我们用线段树维护区间的数字，总变化量，每次更新数字的时候只改变整个区间数字相同的区间。由于我们只更改整个区间数字相同的区间，那么我们每次对单个节点的修改值实际上是固定的，我们可以使用lazy tag来标定这个待传递的修改值。

**struct** Node

{

    ll sum,l,r,num,lazy;

}st[maxn\*4];

sum为区间总变化量

num为区间数字，如果为0代表区间存在多种数字，不为代表仅存在num这一种数字

lazy为待向下传递的单个节点的变化量

#include<bits/stdc++.h>

#define ll long long

#define IOS ios::sync\_with\_stdio(0); cin.tie(0); cout.tie(0);

**using** **namespace** std;

**const** ll maxn=1e5+7;

ll n,m;

**struct** Node

{

    ll sum,l,r,num,lazy;

}st[maxn\*4];

**void** build(ll l,ll r,ll loca)

{

    st[loca].l=l;

    st[loca].r=r;

    st[loca].sum=0;

    st[loca].lazy=0;

**if**(l==r)

    {

        st[loca].num=l;

    }

**else**

    {

        st[loca].num=0;

        ll mid=(l+r)>>1;

        build(l,mid,loca<<1);

        build(mid+1,r,loca<<1|1);

    }

}

**void** pushdown(ll loca)

{

**if**(st[loca].lazy)

    {

        st[loca<<1].sum+=(st[loca<<1].r-st[loca<<1].l+1)\*st[loca].lazy;

        st[loca<<1].lazy+=st[loca].lazy;

        st[loca<<1|1].sum+=(st[loca<<1|1].r-st[loca<<1|1].l+1)\*st[loca].lazy;

        st[loca<<1|1].lazy+=st[loca].lazy;

        st[loca<<1].num=st[loca<<1|1].num=st[loca].num;

        st[loca].lazy=0;

    }

}

**void** pushup(ll loca)

{

    st[loca].sum=st[loca<<1].sum+st[loca<<1|1].sum;

**if**(st[loca<<1].num==st[loca<<1|1].num)

    {

        st[loca].num=st[loca<<1].num;

    }

**else** st[loca].num=0;

}

//void color(ll x,ll add,ll l,ll r,ll index)

//{

//    seg[index].same=true;

//    seg[index].sum+=add\*(r-l+1);

//    seg[index].col=x;

//    seg[index].add+=add;

//}

**void** change(ll l,ll r,ll loca,ll ope)

{

**if**(st[loca].l>=l&&st[loca].r<=r&&st[loca].num)

    {

        st[loca].sum+=(st[loca].r-st[loca].l+1)\*abs(st[loca].num-ope);

        st[loca].lazy+=abs(st[loca].num-ope);

        st[loca].num=ope;

    }

**else**

    {

        pushdown(loca);

        ll mid=(st[loca].l+st[loca].r)>>1;

**if**(l<=mid) change(l,r,loca<<1,ope);

**if**(r>mid) change(l,r,loca<<1|1,ope);

        pushup(loca);

    }

}

ll ask(ll l,ll r,ll loca)

{

**if**(st[loca].l>=l&&st[loca].r<=r) **return** st[loca].sum;

    pushdown(loca);

    ll mid=(st[loca].l+st[loca].r)>>1;

    ll temp=0;

**if**(l<=mid) temp+=ask(l,r,loca<<1);

**if**(r>mid) temp+=ask(l,r,loca<<1|1);

**return** temp;

}

**int** main()

{

    IOS;

    cin>>n>>m;

    build(1,n,1);

**for**(ll i=0;i<m;i++)

    {

        ll cas,l,r,y;

        cin>>cas;

**if**(cas==1)

        {

            cin>>l>>r>>y;

            change(l,r,1,y);

        }

**else**

        {

            cin>>l>>r;

            cout<<ask(l,r,1)<<endl;

        }

    }

}