```cpp
#include <iostream>
#include <queue>
#include <vector>

using namespace std;

struct min_heap{
public:
    int top(){
        return pq[0];
    }

    bool empty(){
        return pq.empty();
    }

    void max_heapify(int i){
        while(2 * i + 1 < pq.size()){
            int exch_idx = 2 * i + 1;
            if(exch_idx + 1 < pq.size() && pq[exch_idx] > pq[exch_idx + 1])
                exch_idx++;
            if(pq[i] < pq[exch_idx]) break;
            swap(pq[i], pq[exch_idx]);
            i = exch_idx;
        }
    }

    void pop(){
        swap(pq[0], pq[pq.size() - 1]);
        pq.pop_back();
        max_heapify(0);
    }

    void swim(int i){
        while(i > 0 && pq[(i - 1) / 2] > pq[i]){
            int parent = (i - 1) / 2;
            swap(pq[parent], pq[i]);
            i = parent;
        }
    }

    void push(int e){
        pq.push_back(e);
        swim(pq.size() - 1);
    }

private:
    vector<int> pq;
};
```

```cpp

void display_array(const vector<int>& A){
    for(int e : A) cout << e << " ";
    cout << endl;
}

// O(n * log(k))
void k_index_away(vector<int>& A, int k){
    min_heap pq;
    for(int i = 0;i < k;i++)
        pq.push(A[i]);
    int idx = 0;
    for(int i = k;i < A.size();i++){
        pq.push(A[i]);
        A[idx++] = pq.top();
        pq.pop();
    }
    while(!pq.empty()){
        A[idx++] = pq.top();
        pq.pop();
    }
}

int main() {
    vector<int> A = {2, 8, 0, 17, 5, 12};
    k_index_away(A, 2);
    display_array(A);
    vector<int> B = {9, 8, 7, 6, 5, 4, 3, 2, 1};
    k_index_away(B, 8);
    display_array(B);
}
```