```cpp
class Treap {
    static minstd_rand generator;

    struct Node {
        int key, priority;
        Node *l = nullptr, *r = nullptr;
        Node (int key): key(key), priority(generator()) {}
    } *root = nullptr;

    static Node *merge(Node *a, Node *b){
        if (!a || !b){
            return a ? a : b;
        }
        if (a->priority > b->priority){
            a->r = merge(a->r, b);
            return a;
        }
        else {
            b->l = merge(a, b->l);
            return b;
        }
    }

    static void split(Node *n, int key, Node *&a, Node *&b){
        if (!n){
            a = b = nullptr;
            return ;
        }
        if (n -> key < key){
            //      a = n
            // n->l        a' b'
            split(n->r, key, n->r, b);
            a = n;
        }
        else {
            split(n->l, key, a, n->l);
            b = n;
        }
    }

public:

    bool find(int key){
        Node *less, *equal, *greater;
        split(root, key, less, greater);
        split(greater, key + 1, equal, greater);
        bool result = equal;
        root = merge(merge(less, equal), greater);
        return equal;
    }

    void insert(int key){
        Node *greater, *less;
        split(root, key, less, greater);
        less = merge(less, new Node(key));
        root = merge(less, greater);
    }

    int min(Node *n) const {
        if (!n)
            return -1;
        while (n -> l){
            n = n -> l;
        }
        return n -> key;
    }

    int next(int key){
        Node *greater, *less;
        split(root, key, less, greater);
        int ans = min(greater);
        root = merge(less, greater);
        return ans;
    }

};
```