```cpp
//
// There are N stones, numbered 1,2,…,N. For each i (1≤i≤N), the height of Stone i is h[i]
// There is a frog who is initially on Stone 1. He will repeat the following action
// some number of times to reach Stone N:
//    - If the frog is currently on Stone i, jump to Stone i+1 or Stone i+2.
//      Here, a cost of | h[i] - h[j] | is incurred, where j is the stone to land on.
// Find the minimum possible total cost incurred before the frog reaches Stone N.
//
//   Time Complexity: O(N)
//

#include <bits/stdc++.h>

using namespace std;

int main() {
    int n;
    cin >> n;

    vector<int> heights(n);
    for (int i = 0; i < n; i++) {
        cin >> heights[i];
    }

    vector<int> dp(n); // dp[i] = minimum cost of getting to i-th stone from the first
    dp[0] = 0;

    // O(n)
    for (int i = 1; i < n; i++) {
        // what is the minimum cost of getting to the previous stone + abs(diff)
        int cur_ans = dp[i-1] + abs(heights[i] - heights[i-1]);
        if (i - 2 >= 0) {
            cur_ans = min(cur_ans, dp[i-2] + abs(heights[i] - heights[i-2]));
        }

        dp[i] = cur_ans;
    }

    cout << dp[n-1] << endl;
    return 0;
}
```