A. Maxim and Discounts

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Maxim always goes to the supermarket on Sundays. Today the supermarket has a special offer of discount systems.

There are m types of discounts. We assume that the discounts are indexed from 1 to m. To use the discount number i, the customer takes a special basket, where he puts exactly q_i items he buys. Under the terms of the discount system, in addition to the items in the cart the customer can receive at most two items from the supermarket for free. The number of the "free items" (0, 1 or 2) to give is selected by the customer. The only condition imposed on the selected "free items" is as follows: each of them mustn't be more expensive than the cheapest item out of the q_i items in the cart.

Maxim now needs to buy n items in the shop. Count the minimum sum of money that Maxim needs to buy them, if he use the discount system optimally well.

Please assume that the supermarket has enough carts for any actions. Maxim can use the same discount multiple times. Of course, Maxim can buy items without any discounts.

Input
The first line contains integer m ($1 \le m \le 10^5$) — the number of discount types. The second line contains m integers: $q_1, q_2, ..., q_m$ ($1 \le q_i \le 10^5$).

The third line contains integer n ($1 \le n \le 10^5$) — the number of items Maxim needs. The fourth line contains n integers: $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^4$) — the items' prices.

The numbers in the lines are separated by single spaces.

Output
In a single line print a single integer — the answer to the problem.

Sample test(s)

input
```
1
2
4
50 50 100 100
```
output
```
200
```

input
```
2
2 3
5
50 50 50 50 50
```
output
```
150
```

input
```
1
1
7
1 1 1 1 1 1 1
```
output
```
3
```

Note
In the first sample Maxim needs to buy two items that cost 100 and get a discount for two free items that cost 50. In that case, Maxim is going to pay 200.

In the second sample the best strategy for Maxim is to buy 3 items and get 2 items for free using the discount. In that case, Maxim is going to pay 150.

B. Maxim and Restaurant

Maxim has opened his own restaurant! The restaurant has got a huge table, the table's length is p meters.

Maxim has got a dinner party tonight, n guests will come to him. Let's index the guests of Maxim's restaurant from 1 to n. Maxim knows the sizes of all guests that are going to come to him. The i-th guest's size (a_i) represents the number of meters the guest is going to take up if he sits at the restaurant table.

Long before the dinner, the guests line up in a queue in front of the restaurant in some order. Then Maxim lets the guests in, one by one. Maxim stops letting the guests in when there is no place at the restaurant table for another guest in the queue. There is no place at the restaurant table for another guest in the queue, if the sum of sizes of all guests in the restaurant plus the size of this guest from the queue is larger than p. In this case, not to offend the guest who has no place at the table, Maxim doesn't let any other guest in the restaurant, even if one of the following guests in the queue would have fit in at the table.

Maxim is now wondering, what is the average number of visitors who have come to the restaurant for all possible $n!$ orders of guests in the queue. Help Maxim, calculate this number.

Input

The first line contains integer n $(1 \le n \le 50)$ — the number of guests in the restaurant. The next line contains integers $a_1, a_2, ..., a_n$ $(1 \le a_i \le 50)$ — the guests' sizes in meters. The third line contains integer p $(1 \le p \le 50)$ — the table's length in meters.

The numbers in the lines are separated by single spaces.

Output

In a single line print a real number — the answer to the problem. The answer will be considered correct, if the absolute or relative error doesn't exceed 10^{-4}.

Sample test(s)

input
3 1 2 3 3

output
1.3333333333

Note

In the first sample the people will come in the following orders:

- $(1, 2, 3)$ — there will be two people in the restaurant;
- $(1, 3, 2)$ — there will be one person in the restaurant;
- $(2, 1, 3)$ — there will be two people in the restaurant;
- $(2, 3, 1)$ — there will be one person in the restaurant;
- $(3, 1, 2)$ — there will be one person in the restaurant;
- $(3, 2, 1)$ — there will be one person in the restaurant.

In total we get $(2 + 1 + 2 + 1 + 1 + 1) / 6 = 8 / 6 = 1.(3)$.

C. Maxim and Matrix

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Maxim loves to fill in a matrix in a special manner. Here is a pseudocode of filling in a matrix of size $(m+1) \times (m+1)$:

```
1.  for (j = 1; j <= m; j = j + 1) {
2.      a[1][j] = 0
3.  }
4.  a[1][m + 1] = 1
5.
6.  for (i = 2; i <= m + 1; i = i + 1) {
7.      for (j = 1; j <= m + 1; j = j + 1) {
8.          if (j equal 1) {
9.              a[i][j] = a[i - 1][j + 1]
10.         } else {
11.             if (j equal m + 1) {
12.                 a[i][j] = a[i - 1][j - 1]
13.             } else {
14.                 a[i][j] = a[i - 1][j - 1] xor a[i - 1][j + 1]
15.             }
16.         }
17.     }
18. }
```

Maxim asks you to count, how many numbers m $(1 \le m \le n)$ are there, such that the sum of values in the cells in the row number $m+1$ of the resulting matrix equals t.

Expression $(x\ xor\ y)$ means applying the operation of bitwise excluding "OR" to numbers x and y. The given operation exists in all modern programming languages. For example, in languages *C++* and *Java* it is represented by character "^", in *Pascal* — by "xor".

Input

A single line contains two integers n and t $(1 \le n, t \le 10^{12}, t \le n+1)$.

Please, do not use the `%lld` specifier to read or write 64-bit integers in *C++*. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

Output

In a single line print a single integer — the answer to the problem.

Sample test(s)

input
1 1
output
1

input
3 2
output
1

input
3 3
output
0

input
1000000000000 1048576
output
118606527258

D. Maxim and Increasing Subsequence

time limit per test: 6 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Maxim loves sequences, especially those that strictly increase. He is wondering, what is the length of the longest increasing subsequence of the given sequence a?

Sequence a is given as follows:

- the length of the sequence equals $n \times t$;
- $a_i = b_{((i-1) \mod n)+1}$ $(1 \le i \le n \times t)$, where operation $(x \mod y)$ means taking the remainder after dividing number x by number y.

Sequence s_1, s_2, ..., s_r of length r is a *subsequence* of sequence a_1, a_2, ..., a_n, if there is such increasing sequence of indexes $i_1, i_2, ..., i_r$ $(1 \le i_1 < i_2 < ... < i_r \le n)$, that $a_{i_j} = s_j$. In other words, the subsequence can be obtained from the sequence by crossing out some elements.

Sequence s_1, s_2, ..., s_r is *increasing*, if the following inequality holds: $s_1 < s_2 < ... < s_r$.

Maxim have k variants of the sequence a. Help Maxim to determine for each sequence the length of the longest increasing subsequence.

Input

The first line contains four integers k, n, $maxb$ and t $(1 \le k \le 10; 1 \le n, maxb \le 10^5; 1 \le t \le 10^9; n \times maxb \le 2 \cdot 10^7)$. Each of the next k lines contain n integers $b_1, b_2, ..., b_n$ $(1 \le b_i \le maxb)$.

Note that for each variant of the sequence a the values n, $maxb$ and t coincide, the only arrays bs differ.

The numbers in the lines are separated by single spaces.

Output

Print k integers — the answers for the variants of the sequence a. Print the answers in the order the variants follow in the input.

Sample test(s)

input
3 3 5 2
3 2 1
1 2 3
2 3 1

output
2
3
3

E. Maxim and Calculator

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Maxim has got a calculator. The calculator has two integer cells. Initially, the first cell contains number 1, and the second cell contains number 0. In one move you can perform one of the following operations:

1. Let's assume that at the current time the first cell contains number a, and the second cell contains number b. Write to the second cell number $b + 1$;
2. Let's assume that at the current time the first cell contains number a, and the second cell contains number b. Write to the first cell number $a \cdot b$.

Maxim is wondering, how many integers x ($l \leq x \leq r$) are there, such that we can write the number x to the first cell of the calculator, having performed at most p moves.

Input

The first line contains three integers: l, r, p ($2 \leq l \leq r \leq 10^9$, $1 \leq p \leq 100$).

The numbers in the line are separated by single spaces.

Output

In a single line print a single integer — the answer to the problem.

Sample test(s)

input

```
2 10 3
```

output

```
1
```

input

```
2 111 100
```

output

```
106
```

input

```
2 111 11
```

output

```
47
```