[**树状数组实现查找K小的元素**](http://www.cnblogs.com/zgmf_x20a/archive/2008/11/15/1334109.html)

回顾树状数组的定义，注意到有如下两条性质：  
一，c[ans]=sum of A[ans-lowbit(ans)+1 ... ans];  
二，当ans=2^k时，  
 c[ans]=sum of A[1 ... ans];

下面说明findK(k)如何运作：  
1，设置边界条件ans,ans'<maxn且cnt<=k；  
2，初始化cnt=c[ans]，其中ans=2^k且k为满足边界条件的最大整数；  
3，找到满足边界条件的最大的ans'使得ans'-lowbit(ans')=ans，即ans'满足c[ans']=A[ans+1 .. ans']（根据性质一），只要将c[ans']累加到cnt中（此时cnt=sum of A[1 ... ans']，根据性质二），cnt便可以作为k的逼近值；  
4，重复第3步直到cnt已无法再逼近k，此时ans刚好比解小1，返回ans+1。

因此findk(k)的实质就是二分逼近

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
树状数组实现查找K小的元素  
                  经典。  
限制：数据范围在1<<20 以内  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
#include <iostream>  
using namespace std;  
  
#define maxn 1<<20  
int n,k;  
int c[maxn];  
  
int lowbit(int x){  
    return x&-x;  
}  
  
void insert(int x,int t){  
       while(x<maxn){  
          c[x]+=t;  
          x+=lowbit(x);      
       }  
}  
int find(int k){  
    int cnt=0,ans=0;  
    for(int i=20;i>=0;i--){  
        ans+=(1<<i);  
        if(ans>=maxn || cnt+c[ans]>=k)ans-=(1<<i);  
        else cnt+=c[ans];  
    }  
    return ans+1;  
}  
void input(){  
       memset(c,0,sizeof(c));  
       int t;  
       scanf("%d%d",&n,&k);  
       for(int i=0;i<n;i++){      
            scanf("%d",&t);  
            insert(t,1);  
       }  
       printf("%d\n",find(k));  
}  
int main(){  
    int cases;  
    scanf("%d",&cases);  
    while(cases--){  
        input();  
    }  
    return 0;  
}