TÀI LIỆU BỒI DƯỠNG HỌC SINH GIỎI
(Cấp tỉnh)

Huỳnh Tấn Thông
Trường THPT chuyên Nguyễn Quang Diêu
(Biên soạn và tổng hợp từ nhiều nguồn)

I- Kỹ thuật đánh dấu – Kỹ thuật nhớ:

- Kỹ thuật đánh dấu: Kỹ thuật đánh dấu thường được sử dụng khi giải các bài toán cần đến chọn những phần tử theo yêu cầu nào đó. Để chọn phần tử ta thực hiện đánh dấu phần tử đó bằng cách: sử dụng một mảng M, đánh dấu M[i]:=true để chọn phần tử i, M[i] := false để không chọn phần tử i.

- Kĩ thuật nhớ: Việc lưu trữ thông tin cần nhớ sao cho có thể lấy lại chúng một cách nhanh nhất là một trong các kỹ năng cơ bản đầu tiên để có thể có một chương trình hiệu quả. Tuy vậy việc nhớ cái gì và lưu trữ như thế nào lại đòi hỏi sự nhạy bén toán học của học sinh. Điều này lại chỉ được hình thành sau khi học sinh tiếp xúc với một hệ thống các bài toán được tổ chức cẩn thận. Kĩ thuật này giúp học sinh xây dựng được các thói quen tư duy cơ bản cũng như các kỹ thuật cơ bản trong lập trình.

Bài toán 1: Bình chọn qua điện thoại (Đề thi THCS tỉnh Đồng Tháp năm 2015)

Trong cuộc thi Hoa hậu có **n** thí sinh dự thi, được đánh số báo danh theo thứ tự từ **1** đến **n**. Khán giả xem truyền hình có thể bình chọn cho thí sinh mình yêu thích, bằng cách dùng điện thoại di động gửi tin nhắn, nội dung tin nhắn là số báo danh của thí sinh gửi đến số tổng đài của Ban tổ chức. Ban tổ chức đã nhận được **m** tin nhắn hợp lệ, khán giả thứ **i** bình chọn cho thí sinh có số báo danh là $\mathbf{a_i}$.

Yêu cầu: Biết trước số thí sinh dự thi là n, số lượng tin nhắn hợp là **m**. Hãy liệt kê số báo danh của thí sinh được khán giả bình chọn nhiều nhất. Nếu có nhiều thí sinh cùng có số lượt bình chọn nhiều nhất thì liệt kê số báo danh theo thứ tự tăng dần.

Dữ liệu vào: Cho từ tệp văn bản BL3.INP có cấu trúc:

- Dòng thứ nhất ghi số nguyên dương **n** ($0 < n \le 100$).

- Dòng thứ hai ghi số nguyên dương **m** ($0 < m \le 10^6$).

- Dòng thứ i trong số **m** dòng tiếp theo ghi số nguyên dương $\mathbf{a_i}$ là số báo danh của thí sinh mà khán giả thứ **i** bình chọn ($0 < a_i \le n$; $1 \le i \le n$).

Kết quả: Ghi vào tệp văn bản BL3.OUT chỉ có một dòng ghi số báo danh của các thí sinh được khán giả bình chọn nhiều nhất.

Các số ghi trên một dòng cách nhau một kí tự trắng.

Ví dụ:

BL3.INP	BL3.OUT
3 5 3 1 3	2 3

2 2	

Hạn chế kĩ thuật:
- Có 70% số test tương ứng 70% số điểm của bài có $m \leq 10^4$
- Có 30% số test tương ứng 30% số điểm của bài có $m \leq 10^6$

Thuật toán 1: Thuật toán tự nhiên, đạt 50-70% số điểm (tùy theo dùng thuật toán sắp xếp).

- B1. Đọc tệp ghi dữ liệu vào mảng A.
- B2. Sắp xếp mảng A
- B3. Tìm đoạn i, j có dữ liệu bằng nhau dài nhất trên mảng A, A[i] là kết quả bài toán. (duyệt hoặc dùng quy hoạch động)

Thuật toán 2: Dùng kĩ thuật mảng nhớ, thời gian O(n) đạt 100% số điểm

- Dùng mảng nhớ D theo thứ tự là số báo danh của thí sinh (đề cho không quá 100 thí sinh)
- Đọc số trong tệp số có giá trị a[i] thì mảng d[a[i]] tăng 1 đơn vị.
- Tìm Max trong mảng D, xuất chỉ số i có D[i]=Max.

Chương trình tham khảo:

```
Const fi='BL3.inp'; fo='BL3.out';
Var d: array[1..100] of longint;
   f, g: text;
   n, m, i,x,  max: longint;
Begin
   Assign(f,fi); reset(f);
   Assign(g, fo); rewrite(g);
   readln(f, n);
   readln(f, m);
   for i:=1 to m do
     begin
      read(f,x);
      d[x]:=d[x]+1;
     end;
     max:=d[1];
     for i:=2 to n do
        if d[i]>max then max:=d[i];
   for i:=1 to n do if d[i]=max then write(g, i,' ');
   close(f); close(g);
end.
```

Bài toán 2: Đánh số trang sách (Đề thi THPT tỉnh Đồng Tháp năm 2015)

Cuộc thi Khoa học kỹ thuật dành cho học sinh Trung học được ngành giáo dục tổ chức hằng năm. Sau nhiều ngày nghiên cứu, An đã viết hoàn thành báo cáo dự án để tham gia cuộc thi này. Báo cáo dự án có N trang và được đánh số trang tự động từ 1 đến N bằng phần mềm soạn thảo văn bản. Do là người thích nghiên cứu, An đặt câu hỏi có bao nhiêu chữ số 0, chữ số 1,... , chữ số 9 đã dùng.

Yêu cầu: Cho biết trước số N. Hãy tính số lượng chữ số 0 đã dùng, số lượng chữ số 1 đã dùng, …, số lượng chữ số 9 đã dùng để đánh số trang từ 1 đến N.

Dữ liệu vào: Cho từ tệp văn bản SOTRANG.INP gồm một dòng duy nhất chứa số nguyên N ($0 < N \leq 10^5$).

Dữ liệu ra: Ghi vào tệp văn bản SOTRANG.OUT có cấu trúc gồm 10 dòng, dòng thứ nhất là số 0 và số lượng chữ số 0 đã dùng, dòng thứ hai là số 1 và số lượng chữ số 1 đã dùng,…, dòng thứ mười là số 9 và số lượng chữ số 9 đã dùng.

Hai số ghi trên một dòng cách nhau kí tự trắng.

Ví dụ:

SOTRANG.INP	SOTRANG.OUT
13	0 1 1 6 2 2 3 2 4 1 5 1 6 1 7 1 8 1 9 1

Thuật toán 1: O(n) đạt 100% số test.

B1: Tạo xâu S (ansistring/$H+) bằng cách ghép các số từ 1 đến N (cần đổi số thành xâu);

B2: Dùng thủ tục/ hàm: x:=Copy(S,1,1) , Delete(S,1,1)

B3. Đổi x thành số, dự vào x tăng biến đếm tương ứng:

Thuật toán 2: O(n*5) đạt 100% số test. Dùng mảng nhớ có 10 phần tử từ 0 đến 9.

Chương trình tham khảo:

```
const
     fi='sotrang.inp';
     fo='sotrang.out';
var n,i, x, y: longint;
    dem: array[0..9] of integer;
    f, g:text;

Begin
   assign(f,fi); reset(f);
   assign(g, fo); rewrite(g);
   read(f,n);
   for i:=1 to n do
      begin
         x:=i;
         while x>0 do
           begin
             y:=x mod 10;
             dem[y]:=dem[y] + 1;// tăng số tương ứng
             x:= x div 10;
           end;
```

```
      end;
    for i:=0 to 9 do  writeln(g,i,' ',dem[i]);
    close(f); close(g);

End.
```

Bài toán 3: Thống kê dân số (Đề thi Tin học Trẻ THCS năm 2010 của tỉnh Đồng Tháp)

Để chuẩn bị cho bầu cử Hội đồng nhân dân 3 cấp ở một tỉnh X có khoảng 1 triệu người dân. Sau khi thực hiện điều tra độ tuổi, chính quyền có nhu cầu thống kê theo độ tuổi dân cư của tỉnh.

Yêu cầu: Viết chương trình thống kê dân số của tỉnh X. Biết rằng tuổi một người nằm trong khoảng từ 1 đến 100.

Dữ liệu vào: Tuổi của dân cư được cho từ tệp văn bản TUOI.INP viết liên tiếp nhau, cách nhau ít nhất một dấu cách hoặc một dấu xuống dòng.

Kết quả: Ghi ra tệp văn bản TUOI.OUT, có cấu trúc:

Dòng thứ nhất ghi số người 1 tuổi. Dòng thứ 2 ghi số người 2 tuổi,... dòng thứ 100 ghi số người 100 tuổi.

Ví dụ:

TUOI.INP	TUOI.OUT
1 2 20 4 45 62 3 2 4 55 2 4 5 100	1 1 2 3 3 1 4 3 5 1 ..// từ dòng 6 đến dòng 99 100 1

II- Kỹ thuật cộng dồn:

Trong quá trình tính toán, nếu biết tổ chức dữ liệu có tính toán kế thừa thì số lượng phép tính giảm đi rõ rệt, giảm thời gian thực hiện chương trình.

S: là mảng cộng dồn. Ta có: **S(i,j)=S(0,j)-S(0, i-1)**

Bài toán 4a. Dãy con lớn nhất

Cho dãy số A gồm N số nguyên khác 0. Tìm một dãy con gồm các phần tử liên tiếp của A mà tổng các số trong dãy con là lớn nhất. ($N<=10^4$)

Dữ liệu vào từ tệp input.txt, trong đó ghi dãy số A, kết thúc bởi số 0 (không thuộc dãy A). Bảo đảm rằng dãy A không rỗng và tổng của số lượng bất kỳ các số của A có thể biểu diễn là số nguyên kiểu longint.

Kết quả ra ghi vào tệp output.txt chỉ số của số đầu và số cuối của dãy con và tổng các số của dãy con. Ví dụ

Input.txt	**Output.txt**	**Input.txt**	**Output.txt**
-2 -1 0	2 2 -1	1 2 -3 3 0	1 2 3

```
Const fi='input.txt';
      fo='output.txt';
      nmax=1000;
Var a: array[0..nmax] of longint;
     max, x, n, i, j, dau, cuoi: longint;
    f, g: text;
Begin
    assign(f, fi); reset(f);
    assign(g, fo); rewrite(g);
    n:=0;
    repeat

          read(f, x);
          if x<>0 then
             begin
               n:=n+1;
               a[n]:=a[n-1]+x;
             end;
    until x=0;

        max:=-maxlongint;

     for i:=1 to n do
         for j:=i to n do

          if a[j]-a[i-1]>max then
              begin
                 max:=a[j]-a[i-1];
                 dau:=i;
                 cuoi:=j;
              end;
     writeln(g, dau,' ', cuoi, ' ',max);

close(f); close(g);
end.
```

Bài toán 4. Tổng k số nguyên liên tiếp lớn nhất

Cho *n* số nguyên a1, a2, ... an và số nguyên dương *k*. Hãy tìm tổng k số nguyên liên tiếp của mảng A có giá trị lớn nhất.

Input: Nhập từ file dayso.inp

- Dòng thứ nhất là hai số n, k (k<=n; 0<n <10^5);
Dòng tiếp theo là dãy a1, a2, .. an (-10^3ai<10^3, i=1..n)

Output: Ghi vào tệp dayso.out gồm ba số nguyên i, j, t; Trong đó i là chỉ số đầu, j là chỉ số cuối của đoạn k phần tử, T là tổng lớn nhất.

Ví dụ:

Dayso.inp	Dayso.out
6 3 3 -1 5 - 4 10 3	3 5 11

Ý tưởng:

Bình thường ta phải tính tổng tất cả các đoạn con có *k* phần tử liên tiếp, rồi so sánh các tổng này để tìm ra tổng lớn nhất. Công việc này đòi hỏi *(N-k) ×(k-1)* phép cộng và tổ hợp C_{N-k}^{2} phép so sánh hai tổng. Nếu N và k tương đối lớn thì số lượng phép tính này rất lớn. Độ phức tạp tính toán trung bình cỡ O(*N*×*k*).

Để giải bài toán này, còn cách sau đây có độ phức tạp tính toán trung bình là O(*N*): Ta tạo các tổng $S_i= A_1+A_2+\ldots+A_i=S_{i-1}+A_i$. Sau đó muốn tìm các tổng *k* phần tử liên tiếp bắt đầu từ *j* ta sử dụng công thức:

$A_j+A_{j+1}+\ldots+A_{j+k-1}=(A_1+A_2+\ldots+A_{j+k-1})-(A_1+A_2+\ldots+A_{j-1})=S_{j+k-1}-S_{j-1}$, với $1 \le j \le N-k+1$.

```
Program Tong_K_;
const fi  = 'dayso.in';
      fo  = 'dayso.out';
     nmax=10000;
var   n, k, dau, cuoi, i, j : LongInt;
      max : longint;
      a   : array[1..nmax] of longint;
      s   : array[0..nmax] of longint;
      f, g: text;
Begin

  assign(f,fi); reset(f);
  assign(g, fo); Rewrite(g);

  read(f,n,k);
  s[0]:=0;
  for i:=1 to n do
     begin
         read(f,a[i]);
         s[i]:=s[i-1]+a[i];  // cong don
     end;

  max := -maxlongint;
  for j:= 1 to n-k+1 do
  if max<(s[j+k-1]-s[j-1]) then
      begin
            max := s[j+k-1]-s[j-1];
            dau :=j;
            cuoi := j+k-1;
      end;

  write(g, dau,' ',cuoi,' ',max);
  close(f); close(g);
END.
```

Bài toán 5: VƯỢT SÔNG (HSG tỉnh Đồng Tháp năm 2015)

Nhà của bé Hằng Nga nằm ở bên bờ trái của con sông quê, còn nhà ngoại của bé nằm ở bên bờ phải của sông. Con đường dọc bờ sông còn có rất nhiều nhánh sông bên trái và bên phải, có nhánh chảy về bên trái, có nhánh chảy về bên phải, có nhánh chảy cả về bên trái và bên phải.

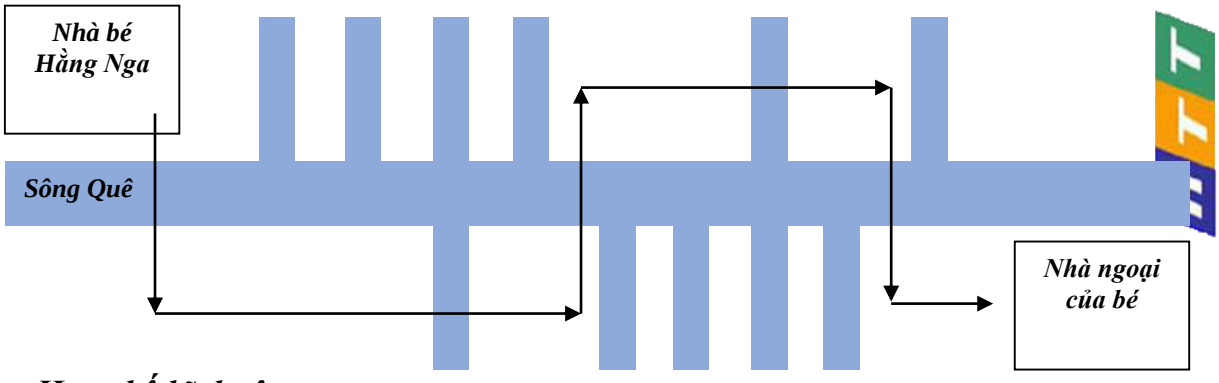
Yêu cầu: Bạn có được sơ đồ đoạn sông từ nhà bé Hằng Nga đến nhà ngoại của bé, hãy chỉ giúp bé Hằng Nga phương án đi đến nhà ngoại sao cho số lần "***vượt sông***" là ít nhất.

Dữ liệu vào: Cho từ tệp văn bản VUOTSONG.INP gồm một xâu kí tự độ dài N ($N \le 10^6$) mô tả bản đồ con sông từ nhà bé Hằng Nga đến nhà ngoại. Kí tự 'L' kí hiệu có nhánh sông bên trái, kí tự 'R' kí hiệu có nhánh sông bên phải, kí tự 'B' kí hiệu có nhánh sông cả bên trái và bên phải.

Dữ liệu ra: Ghi vào tệp văn bản VUOTSONG.OUT một số duy nhất là số lần ít nhất mà bé Hằng Nga phải vượt sông để đến nhà ngoại của bé.

Ví dụ:

VUOTSONG.INP	VUOTSONG.OUT
LLBLRRBRL	5



Hạn chế kĩ thuật:

- Có 50% số test tương ứng 50% số điểm của bài có $N \le 20$
- Có 20% số test tương ứng 20% số điểm của bài có $N \le 250$
- Có 30% số test tương ứng 30% số điểm của bài có $250 < N \le 10^6$

Tham khảo: dùng kỹ thuật cộng dồn kết hợp mảng nhớ.

```
{$h+}

Const fi='vuotsong.inp';
      fo='vuotsong.out';

Var t, p : longint;
     s: string;
     f: text;
     i, kq: longint;
Function min(x, y: longint): longint;
    Begin
       min:=x;
       if x>y then min:=y;
    end;
Begin
    Assign(f, fi);    reset(f);
    read(f, s);
    close(f);
    t:=0; //xuat pat trai
    p:=1;  // xuat pat phai
```

```
    for i:=1 to length(s) do
     begin
       if s[i]='L' then
          begin
               T:=T+1;
               P:=P;
          end
       else if s[i]='R' then
               begin
                   P:=P+1;
                   T:=T;
               end
              else
                  Begin
                    P:=P+1;
                    T:=T+1;
                  end;
              T:=Min(T, P+1);
              P:=Min(p,T+1);


       end;
       kq:=min(T,P);
       if kq=t then kq:=kq+1;

assign(f, fo); rewrite(f);
     write(f, kq);
     close(f);
end.
```

Bài toán 6: Huyền thoại Lục Vân Tiên (Mã bài MINK)

Lục Vân Tiên cũng giống Samurai Jack, bị Quan Thái Sư đẩy vào vòng xoáy thời gian và bị chuyển tới tương lai của những năm 2777 Ở thời đại này, Tráng sỹ phải là người thông thạo máy tính , gõ bàn phím lia lịa như đấu sỹ thời xưa múa kiếm ấy và phải qua một cuộc thi lập trình mới được phong danh hiệu.Để vượt qua vòng loại , Vân Tiên cần tham gia cuộc thi sát hạch . Ban Giám Khảo cuộc thi sát hạch gồm có N người , họ đều là các cao thủ trong giới IT . Các thành viên trong Ban Giám Khảo được đánh số từ 1 -> N và mỗi người lại có một chỉ số sức mạnh gọi là APM (Actions Per Minute) . Các giám khảo sẽ xếp hàng lần lượt từ 1 -> N . Mỗi thí sinh sẽ phải đấu với K vị giám khảo và K vị giám khảo này phải đứng liền thành 1 đoạn (Tức là i , i+1 , i+2 , ... i+K-1) , chỉ cần thắng 1 vị giám khảo thì sẽ vượt qua vòng loại. Tuy nhiên thí sinh kô được chọn xem những giám khảo nào sẽ đấu với mình.

Vân Tiên rất lo vì lỡ may đụng độ với những vị giám khảo nào "khó nhằn" thì sẽ tiêu mất . Nên chiến thuật của Vân Tiên là tập trung hạ vị giám khảo có chỉ số APM thấp nhất trong số K vị . Bạn hãy lập trình để giúp Lục Vân Tiên xác định được ở tất cả các phương án thì chỉ số APM của vị giám khảo thấp nhất sẽ là bao nhiêu (Có tất cả N-k+1 phương án :

Phương án 1 : Vân Tiên phải đấu với vị 1 -> vị k

Phương án 2 : Vân Tiên phải đấu với vị 2 -> vị k+1

…

Phương án N-k+1 : Vân Tiên phải đấu với vị N-k+1 -> vị N) .

(1 <= N <= 17000 , chỉ số APM của 1 giám khảo >= 1 và <= 2 tỉ , 1 <= K <= N) .

Input

Dòng 1 : số T là số test. Tiếp theo là T bộ test , mỗi bộ test có format như sau :

Dòng 1 : N k

Dòng 2 : N số nguyên dương A[1] , … A[N] .

Output

Kết quả mỗi test ghi ra trên dòng , dòng thứ i gồm N-k+1 số , số thứ j tương ứng là chỉ số APM của vị giám khảo yếu nhất trong phương án j .

Input	output
2 4 2 3 2 4 1 3 3 1 2 3	2 2 1 2

III. Kỹ thuật tham lam:

Phương pháp tham lam là kỹ thuật thiết kế thường được dùng để giải các bài toán tối ưu. Phương pháp được tiến hành trong nhiều bước. Tại mỗi bước, theo một lựa chọn nào đó, sẽ tìm một lời giải tối ưu cho bài toán nhỏ tương ứng. Lời giải của bài toán được bổ sung dần từng bước từ lời giải của các bài toán con.

Lưu ý: Phương pháp tham lam đôi lúc chưa cho nghiệm tối ưu:

Bài toán 7: Máy rút tiền ATM:

Trong máy ATM, có sẵn các loại tiền có mệnh giá x1 ngàn đồng, x2 ngàn đồng, .xn ngàn đồng. Giả sử mỗi loại tiền đều có số lượng không hạn chế.

Khi có một khách hàng cần rút một số tiền **S** ngàn đồng. Hãy tìm một phương án trả tiền sao cho trả đủ **S** đồng và số tờ giấy bạc phải trả là ít nhất. (giả thuyết luôn luôn trả đủ tiền cho khách hàng)

Input: ATM.INP có cấu trúc

- Dòng thứ nhất ghi số tiền S
- Dòng thứ 2 ghi mệnh giá các tờ giấy bạc x1, x2, ..xn

Output: ATM.OUT có cấu trúc: gồm n dòng, mỗi dòng ghi hai số, số thứ nhất là mệnh giá tờ giấy bạc, số thứ 2 ghi số tờ trả cho khách hàng

ATM.INP	ATM.OUT
2300 500 100 200	500 4 200 1 100 1

Phân tích bài toán: Để rút số tờ giấy bạc là ít nhất thì số tờ giấy bạc lớn nhất cần trả nhiều nhất.

Chương trình tham khảo:

```
Const fi='ATM.INP'; fo='ATM.OUT';

Var s, i,j, n, t: Longint;
   a, d: array[1..3] of longint;
   f, g: text;

Begin
   Assign(f, fi); reset(f);
   Read(f, s); n:=0;
   while not seekeof(f) do
      begin
         n:=n+1;
         read(f, a[n]);
      end;
   for i:=1 to n-1 do
        for j:=i+1 to n do
           if a[i]<a[j] then
              begin
                 t:=a[i]; a[i]:=a[j];a[j]:=t;
              end;

   For i:=1 to n do
     if s div a[i] <> 0 then
        begin
             d[i]:= s div a[i];
             s:=s mod a[i];
        end;
   Assign(g, fo); rewrite(g);
   for i:=1 to n do
      Begin
        writeln(g, a[i] , ' ', d[i]);
      end;
   close(g);
End.
```

Chú ý: Bài toán ATM có nhiều biến thể như: số tờ giấy bạc không hạn chế số lượng, hay hạn chế số lượng. Ta áp dụng tư tưởng tham cần sắp xếp mảng A theo thứ tự giảm dần. Dùng mảng đánh dấu D để ghi nhận số tờ giấy bạn tương ứng. (D[i]=0; tờ giấy bạc thứ i không được chọn. i=1..N);

Bài 8. Đua ngựa (Bài 1. Vòng 2 tỉnh Đồng Tháp năm 2015 tương tự)

Ở thời Xuân Thu, vua Tề và Điền Kỵ thường hay tổ chức đua ngựa từng cặp với nhau. Mỗi một con ngựa có một hệ số khác nhau. Trong cuộc đua, con ngựa nào có hệ số cao hơn thì sẽ thắng, nếu có hệ số ngang nhau thì sẽ về đích cùng một lúc mỗi một con ngựa chỉ được thi đấu một lượt. Ai có tổng số trận thắng nhiều hơn thì sẽ thắng chung cuộc. Số trận <= 1000 trận. Bạn hãy giúp Điền Kỵ sắp xếp các lượt đấu để đạt số trận thắng cao nhất có thể.

Dữ liệu vào từ file DUANGUA.INP bao gồm:

- Dòng đầu là số lượng ngựa: n
- Dòng thứ hai có n số, số thứ i là hệ số của con ngựa thứ i của Điền Kỵ.
- Dòng thứ ba có n số, số thứ i là hệ số của con ngựa thứ i của vua Tề.

Kết quả ghi vào file DUANGUA.OUT gồm 1 số duy nhất ghi số trận thắng cao nhất mà Điền Kỵ có thể dành được.

DUANGUA.INP	DUANGUA.INP
3 4 6 2 9 3 5	5 3 7 12 5 8 13 5 9 14 6
DUANGUA.OUT	DUANGUA.OUT
2	3

Ý tưởng: Với mục tiêu dành nhiều trận thắng nhất có thể nên Điền Kỵ phải cố gắng đưa ra con ngựa thi đấu sao cho nó sẽ đấu với đối thủ mạnh nhất có thể của vua Tề mà vẫn dành được phần thắng

Để thực hiện được điều này ta sắp xếp hệ số của các con ngựa của cả Điền Kỵ và Vua Tề theo thứ tự giảm dần. Khi đó, con ngựa mạnh nhất sẽ được đưa ra thi đấu trước và nó sẽ thi đấu với con ngựa đầu tiên tìm được (Từ đầu dãy ngựa của Vua Tề trở đi) mà nó có thể thắng. Lần lượt như thế cho đến khi các chú ngựa của Điện Kỵ thi đấu hết.

Với Input:

3
4 6 2
9 3 5

Ta sắp xếp hai dãy số thành

6 4 2 và 9 5 3

Khi đó con ngựa có hệ số 6 của Điền Kỵ sẽ đấu với con ngựa hệ số 5 của Vua Tề và được một trận thắng. Con ngựa có hệ số 4 của Điền Kỵ sẽ đấu với con ngựa hệ số 3 của Vua Tề và được trận thắng thứ hai. Cặp ngựa còn lại Điền Kỵ bị thua và số trận thắng nhiều nhất có thể là 2.

Chương trình tham khảo:

```
const   fi='duangua.inp';
        fo='duangua.out';
        nmax=trunc(1e6);
        Type mang=array[1..nmax] of longint;

var     a,b,d:mang;
       k, n:longint;
        f, g: text;
procedure       enter;
var     i:longint;
begin
        assign(f,fi);reset(f);  assign(g, fo); rewrite(g);
        readln(f,n);
        for i:=1 to n do
                read(f,b[i]);
        readln(f);
        for i:=1 to n do
                read(f,a[i]);
        close(f);
end;
```

```
procedure  quicksort(var x:mang;l,h:longint);
var     i,j,key,tg:longint;
begin
       i:=l;
       j:=h;
       key:=x[(l+h) shr 1];
       repeat
             while x[i]<key do inc(i);
             while x[j]>key do dec(j);
             if i<=j then
                   begin
                         tg:=x[i];
                         x[i]:=x[j];
                         x[j]:=tg;
                         inc(i);
                         dec(j);
                   end;
       until i>j;
       if j>l then quicksort(x,l,j);
       if i<h then quicksort(x,i,h);
end;
procedure       main;
var i,j,dem:longint;f:text;
Begin
     dem:=0;  for i:=1 to n do d[i]:=0;
     for i:=1 to n do
          for j:=1 to n do
            if (a[i]<b[j]) and (d[j]=0) then
               begin
                  dem:=dem+1;
                  d[j]:=1;
                  break;
               end;
     write(g,dem);
     close(g);
end;

Begin
       enter;
       quicksort(a,1,n);
       quicksort(b,1,n);
       main;
end.
```

Nhận xét: Độ phức tạp thuật toán phụ thuộc vào thuật toán sắp xếp.

Bài 9. Trò chơi (Đề thi HSG tỉnh Nghệ An 2013)

Nhân dịp lễ giáng sinh, công viên trung tâm tổ chức trò chơi "con số may mắn". Mỗi em nhỏ đến tham dự sẽ được phát một số nguyên dương. Công viên có một thiết bị

quay số, mỗi lần quay sẽ tạo ngẫu nhiên một số nguyên dương có giá trị tuyệt đối không vượt quá 10^4. Người dẫn chương trình sẽ thực hiện N lần quay số. Số nào xuất hiện nhiều nhất trong N lần quay được gọi là con số may mắn và em nhỏ nào có con số may mắn thì sẽ được phần thưởng.

Yêu cầu: Cho N con số xuất hiện trong N lần quay. Bạn hãy giúp người dẫn chương trình xác định số lần xuất hiện của con số may mắn.

Dữ liệu vào từ file văn bản TC.inp:

- Dòng đầu là số N ($1 < N < 10^4$).
- Dòng tiếp theo có N số là các số xuất hiện trong N lần quay.

Kết quả ghi ra file văn bản TC.out:

Gồm một dòng duy nhất ghi số lần xuất hiện của con số may mắn.

Ví dụ:

TC.inp	TC.out	TC.inp	TC.out
5 4 3 4 4 15	3	7 12 5 10 5 8 10 9	2

Ý tưởng:

Ngoài phương pháp sử dụng kỹ thuật đánh dấu bài toán trên còn có thể sử dụng thuật toán sắp xếp như sau:

- Sắp xếp dãy theo chiều tăng dần
- Max:=0;
- Đếm các phần tử cạnh nhau và bằng nhau, tìm số lượng phần tử nhiều nhất:

```
i:=1;
repeat
        d:=1;
        while a[i]=a[i+1] do
                begin inc(d); inc(i); end;
        if d > max then Max:=d;
        inc(i);
until i > n;
```

- Đưa ra max.

Bài 10. Siêu thị Co.opmart (Đề thi HSG Trường THPT chuyên Nguyễn Quang Diêu năm 2014-2015)

Bờm được mẹ cho đi siêu thị Co.opmart nhân dịp siêu thị này vừa được khai trương, các thực phẩm đều được giảm giá, mẹ Bồm mua rất nhiều thực phẩm để dự trữ cho những ngày tết sắp đến. Sau khi chọn đủ các gói hàng cần mua, thanh toán tiền

xong và đến lúc cần đóng hàng vào hộp để mang về nhà. Số gói hàng mà hai mẹ con chọn mua là n gói với kích thước $k_1, k_2, ...,k_n$. Bờm có nhiệm vụ giúp mẹ đóng những gói hàng này vào những chiếc hộp giấy bìa cứng. Biết rằng siêu thị chỉ còn những chiếc hộp có kích thước m thỏa mãn $k_i \le m$ (i = 1,2,...,n). Hỏi Bờm cần ít nhất bao nhiêu hộp để có thể đóng đủ các gói hàng mang về?

Input: Vào từ file văn bản OPMART.INP

- Dòng 1: Chứa hai số nguyên n và m ($1 \le n \le 100, m \ge 1000$)
- Dòng 2: Chứa n số nguyên dương $k_1, k_2, ..., k_n$ ($1 \le k_i \le 1000$, với mọi i = 1,2,...n)

Output: Ghi ra file văn bản OPMART.OUT gồm một số nguyên duy nhất là số hộp ít nhất cần phải lấy.

Các số trên một dòng của Input files được ghi cách nhau ít nhất một dấu cách.

Ví dụ:

OPMART.INP	OPMART.OUT
6 200 30 70 150 80 120 75	3

Bài 11:. Bài toán cái Ba lô (Túi xách)

Một người đi du lịch có n loại đồ vật có trọng lượng và giá trị khác nhau. Nhưng anh ta chỉ có một túi xách có dung lượng w (có thể chứa được một số đồ vật sao cho tổng trọng lượng của các đồ vật này nhỏ hơn hoặc đúng bằng w).

Bạn hãy viết chương trình giúp người đi du lịch phải chọn lựa một danh sách các đồ vật mang đi như thế nào để tổng giá trị đồ vật mang đi là lớn nhất. Giả thiết mỗi loại đồ vật có đủ nhiều.

Dữ liệu: File vào gồm 3 dòng. Dòng đầu tiên của file vào chứa hai số nguyên dương n và w ($n, w \le 1000$). Dòng thứ hai ghi n số nguyên dương a_i ($a_i < 1000$, i = 1, 2, ..., n). Dòng cuối cùng ghi n số nguyên dương c_i ($c_i < 100.000$, i = 1, 2, ..., n). Các số trên một dòng cách nhau bởi một dấu cách.

Kết quả: File ra gồm 2 dòng. Dòng thứ nhất ghi tổng giá trị đồ vật mang đi lớn nhất. Dòng thứ hai ghi n số nguyên cách nhau bởi dấu cách, trong đó số thứ i là số lượng đồ vật i cần mang theo (i = 1, 2, ..., n). Nếu có nhiều cách mang đồ vật đều cho tổng giá trị lớn nhất thì ghi ra một cách bất kỳ trong chúng.

Ví dụ:

tuixach.in	tuixach.out
5 20 1 2 3 4 6 1 2 9 8 16	56 2 0 6 0 0

Lưu ý có các biến thể bài toán: số mỗi loại đồ vật có số lượng không hạn chế hay số lượng nhất định.

IV. Kĩ thuật tìm kiếm nhị phân:

Chia để trị là một tư tưởng rất phổ biến trong cuộc sống và được áp dụng rất hiệu quả trong Tin học. Tư tưởng cơ bản của phương pháp chia để trị là **Người ta phân bài**

toán thành các bài toán con, các bài toán con lại tiếp tục được phân thành các bài toán con nhỏ hơn, cứ tiếp tục như thế cho đến khi ta nhận được bài toán con đã có thuật giải hoặc có thể dễ dàng đưa ra thuật giải. Sau đó kết hợp nghiệm của các bài toán con để nhận được nghiệm của bài toán con lớn hơn để cuối cùng nhận được nghiệm của bài toán cần giải. Thông thường các bài toán con được phân chia là cùng dạng với bài toán ban đầu chỉ có cỡ của chúng là nhỏ hơn. sắp xếp nhanh (Quick Sort) và tìm kiếm nhị phân là hai thuật toán kinh điển thể hiện rõ tư tưởng chia để trị.

- Thuật toán sắp xếp nhanh (Quick Sort): Xem thêm SGK Tin 10, Chương trình dịch Free Pascal cung cấp sẵn mã nguồn.

- Thuật toán tìm kiếm nhị phân là một trong những thuật toán được áp dụng nhiều trong khoa học cũng như trong thực tế. Trong các kỳ thi học sinh giỏi các cấp của môn Tin học thì bài toán tìm kiếm nhị phân là một trong những bài toán thường được các tác giả chọn làm đề bài của mình.

Đã có rất nhiều tác giả viết về thuật toán tìm kiếm nhị phân tuy nhiên với kinh nghiệm của mình tôi muốn đưa ra một cách tiếp cận các bài toán tìm kiếm nhị phân từ đơn giản đến phực tạp để giúp học sinh có thể tiếp thu dễ dàng hơn khi gặp phải bài toán tìm kiếm nhị phân.

1. Thuật toán tìm kiếm nhị phân cơ bản

Bài toán 12: Cho dãy A gồm N phần tử nguyên từ $A_1, A_2,\ldots, A_N$ được sắp xếp tăng dần và một số nguyên X. Hãy tìm một vị trí trong dãy A có giá trị bằng X.

Thuật toán:

```
Function  Tknpcb(X:longint): longint;
Var     d, c, g: Longint;
Begin
        d:=1;
        c:=N;
        While d<=c Do
              Begin
                  g:=(d + c) Div 2;
                  if A[g]=X then exit(g);
                  if  A[g]<X  then  d:=g +1
                  Else c:=g-1;
              End;
 Exit(0);
End;
```

- Thuật toán trên sẽ trả lại chỉ số của một phần tử có giá trị bằng X, nếu không có phần tử nào thỏa mãn thì hàm sẽ nhận giá trị bằng 0.

- Thuật toán có độ phức tạp là
  O(lg(n)).

- Lưu ý: Cận bài toán TKNP.

Bài toán 13: Bài toán cổ

Vừa gà, vừa chó bó lại cho tròn

N con, m chân

Gacho.inp	Gacho.out
36 100	22 14 1

Với n, m nhập từ bàn phím. In ra màn hình, số gà và số chó thỏa mản điều kiện trên, nếu không có thì in ra số 0.

Ý tưởng: Với x số gà ta tìm kiếm nhị phân số chó trong đoạn 1..m

```
uses crt;
Const fi='gacho.inp';
      fo='gacho.out';
Var  n, m, i: longint;
   x, dem: longint;

Function tknp(x: Longint): boolean;
Var d, c, g: longint;
   begin
      d:=1;
      c:=n;
      while d<=c do
        begin
          g:= (d+c) div 2;
          if (x+g=n) and (x*2 + 4*g=m) then
            begin
              dem:=dem+1;
              writeln('Ga: ', x, ' cho: ', g);
              exit(true);
            end;
            if x*2+ 4*g> m then C:=G-1
            else  d:=G+1;
       end;
   end;

   Begin
     clrscr;
     readln(n, m);
     dem:=0;
     for x:=1 to n do TKNP(x);
     write('so bo la: ', dem);
     readln;
   end.
```

Bài toán 13: Nhập vào một số nguyên dương N và một số nguyên tố K (K<N). Liệt kê tất cả các số nguyên tố nhỏ hơn N lưu vào dãy A. Xóa số nguyên tố K trong dãy.

Ý tưởng:

- Dùng hàm kiểm tra nguyên tố của một số nguyên dương để kiểm tra tính nguyên tố của các số có giá trị từ 2 đến N, lưu các số nguyên tố này vào dãy A..
- Do duyệt các số lần lượt từ 2 đến N nên dãy A thu được là dãy tăng.
- Thực hiện tìm kiếm nhị phân trên dãy A để tìm vị trí mà A_{vt}=K.
- Xóa phần tử ở vị trí vt khỏi dãy A.

```
Chương trình tham khảo
Program ngto;
var n,i,j,k,vt:longint;
      a:array[1..1000] of longint;
Function nt(x:longint):boolean;
var k:longint;
begin
if (x=2) OR (x=3) then exit(true);
if (x=1) or (x mod 2=0) or (x mod 3=0) then exit(False);
k:=-1;
repeat
      inc(k,6);
      if (x mod k=0) or (x mod (k+2) =0) then break;
until (k>trunc(sqrt(x)));
exit(k>trunc(sqrt(x)));
end;
procedure      main;
var    first,mid,last,vt,i:longint;
begin

  first:=1;
  last:=j;
  while first<=last do
        begin
           mid:=(first+last) div 2;
           if (a[mid]=K)  then
              begin
                vt:=mid;
                break;
              end

           else
              if a[mid]>K then first:=mid+1
           else last:=mid-1;
        end;
  for i:=vt to j do a[i]:=a[i+1];
  dec(j);
end;

begin
      write('nhap n,k: ');readln(n,k);
```

```
            j:=0;
            for i:=1 to n do
                  if nt(i) then begin inc(j);a[j]:=i;end;
            Writeln('Day so nguyen to la: ');
            For i:=1 to j do write(a[i],' ');
            writeln;
            main; writeln(vt);
            Writeln('Day nguyen to sau khi xoa ',K,' la: ');
            For i:=1 to j do
                  write(A[i],' ');
            readln
      end.
```

Trên thực tế không phải bao giờ người ta cũng yêu cầu tìm kiếm một phần tử bằng X mà có thể yêu cầu tìm phần tử gần bằng X nhất. Khi đó ta phải chỉnh sửa thuật toán trên ở một số bước để phù hợp với yêu cầu của bài toán. Cụ thể, với bài toán này ta chia thành hai bài toán con:

2. Tìm phần tử lớn nhất nhưng nhỏ hơn hoặc bằng X

Như vậy khi ta so sánh phần tử giữa với X, nếu nó nhỏ hơn hoặc bằng X ta sẽ xác nhận kết quả tạm thời rồi tìm đoạn sau để có nghiệm tốt hơn (gần bằng X hơn).

```
Function      Tknp1(X:longint):longint;
Var     d, c, g, kq: Longint;
Begin
        kq:=0;          d:=1; c:=N;
        While d<=c Do
                Begin
                g:=(d + c) Div 2;
                If A[g]<=X then
                        begin
                                kq:=g;
                                d:=g +1;
                        end;
                Else c:=g-1; End;
        Exit(kq);
End;
```

3. Tìm phần tử nhỏ nhất nhưng lớn hơn hoặc bằng X

Ngược lại với thuật toán trên ta thấy nếu phần tử giữa lớn hơn hoặc bằng X thì ta cập nhật kết quả hiện thời rồi tìm đoạn trước đó để có thể có kết quả tốt hơn.

```
Function      Tknp2(X:longint):longint;
 Var     d, c, g, kq: Longint;
Begin
        kq:=0;
        While d<=c Do
                Begin
                        g:=(d + c) Div 2;
                        If A[g] >= X then
                                begin
```

```
                                    kq:=g;
                                    c:=g - 1;
                                end;
                        Else d:=g + 1; End;
        Exit(kq);
End;
```

Bài 14. Dãy con tổng bằng K

Cho một dãy gồm N số nguyên dương, xác định dãy con liên tiếp có tổng bằng K? Nếu không tồn tại xuất ra số 0.

Dayso.inp	Dayso.out
6 10 1 5 3 2 4 6	2 4

Với bài toán này ta có thể thực hiện bằng cách duyệt tất cả các dãy con có thể có từ dãy N phần tử và thực hiện tính tổng của dãy con đó rồi so sánh với K. Giải thuật có thể thực hiện như sau:

```
For i:=1 to N do
  Begin
        S:=0;
        For j:=i to N do S:=S+A[j];
        If s=k then begin write('tim thay');break; end;
End;
```

Thuật toán trên cho ta độ phức tạp cỡ $O(n^2)$. Ta có thể cải tiến thuật toán bằng nhận cách sau:

- Tạo dãy S trong đó $S_i=A_1+...+A_i$;
- Nhận thấy do dãy A gồm các số nguyên dương nên dãy S là tăng dần.
- Để xác định đoạn $[A_i,A_j]$ có tổng các phần tử bằng K hay không ta thực hiện tìm kiếm nhị phân trên đoạn từ vị trí i đến vị trí j trong dãy S mà $S_j-S_i=K$.

Thuật toán như sau:

Bước 1. Tạo dãy S từ dãy A:

S[0]:=0;

Với mỗi i nhận giá trị từ 1 đến n làm S[i]:=S[i-1]+A[i];

Bước 2. Res=-1; {Res dùng để đánh dấu vị trí cuối của dãy con cần tìm}

Bước 3. Với mỗi i nhận giá trị từ 1 đến N làm:

3.1 Cho first:=i; last:=n;

3.2 chừng nào first<=last làm:

3.2.1 mid:=(first+last) shr 1;

3.2.2 Nếu s[mid]-s[i-1]=k thì

res:=mid ; dau:=firt; cuoi:=giua;

thoát khỏi vòng lặp;

Nếu không thì

Nếu s[mid]-s[i-1]>k thì last:=mid-1;

first:=mid+1;

3.3. Nếu Res<>-1 thì xuất giá trị dau, cuoi và kết thúc.

Bước 4. Đưa ra kết luận không tìm thấy và kết thúc.

```
const   fi='Dayso.inp'; fo='dayso.out';
        nmax=100000;
var     dem, dau, cuoi, i, n,k:longint;
        a:array[1..nmax]of longint;
        s:array[0..nmax]of longint;
        f1, f2: text;

procedure tknp(i: Longint);
Var d, c, g: longint;
    begin
        d:=i;
        c:=n;
        while d<=c do
          begin
             g:= (d+c) div 2;
             if s[g]-s[d-1]=k then
                begin
                  dem:=dem+1;
                   dau:=d; cuoi:=g;
                   exit;
                end;
              if s[g]-s[d-1]> k then C:=G-1
              else  d:=G+1;
         end;
    end;

BEGIN
        assign(f1, fi); reset(f1);
        assign(f2, fo); rewrite(f2);
        read(f1, n, k); dem:=0;
        for i:=1 to n do
           begin
               read(f1, a[i]);
               s[i]:=s[i-1]+a[i];
           end;
        for i:=1 to n do tknp(i);
        if dem=0 then writeln(f2, dem)
        else write(f2, dau,' ', cuoi);

        close(f1); close(f2);
END.
```

Bài toán 15: Trò chơi với dãy số (Đề HSG quốc gia năm 2008)

Hai bạn học sinh trong lúc nhàn rỗi nghĩ ra trò chơi sau đây. Mỗi bạn chọn trước một dãy số gồm n số nguyên. Giả sử dãy số mà bạn thứ nhất chọn là: $B_1,B_2,..,B_n$, còn dãy số mà bạn thứ hai chọn là $C_1,C_2,\ldots,C_n$.

Mỗi lượt chơi, mỗi bạn đưa ra một số hạng trong dãy số của mình. Nếu bạn thứ

nhất đưa ra số hạng B_i, còn bạn thứ hai đưa ra số hạng C_j thì giá trị của lượt chơi đó là $|B_i$

$+ C_j|$.

Hãy xác định giá trị nhỏ nhất của một lượt chơi trong số các lượt chơi có thể.

Ví dụ: Giả sử dãy số bạn thứ nhất chọn là 1, -2; còn dãy số mà bạn thứ hai chọn là 2, 3. Khi đó các khả năng có thể của một lượt chơi là (1,2), (1,3), (-2,2), (-2,3). Như vậy, giá nhỏ nhất của một lượt chơi trong số các lượt chơi có thể là 0 tương ứng với giá của lượt chơi (-2,2).

Yêu cầu

Hãy xác định giá nhỏ nhất của một lượt chơi trong số các lượt chơi có thể.

Dữ liệu

- Dòng đầu tiên chứa số nguyên dương N ($N \le 10^5$)
- Dòng thứ hai chứa dãy số nguyên $b_1, b_2, \ldots, b_n$ ($|b_i| \le 10^9, i=1, 2, \ldots, N$)
- Dòng thứ ba chứa dãy số nguyên $c_1, c_2, \ldots, c_n$ ($|c_i| \le 10^9, i=1, 2, \ldots, N$)

Hai số liên tiếp trong một dòng được ghi cách nhau bởi dấu cách.

Kết quả

Ghi ra giá nhỏ nhất tìm được.

Ràng buộc

60% số test ứng với 60% số điểm của bài có $1 \le N \le 1000$

Ví dụ

SEQGAME.IN	SEQGAME.OUT
2 1 -2 2 3	0

Ý tưởng:

Ta có $|B_i + C_j|$ đạt giá trị nhỏ nhất khi mà C_j gần bằng $-B_i$ nhất. Vậy bài toán thực chất yêu cầu: Với mỗi phần tử B_i ta tìm kiếm nhị phân một phần tử C_j gần bằng phần tử B_i nhất. Để thực hiện được yêu cầu này ta chỉ cần sắp xếp tăng dần mảng C rồi sử dụng hai thuật toán tìm kiếm nhị phân đã đề xuất ở trên.

```
const fi='seoque.inp';
      fo='seoque.out';
      nn=100000;
Type bg= array[1..nn] of longint;

Var b, c: bg;
    n, k, kq: longint;
```

```
    f, g: text;
Procedure nhapdl;
Var i: longint;
Begin
      assign(f, fi); reset(f);
      read(f, n);
      for i:=1 to n do  read(f, b[i]);
      for i:=1 to n do read(f, c[i]);
      close(f);
end;


procedure qs1(var c: bg; l,r: longint);
      var
         i,j,x,y: longint;
      begin
         i:=l;
         j:=r;
         x:=c[(l+r) div 2];
         repeat
           while c[i]<x do
            inc(i);
           while x<c[j] do
            dec(j);
           if not(i>j) then
             begin
                y:=c[i];
                c[i]:=c[j];
                c[j]:=y;
                inc(i);
                j:=j-1;
             end;
         until i>j;
         if l<j then
           qs1(c, l,j);
         if i<r then
           qs1(c, i,r);
      end;

      function min(x, y: longint): longint;
      Begin
           min:=x;
           if x>y then min:=y;

      end;
    function TKNP(x: longint): longint;
       Var dau, cuoi, giua, kqtam, kq1, kq2: Longint;
       Begin
       // tim so c[i] lon hon x
           dau:=1;
           cuoi:=n;


           while dau<=cuoi do
              begin
                giua:=(dau+cuoi) div 2;
                if (c[giua]>=x) then
                  begin
```

```
                    kq1:= (x+c[giua]);

                     cuoi:=giua-1;
                end
             else dau:=giua+1;

            end;
         // tim so c[i] nho hon x
         dau:=1;
         cuoi:=n;


         while dau<=cuoi do
            begin
              giua:=(dau+cuoi) div 2;
              if (c[giua]<=x) then
                begin
                    kq2:= (x+c[giua]);

                     dau:=giua+1;
                end
             else cuoi:=giua-1;

            end;
            if kq1>kq2 then kqtam:=kq2 else kqtam:=kq1;
          exit(kqtam);
   end;
Begin

    kq:=Trunc(1E9);

   nhapdl;

   qs1(b, 1,n);
   qs1(c, 1,n);
   for k:=1 to n do kq:= min(kq, TKNP(b[k]));
   assign(g, fo); rewrite(g);
   write(g,kq);

   close(g);
end.
```

Bài toán 16: Tư vấn tuyển sinh (Đề Chọn đội tuyển 2016 tỉnh Đồng Tháp)

Trong đợt tư vấn tuyển sinh vào các trường Đại học, Cao đẳng có M học sinh muốn được tư vấn. Ban tổ chức bố trí N bàn để tư vấn, thời gian tư vấn của mỗi bàn thứ i là t_i (i=1..N). Biết rằng tại một thời điểm thì mỗi bàn chỉ có thể tư vấn cho một học sinh.

- Yêu cầu: Hãy tìm cách sắp xếp học sinh vào các bàn tư vấn sao cho thời gian hoàn thành buổi tư vấn của ban tổ chức là ít nhất.

- Dữ liệu vào: Cho từ tệp văn bản TUVAN.INP có cấu trúc:

- Dòng thứ nhất ghi hai số nguyên dương N và M lần lượt là số bàn tư vấn và số học sinh tham gia tư vấn ($1 \le N \le 10^5$; $1 \le M \le 10^9$).
- Dòng thứ i trong N dòng tiếp theo, mỗi dòng ghi một số nguyên dương t_i là thời gian để tư vấn xong cho một học sinh của bàn thứ i ($1 \le t_i \le 10^9$; i=1..N).

- Kết quả: Ghi vào tệp văn bản TUVAN.OUT gồm một dòng chứa một số nguyên dương duy nhất là thời gian ít nhất để hoàn thành buổi tư vấn.

- Ví dụ :

TUVAN.INP	TUVAN.OUT
7 10 3 8 3 6 9 2 4	8
2 6 7 10	28

Hạn chế kĩ thuật:

+ Có 60% số test tương ứng 60% số điểm có $1 \le N, M \le 10^4$

+ Có 40% số test tương ứng 40% số điểm có $10^4 < N \le 10^5; 10^4 < M \le 10^9$

Thuật toán 1: Duyệt $O(n^2)$ đạt 60% test

Mảng A lưu thời gian các bàn, mảng B lưu thời gian phân bổ các học sinh.

- xét hs 1: tìm Min(A[i]+B[i]) lưu vị trí VT cập nhật mảng B[VT]:=A[vt]+B[vt];

- Tìm Max của mảng B-> được kết quả bài toán.

Thuật toán 2: Chặt nhị phân theo thời gian O(nlgn) đạt 100% test.
Với một thời gian t, t thỏa mãn yêu cầu của đề bài thì tổng t/a[i] với a[i] là thời bàn i cần để tư vấn xong cho một người phải lớn hơn hoặc bằng số người được tư vấn.

Vậy nếu thời gian này thỏa mãn thì ta chặt xuống vì đề bài yêu cầu thời gian nh ỏ nhất, ngược lại sẽ chặt lên.

```
const maxn=trunc(1e5) +1;
Var
dau, cuoi, kq, n, m, giua : int64;
a: array[0..maxn - 1] of int64;
f, g:text;
i : longint;

function ok( p: int64): boolean;
var kq1:int64;
i : longint;
begin
```

```
    kq1 := 0;
    for i := 1 to n do
    kq1 := kq1 + p div a[i];
    if(kq1>=m) then exit(true);
     exit(false);
end;

begin
     dau := 0;
     cuoi := trunc(1e18);
     kq := 0;
        assign(f, 'tUVAN.inp');  reset(f);
     assign(g, 'TUVAN.out');  rewrite(g);
    read(f, n, m);
    for i := 1 to n do read(f, a[i]);
    while(dau<=cuoi)  do
    begin
        giua:=(dau+cuoi) div 2 ;

        if ok(giua) then
        begin
            cuoi := giua-1;
            kq := giua;
        end
        else dau := giua+1;
    end;
    write(g, kq);
     close(f);
     close(g);
end.
```

Bài 17. THẢ DIỀU (HSG tỉnh Đồng Tháp năm 2015)

Trong một cuộc thi thả diều, Ban giám khảo căn cứ vào độ cao của mỗi chiếc diều đạt được để xếp hạng cho chiếc diều đó. Nhằm tạo ra hứng thú cho cuộc thi thả diều, Ban tổ chức đưa ra thể lệ cuộc thi theo một cách đặc biệt: Những chiếc diều không được thả cùng một lúc, mà thả theo trình tự từng chiếc một. Khi một chiếc diều được thả lên trời, Ban giám khảo căn cứ vào độ cao của chiếc diều và xếp hạng cho chiếc diều đó bằng cách so độ cao của nó với độ cao của những chiếc diều đã thả trước đó. Chẳng hạn: Độ cao của 6 chiếc diều theo thứ tự được thả như sau: 78 24 68 40 39 89. Chiếc diều đầu tiên được xếp hạng nhất vì trước nó chưa có chiếc diều nào được thả. Chiếc diều thứ hai xếp hạng 2 vì 24 < 78. Chiếc diều thứ ba cũng xếp hạng 2 vì 24 < 68 < 78. Chiếc diều thứ tư xếp hạng 3 vì 24 <40 <68 <78. Chiếc diều thứ năm xếp hạng 4 vì 24 <39 <40 <68 <78 và chiếc diều cuối cùng xếp hạng nhất với độ cao 89 vì 24 <39 < 40 <68 <78 <89. Như vậy trình tự dãy số xếp hạng được công bố sẽ là: 1 2 2 3 4 1

Yêu cầu: Có N chiếc diều lần lượt được thả lên trời, em hãy cho biết dãy số biểu diễn giá trị xếp hạng của N chiếc diều.

Dữ liệu vào: Cho từ tệp văn bản THADIEU.INP có cấu trúc:

- Dòng thứ nhất ghi số nguyên dương N cho biết số chiếc diều tham gia dự thi.

- N dòng tiếp theo, mỗi dòng ghi một số nguyên dương mô tả độ cao của một chiếc diều, theo thứ tự mà nó được thả lên.

Dữ liệu ra: Ghi vào tệp văn bản THADIEU.OUT có cấu trúc gồm N dòng, dòng thứ i ghi số nguyên biểu diễn giá trị xếp hạng của chiếc diều thứ i tại thời điểm nó được thả lên.

Ví dụ:

THADIEU.INP	THADIEU.OUT
6 78 24 68 40 39 89	1 2 2 3 4 1

Hạn chế kĩ thuật:

- N ≤ 45000 và không có hai chiếc diều nào có cùng độ cao.
- Độ cao của mỗi chiếc diều là một số nguyên dương không vượt quá 2^{31}
- Có 60% số test với N ≤ 8000

Thuận toán 1: Duyệt tìm hạng lưu vào mảng B, $O(n^2)$ đạt 60% test

```
const fi='thadieu.inp'; fo='thadieu.out';
      nmax=45000;
Var i,j, n: longint;
    f, g: text;
    a, b: array[1..nmax] of longint;
Begin
   assign(f, fi); assign(g, fo);
   reset(f); rewrite(g);
   read(f, n);
   for i:=1 to n do
      begin
         read(f, a[i]);
         b[i]:=i;//khoi tao voi b[i] co hang la i;
      end;
   close(f);
   for i:=1 to n do
        for j:=i downto 1 do
              if a[i]>a[j] then b[i]:=b[i]-1;

for i:=1 to n do writeln(g, b[i]);
close(g);
End.
```

*** Thuật toán 2 (Tổ chức dữ liệu kết hợp tìm kiếm nhị phân):**

Đây là bài 3 của đề dùng để phân hóa học sinh, có nhiều cách giải có O(nlgn) đạt 100% số test. Phương pháp chặt nhị phân như sau:

Tổ chức lưu trữ dãy số input trong một bảng $\sqrt{N} * \sqrt{N}$ phần tử. Trên mỗi dòng trong bảng các số được lưu theo trật tự giá trị giảm dần. Sử dụng *Tìm kiếm nhị phân* để tìm vị trí thích hợp của phần tử khi lưu trữ và xác định thứ hạng của phần tử. Độ phức tạp thời gian của thuật toán tương đương với $N * \sqrt{N} * \log(N) < N^2$, thuật toán này tốt hơn đáng kể so với thuật toán tầm thường.

```
program kite;
const
    fi='thadieu.inp';
    fo='Thadieu.out';
    maxn=45000;
    maxm=250;
type bang=array[0..maxm,0..maxm]of longint;
var a : bang;
    next: array[0..maxm]of longint;
    i, n, x, y, m: longint;
    f, g: text;
   d:longint;
function findrowrank(i:longint; x: longint; l,r: longint):
longint;
var mid:longint;
begin
    if x > a[i,l] then exit(l);
    if x < a[i,r] then exit(r+1);
    mid:= (l+r) div 2;
    if x > a[i,mid] then
        if x < a[i,mid-1] then exit(mid)
        else exit(findrowrank(i,x,l,mid-1));
    if x < a[i,mid] then
        if x > a[i,mid+1] then exit(mid+1)
        else exit(findrowrank(i,x,mid+1,r));
end;

function rank(x: longint): longint;
var i:longint; b:longint;
begin
    b := 0;
    for i:= 1 to m do b := b + findrowrank(i,x,1,next[i]-1)-1;
    rank := b + 1;
end;

procedure insertx(x:longint);
var i,j,k:longint;
begin
    i:=1;
    while next[i]>m do inc(i);
    j:=1;
    while a[i,j]>x do inc(j);
    for k:=next[i]+1 downto j+1 do a[i,k]:=a[i,k-1];
    a[i,j]:=x;
    inc(next[i]);
end;

begin
     d:=0;
    fillchar(a,sizeof(a),0);
    assign(f,fi); reset(f);
```

```
    assign(g,fo); rewrite(g);
    readln(f,n);
    m:=trunc(sqrt(n))+1;
    for i:=1 to m do next[i]:=1;
    for i:=1 to n do
    begin
        readln(f,x);
        y := rank(x);
        insertx(x);
        writeln(g,y);
    end;
    close(f);
    close(g);
end.
```

V. Duyệt xuôi và duyệt ngược

Giai đoạn duyệt xuôi: từ điểm xuất phát, dựa trên những nhận xét hợp lý nào đó sẽ lần lượt tìm ra những điểm trung gian cần phải qua trước khi tới đích trên hành trình ngắn nhất. Trong giai đoạn tìm kiếm theo chiều thuận này, người ta lưu lại vết của hành trình vào một mảng *trace* mà trace[i]=j có ý nghĩa điểm j là điểm cần phải qua ngay trước điểm i trong hành trình ngắn nhất.

Giai đoạn duyệt ngược: Với mảng *trace,* từ trace[kt]=i_1 biết được trước khi đến điểm đích *kt* phải qua điểm i_1. Tương tự, từ trace[i_1]=i_2 biết được trước khi đến điểm i_1 phải qua điểm i_2…, cuối cùng, từ trace[i_k]=xp biết được trước khi đến điểm i_k phải qua điểm xuất phát *xp.* Suy ra hành trình ngắn nhất là $xp \to i_k \to \ldots \to i_2 \to i_1 \to kt$.

Bài toán 1. Phân tích số thành tổng hai lập phương

Phân tích một số nguyên dương N thành tổng hai lập phương của hai số nguyên dương. Có bao nhiêu cách khác nhau?

Input: Số N nhập từ bàn phím

Output: Đưa kết quả ra màn hình, mỗi cách trên một dòng

Phân tích.

Bình thường có thể xét mọi cặp số nguyên dương i và j có giá trị tăng dần để chọn ra những cặp (i,j) mà $i^3+j^3=N$.

Tuy nhiên nhận thấy nếu i tăng thì j giảm nên ta có thể dùng phương pháp duyệt đồng thời ngược và xuôi (i: tăng dần, j: giảm dần) như sau: Ban đầu chọn j có giá trị cao nhất, còn i =1. Kiểm tra $k=i^3+j^3$, nếu k=N thì cặp (i,j) này là một kết quả, tiếp tục tăng i và giảm j, nếu k>N thì giảm j, nếu k<N thì tăng i. Công việc này được lặp cho đến khi i>j. Cách duyệt này hiệu quả hơn cách bình thường.

Văn bản chương trình.

```
uses   Crt;
var    i,j,count,N,k  : Integer;
BEGIN
   write('Nhập N = '); Readln(N);
   count:= 0;
```

```
  i     := 1;
  j     := 1;
  while (j*j*j+1<N) do Inc(j);
  repeat
     k :=i*i*i+j*j*j;
     if k=N then begin
         Inc(count);
         Write(i:4,j:4); Inc(i) ; Dec(j);
     end;
     if k < N then Inc(i);
     if k > N then Dec(j);
  until i >j;
  writeln('Co ',count,' Cach phan tich ' );
  readln
END.
```

Bài toán 18. Trò chơi với dãy số - Seqgame (HSG QG 2008)

Hai bạn học sinh trong lúc nhàn rỗi nghĩ ra trò chơi sau đây. Mỗi bạn chọn trước một dãy số gồm N số nguyên. Giả sử dãy số mà bạn thứ nhất chọn là:

$b_1, b_2, \ldots, b_n$

còn dãy số mà bạn thứ hai chọn là:

$c_1, c_2, \ldots, c_n$

Mỗi lượt chơi mỗi bạn đưa ra một số hạng trong dãy số của mình. Nếu bạn thứ nhất đưa ra số hạng b_i ($1 \le i \le N$), còn bạn thứ hai đưa ra số hạng c_j ($1 \le j \le N$) thì giá của lượt chơi đó sẽ là $|b_i+c_j|$.

Ví dụ: Giả sử dãy số bạn thứ nhất chọn là 1, -2; còn dãy số mà bạn thứ hai chọn là 2, 3. Khi đó các khả năng có thể của một lượt chơi là (1,2), (1,3), (-2,2), (-2,3). Như vậy, giá nhỏ nhất của một lượt chơi trong số các lượt chơi có thể là 0 tương ứng với giá của lượt chơi (-2,2).

Yêu cầu

Hãy xác định giá nhỏ nhất của một lượt chơi trong số các lượt chơi có thể.

Dữ liệu

- Dòng đầu tiên chứa số nguyên dương N ($N \le 10^5$)
- Dòng thứ hai chứa dãy số nguyên $b_1, b_2, \ldots, b_n$ ($|b_i| \le 10^9, i=1, 2, \ldots, N$)
- Dòng thứ ba chứa dãy số nguyên $c_1, c_2, \ldots, c_n$ ($|c_i| \le 10^9, i=1, 2, \ldots, N$)

Hai số liên tiếp trong một dòng được ghi cách nhau bởi dấu cách.

Kết quả

Ghi ra giá nhỏ nhất tìm được.

Ràng buộc

60% số test ứng với 60% số điểm của bài có $1 \le N \le 1000$

Ví dụ

SEQGAME.IN	SEQGAME.OUT
2 1 -2 2 3	0

Phân tích

+ Sắp xếp tăng từng dãy số

+ Dùng 2 biến chạy xuôi và ngược là *i* và *j*. Biến *i* dùng duyệt xuôi mảng b(N) bắt đầu từ vị trí 1, biến *j* dùng duyệt ngược mảng c(N) bắt đầu từ vị trí N.

+ Vòng lặp thực hiện trong khi (i<=N) và (j>=1):

- Tính $v=b_i+c_j$

- Nếu v>=0 thì giảm j (để |v| giảm đi)

Ngược lại nếu v<0 thì tăng i (vì số âm khi được tăng lên thì trị tuyệt đối của nó giảm đi.

```
Program NKSGAME;
Type
  Table =Array[1..100001] of LongInt;
Var
  n :LongInt;
  A,B :Table;

  procedure Enter;
  var
    i :LongInt;
  begin
    ReadLn(n);
    for i:=1 to n do Read(A[i]);
    for i:=1 to n do Read(B[i]);
  end;

  procedure Sort(var x :Table; l,h :LongInt);
  var
    i,j,k,tmp :LongInt;
  begin
    if (l>=h) then Exit;
    i:=l; j:=h; k:=X[(l+h) div 2];
    repeat
      while (X[i]<k) do Inc(i);
      while (X[j]>k) do Dec(j);
      if (i<=j) then
        begin
          if (i<j) then
            begin
              tmp:=X[i]; X[i]:=X[j]; X[j]:=tmp;
            end;
```

```
          Inc(i); Dec(j);
        end;
    until (i>j);
    Sort(X,l,j); Sort(X,i,h);
  end;

  function Min :LongInt;
  var
    i,j,k :LongInt;
  begin
    Min:=High(LongInt);
    i:=1; j:=n;
    repeat
      k:=A[i]+B[j];
      if (k=0) then Exit(k);
      if (Min>Abs(k)) then Min:=Abs(k);
      if (k<0) then Inc(i) else Dec(j);
    until (i>n) or (j<1);
  end;

Begin
  Assign(Input,''); Reset(Input);
  Assign(Output,''); Rewrite(Output);
  Enter;
  Sort(A,1,n);
  Sort(B,1,n);
  Write(Min);
  Close(Input); Close(Output);
End.
```

VI. Kỹ thuật lùa bò vào chuồng

Ngoài việc dùng kĩ thuật đánh dấu trong khi duyệt, người ta còn hay dùng kĩ thuật "*lùa bò vào chuồng*". Nội dung của kĩ thuật này là: khi duyệt, những thành phần nào có đặc điểm giống nhau thì lưu vào cùng một chỗ. Kĩ thuật này có cái tên ngộ nghĩnh như vậy vì quá trình thực hiện tương tự như công việc lùa những con bò cùng loại vào cùng một chuồng.

Ví dụ xét bài toán: cho mảng A(1..N) một chiều gồm N phần tử là các số nguyên không âm đánh số từ 1 đến N, có giá trị không vượt quá M, hãy tìm số nguyên không âm nhỏ nhất chưa có mặt trong mảng A.

Giải bài toán này ta làm như sau: Dùng mảng một chiều B[0..M+1] để làm vai trò dãy chuồng bò. Duyệt mảng A[1..N], cho "con bò" A[i] vào "chuồng" B[j] với chỉ số j=A[i], nghĩa là tăng B[A[i]] lên một đơn vị. Sau đó duyệt lại B theo chỉ số tăng dần từ 0, gặp phần tử đầu tiên bằng 0 thì chỉ số của phần tử này chính là số nguyên không âm nhỏ nhất chưa có mặt trong mảng A (loại bò này không có mặt trong đàn bò). Kỹ thuật này có tốc độ tuyến tính (chỉ cần duyệt mảng A một lần).

Bài 19. Mã nhân viên.

Tổng Giám đốc công ty X nổi tiếng là một người kĩ lưỡng. Ông ta thực hiện việc quản lý nhân viên của mình bằng cách gán cho mỗi nhân viên một mã số khác nhau.

Công ty có N nhân viên, nhân viên i (i=1, 2, …N) có mã số A_i. Do bận đi công tác nước ngoài một thời gian dài nên ông trao quyền quản lý công ty cho một người khác. Khi ông trở về, công ty đã có sự thay đổi về số lượng nhân viên. Khi tiếp nhận thêm nhân viên mới, ông yêu cầu muốn biết mã số nhỏ nhất có thể gán cho nhân viên mới.

Yêu cầu: Cho N mã số của nhân viên trong công ty. Hãy tìm mã số nhỏ nhất chưa xuất hiện trong N mã số đã cho

Dữ liệu vào cho trong tệp văn bản MASO.IN: Dòng đầu là số N ($1<N\leq 30000$). N dòng tiếp theo, dòng thứ i ghi số Ai ($i=1..N; 1\leq Ai\leq 109$)

Kết quả ghi vào tệp văn bản MASO.OUT một số duy nhất là mã số tìm được.

Ví dụ

MASO.IN	MASO.OUT
6	2
7	
5	
6	
1	
3	
4	

Phân tích Đây là bài sử dụng kỹ thuật "lùa bò vào chuồng".

Chương trình tham khảo

```
const fi='MASO.in';
      fo='MASO.out';
var   n, k : longint;
      b : array[0..30001] of longint;
      ai : longint;
Procedure readfile;
var f : text; i : longint;
begin
  assign(f,fi); reset(f);
  readln(f,n);
  for i:=1 to n do b[i] := 0;
  for i:=1 to n do begin
    read(f,ai);
    if ai<=n+1 then inc(b[ai]);
  end;
  close(f);
end;
Procedure process;
var i : longint;
begin
  k := 1;
  for i:=1 to n+1 do
  if b[i]=0 then begin
     k := i;
     exit;
  end;
end;
Procedure writefile;
var f : text; i : longint;
begin
  assign(f,fo); rewrite(f);
  writeln(f,k);
  close(f);
end;
BEGIN
```

```
  readfile;
  process;
  writefile;
END.
```

Bài 20. Tìm tổng Nim

Cho bảng G hai chiều có dòng tiêu đề (dòng đầu tiên) là các giá trị của x (nguyên không âm) và cột tiêu đề (cột bên trái) là các giá trị của y (nguyên không âm). Giao giữa dòng giá trị của y và của x là một số nguyên z không âm gọi là tổng Nim của x và y. Quy tắc tìm tổng Nim z của x và y như sau: z là số nguyên không âm nhỏ nhất khác với các số bên trái nó và phía trên nó. Hãy lập trình hiện trên màn hình bảng G có các giá trị z là tổng Nim của x và y với $0 \le x, y \le 9$.

Phân tích Dùng kỹ thuật "lùa bò vào chuồng".

Chương trình tham khảo

```
Program tim_NIM;
var   g  : array[0..9,0..9] of integer;
      c  : array[0..100] of integer;
      i,j,k : integer;
begin
    fillchar(g,sizeof(g),0);
    g[0,0] := 0;
    for i:=0 to 9 do g[i,0] := i;
    for j:=0 to 9 do g[0,j] := j;
    for i:=1 to 9 do
    for j:=1 to 9 do begin
        fillchar(c,sizeof(c),0);
        for k:=0 to i-1 do c[g[k,j]] := 1;
        for k:=0 to j-1 do c[g[i,k]] := 1;
        k:=0;
        while c[k]=1 do inc(k);
        g[i,j] := k;
    end;
    for i:=0 to 9 do begin
        for j:=0 to 9 do write(g[i,j]:3);
        writeln;
    end;
    readln
end.
```

VII. Kĩ thuật Qui hoạch động

Bài 21. Dãy con tăng dần dài nhất

Cho một dãy số nguyên gồm N phần tử A[1], A[2], … A[N]. Biết rằng dãy con tăng đơn điệu là 1 dãy $A[i_1]$,… $A[i_k]$ thỏa mãn:

$i_1 < i_2 < \ldots < i_k$ và $A[i_1] < A[i_2] < .. < A[i_k]$.

Hãy cho biết dãy con tăng đơn điệu dài nhất của dãy này có bao nhiêu phần tử ?

Input

- Dòng 1 gồm 1 số nguyên là số N ($1 \le N \le 1000$).
- Dòng thứ 2 ghi N số nguyên A[1], A[2], .. A[N] ($1 \le A[i] \le 10000$).

Output

Ghi ra độ dài dãy con tăng dài nhất.

Ví dụ:

Input	**Output**
6 1 2 5 4 6 2	4

Giải thích : Dãy con tăng dài nhất là dãy A[1] = 1, A[2] = 2, A[4] = 4, A[5] = 6.

Hướng dẫn:

Gọi F[i] là độ dài dãy con tăng dài nhất bắt đầu từ i. Khởi tạo phần tử lớn nhất A[n+1] = +∞ và phần tử bé nhất A[0] = -∞. Khởi tạo F[n+1] = 1. Tại mỗi bước ta tìm jMax là chỉ số của phần tử lớn hơn i và có độ dài dãy con tăng lớn nhất trong số các chỉ số j (i+1 ≤ j ≤ n+1). Ban đầu jMax = n+1 với ý nghĩa giả sử dãy con tăng chỉ bao gồm 2 phần tử là A[i] và A[n+1]. F[0] có tác dụng duyệt lại tất cả các dãy con tăng xem dãy nào là dài nhất.

```
Program LIQ;
Const
maxN =1000; minValue =0; maxValue =10001;
Var
n,maxV :SmallInt;
A,F :Array[0..maxN+1] of SmallInt;

procedure Enter;
var
i :SmallInt;
begin
ReadLn(n);
for i:=1 to n do Read(A[i]);
A[0]:=minValue; A[n+1]:=maxValue;
end;

procedure Optimize;
var
i,j,jMax :SmallInt;
begin
F[n+1]:=1;
for i:=n downto 0 do
begin
jMax:=n+1;
for j:=n downto i+1 do
if (A[i]<A[j]) and (F[jMax]<F[j]) then jMax:=j;
F[i]:=F[jMax]+1;
end;
end;

Begin
Assign(Input,''); Reset(Input);
Assign(Output,''); Rewrite(Output);
```

```
    Enter;
    Optimize;
    Write(F[0]-2);
    Close(Input); Close(Output);
    End.
```

Bài 22: DÃY HÌNH CHỮ NHẬT LỒNG NHAU

Trên mặt phẳng tọa độ cho N hình chữ nhật với các cạnh song song với hệ trục tọa độ, các hình chữ nhật được đánh số từ 1 tới N. Hình chữ nhật thứ *i* được cho bởi 4 số nguyên dương x_{i1}, y_{i1}, x_{i2}, y_{i2}, trong đó (x_{i1}, y_{i1}) là tọa độ đỉnh trái dưới, còn (x_{i2}, y_{i2}) là tọa độ đỉnh phải trên. Ta nói rằng hình chữ nhật thứ i nằm trong hình chữ nhật thứ j nếu trên mặt phẳng tọa độ, mọi điểm của hình chữ nhật i đều thuộc hình chữ nhật j.

Yêu cầu: Với N hình chữ nhật cho trước, hãy tìm K hình chữ nhật với chỉ số i_1, i_2, …, i_K sao cho hình i_1 nằm trong hình i_2, hình i_2 nằm trong hình i_3, …, hình i_{K-1} nằm trong hình i_K và K là lớn nhất.

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

- Dòng đầu tiên chứa số nguyên dương N ($1 \le N \le 100$).
- N dòng tiếp theo, dòng thứ i chứa 4 số nguyên dương x_{i1}, y_{i1}, x_{i2}, y_{i2} có giá trị không vượt quá 200.

Kết quả: Ghi ra file văn bản **HCN.OUT** số K tìm được.

Ví dụ:

HCN.INP
3 1 1 7 4 3 1 6 6 2 2 5 4

HCN.OUT
2

Hướng dẫn:

Bước 1, ta sắp xếp các hình chữ nhật theo thứ tự giảm dần của diện tích. Điều này đảm bảo một hình chữ nhật chỉ có thể bị bao bởi một hình chữ nhật xuất hiện phía trước nó. Đoạn chương trình sau thực hiện việc sắp xếp:

```
for i:=1 to n-1 do
    for j:=i+1 to n do
        if (dientich(h[i])<dientich(h[j])) then
        begin
            tam:=h[i];h[i]:=h[j];h[j]:=tam;
        end;
```

Giả sử các hình chữ nhật đã được sắp xếp là h_1, h_2, ..., h_n. Ta gọi f[i] là độ dài lớn nhất của chuỗi hình chữ nhật lồng nhau kết thúc tại hình h_i. Để tính f[i], ta đi tìm các vị trí j sao cho j < i và hình chữ nhật h_j bao hình chữ nhật h_i, với mỗi j, khi đó f[i]=max(f[i],f[j]+1).

```
uses math;
const
   finp='hcn.inp';
   fout='hcn.out';
   MAXN=100;
type hcn=record x1,y1,x2,y2: longint; end;
var
  h: array[1..MAXN] of hcn;
  f: array[1..MAXN] of longint;
  i,j,n,kq: longint;
  tam:hcn;
function dientich(a:hcn):longint;
begin
   with a do dientich:=(x2-x1)*(y2-y1);
end;

function bao(a,b:hcn):boolean;
begin
   bao:=(a.x1<=b.x1) and (b.x2<=a.x2) and
     (a.y1<=b.y1) and (b.y2<=a.y2);
end;

begin
   assign(input,finp);
   reset(input);
   assign(output,fout);
   rewrite(output);

   readln(n);
   for i:=1 to n do
      with h[i] do
       readln(x1,y1,x2,y2);
   for i:=1 to n-1 do
      for j:=i+1 to n do
         if (dientich(h[i])<dientich(h[j])) then
begin
             tam:=h[i];h[i]:=h[j];h[j]:=tam;
            end;
   kq:=0;
   for i:=1 to n do
   begin
      f[i]:=1;
      for j:=i-1 downto 1 do
        if bao(h[j],h[i]) then
         f[i]:=max(f[i],f[j]+1);
       if (f[i]>kq) then kq:=f[i];
   end;
   writeln(kq);
   close(input);    close(output);
end.
```

Bài 23: XẾP HÀNG (Đề chọn đội quốc gia năm 2015)

Trong buổi lễ tổng kết năm học, có N học sinh được xếp thành một hàng. Cách xếp hàng của các học sinh chưa được đẹp mắt nên ban tổ chức xếp lại hàng theo chiều cao không giảm. Cách xếp hàng của ban tổ chức như sau: di chuyển một học sinh bất kỳ về đầu hàng hoặc cuối hàng cho đến khi được một hàng có chiều cao không giảm.

- **Yêu cầu:** Bạn hãy giúp ban tổ chức xếp hàng sao cho số lần di chuyển học sinh là ít nhất.

- **Dữ liệu vào:** Cho từ tệp văn bản XEPHANG.INP có cấu trúc:
  - Dòng thứ nhất ghi số nguyên dương N là số học sinh ($1 < N \le 10^5$).
  - Dòng thứ hai ghi N số nguyên dương H_1, H_2, ...H_n là chiều cao của N học sinh đang xếp hàng ($1 \le H_i \le 10^9$; i=1..N).

  Các số trên một dòng được ghi cách nhau bởi kí tự trắng

- **Kết quả:** Ghi vào tệp văn bản XEPHANG.OUT có một số nguyên dương duy nhất là số lần di chuyển học sinh ít nhất.

Ví dụ :

XEPHANG.INP	XEPHANG.OUT	Giải thích
4 2 1 3 5	1	*Di chuyển học sinh chiều cao 1 về đầu hàng: 1 2 3 5*
3 3 2 1	2	*Di chuyển học sinh chiều cao 1 về đầu hàng: 1 3 2* *Di chuyển học sinh chiều cao 3 về cuối hàng: 1 2 3*
5 3 7 2 6 9	3	*3 2 6 9 7* *2 3 6 9 7* *2 3 6 7 9*

Hạn chế kĩ thuật:

+ Có 70% số test tương ứng 70% số điểm có $N \le 10^3$

+ Có 30% số test tương ứng 30% số điểm có $10^3 < N \le 10^5$

Hướng dẫn: Thuật toán qui hoạch động theo chỉ số

```
const fi='xephang.inp';
    fo='xephang.out';
var f1,f2:text;
    n,max:longint;
    a,vt,f:array[1..1000000]of longint;
procedure nhap;
    var i:longint;
    begin
        assign(f1,fi);
        reset(f1);
        readln(f1,n);
        for i:=1 to n do
            begin
                read(f1,a[i]);
                vt[i]:=i;
            end;
    end;

procedure swap(var x,y:longint);
    var tg:longint;
    begin
```

```
        tg:=x;
        x:=y;
        y:=tg;
    end;


procedure qsort(l,r:longint);
    var i,j,key:longint;
    begin
        i:=l;
        j:=r;
        key:=a[(i+j) div 2];
        repeat
            while key>a[i] do inc(i);
            while key<a[j] do dec(j);
            if i<=j then
                begin
                    if a[i]>a[j] then
                        begin
                            swap(a[i],a[j]);
                            swap(vt[i],vt[j]);
                        end;
                    inc(i);
                    dec(j);
                end;
        until i>j;
        if i<r then qsort(i,r);
        if l<j then qsort(l,j);
    end;
procedure xuli;
    var i,j:longint;
    begin
        for i:=1 to n do f[i]:=1;
        for i:=1 to n-1 do
            if vt[i+1]>=vt[i] then
                f[i+1]:=f[i]+1;
        max:=f[1];
        for i:=2 to n do
            if max<f[i] then max:=f[i];
    end;

procedure xuat;
    begin
        assign(f2,fo);
        rewrite(f2);
        write(f2,n-max);
        close(f2);
    end;
begin
    nhap;
    qsort(1,n);
    xuli;
    xuat;
end.
```

Bài 24. Xếp hàng mua vé

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 một vé, song người bán vé được phép bán cho mỗi người tối đa hai vé. Vì thế, một số người có thể rời hàng và nhờ người đứng trước mình mua hộ vé. Biết t_i 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 hộ vé thì thời gian để người thứ i mua được vé cho cả hai người là r_i.

Yêu cầu: Xác 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.

Input

- Dòng đầu tiên chứa số N ($1 \le N \le 60000$).
- Dòng thứ 2 ghi N số nguyên dương $t_1, t_2, ..., t_N$. ($1 \le t_i \le 30000$)
- Dòng thứ ba ghi N-1 số nguyên dương $r_1, r_2, ..., r_{N-1}$. ($1 \le r_i \le 30000$)

Output: **In ra tổng thời gian phục vụ nhỏ nhất.**

Ví dụ:

Dữ liệu: 5 2 5 7 8 4 4 9 10 10 Kết qủa 18	Dữ liệu: 4 5 7 8 4 50 50 50 Kết qủa 24

Hướng dẫn:

Gọi F[i] là thời gian ít nhất để mua vé cho i người đầu tiên. Ta có công thức truy hồi như sau : F[i] = Min(F[i - 2] + R[i - 1], F[i - 1] + T[i]).

```
Program NKTICK;
Uses Math;
Var
n :LongInt;
F,T,R :Array[1..60000] of LongInt;

procedure Enter;
var
i :LongInt;
begin
ReadLn(n);
for i:=1 to n do Read(T[i]);
for i:=1 to n-1 do Read(R[i]);
Fillchar(F,SizeOf(F),0);
end;

procedure Optimize;
var
i :LongInt;
```

```
begin
F[1]:=T[1];
for i:=2 to n do F[i]:=Min(F[i-2]+R[i-1],F[i-1]+T[i]);
end;

Begin
Assign(Input,''); Reset(Input);
Assign(Output,''); Rewrite(Output);
Enter;
Optimize;
Write(F[n]);
Close(Input); Close(Output);
End.
```

---------HẾT---------