```
#include<bits/stdc++.h>
using namespace std;
const int MAX_N = (1 << 21) - 1;
const int inf = numeric_limits<int>::max();

int tree[MAX_N];

// call init(0,0,n) to init
// node k => [l,r)
void init(int k,int l,int r){
  if(r - l == 1){
    tree[k] = inf; // init leaf
  }else{
    int chl = k*2+1;
    int chr = k*2+2;
    tree[k] = inf; // init internal-node
    init(chl, l, (l+r)/2);
    init(chr, (l+r)/2, r);
  }
}

// call update(i,x,0,0,n) to update
// node k => [l,r)
void update(int i,int x,int k,int l, int r){
  if(l <= i && i < r){
    if(r - l == 1){
      tree[k] = x;
    }else{
      int chl = k*2+1;
      int chr = k*2+2;
      update(i,x,chl,l,(l+r)/2);
      update(i,x,chr,(l+r)/2,r);
      tree[k] = min(tree[chl],tree[chr]);
    }
  }
}

// call find(a,b,0,0,n) to find min-value in [a,b)
// node k => [l,r)
int find(int a,int b,int k,int l,int r){
  // not cross
  if(r <= a || b <= l)return inf;
  // [a,b) contain [l,r)
  else if(a <= l && r <= b)return tree[k];
  // otherwise
  else{
    int chl = k*2+1;
    int chr = k*2+2;
    int vl = find(a,b,chl,l,(l+r)/2);
    int vr = find(a,b,chr,(l+r)/2,r);
    return min(vl,vr);
  }
}

int main(void){
  int n,q;
  cin >> n >> q;
  init(0,0,n);
  for(int i = 0;i < q;++i){
    int c,x,y;
    cin >> c >> x >> y;
    if(c == 0){
      update(x,y,0,0,n);
    }else{
      cout << find(x,y+1,0,0,n) << endl;
    }
  }
  return 0;
}
```