```
#include<bits/stdc++.h>
using namespace std;
#define ll long long
#define mx 100005
#define MOD 1000000007LL
ll a[mx],b[mx],lft[mx],rgt[mx],st[mx][20];
void sparsetable(ll m){
    for(int i=1; i<=m; i++)st[i][0] = i;
    for(int j=1; (1<<j)<=m; j++){
        for(int i=1; (i+(1<<j)-1)<=m; i++){
            int x = st[i][j-1];
            int y = st[i+(1<<(j-1))][j-1];
            if(b[x]>=b[y])st[i][j] = x;
            else st[i][j] = y;
        }
    }
}
ll query(int l,int r){
    if(l>r)swap(l,r);
    int p = (int)log2(r-l+1);
    if(b[st[l][p]] >= b[st[r-(1<<p)+1][p]])
    return st[l][p];
    return st[r-(1<<p)+1][p];
}
int main(){
    int tt; scanf("%d",&tt);
    for(int ks=1; ks<=tt; ks++){
        ll n,k; scanf("%lld%lld",&n,&k);
        for(int i=1; i<=n; i++) scanf("%lld",&a[i]);

        a[0] = 0;
        lft[0] = 0;
        for(int i=1; i<=n; i++){
            if(a[i]>a[i-1]) lft[i] = min(k,lft[i-1]+1);
            else lft[i] = 1;
        }

        a[n+1] = 100000005;
        rgt[n+1] = 0;
        for(int i=n; i>=1; i--){
            if(a[i]<a[i+1]) rgt[i] = min(k,rgt[i+1]+1);
            else rgt[i] = 1;
        }

        ll sum = 0;
        for(int i=1; i<=k; i++){
            sum += lft[i];
        }

        b[1] = sum;
        for(int i=k+1; i<=n; i++){
            sum -= rgt[i-k];
            sum += lft[i];
            b[i-k+1] = sum;
        }

        ll m = n-k+1;

        sparsetable(m);

        ll ans = 0;
        ll tot = 0;
        for(int i=1; i<=m; i++){
            int lo = i, hi = m;
            int id = i;
            while(lo<=hi){
                int md = (lo+hi)/2;
                int x = query(lo,md);
                if(b[x]<=b[i]){
                    id = md;
                    lo = md+1;
                }else{
                    hi = md-1;
                }
            }
            ll p = id-i+1;

            lo = 1, hi = i-1;
            id = i;
            while(lo<=hi){
                int md = (lo+hi)/2;
                int x = query(md,hi);
                if(b[x]<b[i]){
                    id = md;
                    hi = md-1;
                }else{
                    lo = md+1;
                }
            }

            ll q = max(1,i-id+1);
            ll pq = (p*q)%MOD;

            ans += (b[i]*pq);
            ans %= MOD;
        }
        printf("%lld\n",ans);
    }
    return 0;
}
/*
2
16 5
2 4 5 3 2 8 10 10 11 8 2 20 21 22 23 5
*/
```

Biswa and Restaurant Business

Limits: 3s, 512 MB

After huge success in Borhani business, Biswa has recently started restaurant business. He’s got the information about the numbers of customers for **N** days, a sequence of **N** integers a_1, a_2, …, a_N. He wants to calculate the popularity of his restaurant from this data. First he’ll choose a window of size **K (≤ N)**. Then for each K-sized window from the original sequence (first K-sized window will contain a_1, a_2, …, a_K, second one will contain a_2, a_3, …, a_{K+1}, and so on), he’ll calculate the number of all possible continuous increasing sequences in it and write them down in order. It’ll create a sequence of **N - K + 1** integers b_1, b_2, …, b_{N-K+1}. Finally he’ll run the following function on the later found sequence:

$$\sum_{i=1}^{N-K+1} \sum_{j=i}^{N-K+1} \mathbf{max}(b_i, b_{i+1}, \ldots, b_j)$$

Output of this function is the popularity that Biswa desires to calculate.

A sequence is increasing if for every two adjacent elements in it, the later one is greater than the previous one. A sequence containing a single element is considered to be a increasing sequence as well.

Input

First line of the test case contains a positive integer **T (≤ 10)** denoting the number of scenarios. For each scenario the first line will contain two positive integers **N** and **K** ($\mathbf{1 \le K \le N \le 10^5}$). The second line contains a sequence a_1, a_2, …,a_N ($\mathbf{1 \le a_i \le 10^5}$), where a_i equals to the number of customers on i-th day.

Output

For each scenario, output a single integer - the popularity in a separate line. The answer may be large so compute it modulo $\mathbf{10^9+7}$.

Samples

Input	Output

1
4 3
1 2 3 2
	16

Note: In the sample test the sequence is: 1, 2, 3, 2 and the window size is 3. The first window will contain 1, 2, 3 and 6 possible continuous increasing sequences can be found there: {1, 2, 3}, {1, 2}, {2, 3}, {1}, {2}, {3}. Similarly the second window will contain 4 continuous increasing sequences: {2, 3}, {2}, {3}, {2}. So the sequence b is: 6, 4 and if we run that function on it we’ll get 16 which is also 16 in modulo 10^9+7.