```
#include<bits/stdc++.h>
using namespace std;
#define ll int
struct st
{
   ll zero,neg;
} tree[100005 * 4],arr[100005];

void build(ll node , ll b, ll e)
{
   if(b== e)
   {
      if(arr[b].zero == 1)
      {
         tree[node].zero = 1;
         tree[node].neg = 0;
      }
      else if(arr[b].neg == 1)
      {
         tree[node].neg = 1;
         tree[node].zero = 0;
      }
      else
      {
         tree[node].neg = 0;
         tree[node].zero = 0;
      }
      return;
   }
   ll left = node * 2;
   ll right = node * 2 + 1;
   ll mid = (b + e)/2;

   build(left, b, mid);
   build(right,mid+1,e);

   tree[node].zero = tree[left].zero +
tree[right].zero;
   tree[node].neg = tree[left].neg +
tree[right].neg;

}
void update( ll node, ll b , ll e, ll i, ll val)
{
   if(e<i || b>i) return ;

   if(b == e && b == i)
   {
      if(arr[b].zero == 1 && val<0)
      {
         tree[node].zero =  -1;
         tree[node].neg = 1;
         arr[b].zero = 0;
         arr[b].neg = 1;
      }
      else if(arr[b].zero == 1 && val>0)
      {
         tree[node].zero--;
         tree[node].neg = 0;
         arr[b].zero = 0;
         arr[b].neg = 0;
      }
      else if(arr[b].neg == 1 && val== 0)
```

```
{
   tree[node].zero++;
   tree[node].neg--;
   arr[b].zero = 1;
   arr[b].neg = 0;
}
else if(arr[b].neg == 1 && val<0)
{
   arr[b].zero = 0;
   arr[b].neg = 1;
}
else if(arr[b].neg == 1 && val>0)
{
   tree[node].neg --;
   tree[node].zero = 0;
   arr[b].zero = 0;
   arr[b].neg = 0;
}
else if(arr[b].neg!=1 && arr[b].zero!=1)
{
   if(val>0)
   {
      tree[node].neg = 0;
      tree[node].zero = 0;
      arr[b].zero = 0;
      arr[b].neg = 0;
   }
   else if(val == 0)
   {
      tree[node].neg = 0;
      tree[node].zero = 1;
      arr[b].zero = 1;
      arr[b].neg = 0;
   }
   else
   {
      tree[node].neg = 1;
      tree[node].zero = 0;
      arr[b].zero = 0;
      arr[b].neg = 1;
   }
  }
   return;
  }
  ll left = node * 2;
  ll right = node * 2 + 1;
  ll mid = (b + e)/2;

  update(left, b, mid, i, val);
  update(right,mid+1,e, i, val);
  tree[node].zero = tree[left].zero +
tree[right].zero;
  tree[node].neg = tree[left].neg +
tree[right].neg;

}
st query( ll node, ll b , ll e, ll i, ll j)
{
  if(e<i || b>j)
  {
     st rt;
     rt.zero =  0;
     rt.neg = 0;
```

```
        return rt;
    }
    if(b>=i && e<=j)
    {
        return tree[node];
    }

    ll left = node * 2;
    ll right = node * 2 + 1;
    ll mid = (b + e)/2;

    st x = query(left, b, mid, i, j);
    st y = query(right,mid+1,e, i, j);
    st last;
    last.zero = x.zero + y.zero;
    last.neg = x.neg + y.neg;
    return last;
}

int main()
{
    ll n,q,x;
    while(scanf("%d %d",&n,&q) == 2)
    {
//      memset(tree,0,sizeof(tree));
//      memset(arr,0,sizeof(arr));
        for(int k=1; k<=n; k++)
        {
            scanf("%d",&x);
            if(x == 0)
            {
                arr[k].zero = 1;
                arr[k].neg = 0;
            }
            else if(x<0)
            {
                arr[k].zero = 0;
                arr[k].neg = 1;
            }
            else
            {
                arr[k].zero = 0;
                arr[k].neg = 0;
            }

        }
        build(1,1,n);
        ll ii,jj;
        char stt,pro[100005];
        getchar();
        int track = 0;
        while(q--)
        {
            scanf("%c",&stt);
            if(stt == 'P')
            {
                scanf("%d %d",&ii,&jj);
                st ans = query(1,1,n,ii,jj);
                // cout<<ans.zero<<"
"<<ans.neg<<endl;
                if(ans.zero>=1) pro[track] = '0';
                //printf("0");
```

```
            else if(ans.neg>=0 && ans.neg%2
== 1) pro[track] = '-';
            //  printf("-");
            else pro[track] = '+';
            track++;
            //printf("+");
         }
         else
         {
            scanf("%d %d",&ii,&jj);
            update(1,1,n,ii,jj);
         }
         getchar();
      }
      pro[track] = '\0';
      puts(pro);
   }

}
/*
4 6
-2 6 0 -1
C 1 10
P 1 4
C 3 7
P 2 2
C 4 -5
P 1 4
5 9
1 5 -2 4 3
P 1 2
P 1 5
C 4 -5
P 1 5
P 4 5
C 3 0
P 1 5
C 4 -5
C 4 -5
*/
```