```cpp
#include <iostream>
#include <random>
#include <vector>
#include <chrono>

using namespace std;

bool get_line(const string& prompt, string& userinput){
    cout << prompt;
    getline(cin, userinput);
    return !userinput.empty();
}

int partition(vector<int> &arr, int lo, int hi, int pivot_idx){
    int pivot_value = arr[pivot_idx];
    int left = lo - 1;
    int right = hi;
    swap(arr[pivot_idx], arr[right]);
    while(left < right){
        while(arr[++left] < pivot_value) if(left == right) break;
        while(arr[--right] > pivot_value) if(left == right) break;
        if(left >= right) break;
        swap(arr[left], arr[right]);
    }
    swap(arr[left],arr[hi]);
    return left;
}

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

int quick_select_idx(vector<int> &arr, int k){
    unsigned int seed = chrono::steady_clock::now().time_since_epoch().count();
    mt19937 gen(seed);
    int lo = 0, hi = arr.size() - 1;
    while(lo < hi){
        int pivot_idx = uniform_int_distribution<int>{lo, hi}(gen);
        int new_pivot_idx = partition(arr, lo, hi, pivot_idx);
        if(new_pivot_idx == k - 1) return new_pivot_idx;
        else if(new_pivot_idx < k - 1) lo = new_pivot_idx + 1;
        else hi = new_pivot_idx - 1;
    }
    return lo;
}

vector<int> max_k_numbers(vector<int>& arr, int k){
    int starting_idx = quick_select_idx(arr, arr.size() -
k + 1);
    vector<int> result;
    for(int i = starting_idx;i < arr.size();i++)
        result.push_back(arr[i]);
    return result;
}

int main(){
    unsigned int seed = chrono::steady_clock::now().
time_since_epoch().count();
    mt19937 gen(seed);
    string userinput;
    while(get_line("(part 1) Enter positive integer n: ",
userinput)){
        int n = stoi(userinput);
        uniform_int_distribution<int>
uniform_int_distribution(-100, 100);

        vector<int> A;
        for(int i = 0;i < n;i++)
            A.push_back(uniform_int_distribution(gen));

        display_array(A);

        string second_input;
        get_line("Enter a number between 1 to n: ",
second_input);
        int kth = stoi(second_input);
        int idx = quick_select_idx(A, kth);
        cout << A[idx] << endl;
    }

    while(get_line("(part 2) Enter positive integer n: ",
userinput)){
        int n = stoi(userinput);
        uniform_int_distribution<int>
uniform_int_distribution(-100, 100);

        vector<int> A;
        for(int i = 0;i < n;i++)
            A.push_back(uniform_int_distribution(gen));

        display_array(A);

```

```cpp
        string second_input;
        get_line("Enter a number between 1 to n: ", second_input);
        int kth = stoi(second_input);
        auto result = max_k_numbers(A, kth);
        display_array(result);
    }
}

```