Probablistic OR

Everyone knows OR operation. Let us define new operation which we will call Probabilistic OR. We will denote this operation as #. For given real number p (0 <= p <= 1) and two bits a and b:

- if a = 1 and b = 1, then #(a, b) = 1;
- if a = 0 and b = 0, then #(a, b) = 0;
- else #(a, b) = 0 with probability p, #(a, b) = 1 with probability 1-p.

Now for two given non-negative integers x and y we can define bitwise Probabilistic OR operation. The result of this operation is a number received by performing # operation for each pair of bits of x and y in same positions. For example, for p= 0.5, x = 2, and y = 4, we will get 0, 2, 4 or 6 each with probability 0.25. You will be given a list of non-negative integers. You have to implement a program which will calculate the expected value of the result of performing bitwise probabilistic OR operation on all these numbers given some p. The numbers will be taken from left to right.

Input

Input file starts with real number p (0 <= p <= 1) with exactly two digits after the decimal point. Integer n follows (1 <= n <= 100). Next line contains n numbers ai in the order they are taking pert in the operation (0 <= ai <= 10^9).

Output

Output the expected value of performing Probabilistic OR operation on the given numbers for given p. Print the result with two digits after the decimal point.

Example

Input:

```
0.25 4
1 2 3 4
```

Output:

```
5.11
```