In Tobyland, dogs communicate with each other through barking. These barks are like symbols in a complex language, and naturally different symbols have different probabilities of appearing. Toby enumerates the symbols (barks) from 1 to N in the dog alphabetic order, so symbol 1 is lexicographically smaller than 2 and so on. Toby also counts in many dog conversations the frequency of each symbol to estimate its probabilities.

Now Toby wants to put these symbols (barks) in a fast data structure (Toby is the only dog that studies computer science), and he chose a binary search tree (BST). As in any BST, all nodes that are to the left of some node with symbol X must be smaller lexicographically than X and nodes to the right must have symbols bigger than X. Toby is incredibly concerned with efficiency; he defines the cost of searching a symbol as the number of nodes that have to be visited from the root to the node with the desired symbol—if the searched symbol is at the root the cost is 1. Now Toby wants to find the BST that minimizes the expected cost of searching a symbol.

Input

The input consist of several test cases. Each test case begins with a line containing the number N, which is the number of symbols in the dog alphabet, the second line contains N numbers p_i stating the probability of receiving the symbol i. It is guaranteed that the sum of the probabilities is 1.

$1 \leq N \leq 100$; $0 \leq p_i \leq 1$; $\sum_{i=1}^{N} p_i = 1$

Output

For each test case write the minimum expected cost of searching a symbol in the BST. Print one line per test case. Answers with a relative or absolute error less than 10^{-4} are considered correct.

Sample Input

```
3
0.33 0.34 0.33
3
0.8 0.15 0.05
4
0.23 0.4 0.17 0.2
```

Sample Output

```
1.6600
1.2500
1.7700
```