```
#include <bits/stdc++.h>
#define maxn 100001
#define maxT 11000111

using namespace std;

struct Tnode {
    int left, right;
    int sum;
    Tnode() {}
    Tnode (int left, int right, int sum) : left (left), right (right), sum (sum) {}
};

int n, m, a[maxn], x[maxn];
Tnode it[maxT];
int nT = 0;
int ver[maxn];

int update (int l, int r, int u, int oldver) {
    if (l == r) {
        ++nT;
        it[nT].sum = it[oldver].sum + 1;
        it[nT].left = it[nT].right = 0;
        return nT;
    }

    int mid = (l + r) / 2;
    int cur = ++nT;

    if (u <= mid) {
        it[cur].left = update (l, mid, u, it[oldver].left);
        it[cur].right = it[oldver].right;
    } else {
        it[cur].right = update (mid + 1, r, u, it[oldver].right);
        it[cur].left = it[oldver].left;
    }
    it[cur].sum = it[it[cur].left].sum + it[it[cur].right].sum;
    return cur;
}

int get (int ver, int l, int r, int u, int v) {
    if (v < l || u > r)
        return 0;
    if (u <= l && r <= v)
        return it[ver].sum;
    int mid = (l + r) / 2;
    int t1 = get (it[ver].left, l, mid, u, v);
    int t2 = get (it[ver].right, mid + 1, r, u, v);
    return t1 + t2;
}

int InitT (int l, int r) {
    if (l == r) {
        ++nT;
        it[nT] = Tnode (0, 0, 0);
        return nT;
    }

    int mid = (l + r) / 2;
    int cur = ++nT;
    it[cur].sum = 0;
    it[cur].left = InitT (l, mid);
    it[cur].right = InitT (mid + 1, r);
    return cur;
}

int query (int u, int v, int k) {
    int lo = 0, hi = n;
    while (hi - lo > 1) {
        int mid = (lo + hi) / 2;
        int res = get (ver[v], 1, n, 1, mid) - get (ver[u - 1], 1, n, 1, mid);
        if (res < k)
            lo = mid;
        else
            hi = mid;
    }
    return x[hi];
}

int main() {
#ifndef ONLINE_JUDGE
    freopen ("inp.txt", "r", stdin);
    freopen ("out.txt", "w", stdout);
#endif // ONLINE_JUDGE
    scanf ("%d %d", &n, &m);
    for (int i = 1; i <= n; ++i)
        scanf ("%d", &a[i]);
    for (int i = 1; i <= n; ++i)
        x[i] = a[i];
    sort (x + 1, x + n + 1);
    for (int i = 1; i <= n; ++i)
        a[i] = lower_bound (x + 1, x + n + 1, a[i]) - x;

    ver[0] = InitT (1, n);

    for (int i = 1; i <= n; ++i)
        ver[i] = update (1, n, a[i], ver[i - 1]);

    while (m--) {
        int u, v, k;
        scanf ("%d %d %d", &u, &v, &k);
        printf ("%d\n", query (u, v, k));
    }

}
```