Find the (min / max) value for each continuous subArray of size k:

```cpp
//1D-Array
vector<int> get_window_1d(const vector<int> &a, int k)
{
    const int n = a.size();
    vector<int> b(n - k + 1);
    deque<int> mono;
    for (int i = 0; i < n; ++i)
    {
        //change this condition to ⩾ for (min / max)
        while (!mono.empty() && a[mono.back()] ⩽ a[i])
            mono.pop_back();

        mono.push_back(i);

        if (mono.front() ⩽ i - k) mono.pop_front();

        if (i + 1 ⩾ k) b[i + 1 - k] = a[mono.front()];
    }
    return b;
}

//2D-matrix Find the (min / max) value for each continuous subMatrix of size k × l:
vector<vector<int>> get_window_2d(const vector<vector<int>> &a, int k, int l)
{
    //change the condition in the 1d function to change it here
    const int n = a.size(), m = a[0].size();
    vector<vector<int>> b(m - l + 1, vector<int>(n));
    for (int i = 0; i < n; ++i)
    {
        const auto tmp = get_window_1d(a[i], l);
        for (int j = 0; j < m - l + 1; ++j)
        {
            b[j][i] = tmp[j];
        }
    }
    vector<vector<int>> c(n - k + 1, vector<int>(m - l + 1));
    for (int j = 0; j < m - l + 1; ++j)
    {
        const auto tmp = get_window_1d(b[j], k);
        for (int i = 0; i < n - k + 1; ++i)
        {
            c[i][j] = tmp[i];
        }
    }
    return c;
}
```