

```

1  /*** SPOJ - MKTHNUM K-th Number: Given an array  $a[1 \dots n]$  of different integer numbers,
2   your program must answer a series of questions  $Q(i, j, k)$  in the form:
3   "What would be the  $k$ -th number in  $a[i \dots j]$  segment, if this segment was sorted?" ***/
4  -----
5  #define mx 100005
6  int a[mx+5],ar[mx+5]; vector<int> tree[4*mx+5];
7  void MergeNode(int nd,int lf,int rg){
8      int n1=tree[lf].size(); int n2=tree[rg].size();
9      int i=0,j=0;
10     while(i<n1 && j<n2){
11         if(tree[lf][i]<=tree[rg][j]){
12             tree[nd].push_back(tree[lf][i]); i++;
13         } else{
14             tree[nd].push_back(tree[rg][j]); j++;
15         }
16     }
17     while(i<n1){ tree[nd].push_back(tree[lf][i]); i++; }
18     while(j<n2){ tree[nd].push_back(tree[rg][j]); j++; }
19 }
20 void init(int nd,int b,int e){
21     if(b==e){ tree[nd].push_back(a[b]); return; }
22     int lf=2*nd, rg=2*nd+1, md=(b+e)/2;
23     init(lf,b,md);
24     init(rg,md+1,e);
25     MergeNode(nd,lf,rg);
26 //merge(tree[lf].begin(),tree[lf].end(),tree[rg].begin(),tree[rg].end(),back_inserter(tree[nd]));
27 }
28 int LowerBound(int nd,int v){
29     int lo=0; int hi=tree[nd].size()-1; int cnt=0;
30     while(lo<hi){
31         int md=(lo+hi)/2;
32         int u = tree[nd][md];
33         if(u<v){ cnt=md+1; lo=md+1; }
34         else{ hi=md-1; }
35     }
36     return cnt;
37 }
38 int query(int nd,int b,int e,int x,int y,int v){
39     if(b>y||e<x) return 0;
40     if(b>=x&&e<=y){
41         int cnt = LowerBound(nd,v); return cnt;
42         // return Lower_bound(tree[nd].begin(),tree[nd].end(),v)-tree[nd].begin();
43     }
44     int lf=2*nd, rg=2*nd+1, md=(b+e)/2;
45     return query(lf,b,md,x,y,v) + query(rg,md+1,e,x,y,v);
46 }
47 int solve(int n,int x,int y,int k){
48     int lo=1; int hi=n; int ans = -1;
49     while(lo<hi){
50         int md = (lo+hi)/2;
51         int v = ar[md];
52         int res = query(1,1,n,x,y,v);
53         if(res<k){ ans=v; lo=md+1; }
54         else{ hi=md-1; }
55     }
56     return ans;
57 }
58 int main(){
59     int n,q;
60     while(scanf("%d%d",&n,&q)==2){
61         for(int i=1; i<=n; i++){
62             scanf("%d",&a[i]); ar[i] = a[i];
63         }
64         init(1,1,n);
65         sort(ar+1,ar+n+1);
66         for(int qq=1; qq<=q; qq++){
67             int x,y,k; scanf("%d%d%d",&x,&y,&k);
68             int ans = solve(n,x,y,k);
69             printf("%d\n",ans);
70         }
71         for(int i=0; i<4*mx; i++)tree[i].clear();
72     }
73 }
```