```cpp
//
//  There are N slimes lining up in a row. Initially, the i-th slime from the left has a size of a[i].
//  Taro is trying to combine all the slimes into a larger slime.
//  He will perform the following operation repeatedly until there is only one slime:
//     - Choose two adjacent slimes, and combine them into a new slime.
//       The new slime has a size of x+y, where x and y are the sizes
//       of the slimes before combining them.
//       Here, a cost of x+y is incurred. The positional relationship
//       of the slimes does not change while combining slimes.
//  Find the minimum possible total cost incurred.
//
//    Time Complexity: O(N^3)
//

#include <bits/stdc++.h>
#define ll long long

using namespace std;

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

    vector<int> s(n);
    vector<ll> prefix(n);
    for (int i = 0; i < n; i++) {
        cin >> s[i];
        prefix[i] = (i-1 >= 0 ? prefix[i-1]: 0LL) + s[i];
    }

    vector<vector<ll>> dp(n, vector<ll>(n, 0));

    for (int size = 1; size <= n; size++) {
        for (int start = 0; start < n - size; start++) {
            int L = start;
            int R = start + size;

            ll best = LLONG_MAX;
            for (int i = 0; i < R - L; i++) {
                best = min(best, dp[L][L+i] + dp[L+i+1][R]);
            }

            dp[L][R] = best + (prefix[R] - (L-1 >= 0 ? prefix[L-1] : 0));
        }
    }

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