```
//seg tree
#include <iostream>

using namespace std;

const int N = 1e5 + 2;

int a[N];
int n,q,l,t,r;
int tree[4*N];

int mrg(int x, int y){
    return max(x,y);
}

void build(int id=0,int ns = 0, int ne = n-1){
    if(ns==ne){
        tree[id] = a[ns];
        return ;
    }
    int l = 2*id+1;
    int r = l+1;
    int md = ns+(ne-ns)/2;
    build(l, ns, md);
    build(r, md+1, ne);
    tree[id] = mrg(tree[l],tree[r]);
}


int query(int qs, int qe, int id=0, int ns=0, int ne=n-1){
    if(ns>qe || qs>ne){
        return -1e7; ///infnity
    }
    if(qs<=ns && qe>=ne){
        return tree[id];
    }
    int l = 2*id+1;
    int r = l+1;
    int md = ns+(ne-ns)/2;
    return mrg(query(qs, qe, l, ns, md), query(qs, qe,r, md+1,ne));
}

void upd(int pos , int val , int id=0, int ns=0,int ne=n-1){
    if(ns>pos || pos>ne){
        return;
    }
    if(ns==ne){
        tree[id]=val;
        return ;
    }
```

```
    int l = 2*id+1;
    int r = l+1;
    int md = ns+(ne-ns)/2;
    upd(pos, val,l, ns, md);
    upd(pos, val, r, md+1, ne);
    tree[id] = mrg(tree[l],tree[r]);
}



int main()
{
    cin >> n >> q;
    for(int i=0;i<n;i++){
        cin >> a[i];
    }
    build();
    while(q--){
        cin >> t >> l >> r;
        switch (t){
        case 1: // upd l=x, r=y;
            upd(--l,r);
            break;
        case 2:
            cout << query(--l,--r) << endl;
            break;
        }
    }
    return 0;
}
```