低分直接暴力，或者暴力 + 优化即可。

80pt

考虑每两个数字的差值对答案的贡献。假设选择三个数字$(k = 3)$，那么对于 a_1, a_2 这一对差值来说，在 $[a_1, a_2, a_3]$ 中会有这一对差值， $[a_1, a_2, a_4]$ 中也会有这一对差值，一直到 $[a_1, a_2, a_n]$ 都会有这一对差值，共有 $n - 2$ 种包含这一对差值的数组。所以这一对差值会使得答案增加 $|a_1 - a_2| * (n - 2)$。

然后进行找规律，如果 $k = 4$ 或更多的话会有多少种数组包含 a_1, a_2 的差值，答案是 C_{n-2}^{k-2}，其中 C 是组合数符号。

所以我们可以预处理好组合数或者预处理好 C_{n-2}^{k-2} 的值，枚举每一对差值对答案的贡献，以 $O(n^2)$ 的方法解决这个问题。

```cpp
#include <bits/stdc++.h>
using namespace std;
int mod = 1e9 + 7;
const int maxn = 2e5 + 5;
#define ll long long
ll F[maxn], Finv[maxn], inv[maxn], a[maxn], n, k, ans = 0, premax[maxn];
void init(){
    inv[1] = 1;
    for(int i = 2; i < maxn; i ++){
        inv[i] =(mod - mod / i) * 1ll * inv[mod % i] % mod;
    }
    F[0] = Finv[0] = 1;
    for(int i = 1; i < maxn; i ++){
        F[i] = F[i-1] * 1ll * i % mod;
        Finv[i] = Finv[i-1] * 1ll * inv[i] % mod;
    }
}
ll comb(int n, int m){//comb(n, m)就是C(n, m)
    if(m < 0 || m > n) return 0;
    return F[n] * 1ll * Finv[n - m] % mod * Finv[m] % mod;
}

ll A(int n, int m){
        if(m < 0 || m > n) return 0;
    return F[n] * 1ll * Finv[n-m] % mod;
}
int main(){
        init();
        cin >> n >> k;
        for(int i = 1; i <= n; i++){
                cin >> a[i];
        }
        for(int j = 1; j <= n; j++)
                for(int i = j + 1; i <= n; i++){
                        ans = (ans + abs(a[i] - a[j]) * comb(n-2, k-2)) % mod;
                }
        cout << ans << endl;
}
```

100pt

我们发现每一对差值对答案的贡献都是：差值 * C_{n-2}^{k-2}，所以我们只需要算出所有数字的差值之和，再乘以 C_{n-2}^{k-2} 就是答案了。

算一个数组中所有差值的和可以用排序 + 前缀和来完成。

```cpp
#include <bits/stdc++.h>
using namespace std;
int mod = 1e9 + 7;
const int maxn = 2e5 + 5;
#define ll long long
ll F[maxn], Finv[maxn], inv[maxn], a[maxn], n, k, ans = 0, premax[maxn];
void init(){
    inv[1] = 1;
    for(int i = 2; i < maxn; i ++){
        inv[i] =(mod - mod / i) * 1ll * inv[mod % i] % mod;
    }
    F[0] = Finv[0] = 1;
    for(int i = 1; i < maxn; i ++){
        F[i] = F[i-1] * 1ll * i % mod;
        Finv[i] = Finv[i-1] * 1ll * inv[i] % mod;
    }
}
ll comb(int n, int m){//comb(n, m)就是C(n, m)
    if(m < 0 || m > n) return 0;
    return F[n] * 1ll * Finv[n - m] % mod * Finv[m] % mod;
}

ll A(int n, int m){
        if(m < 0 || m > n) return 0;
    return F[n] * 1ll * Finv[n-m] % mod;
}
int main(){
        init();
        cin >> n >> k;
        for(int i = 1; i <= n; i++){
                cin >> a[i];
        }
        sort(a+1, a+1+n);
        for(int i = 1; i <= n; i++){
                premax[i] = premax[i-1] + a[i];
        }
        for(int i = 1; i <= n; i++){
                ans = (ans + (a[i] * (i - 1) - premax[i-1]) % mod * comb(n-2, k-2)) % mod;
        }
```

```
        cout << ans << endl;
}
```