// CPP code to count the change required to

// convert the array into non-increasing array

#include <bits/stdc++.h>

using namespace std;

int DecreasingArray(int a[], int n)

{

    int sum = 0, dif = 0;

    // min heap

    priority\_queue<int, vector<int>, greater<int> > pq;

    // Here in the loop we will

    // check that whether the upcoming

    // element of array is less than top

    // of priority queue. If yes then we

    // calculate the difference. After

    // that we will remove that element

    // and push the current element in

    // queue. And the sum is incremented

    // by the value of difference

    for (int i = 0; i < n; i++) {

        if (!pq.empty() && pq.top() < a[i]) {

            dif = a[i] - pq.top();

            sum += dif;

            pq.pop();

            pq.push(a[i]);

        }

        pq.push(a[i]);

    }

    return sum;

}

// Driver Code

int main()

{

    int a[] = { 3, 1, 2, 1 };

    int n = sizeof(a) / sizeof(a[0]);

    cout << DecreasingArray(a, n);

    return 0;

}