可持久化线段树

一道例题

洛谷p3919

你需要维护一个长度为N的数组，支持对历史版本单点的修改和查询。

每次操作都把历史版本的线段树复制一遍？时间复杂度$O(nm)$显然会超时。

我们发现，其实大部分的节点都是可以重复利用的，只需要对修改节点时所途径的节点复制即可。这大概就是可持久化的思想。

被卡常的代码

```
#include <stdio.h>
#include <iostream>
#define mid ((l+r)>>1)
using namespace std;
    int rt[2000001],T[20000001],L[20000001],R[20000001];
int cnt;
int build(int l,int r)
{
    int root=++cnt;
    if (l==r)
    {
        scanf("%d",&T[root]);
        return root;
    }
    L[root]=build(l,mid);
    R[root]=build(mid+1,r);
    return root;
}
int update(int pre,int l,int r,int x,int c)
{
    int root=++cnt;
    if (l==r)
    {
        T[root]=c;
        return root;
    }
    L[root]=L[pre];
    R[root]=R[pre];
    if (x<=mid)
    L[root]=update(L[pre],l,mid,x,c);
    else
    R[root]=update(R[pre],mid+1,r,x,c);
    return root;
}
```

```
void query(int pre,int l,int r,int x)
{
    if (l==r)
    {
    printf("%d\n",T[pre]);
    return;
}
    if (x<=mid)
    query(L[pre],l,mid,x);
    else
    query(R[pre],mid+1,r,x);
}
int main()
{
    cnt=0;
    int n,m;
    scanf("%d%d",&n,&m);
    build(1,n);
    int v,cd,x,y;
    rt[0]=1;
    for (int i=1;i<=m;i++)
    {
        scanf("%d%d%d",&v,&cd,&x);
        if (cd==1)
        {
            scanf("%d",&y);
            rt[i]=update(rt[v],1,n,x,y);
        }
        else
        {
            rt[i]=rt[v];
            query(rt[v],1,n,x);

        }
    }
    return 0;
}
```

可持久化思想的延申

求区间第k大

模板题：洛谷p3834