HDU5306 Gorgeous Sequence (jls线段树)

3个操作： 1.区间最值修改。 2.区间求最值。 3.区间求和。

思路：开一个变量mx,se维护区间最大值和严格次大值，再开一个cnt，维护区间最大值的个数。

时间复杂度：$O((n+m)logn)$

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
const int N=1e6+5,M=2e4+5,inf=0x3f3f3f3f,mod=1e9+7;
#define mst(a,b) memset(a,b,sizeof a)
#define PII pair<int,int>
#define fi first
#define se second
#define pb push_back
template<class T>
inline void read(T &x){
    x=0;int w=1;
    char ch=getchar();
    while(ch<'0'||ch>'9') {if(ch=='-') w=-1;ch=getchar();}
    for(;ch>='0'&&ch<='9';ch=getchar())
        x=(x<<3)+(x<<1)+(ch&15);
    x*=w;
}
int val[N],T,n,m;
struct Segment_Tree{
#define lx x<<1
#define rx x<<1|1
    struct node{
        int l,r,mx,se,cnt,lz;
        ll s;
    }a[N<<2];
    inline void re(int x){
        a[x].s=a[lx].s+a[rx].s;
        a[x].mx=max(a[lx].mx,a[rx].mx);
        a[x].se=max(a[lx].se,a[rx].se);a[x].cnt=0;
```

```
    if(a[lx].mx!=a[rx].mx) a[x].se=max(a[x].se,min(a[lx].mx,a[rx].mx));
    if(a[x].mx==a[lx].mx) a[x].cnt+=a[lx].cnt;
    if(a[x].mx==a[rx].mx) a[x].cnt+=a[rx].cnt;
}
inline void pushtag(int x,int v){
    if(a[x].mx<=v) return;//此种情况没有影响
    a[x].s+=1LL*(v-a[x].mx)*a[x].cnt,a[x].mx=v;//此时更新区间和与区间最大值.
}
inline void pushdown(int x){
    pushtag(lx,a[x].mx),pushtag(rx,a[x].mx);
}
void build(int x,int l,int r){
    a[x].l=l,a[x].r=r,a[x].lz=-1;
    if(l==r){
        a[x].s=a[x].mx=val[l],a[x].cnt=1,a[x].se=-1;return;
    }
    int mid=(l+r)>>1;build(lx,l,mid),build(rx,mid+1,r);
    re(x);
}
void upd(int x,int l,int r,int k){
    if(a[x].r<l||a[x].l>r||a[x].mx<=k) return;
    if(a[x].l>=l&&a[x].r<=r&&k>a[x].se){
        pushtag(x,k);return;    //a[x].se<k<a[x].mx 此情况直接下方标记
    }
    pushdown(x);//无法直接该儿子,就递归搜索,回溯再合并信息
    int mid=(a[x].l+a[x].r)>>1;
    if(l<=mid) upd(lx,l,r,k);
    if(r>mid) upd(rx,l,r,k);
    re(x);
}
ll qsum(int x,int l,int r){
    if(a[x].l>=l&&a[x].r<=r) return a[x].s;
    pushdown(x);
    int mid=(a[x].l+a[x].r)>>1;
    ll ans=0;
    if(l<=mid) ans+=qsum(lx,l,r);
    if(r>mid) ans+=qsum(rx,l,r);
```

```
        return ans;
    }
    int qmx(int x,int l,int r){
        if(a[x].l>=l&&a[x].r<=r) return a[x].mx;
        pushdown(x);int mid=(a[x].l+a[x].r)>>1,ans=-1;
        if(l<=mid)   ans=max(ans,qmx(lx,l,r));
        if(r>mid) ans=max(ans,qmx(rx,l,r));
        return ans;
    }
}seg;
int main(){
    read(T);
    while(T--){
        read(n),read(m);
        for(int i=1;i<=n;i++) read(val[i]);
        seg.build(1,1,n);
        while(m--){
            int op,l,r,k;
            read(op),read(l),read(r);
            if(!op){
                read(k),seg.upd(1,l,r,k);
            }
            else if(op==1){
                printf("%d\n",seg.qmx(1,l,r));
            }
            else {
                printf("%lld\n",seg.qsum(1,l,r));
            }
        }
    }
    return 0;
}
```