```cpp
// https://www.geeksforgeeks.org/count-smaller-elements-on-right-side-and-greater-elements-on-left-side-using-binary-index-tree/

#include <bits/stdc++.h>
using namespace std;

int getSum(int BITree[], int index)
{
        int sum = 0;

        while (index > 0) {
                sum += BITree[index];

                index -= index & (-index);
        }
        return sum;
}

void updateBIT(int BITree[], int n, int index, int val)
{
        while (index <= n) {

                BITree[index] += val;

                index += index & (-index);
        }
}

void convert(int arr[], int n)
{

        int temp[n];
        for (int i = 0; i < n; i++)
                temp[i] = arr[i];
        sort(temp, temp + n);

        for (int i = 0; i < n; i++) {

                arr[i] = lower_bound(temp, temp + n, arr[i]) - temp + 1;
        }
}

int findElements(int arr[], int n)
{
        convert(arr, n);

        int BIT[n + 1];
        for (int i = 1; i <= n; i++)
                BIT[i] = 0;

        int smaller_right[n], greater_left[n];

        for (int i = n - 1; i >= 0; i--) {

                smaller_right[i] = getSum(BIT, arr[i] - 1);

                updateBIT(BIT, n, arr[i], 1);
        }

        for (int i = 1; i <= n; i++)
                BIT[i] = 0;

        for (int i = 0; i < n; i++) {

                greater_left[i] = i - getSum(BIT, arr[i]);
```

```
                updateBIT(BIT, n, arr[i], 1);
        }

        int maxdiff = INT_MIN;

        for (int i = 0; i < n; i++) {

      maxdiff = max(maxdiff, abs(greater_left[i] - smaller_right[i]));

    }

    return maxdiff;
}

int main()
{
        int arr[] = {1, 4, 2, 7};

        int n = sizeof(arr) / sizeof(arr[0]);

        cout << findElements(arr, n);

        return 0;
}
```