```cpp
#include <iostream>
#include <cmath>
#include <algorithm>

int dumbMax(unsigned [], int);
int smartMax(unsigned [], int);
int cont(unsigned [], int, int, int);
void compute_intervals(unsigned [], unsigned, unsigned, int);


/////////////////////////////////////////////////////////////////////////
//                   WHERE THE MAGIC HAPPENS
/////////////////////////////////////////////////////////////////////////


// Dumb version:
// Simply look for the maximum element...

int dumbMax(unsigned v[], int n) {
    int max = v[0];
    for (auto i = 0; i < n; i++)
        if (max < v[i])
            max = v[i];
    return max;
}


// Better version:
// Use sort() and extract the last element in the sorted array

int smartMax(unsigned v[], int n) {
    std::sort(v, v + n);
    return v[n - 1];
}


// This is not very intelligent...
// For every single element I will have to scan the entire array.
// There are probably better ways to do it.

int cont(unsigned v[], int m, int M, int n) {
    int tot = 0;
    for (auto i = 0; i < n; i++) {
        if (v[i] >= m && v[i] < M)
            tot++;
    }
    return tot;
}


void compute_intervals(unsigned v[], unsigned n, unsigned l, int M) {
    int k, i;
    k = ceil((double)M / l);
    for (i = 0; i < (l - 1); i++)
        std::cout << i * k << " : " << cont(v, (i * k), ((i + 1) * k), n) << "\n";

    std::cout << i * k << " : " << cont(v, (i * k), (M + 1), n) << "\n";
}
```