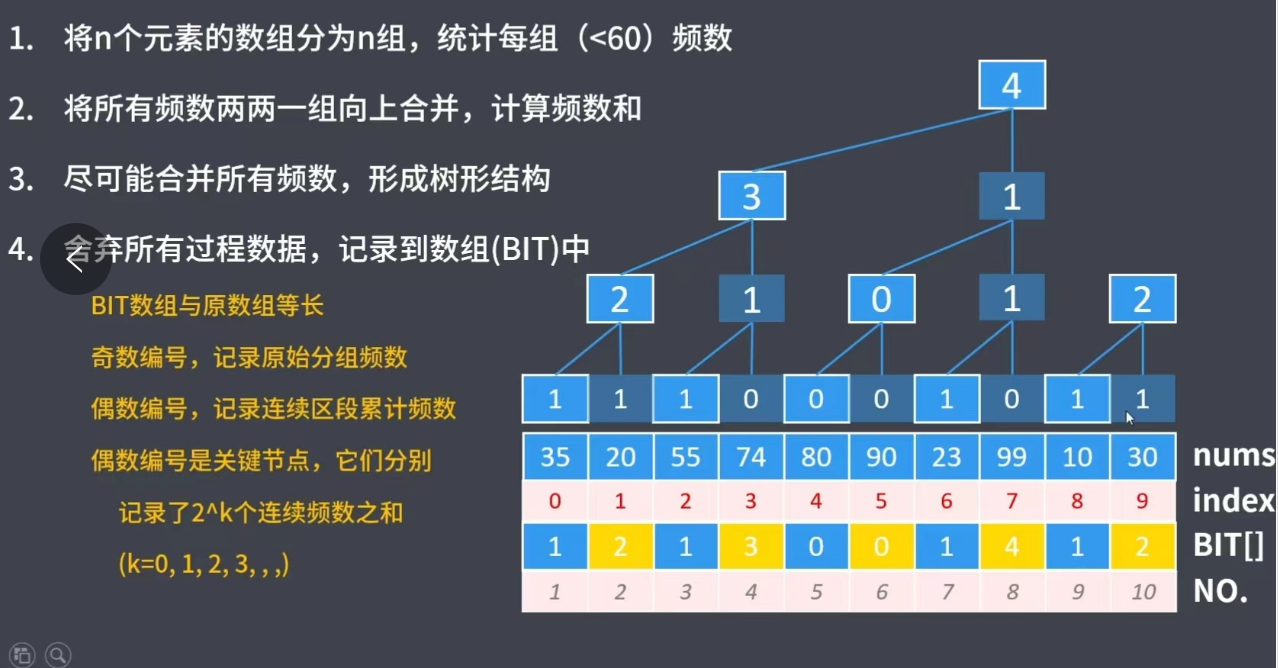
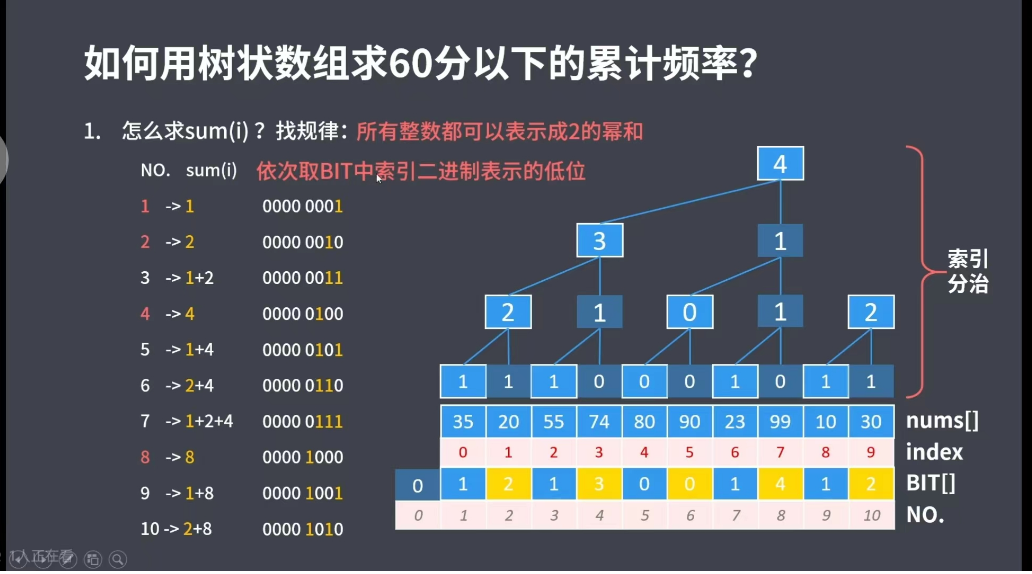
树状数组（适用）





关键：根据二进制的特性来构建树状存储的数组

根据二进制来划分区间

/构建树状树组的代码

struct BIT{

    vector<int> a;

    int n;

    BIT(int \_n):n(\_n),a(\_n+1){}

    static int lowbit(int x){//取最低位的1 所代表的值

        return x&(-x);

    }

    int query(int x) const{ 查询前缀和

        int ret =0;

        while(x){

            ret +=a[x];

            x-=lowbit(x);

        }

        return ret;

    }

    void update(int x,int dt){//更新对应区间的值

        while(x<=n){

            a[x]+=dt;

            x+=lowbit(x);

        }

    }

};

Sum[i] 由BIT[i] 加多个关键节点的和构成

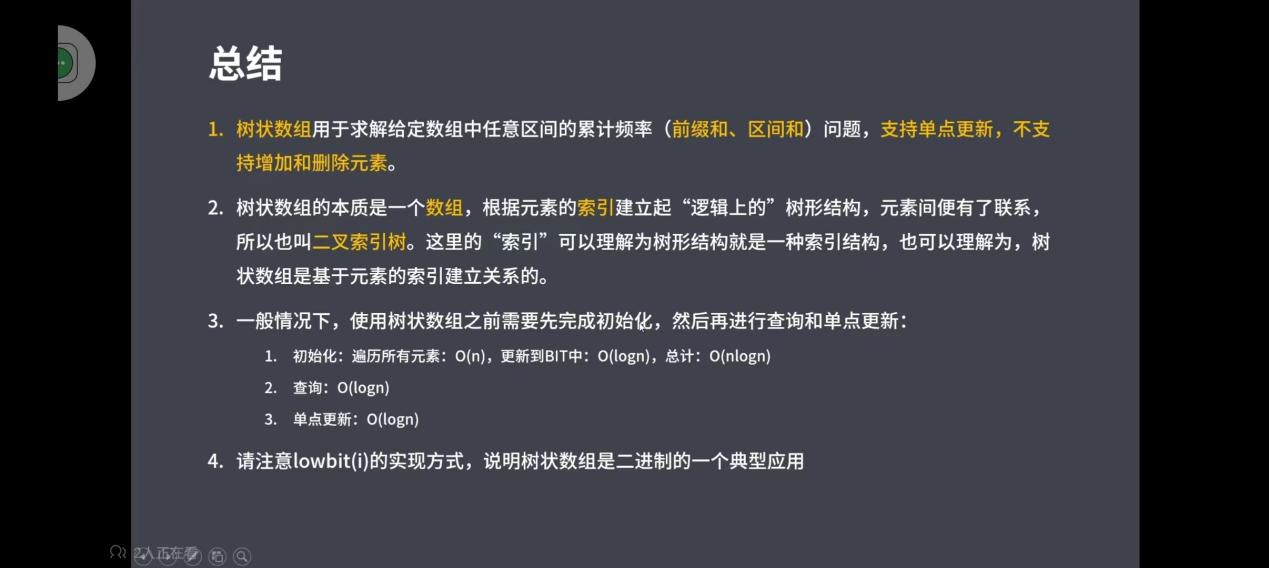
初始化树状数组需要用到update;

 for(int i=1;i<=m;++i){

         pos[i]=n+i;

         bit.update(n+i,1);//初始化树状数组

     }

****