toán học người pháp . Tháp là một cọc đĩa đường kính giảm dần từ dưới lên . Bài toán đặt ra là cần chuyển chồng đĩa sang một cọc khác sử dụng một cọc trung gian sao cho trong quá trình chuyển đĩa không có đĩa nào có đường kính lớn hơn lại bị đặt trên đĩa có đường kính nhỏ hơn .

Yêu Cầu : giải bài toán tháp Hà Nội tổng quát : cho M cọc và tháp gồm N đĩa (3 <= M , N <= 30) , hãy xác định số lần chuyển đĩa tối thiểu cần thực hiện để chuyển chồng đĩa từ cọc xuất phát sang một cọc đích sử dụng M-2 cọc còn lại như cọc trung gian .

Dữ liệu : vào từ file Hantower.Inp chứa hai số nguyên N , M được ghi cách nhau theo thứ tự là số cọc đĩa và số cọc trong bài toán tháp Hà Nội
Kết Quả : ghi ra file Hantower.out số lần tối thiểu chuyển đĩa cần thực hiện
Ví dụ :
Hướng Dẫn :

Gọi A[i,j] là số bước chuyển nhỏ nhất nếu ta có i là cái cọc và có j cái đĩa cần chuyển từ cọc 1 đến cọc j .

Trước tiên ta xét cách chuyển :

Đầu tiên ta sẽ chuyển 1 lượng l đĩa (l<=i) từ cọc 1 đến coc (i - 1) sử dụng cả i cọc
Sau đó chuyển (j-l) cái còn lại ở cọc 1 sang cọc i và sử dụng i-1 cọc (trừ cọc i-l ra)
Sau đó chuyển l cái ở cọc i-1 sang cọc i sử dụng i cọc

Do đó ta có :

A[i,j]:=Min2*A[i,j-t]+A[i-1,t]; t=1,..j-1 ;

Bài toán 36 : Khoảng cách hai từ

Đề Bài :

Trong bài này mọi xâu ký tự đều chỉ gồm các chữ cái thường. Đối với một xâu ký tự, ta có thể tiến hành các phép biến đổi sau:
1. Thay một chữ cái bất kỳ bởi một chữ cái khác, ví dụ: test -> text;
2. Bỏ đi một chữ cái bất kỳ, ví dụ: text -> ext hoặc text -> txt;
3. Thêm một chữ cái bất kỳ vào một vị trí bất kỳ, ví dụ: each -> peach, bat -> boat, test -> tests;
4. Đổi chỗ hai chữ cái liền kề, ví dụ: cat -> act.
Với hai xâu S1 và S2, ta định nghĩa *khoảng cách* từ S1 đến S2 bằng số lượng ít nhất các phép biến đổi thuộc bốn loại trên mà khi áp dụng liên tiếp vào S1 ta sẽ nhận được S2.
Dữ liệu: Vào được cho bởi:
- File văn bản TEXT.DAT gồm một số dòng, mỗi dòng không quá 80 ký tự, trên mỗi dòng ghi một số xâu, hai xâu liên tiếp cách nhau một ký tự rỗng.

File văn bản LEX.DAT (từ điển), dòng thứ nhất ghi số nguyên dương M<=4000 (số lượng từ trong từ điển), trong M dòng tiếp theo, mỗi dòng ghi một xâu có độ dài không quá 20, các xâu này được viết theo thứ tự từ điển.

Chương trình cần ghi ra file văn bản NEAREST.SOL như sau: Với mỗi xâu S đọc được từ file TEXT.DAT, ghi một nhóm dòng: dòng thứ nhất ghi khoảng cách H nhỏ nhất từ S đến các xâu trong từ điển, dòng thứ hai ghi số lượng K các xâu trong từ điển có khoảng cách đến S bằng H, trong K dòng tiếp theo, mỗi dòng ghi một xâu trong K xâu nói trên.
Hướng Dẫn :

Theo định nghĩa, với hai xâu S1 và S2, khoảng cách từ S1 đến S2 cũng bằng khoảng cách từ S2 đến S1. Đoạn chương trình sau giải quyết việc tính khoảng cách giữa hai xâu ký tự cho dưới dạng hàm CheckDistance.

Distance[i,j] là số phép biến đổi của xâu con 1..i của S1 và xâu con 1...j của S2 cần biến đổi về nhau :

Distance[i,j]:=MinDistance[i,j-1]+1 ; Distance[i-1,j]+1 ; Distance[i-t,j-t]+ h

Trong đó t và h là các giá trị chạy xuống vị trí mà hai xâu đổi chỗ (động tác 4) . Đoan trình sau sẽ trình bày rõ :

```
const
  MaxL = 20;
  MaxN = 4000;
type
  Words    = string[MaxL];
  matrix   = array [0..MaxL,0..MaxL] of word;
var
  Distance          : matrix;
function Change (a,b: char): word;
begin
  if (a=b) then
    Change:=0
  else Change:=1;
end;
function CheckDistance (var A: Words; var B: Words): word;
  var
    i,j,t  : word;
    nA,nB  : byte;
    m1,m2  : word;
begin
  FillChar(Distance,sizeof(Distance),0);
  Distance[0,0]:=0;
  nA:=Length(A);
  nB:=Length(B);
  for i:=1 to nA do
    Distance[i,0]:=Distance[i-1,0]+1;
  for i:=1 to nB do
    Distance[0,i]:=Distance[0,i-1]+1;
  for i:=1 to nA do
    for j:=1 to nB do
      begin
        Distance[i,j]:=Distance[i-1,j-1]+Change(A[i],B[j]);
        m1:=Distance[i-1,j]+1;
        m2:=Distance[i,j-1]+1;
```

```
      if (m1<Distance[i,j]) then Distance[i,j]:=m1;
      if (m2<Distance[i,j]) then Distance[i,j]:=m2;
      if (i>1) and (j>1) and (A[i]<>B[j]) then
        begin
          if (A[i]=B[j-1]) then
            begin
              t:=i-1;
              while (t>0) and (B[j]<>A[t]) do
                dec(t);
              if (t>0) and (Distance[i,j]>Distance[t-1,j-2]+(i-t)) then
                Distance[i,j]:=Distance[t-1,j-2]+(i-t);
            end;
          if (A[i-1]=B[j]) then
            begin
              t:=j-1;
              while (t>0) and (B[t]<>A[i]) do
                dec(t);
              if (t>0) and (Distance[i,j]>Distance[i-2,t-1]+(j-t)) then
                Distance[i,j]:=Distance[i-2,t-1]+(j-t);
            end;
        end;
    end;
  CheckDistance:=Distance[nA,nB];
end;
```

Bài toán 37 : Trò Chơi Poli

Đề Bài :

Polygon là một trò chơi cho một người mà bắt đầu bởi một đa giác N đỉnh (3<=N<=50) , xem hình 1 với N=4 . Mỗi đỉnh được đánh nhãn với một số nguyên . Mỗi cạnh được đánh nhãn bằng hoặc là dấu cộng hoặc là dấu * , và được đánh số từ 1 đến N .

Hình 1 :

```
                    2
                    +

          1 +                   *3

                    *
                    4
```

Trò chơi như sau :

Bước đi đầu tiên , một trong các cạnh được tách ra

Sau đó , lần lượt đi theo các bước sau :

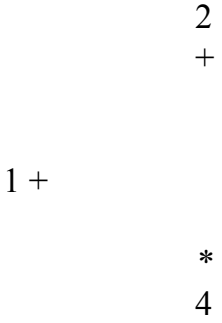
+ Chọn một cạnh E và hai đỉnh V1,V2 là hai đầu mút của cạnh E .

+ Thay thế cạnh E bởi một đỉnh mới được đánh nhãn bằng số là kết quả của việc biểu diễn phép toán biểu thị ở cạnh E của các số được đánh nhãn ở V1,V2 .

Trò chơi kết thúc khi không còn một cạnh nào , và điểm được tính là nhãn của một đỉnh còn lại .

Ví Dụ : Xét đa giác như hình 1 . Người chơi bắt đầu bằng cách tách cạnh 3

Hình 2 : *Tách cạnh 3*



Sau đó chọn cạnh 1

Hình 3 : *Chọn cạnh 1 (-2 = -7 + 5)*

Rồi đến cạnh 4

Hình 4 : *Chọn cạnh 4 (-4 = (2)*2)*

Và cuối cùng là chọn cạnh 2 . Điểm thu được trong trò chơi theo cách này là 0

Hình 5 : *Chọn cạnh 2 (0 = 4 + (-4))*

Yêu cầu : Cho đa giác , viết chương trình tính điểm cao nhất có thể và danh sách tất cả các cạnh tách đầu tiên (mà nếu dựa vào việc tách cạnh đầu tiên đó thì có thể dẫn đến điểm cao nhất)

Dữ liệu : Vào từ file Polygon.Inp Miêu tả đa giác với N đỉnh , gồm 2 dòng :

Dòng đầu tiên là số N (số đỉnh của đa giác)

Dòng thứ hai là nhãn của các cạnh 1,...N xen lẫn với nhãn của các đỉnh (đầu tiên là giữa hai cạnh 1 và 2 , sau đó là 2 và 3 ,..Cuối cùng là n và 1) . tất cả cách nhau 1

dấu cách . Nhãn của một cạnh hoặc là t (tương ứng với dấu + hoặc là x tương ứng với *)

Kết quả : Ghi ra file Polygon.Out :

Dòng đầu tiên ghi được số điểm cao nhất nhận từ dữ liệu vào

Dòng thứ hai là danh sách các cạnh được tách đầu tiên mà nếu dựa vào việc tách các cạnh đầu tiên đó thì có thể cho số điểm cao nhất đạt được .

Ví Dụ :

POLYGON.INP	POLYGON.OUT
4 t -7 t 4 x 2 x 5	33 1 2

Hướng Dẫn :

Bài toán 38 : In Thông Tin

Đề Bài :

Một máy in kiểu cổ có các khuôn chữ được khắc trên một cái vòng, một chữ cái có thể được khắc trên vòng tại 0,1 hoặc nhiều vị trí. Người ta có thể xoay vòng khuôn chữ quanh một trục cố định. Để in một ký, người ta phải xoay vòng khuôn chữ sao cho ký tự cần in đến được vị trí nằm ngay trên băng mực roòi mới ấn nút I (in). Để xoay vòng chữ sang trái một vị trí, ta ấn nút T (trái). Để xoay vòng chữ sang phải một vị trí, ta ấn nút P (phải). Để tạo một dấu cách, ta ấnnút C (cách). Các nút I và C không làm xoay vòng khuôn. Tìm chuỗi thao tác ngắn nhất để in một chuỗi ký tự.

Dữ liệu: Vào File B2.INP

Bộ dữ liệu gồm 2 dòng. Dòng thứ nhất gồm các chữ cái được khắc trên vòng khuôn in (chữ cái đầu tiên là chữ cái hiện đang nằm trên băng mục). Dòng thứ hai là chuỗi văn bản cần in.

Dữ liệu Ra File B2.OUT

Dãy ngắn nhất gồm cáclệnh L, P, I, C mà có thể được sử dụng để in chuỗi văn bản đã nhập. Nếu không thể in chuỗi văn bản đó với vòng khuôn in đã cho, in dòng chữ "không the in đuoc"

Ví dụ:	Dữ liệu vào	Dữ liệu ra
	abaed aba dac	I P I T I C T I T T I P I

Hướng Dẫn :

Gọi A[i,j] là giá trị ít nhất các thao tác để in S[i] tại địa điểm j trên máy in .

Tức là :

Nếu S1[i]=#32 then A[i,j]:=A[i-1,j]+1 ;

Nếu S1[i]<>S2[j] then A[i,j]:=Maxint ;

Nếu s1[i]=S2[j] then A[i,j]:=MinA[i-1,t]+số bước quay ít nhất từ t đến j ;

Bài toán39 : Cắt

Đề Bài :

Ta cần cắt một hình chữ nhật có kích thước MxN (m,N nguyên dương không lớn hơn 100) thành một số ít nhất các hình chữ nhật vuông có kích thước nguyên dương và có cạnh song song với cạnh hình chữ nhật ban đầu . Máy khi cắt hình chữ nhật bất kì chỉ cắt được một nhát theo phương song song với một trong hai cạnh của hình chữ nhật .

Yêu cầu : Hãy cắt các nhát cắt sao cho tạo ít nhát các hình vuông .

Dữ liệu : Vào cho bởi file Cut.Inp trong đó :

Dòng thứ nhất ghi số M , N

Kết quả : Ghi ra file Cut.Out :

Dòng thứ nhất ghi số hình vuông

Dòng tiếp theo ghi các kích thước của các hình vuông đó

Ví dụ :

Cut.Inp

5 6

Cut.Out

5

3 3 2 2 2

Hướng dẫn :

Thuật toán : Quy hoạch động

Trước tiên ta thấy tìm ớc chung lớn nhất của M và N . dặt M1=M div Ucln(M,N) và N1=N div ucln(M,N) thì ta chỉ cần tìm hình chữ nhật có cạnh M1 , N1 phải chia làm bao nhiêu hình (ít nhất) . Thì số hình vuông ít nhất của hình chữ nhật có cạnh (M,N) sẽ có số hình vuông gấp Ucln(M,N)2 hình vuông có cạnh (M1,N1) .

Gọi Shv [i,j] là số hình vuông ít nhất khi cắt hình chữ nhật có cạnh là i,j . Ta có công thức truy hồi :

```
fillchar(shv,sizeof(shv),0);
shv[i,1]:=i;
shv[1,i]:=i;
If i=j then Shv[i,j]:=1 ;
If I<>J then
Begin
   Shv[i,j]:=MinShv[k,j]+Shv[i-K,j] ; K=1,..I div 2;
   Shv[i,j]:=MinShv[i,k]+Shv[i,j-K] ; K=1,..j div 2;
End ;
```

Bài toán 40 : Ghép tam giác.

Đề bài :

Cho N đoạn thẳng (N <= 40) có tổng chiều dài <= 600 . Hãy tìm cách ghép các đoạn thẳng này thành các cạnh của một tam giác có diện tích lớn nhất với các điều kiện sau:

+ Tất cả các đoạn thẳng phải được sử dụng.
+ Mỗi cạnh của tam giác phải là ghép nguyên một số đoạn thẳng.

Dữ liệu : Vào từ file Ghep.Inp :
Dòng đầu tiênlà số N
N dòng tiếp theo, dòng thứ i ghi si là chiều dài đoạng thẳng thứ i.

Kết quả : Ghi ra file GHep.Out
Dòng đầu ghi diện tích lớn nhất tìm được, chính xác đến 2 chữ số sau dấu phẩy.
3 dòng tiếp theo, mỗi dòng ghi danh sách tên các đoạn thẳng được chọn để làm một cạnh của tam giác.

Hướng Dẫn :

Ta dùng phương pháp qui hoạch động:
Gọi C(k,i,j,)l có giá trị cho biết có thể phân tích tập các đoạn thẳng từ 1 đến k thành 3 tập trong đó tổng độ dài là i, j và l.
Khi đó, ta dễ dàng có thẻ tính các C(k,i,j,l) từ các C(k-1,x,yz) đơn giản bằng các phép thử ghép đoạn k vào các tập i , j , l.
Khi đó nghiệm của của bài toán sẽ đươc tìm thấy trên cơ cở duyệt qua các C(n,x,y,z) xem bộ nào xây dựng nên 3 cạnh tam giác có diện tích lớn nhất.
Chú ý:
+ Tính C(k,i,j,l) thực chất là chỉ cần C(k,i,j) với i <= j, vì i+j+l bằng tổng độ dài các đoạn từ 1 đến k.
+ Rõ ràng C(k,i,j) chỉ cần tính từ các C(k-1,x,y) nên không cần phải lưu toàn bộ C(k,i,j) mà chỉ cần lưu 2 màng C(i,j) tính lẫn nhau.

Bài toán 41 : 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 : Vào từ file Nhiphan.Inp
Một dòng ghi 2 số N, M (N <= 40, M đảm bảo có nghiệm)

Kết quả : Ghi ra file Nhiphan.Out
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 như sau:

```
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 42 : Đoạn 1

Đề bài :

Cho hai số nguyên dương N và k, tập S gồm các dãy nhị phân độ dài N thoả mãn số các đoạn 1 liên tiếp bằng k. Ví dụ N=5, k = 2 tập S gồm các dãy 00101, 01011, 11011,… . Các phần tử của S đực sắp xếp theo thú tự tăng dần của số nguyên tương ứng. Hãy cho biết dãy nhị phân thứ M của S.

Dữ liệu : Cho từ file Doan1.Inp

Một dòng 3 số nguyên N, k , M (N <= 50). Các số cho đảm bảo có nghiệm.

Kết quả : Ghi ra file Doan1.Out

Một dòng gồm các số 0, 1 ghi liền nhau thể hiện dãy nhị phân tìm được.

Hướng Dẫn :

Gọi C(i,j) là số các dãy nhị phân độ dài i có j đoạn 1 liên tiếp, ta dùng công thức sau đề tính C(i,j)

```
 C(i,j) = 0;
 For u = i downto 1 do C(i,j) := C(i,j) + C(u-2,j-1) + C(u-1,j);
```

Sau khi tính các C(i,j), ta xây dựng dãy thứ M như sau:

```
i := 1;
for j := k downto 1 do
 begin
  if M <= C(N-i,j) then
   begin
    Bit[i] := 0;
    i := i + 1;
   end
  else
   begin
    M := M - C(N-i,j);
    repeat
     Bit[i] := 1;
     i := i+1;
     if M > C(N-i,j-1) then M := M-C(N-i,j-1)
     else Break;
    until False;
```

```
   end;
 end;
for j := i to N do Bit[j] := 0;
```

Bài toán 43 : Chi phí đường đi

Đề bài :

Một Công ty vận tải đường bộ muốn xác định chi phí nhỏ nhất để di chuyển từ một thành phố này đến một thành phố khác. Công ty có danh sách những trạm xăng trên đoạn đường di chuyển giữa hai thành phố. Danh sách bao gồm vị trí của các trạm xăng và giá bán mỗi lít xăng ở các trạm.

Để đơn giản cho cách tính chi phí, Công ty sử dụng các quy tắc về hành vi của người lái xe:

Lái xe không dừng lại trạm xăng để nạp xăng cho xe chừng nào xăng tỏng bình xăng của xe vẫn còn nhiều hơn một nửa dung tích của bình và vẫn đủ để đạt trạm xăng tiếp theo.

Mỗi khi dừng xe để nạp xăng, lái xe luôn nạp đầy bình xăng.

Mỗi khi dừng xe nạp xăng ở trạm xăng lái xe luôn uống nớc mất 2 USD.

ở điểm xuất phát bình xăng luôn được đổ đầy.

Cần viết chương trình tính chi phí tối thiểu để lái xe mua xăng và uống nớc trên dọc đường đi.

Dữ liệu: Vào từ file văn bản MINBT.INP:

Thông tin về đoạn đường cầu đi bao gồm các dòng sau:

Dòng 1: Chứa một số thực là khoảng cách (km) giữa điểm xuất phát và đích.

Dòng 2: Gồm 3 số thực và một số nguyên:

V - dung tích của bình xăng (lít);

C - khoảng cách xe có thể đi nhờ sử dụng 1 lít xăng (km);

P - chi phí đổ đầy xăng ở điểm xuất phát (USD);

n - số lượng trạm xăng trên tuyến đường (n < 101).

Mỗi dòng thứ i trong số n dòng tiếp theo chứa hai số thực.

di - khoảng cách từ điểm xuất phát đến trạm xăng thứ i

pi - giá một lít xăng (đơn vị tính: cent = 0,01 USD), i = 1,2,...n

Giả thiết rằng các trạm xăng được sắp xếp theo thứ tự tăng dần của khoảng cách từ điểm xuất phát đến chúng và các trạm xăng được phân bố ở các vị trí thích hợp để xe có thể đạt đích mà không thiếu nhiên liệu.

Kết quả: Ghi ra file MINBT.OUT tổng chi phí (USD) (có tính cả chi phí đổ xăng ở điểm xuất phát). Quy ớc làm tròn chi phí mỗi lần chi ở trạm xăng đến cent.

Ví dụ: File MINBT.INP và MINBT.OUT có thể

MINBT.INP

```
475,6
11,9 27,4   14,98  6
102.0  99.9
220.0  132.9
```

256.3 147.9
275.0 102.9
277.6 112.9
381.8 100.9
MINBT.OUT
27.31 USD

Hướng Dẫn :

Gọi Mind[i] là số tiền ít nhất khi dừng ở i . Ta có :
Mind[i]:=MinMind[j]+Tiền để j đến i
Bài toán này giống bài toán 6 ở dạng 1 trên .

Bài toán 44 : Quân cờ Đôminô.

Đề bài :

Trong bài toán này, một quân cờ đôminô là hình có kích thước 2*1, mỗi ô ghi một số nguyên dương không quá 10. Có N quân cờ đôminô xếp thành hàng ngang như hình vẽ dưới đây.

1	6	6	8	6	4	4	3	8	9
6	10	2	8	5	2	1	6	9	5

Yêu cầu : Cho hiện trạng của các quân Đôminô hãy tìm cách lật các quân Đôminô(đổi chỗ ô trên với ô dưới) sao cho chêch lệch của tổng các số hàng trên với hàng dưới là nhỏ nhất với một số ít nhất các phép lật.

Dữ liệu : Cho từ file Domino.Inp
Dòng đầu tiên ghi số nguyên dương N (N <= 50)
Dòng thứ hai ghi các số ở hàng trên.
Dòng thứ ba ghi các số hàng dưới.

Kết quả : Ghi ra file Domino.Out
Dòng đầu ghi chêch lệch nhỏ nhất tìm được và số các đôminô cần chuyển.
Dòng thứ hai ghi danh sách thứ tự các quân đô minô cần chuyển.

Hướng Dẫn :

Ta giải bài toán bằng phương pháp qui hoạch động.
Gọi C(i,j) là số các phép lật nhỏ nhất vơi các quân từ 1 đến i, để hàng trên có tổng là j
C(i,j) = min{ C(i-1,j-Tren(i)) , C(i-1,j-Duoi(i)) + 1}
Từ các C(i,j) ta dễ dàng tìm ra chêch lệch tối thiểu và số lật nhỏ nhất.

Bài toán 45 : Lắp ráp linh kiện

Đề bài :

Trong dây chuyền lắp ráp, một rôbôt phải lắp ráp lần lượt N linh kiện theo thứ tự từ 1 đến N . Rôbốt có M công cụ để lắp ráp. Biết C(i,j) là thời gian rôbôt lắp linh kiện i bằng công cụ j (1 <= i <= N ; 1 <= j <= M), đồng thòi nếu i, j là hai công cụ được dùng trong lắp ráp hai linh kiện liên tiếp nhau thì rôbôt phải mất thêm D(i,j) thời gian (1 <=

i,j <= N). Hãy chỉ ra cho rôbôt lịch trình thực hiện lắp ráp các linh kiện sao cho tổng thời gian là nhỏ nhất.

Dữ liệu : Cho từ file laprap.Inp

Dòng dầu ghi hai số N, M (N, M <= 100)

N dòng tiếp theo mô tả ma trận C.

M dòng cuối mỗi dòng M số mô tả ma trận D

(Các số cho trong input đều là các số nguyên dương <= 32000)

Kết quả : Ghi ra file Laprap.Out

Dòng đầu ghi T là thời gian nhỏ nhất tìm được.

Dòng hai ghi N số, trong đó số thứ i là công cụ thực hiện lắp ráp linh kiện i.

Hướng Dẫn :

Ta giải bài toán trên bằng phương pháp Qui hoạch động: Gọi F(i,j) là thời gian tối thiểu cần để lắp ráp i linh kiện đầu tiên trong đó sử dụng công cụ j thực hiện lắp ráp linh kiện i. Ta tính F(i,j) theo công thứ sau :

$$F(i,j) = \min_{1 <= k <= M} \{F(i-1,k) + D(k,j) + C(i,j)\}$$

Bài toán 46 : Đòng chéo đa giác lồi

Đề bài :

Cho một đa giác lồi N đỉnh trên mặt phẳng, các đỉnh theo thứ tự vòng quanh có tên là 1, 2, ..., N, N 20.

1) Hãy chia đa giác này thành N-2 tam giác bởi các đường chéo không cắt nhau tại các điểm khác đỉnh sao cho tổng độ dài các đường chéo dùng để chia nhỏ nhất.

2) Hãy chia đa giác này thành N-2 tam giác bởi các đường chéo không cắt nhau tại các điểm khác đỉnh sao cho đường chéo dài nhất trong các đường chéo dùng có độ dài nhỏ nhất.

Chú ý rằng nói chung hai cách chia ở hai câu không giống nhau.

Dữ liệu : Với 1 i N, đỉnh i được cho bởi toạ độ (là các số thực) Ai, Bi ghi ở dòng thứ i của file DG.INP. Dữ liệu đúng như mô tả (là các đỉnh liên tiếp của một đa giác lồi).

Kết quả : Ghi ra file DG.OUT, dòng thứ nhất ghi giá trị tổng các đường chéo, trong N-2 dòng tiếp theo ghi mỗi dòng một đường chéo bằng cách ghi tên hai đỉnh đầu mút. Sau đó là dòng ghi độ dài đường chéo dài nhất, trong N-2 dòng tiếp theo ghi mỗi dòng một đường chéo theo quy cách như câu 1. Hai số liên tiếp trên một dòng cách nhau ít nhất một dấu trống. Số thực viết với 3 số lẻ sau dấu phẩy.

Ví dụ:

DG.INP		DG.OUT
0	0	1.412
1	0	1 3
1	1	1.412

0 1

Hướng Dẫn :

Câu1

- Các đỉnh có tên K=0,1,2, . . ., N-1 sẽ luôn hiểu là K mod N.
- Với L = 1, 2, . . , N-2, xét đa giác gồm L đỉnh liên tiếp P, P+1, . ., P+L-1 xem cạnh (P,P+L-1) cũng là một đường chéo. Ký hiệu S(P,P+L-1) là tổng nhỏ nhất của các đường chéo khi chia L thành các tam giác. Ta có S(P,P)=S(P,P+1)=0; S(P,P+2)=d(P,P+2) (độ dài cạnh (P,P+2).
- Giả sử ta đã biế(P,P+L-1) với P=0, . ., N-1 và L=1,2, . ., K. Ta sẽ có hệ thức đệ quy

S(P,P+K) = d(P,P+K) + Min {S(P,i)+S(i,K)} (*)

trong đó Min lấy theo mọi i mà P+1<=i<=P+K-1.

Câu 2

Tương tự với S(P,P+L-1) hiểu là Min của Max của độ dài các đường chéo và (*) trở thành hệ thức sau:

S(P,P+K) = Min Max{d(P,P+K), S(P,i), S(i,K)} (**)

Bài toán 47 : Cấp điện

Đề bài :

Trong một đợt cắm trại , có N trại được cắm . Vị trí của mỗi trại , coi như một điêm trên mặt phẳng , lần lượt là các đỉnh của một đa giác lồi. Người ta cần tiến hành cấp điện cho các trại. Máy phát điện phải đặt tại trại 1 (trại chỉ huy), có một đường dây điện đi qua tất cả N trại , bắt đầu từ trại 1 có máy phát . Yêu cầu : tìm cách mắc điện cho các trại sao cho dây điện cần mắc qua tất cả N trại có độ dài nhỏ nhất.

Dữ liệu :

N (N <= 200)

N dòng tiếp theo , dòng thứ i ghi hai số x , y cho biết toạ đọ trên mặt phẳng của trại thứ i

Kết Quả :

Dòng đầu là độ dài dây tìm được .

Dòng 2 ghi N số là thứ tự của các trại trên đường dây , bắt đầu từ trại 1.

Hướng Dẫn :

Ta giải bài toán trên bằng qui hoạch động : Nếu gọi

A(i,j) là độ dài dây ngắn nhất cần mắc cho các trại i , i+1 , … j (các đỉnh theo chiều đánh số tức là sau n là 1) , và bắt đầu từ i

B(i,j) cũng định nghĩa tương tự nhưng thay vì bắt đầu từ i phải bắt đầu từ j.

Giả sử ta tính được hết các A(i,j) , B(i,j) mà j-i <k , ta sẽ tiếp tục tính trong trường hợp j-i=k.

Ta chú ý rằng vì đa giác là lỗi nên

tập đỉnh i , … j cùng tạo thành đa giác lồi , do đó đường đi ngắn nhất phải có dạng :

Hoặc đi dến i+1 : độ dài ngắn nhất = độ dài(i,i+1) + A(i+1,j)

Hoặc đi đến j : = độ dài(i,j) + B(i+1,j)

Chú ý phải là B(i+1,j) vì khi đó đường đi tiếp bắt đầu từ j.
A(i,j) = min của hai độ dài trên . Tương tự ta tính B(i,j) cùng với A(i,j)
Như vậy cuối cùng độ dài dây nhỏ nhất chính là A(1,n)