TÌM KIẾM NHỊ PHÂN
MÃ: TI16

A. MỞ ĐẦU

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 phức tạp để giúp học sinh có thể tiếp thu dễ dàng hơn khi gặp phải bài toán tìm kiếm nhị phân.

Để so sánh giữa Pascal và C++, trong chuyên đề của mình tôi không sử dụng thuật toán tìm kiếm nhị phân trong hệ thống của C++.

B. NỘI DUNG

I. Thuật toán tìm kiếm nhị phân cơ bản

Bài toán: Cho dãy A gồm N phần tử nguyên từ A_1, A_2,…, 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;
```

```
int Tknpcb(int X)
{
        int left = 1, right = N, mid;
        while(left <=right)
          {
           mid=(left+right)/2;
           if (X==A[mid]) return mid;
            if(X < a[mid]) right = mid - 1;
           else left= mid + 1;
          }
          return 0;
}
```

- 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)).

II. Thuật toán tìm kiếm một phần tử có giá trị gần bằng X

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:

1. 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).

Thuật toán:

```
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;
```

```
int Tknp1(int X)
{
        int left = 1, right = N, kq = 0, mid;
        while(left <= right)
          {
           mid=(left + right)/2;
           if (A[mid] <= X)
             {
               Kq = mid;
               left = mid +1;
             }
              Else right=mid-1;
          }
          return kq;
}
```

2. 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.

Thuật toán:

```
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;
```

```
int Tknp2(int X)
{
        int left = 1, right = N, kq = 0, mid;
        while(left <= right)
          {
           mid=(left + right)/2;
           if(A[mid] >= X)
             {
               Kq = mid;
               right = mid - 1;
             }
              Else left=mid + 1;
          }
          return kq;
}
```

3. Bài tập áp dụng

Bài toán 1: Trò chơi với dãy số (Đề thi học sinh giỏi quốc gia năm 2007 – 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.

Giải thuật:

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.

Bài toán 2: Bước nhảy xa nhất (Đề thi lớp 11 Trại hè Hùng Vương 2014)

Cho dãy A gồm N số nguyên không âm $A_1, A_2,\ldots, A_N$. Một bước nhảy từ phần tử A_i đến phần tử A_j được gọi là bước nhảy xa nhất của dãy nếu thỏa mãn các điều kiện sau:

+ $1 \le i < j \le N$.

+ $A_j - A_i \ge P$.

+ $j - i$ lớn nhất (Khi đó $j - i$ được gọi là độ dài bước nhảy xa nhất của dãy).

Yêu cầu: Tìm độ dài bước nhảy xa nhất của dãy A.

Dữ liệu vào: Từ tệp JUMP.INP có cấu trúc như sau:

- Dòng 1: Gồm hai số nguyên N và P ($1 \le N \le 10^5$; $0 \le P \le 10^9$).
- Dòng 2: Gồm N số nguyên $A_1, A_2,\ldots, A_N$ ($0 \le A_i \le 10^9$ với $1 \le i \le N$).

Kết quả: Ghi vào tệp JUMP.OUT gồm một số nguyên dương duy nhất là độ dài của bước nhảy xa nhất của dãy (Nếu không có bước nhảy nào thỏa mãn thì ghi kết quả bằng 0).

Ví dụ:

JUMP.INP	JUMP.OUT
6 3 4 3 7 2 6 4	3

Giải thuật:

- Gọi T[i] là phần tử nhỏ nhất từ a_1 đến a_i. Vậy T là dãy giảm dần.
- Duyệt tất cả các chỉ số j từ 1 đến N. Với mỗi j ta tìm kiếm nhị phân trên trên đoạn chỉ số [1, j-1] của dãy T phần tử lớn nhất thỏa mãn ≤ a[j]-p.
- Lưu ý : Ở đây dãy T là dãy giảm dần nên thuật toán có một chút thay đổi so với thuật toán đã giới thiệu ở trên.

III. Sắp xếp và tìm kiếm nhị phân các đối tượng có nhiều thông tin (đối với các bản ghi)

Những bài tập đòi hỏi phải tìm kiếm các bản ghi thường rất phức tạp, thường nhầm lẫn trong quá trình cài đặt chương trình.

1. Xây dựng hàm so sánh hai bản ghi với nhau

Để đơn giản ta phải sử dụng một hàm để định nghĩa việc so sánh hai bản ghi với nhau.

Giả sử ta có hai bản ghi X và Y gồm hai trường p, q trong đó trường p ưu tiên trước được viết như sau:

```
Function      sosanh(X,Y: Banghi):longint;
Begin
        If  X.p <Y.p then exit(-1);
        If  (X.p =Y.p) and (X.q  < Y.q) then exit(-1);
        If   (X.p =Y.p) and (X.q  = Y.q) then exit(0);
        Exit(1);
End;
```

Hàm trên cho kết quả -1 nếu bản ghi X nhỏ hơn bản ghi Y; cho giá trị 0 nếu X bằng Y và cho giá trị 1 nếu X lớn hơn Y.

Với những bản ghi có nhiều trường để so sánh thì ta cũng xây dựng như trên dựa vào thứ tự ưu tiên các trường khi so sánh chúng với nhau.

2. Sắp xếp nhanh (Quicksort) đối với dãy các bản ghi

a. Thuật toán Quicksort với dãy A là dãy số nguyên (hoặc thực)

Đây là một thuật toán cơ bản, có độ phức tạp là O(nlg(n)), bất cứ người nào học tin học cũng đã sử dụng thành thạo thuật toán này.

Giả sử có dãy A nguyên gồm N phần tử thì thuật toán sắp xếp được viết như sau :

```
Procedure     Qs(l, h:longint);
Var     i,j:longint;
        Tg: Longint;
        giua: Longint;
Begin
        i:=l;
        j:=h;
        giua:= A[(l+h) div 2];
        repeat
                while A[i] < giua do inc(i);
                while A[j] > giua do
dec(j);
                if i<=j then
                Begin
                        tg:=A[i];
                        A[i]:=A[j];
                        A[j]:=tg;
                        inc(i);
                        dec(j);
                end;
        until i>j;
        if i < h then qs(i, h);
        if j > l then qs(l, j);
end;
```

```
void Qs(int Left, int Right)
{
        int i = Left, j = Right;
        int pivot = A[(Left + Right) / 2];
        while (i <= j)
        {
                while (A[i] < pivot)    i++;
                while (A[j] > pivot)    j--;
                if (i <= j)
                {
                        int tg=A[i];
                        A[i]=A[j];
                        A[j]:=tg;
                        i++;
                        j--;
                }
        }
        if (Left < j)  QuickSort(Left, j);
        if (i < Right)  QuickSort(i, Right);
}
```

b. Thuật toán Quicksort với dãy A là dãy các bản ghi

Giả sử dãy A gồm N phần tử, các phần tử là một bản ghi, trước hết ta phải xây dựng hàm so sánh các bản ghi như đã giới thiệu ở trên.

Thuật toán Quicksort cũng được viết tương tự như trên tuy nhiên các phép so sánh giữa hai phần tử phải sử dụng hàm định nghĩa ở trên.

```
Procedure     Qs(l, h:longint);
Var     i,j:longint;
        Tg: Banghi;
        giua: Banghi;
Begin
        i:=l;
        j:=h;
        giua:= A[(l+h) div 2];
        repeat
                while    Sosanh(A[i],giua)=-1    do
inc(i);
                while    Sosanh(A[j],giua)    =1    do
dec(j);
                if  i<= j then
                Begin
                        tg:=A[i];
                        A[i]:=A[j];
                        A[j]:=tg;
                        inc(i);
                        dec(j);
                end;
        until i>j;
        if i < h then qs(i, h);
        if j > l then qs(l, j);
end;
```

```
void Qs(int Left, int Right)
{
         int i = Left, j = Right;
         Banghi pivot = A[(Left + Right) /
2];
        while (i <= j)
        {
            while     Sosanh((A[i],pivot)=-1)
i++;
            while Sosanh((A[j],pivot)=1)  j-
-;
                if (i <= j)
                {
                        Banghi tg=A[i];
                        A[i]=A[j];
                        A[j]:=tg;
                        i++;
                        j--;
                }
        }
        if (Left < j)  QuickSort(Left, j);
        if (i < Right)  QuickSort(i, Right);
}
```

3. Tìm kiếm nhị phân dựa trên dãy các bản ghi

Các thuật toán tìm kiếm nhị phân cũng được trình bày tương tự như các thuật toán tìm kiếm nhị phân trên dãy số chỉ lưu ý ở những thao tác phần tử giữa với phần tử cần tìm ta phải sử dụng hàm so sánh như đã giới thiệu ở trên.

4. Một số bài tập áp dụng

Bài toán 1: Con đường Tùng – Trúc (Đề thi học sinh giỏi quốc gia năm học 2013 – 2014)

Địa điểm du lịch Dailai nổi tiếng với con đường Tùng-Trúc. Đó là một con đường dài và thẳng, dọc bên đường người ta trồng rất nhiều cây tùng và cây trúc. Với mục đích tạo điểm nhấn cho con đường, Ban quản lý khu du lịch muốn chọn một đoạn đường mà dọc theo nó có ít nhất a cây tùng và có ít nhất b cây trúc để trang trí. Sau khi khảo sát, Ban quản lý ghi nhận được vị trí của từng cây tùng và cây trúc. Trên con đường có tất cả n cây, không có hai cây nào ở cùng một vị trí. Cây thứ i ở vị trí có khoảng cách đến vị trí bắt đầu của con đường là d_i (i = 1, 2, ..., n). Với kinh phí có hạn, Ban quản lý muốn chọn một đoạn đường thỏa mãn điều kiện đã nêu với độ dài là ngắn nhất.

Yêu cầu

Cho a, b và vị trí của n cây. Hãy tìm đoạn đường có độ dài ngắn nhất mà dọc theo đoạn đường đó có ít nhất a cây tùng và b cây trúc.

Input

- Dòng đầu chứa 3 số nguyên dương n, a, b (a + b <= n)
- Dòng thứ i trong n dòng tiếp theo mỗi dòng chứa hai số nguyên dương d_i (d_i <= 10^9) trong đó d_i là khoảng cách của cây tính từ vị trí bắt đầu của con đường, k_i = 1 nếu cây thứ i là cây tùng, k_i = 2 nếu là cây trúc.
- Các số trên cùng một dòng được ghi cách nhau ít nhất một dấu cách.

Output

Ghi ra một số nguyên là độ dài đoạn đường ngắn nhất tìm được, quy ước ghi số -1 nếu không tồn tại đoạn đường nào thỏa mãn điều kiện đặt ra.

Giới hạn

+ d_i <= 10^9.
+ 30% số test có n <= 300.
+ 30% số test khác có n <= 3000.
+ 40 % số test còn lại có n <= 300000.

Ví dụ:

Input	Output
7 2 2 20 2 30 1 25 1 35 1 60 2 65 2 10 1	35

Giải thuật:

- Trước hết ta tạo ra một mảng cộng dồn số tùng và số trúc.
- Duyệt i chạy từ đầu đến cuối. Với mỗi i ta tìm kiếm nhị phân j lớn nhất sao cho số tùng và số trúc thỏa mãn điều kiện đầu bài.

Chương trình tham khảo:

```
uses math;
        fi='minroad.inp';
        fo='minroad.out';
        maxn=300000;
        oo=trunc(1e9)+1;
Type banghi=record
        td,loai:longint;
        end;
Var     l:array[0..maxn] of banghi;
        st,sb:array[0..maxn] of longint;
        n,a,b,kq:longint;
```

```
Procedure docdl;
Var
        i:longint;
Begin
        Readln(n,a,b);
        For i:=1 to n do
                readln(L[i].td,l[i].loai);

end;
Procedure     qs(l1,h:longint);
Var     i,j:longint;
        tg:banghi;
        giua:longint;
Begin
        i:=l1;
        j:=h;
        giua:= l[(l1+h) div 2].td;
        repeat
                while l[i].td<giua do inc(i);
                while l[j].td>giua do dec(j);
                if i<=j then
                Begin
                        tg:=l[i];
                        l[i]:=l[j];
                        l[j]:=tg;
                        inc(i);
                        dec(j);
                end;
        until i>j;
        if i<h then qs(i,h);
        if j>l1 then qs(l1,j);
end;
Procedure     congdon;
Var     i:longint;
Begin
        st[0]:=0;
        sb[0]:=0;
        For i:=1 to n do
        if l[i].loai=1 then
    Begin
                st[i]:=st[i-1]+1;
                sb[i]:=sb[i-1];
```

```
    end
    else
     Begin
            st[i]:=st[i-1];
            sb[i]:=sb[i-1]+1;
     end;
end;
Function    TKNP(h:longint):longint;
Var         dau,cuoi,giua,kq1:longint;
Begin
      if (st[h]-st[0]<a) or (sb[h]-sb[0]<b) then exit(oo);
      Dau:=1;     kq1:=oo;
      cuoi:=h;
      while dau<=cuoi do
            Begin
                  giua:=(dau+cuoi) div 2;
                  if (st[h]-st[giua-1]>=a) and (sb[h]-sb[giua-1]>=b) then
                        Begin
                              kq1:=l[h].td-l[giua].td;
                              dau:=giua+1;
                        end
                  else cuoi:=giua-1;
            end;
      exit(kq1);
end;
Procedure   xuli;
Var   i:longint;
Begin
      kq:=oo;
      For i:=2 to n do
            kq:=min(kq,tknp(i));
            if kq=oo then kq:=-1;
end;
Procedure ghikq;
Begin
      Write(kq);
end;
BEGIN
      Docdl;
      qs(1,n);
      congdon;
      xuli;
```

```
    ghikq;
END.
```

Bài toán 2: Cắt hình (Đề thi học sinh giỏi quốc gia năm học 2014 – 2015)

Cho A là lưới ô vuông gồm m dòng và n cột. Các dòng của lưới được đánh số từ 1 đến m, từ trên xuống dưới. Các cột của lưới được đánh số từ 1 đến n, từ trái sang phải. Ô nằm trên giao của dòng i và cột j của lưới, được gọi là ô (i, j), chứa số nguyên không âm $a_{i,j}$ có giá trị không vượt quá 10^6.

Các lưới ô vuông như vậy luôn là đối tượng cho nhiều nghiên cứu thú vị. Vừa qua, trong giờ học ôn luyện cho kỳ thi học sinh giỏi Tin học, Hùng được cô giáo giao cho giải quyết bài toán trả lời truy vấn sau đây đối với bảng đã cho:

Cho một hình chữ nhật con có ô trái trên là ô (x,y) và ô phải dưới là ô (u,v), cần đưa ra chênh lệch nhỏ nhất trong số các chênh lệch giữa hai tổng các số trong hai hình chữ nhật thu được bằng việc cắt ngang hoặc cắt dọc hình chữ nhật đã cho dọc theo đường kẻ của lưới. Giả thiết (x,y) và (u,v) là hai ô khác nhau trên lưới.

Bạn hãy giúp Hùng giải quyết bài toán đặt ra.

Yêu cầu: Cho lưới A và k bộ x_q, y_q, u_q, v_q ($q = 1, 2, ..., k$) tương ứng với k truy vấn, hãy đưa ra các câu trả lời cho k truy vấn.

Dữ liệu vào:

- Dòng đầu tiên chứa ba số nguyên m, n, k ($k \le m \times n$);
- m dòng tiếp theo, dòng thứ i chứa n số nguyên không âm $a_{i1}, a_{i2}, ..., a_{in}$;
- k dòng tiếp theo chứa 4 số nguyên x_q, y_q, u_q, v_q ($q = 1, 2, ...,k$).

Dữ liệu ra:

Ghi ra file văn bản MINCUT.OUT gồm k dòng, mỗi dòng chứa một số là câu trả lời cho một truy vấn theo thứ tự xuất hiện trong file dữ liệu vào.

Ràng buộc:

- Có 30% số test ứng với 30% số điểm của bài có $m, n \le 10$.
- Có 30% số test khác ứng với 30% số điểm của bài có $m, n \le 100$.
- Có 40% số test ứng với 40% số điểm còn lại của bài có $m, n \le 1000$.

Ví dụ:

Input	Output
3 3 2 1 1 1 1 1 1 1 1 1 1 1 3 3 1 1 3 2	**3** **0**

Giải thuật:

Với mỗi hình chữ nhật chúng ta cũng chặt nhị phân theo hàng và theo cột. Nếu nửa bên nào có tổng lớn hơn ta sẽ chặt nhị phân trên phần đó.

Chương trình tham khảo:

```
uses math;
const
    fi='MINROAD.INP';
    fo=' MINROAD.OUT';
```

```
        oo=trunc(1e12);
var     m,n,k,x,y,u,v:longint;
        kq,sum:int64;
        a:array[1..1000,1..1000] of longint;
        s:array[0..1000,0..1000] of int64;
procedure tknp1;
var     d,c,g:longint;
        t,t1:int64;
begin
        d:=x;
        c:=u-1;
        while d<=c do
        begin
                g:=(d+c) div 2;
                t:=s[g,v]-s[x-1,v]-s[g,y-1]+s[x-1,y-1];
                t1:=sum-t;
                kq:=min(kq,abs(t-t1));
                if t=t1 then break else
                if t>t1 then c:=g-1 else
                d:=g+1;
        end;
end;
procedure tknp2;
var     d,c,g:longint;
        t,t1:int64;
begin
        d:=y;
        c:=v-1;
        while d<=c do
        begin
                g:=(d+c) div 2;
                t:=s[u,g]-s[x-1,g]-s[u,y-1]+s[x-1,y-1];
                t1:=sum-t;
                kq:=min(kq,abs(t-t1));
                if t=t1 then break else
                if t>t1 then c:=g-1 else
                d:=g+1;
        end;
end;
procedure xuli;
begin
        kq:=oo;
```

```
        tknp1;
        tknp2;
end;
procedure prep;
var     i,j:longint;
begin
        fillchar(s,sizeof(s),0);
        for i:=1 to m do
                for j:=1 to n do s[i,j]:=s[i-1,j]+s[i,j-1]-s[i-1,j-1]+a[i,j];
end;
procedure docdl;
var     f,f1:text;
        i,j:longint;
begin
        assign(f,fi); reset(f);
        assign(f1,fo); rewrite(f1);
        read(f,m,n,k);
        for i:=1 to m do
                for j:=1 to n do read(f,a[i,j]);
        prep;
        for i:=1 to k do
        begin
                read(f,x,y,u,v);
                sum:=s[u,v]-s[x-1,v]-s[u,y-1]+s[x-1,y-1];
                xuli;
                writeln(f1,kq);
        end;
        close(f);
        close(f1);
end;
begin
        docdl;
end.
```

Bài toán 3: Dãy con tăng dài nhất.

Cho một dãy gồm N số nguyên ($1 \leq N \leq 30000$). Hãy tìm dãy con tăng dài nhất trong dãy đó. In ra số lượng phần tử của dãy con. Các số trong phạm vi longint.

Input: File Lis.Inp

- Dòng đầu tiên gồm số nguyên N.
- Dòng thứ hai gồm N số mô tả dãy.

Output: file Lis.Out Gồm một số nguyên duy nhất là đáp số của bài toán

Ví dụ:

Lis.Inp	Lis.Out
5 2 1 4 3 5	**3**

Thuật toán:

Gọi k là độ dài cực đại của dãy con tăng và ký hiệu H[1..k] là dãy có ý nghĩa sau: H[i] là số hạng nhỏ nhất trong các số hạng cuối cùng của các dãy con tăng có độ dài i. Đương nhiên H[1] ≤ H[2] ≤ .. ≤ H[k].

Khi xét thêm một giá trị mới trong dãy A (giả sử A[i]) ta tìm kiếm nhị phân trong H phần tử lớn nhất nhưng nhỏ hơn hoặc bằng A[i], ngoài ra cần chú ý các giá trị trong dãy H và giá trị k cũng tương ứng thay đổi.

```
Res:=1; H[1]:=1;
For i:=2 to n do
Begin
    If A[i] < a[h[1]] then h[1]:=i
    else if a[i] > a[h[res]] then
    Begin
          Inc(Res); H[res]:=i;
    End
    else
    Begin
         d:=1; c:=Res;
        While d<>c do
        begin
             mid:=(d+c+1) div 2;
             If A[i] > a[h[mid]] then d:=mid else c:=mid-1;
        End;
        Mid:=(d+c) div 2;
        If (A[h[mid]] < a[i]) and (a[i]<a[h[mid+1]]) then
        h[mid+1]:=i;
     End;
     End;
     Writeln(Res);
```

IV. Bài tập tự giải:

STT	ĐỊA CHỈ BÀI TẬP
1	http://vn.spoj.com/problems/VOPIG/
2	http://www.spoj.com/PTIT/problems/PTIT126J/
3	http://vn.spoj.com/problems/DTKSUB/
4	http://vn.spoj.com/problems/MOVE12/
5	http://vn.spoj.com/problems/LEM1/
6	http://vn.spoj.com/problems/CRUELL2/
7	http://vn.spoj.com/problems/NDCCARD/
8	http://vn.spoj.com/problems/C11POST/
9	http://vn.spoj.com/problems/DGOLD/
10	http://vn.spoj.com/problems/VOGCDSUM/
11	http://www.spoj.com/PTIT/problems/P144SUMD/
12	http://vn.spoj.com/problems/CARDSHUF/
13	http://www.spoj.com/PTIT/problems/BCFRIEND/
14	http://vn.spoj.com/problems/C11SEQ/
15	http://www.spoj.com/PTIT/problems/P145SUMG/
16	http://vn.spoj.com/problems/VMCANDLE/
17	http://vn.spoj.com/problems/SPSEQ/
18	http://vn.spoj.com/problems/ASSIGN1/
19	http://vn.spoj.com/problems/ORDERSET/
20	http://vn.spoj.com/problems/VOSTR/

C. LỜI KẾT

Trên đây là những kinh nghiệm mà tôi đã nghiên cứu và vận dụng trong quá trình giảng dạy thực tế. Thông qua một số bài tập trên, học sinh được rèn luyện cách xây dựng các biến thể của thuật toán tìm kiếm nhị phân. Từ đó các em có những kinh nghiệm vận dụng linh hoạt một thuật toán rất cơ bản vào từng bài toán cụ thể. Trong quá trình cài đặt các chương trình về sắp xếp và tìm kiếm nhị phân thường ít bị sai sót. Để chuyên đề của tôi được hoàn thiện hơn, việc sử dụng đạt hiệu quả cao hơn, tôi rất mong nhận được những ý kiến đóng góp quí báu của quí thầy cô và các đồng nghiệp.

D. TÀI LIỆU THAM KHẢO.

1. TÀI LIỆU GIÁO KHOA CHUYÊN TIN QUYỂN 1-NHÀ XUẤT BẢN GIÁO DỤC
2. BÀI GIẢNG CỦA LÊ MINH HOÀNG - ĐHSP HÀ NỘI
3. TẠP CHÍ TIN HỌC - NHÀ TRƯỜNG
4. MỘT SỐ ĐỀ THI CHỌN HSG THPT, ĐỀ THI HSG QUỐC GIA…
5.BÀI TẬP TRÊN **https://vn.spoj.pl/problem**