```
#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];
int uniform[MAX_N];

// call init() to init
void init(){
  tree[0] = inf;
  uniform[0] = 1;
}

// call update(a,b,x,0,0,n) to update
// node k => [l,r)
void update(int a,int b,int x,int k,int l, int r){
  // not cross
  if(r <= a || b <= l)return;
  else if(a <= l && r <= b){
    // [a,b) contain [l,r)
    tree[k] = x;
    uniform[k] = 1;
  } else{
    // otherwise
    int chl = k*2+1;
    int chr = k*2+2;
    if(uniform[k]){
      tree[chl] = tree[chr] = tree[k];
      uniform[chl] = uniform[chr] = 1;
    }
    uniform[k] = 0;
    update(a,b,x,chl,l,(l+r)/2);
    update(a,b,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 if(uniform[k])return tree[k];
  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();
  for(int i = 0;i < q;++i){
    int c;
    cin >> c;
    if(c == 0){
      int s,t,x;
      cin >> s >> t >> x;
      update(s,t+1,x,0,0,n);
    }else{
      int s,t;
      cin >> s >> t;
```

```
      cout << find(s,t+1,0,0,n) << endl;
    }
  }
  return 0;
}
```