A. Robot Sequence

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

Calvin the robot lies in an infinite rectangular grid. Calvin's source code contains a list of n commands, each either 'U', 'R', 'D', or 'L' — instructions to move a single square up, right, down, or left, respectively. How many ways can Calvin execute a non-empty contiguous substrings of commands and return to the same square he starts in? Two substrings are considered different if they have different starting or ending indices.

Input

The first line of the input contains a single positive integer, n ($1 \leq n \leq 200$) — the number of commands.

The next line contains n characters, each either 'U', 'R', 'D', or 'L' — Calvin's source code.

Output

Print a single integer — the number of contiguous substrings that Calvin can execute and return to his starting square.

Examples

input

```
6
URLLDR
```

output

```
2
```

input

```
4
DLUU
```

output

```
0
```

input

```
7
RLRLRLR
```

output

```
12
```

Note

In the first case, the entire source code works, as well as the "RL" substring in the second and third characters.

Note that, in the third case, the substring "LR" appears three times, and is therefore counted three times to the total result.

B. Cards

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

Catherine has a deck of n cards, each of which is either red, green, or blue. As long as there are at least two cards left, she can do one of two actions:

- take any two (not necessarily adjacent) cards with different colors and exchange them for a new card of the third color;
- take any two (not necessarily adjacent) cards with the same color and exchange them for a new card with that color.

She repeats this process until there is only one card left. What are the possible colors for the final card?

Input

The first line of the input contains a single integer n ($1 \leq n \leq 200$) — the total number of cards.

The next line contains a string s of length n — the colors of the cards. s contains only the characters 'B', 'G', and 'R', representing blue, green, and red, respectively.

Output

Print a single string of up to three characters — the possible colors of the final card (using the same symbols as the input) in alphabetical order.

Examples

input

```
2
RB
```

output

```
G
```

input

```
3
GRG
```

output

```
BR
```

input

```
5
BBBBB
```

output

```
B
```

Note

In the first sample, Catherine has one red card and one blue card, which she must exchange for a green card.

In the second sample, Catherine has two green cards and one red card. She has two options: she can exchange the two green cards for a green card, then exchange the new green card and the red card for a blue card. Alternatively, she can exchange a green and a red card for a blue card, then exchange the blue card and remaining green card for a red card.

In the third sample, Catherine only has blue cards, so she can only exchange them for more blue cards.

C. Block Towers

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

Students in a class are making towers of blocks. Each student makes a (non-zero) tower by stacking pieces lengthwise on top of each other. n of the students use pieces made of two blocks and m of the students use pieces made of three blocks.

The students don't want to use too many blocks, but they also want to be unique, so no two students' towers may contain the same number of blocks. Find the minimum height necessary for the tallest of the students' towers.

Input

The first line of the input contains two space-separated integers n and m ($0 \le n, m \le 1\,000\,000$, $n + m > 0$) — the number of students using two-block pieces and the number of students using three-block pieces, respectively.

Output

Print a single integer, denoting the minimum possible height of the tallest tower.

Examples

input

```
1 3
```

output

```
9
```

input

```
3 2
```

output

```
8
```

input

```
5 0
```

output

```
10
```

Note

In the first case, the student using two-block pieces can make a tower of height 4, and the students using three-block pieces can make towers of height 3, 6, and 9 blocks. The tallest tower has a height of 9 blocks.

In the second case, the students can make towers of heights 2, 4, and 8 with two-block pieces and towers of heights 3 and 6 with three-block pieces, for a maximum height of 8 blocks.

D. Jerry's Protest

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

Andrew and Jerry are playing a game with Harry as the scorekeeper. The game consists of three rounds. In each round, Andrew and Jerry draw randomly without replacement from a jar containing n balls, each labeled with a distinct positive integer. Without looking, they hand their balls to Harry, who awards the point to the player with the larger number and **returns the balls** to the jar. The winner of the game is the one who wins at least two of the three rounds.

Andrew wins rounds 1 and 2 while Jerry wins round 3, so Andrew wins the game. However, Jerry is unhappy with this system, claiming that he will often lose the match despite having the higher overall total. What is the probability that the sum of the three balls Jerry drew is strictly higher than the sum of the three balls Andrew drew?

Input

The first line of input contains a single integer n ($2 \le n \le 2000$) — the number of balls in the jar.

The second line contains n integers a_i ($1 \le a_i \le 5000$) — the number written on the ith ball. It is guaranteed that no two balls have the same number.

Output

Print a single real value — the probability that Jerry has a higher total, given that Andrew wins the first two rounds and Jerry wins the third. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-6}.

Namely: let's assume that your answer is a, and the answer of the jury is b. The checker program will consider your answer correct, if $\frac{|a-b|}{\max(1,b)} \le 10^{-6}$.

Examples

input

```
2
1 2
```

output

```
0.0000000000
```

input

```
3
1 2 10
```

output

```
0.0740740741
```

Note

In the first case, there are only two balls. In the first two rounds, Andrew must have drawn the 2 and Jerry must have drawn the 1, and vice versa in the final round. Thus, Andrew's sum is 5 and Jerry's sum is 4, so Jerry never has a higher total.

In the second case, each game could've had three outcomes — $10 - 2$, $10 - 1$, or $2 - 1$. Jerry has a higher total if and only if Andrew won $2 - 1$ in both of the first two rounds, and Jerry drew the 10 in the last round. This has probability $\frac{1}{3} \cdot \frac{1}{3} \cdot \frac{2}{3} = \frac{2}{27}$.

E. Simple Skewness

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Define the *simple skewness* of a collection of numbers to be the collection's mean minus its median. You are given a list of n (not necessarily distinct) integers. Find the non-empty subset (with repetition) with the maximum simple skewness.

The mean of a collection is the average of its elements. The median of a collection is its middle element when all of its elements are sorted, or the average of its two middle elements if it has even size.

Input

The first line of the input contains a single integer n ($1 \leq n \leq 200\ 000$) — the number of elements in the list.

The second line contains n integers x_i ($0 \leq x_i \leq 1\ 000\ 000$) — the ith element of the list.

Output

In the first line, print a single integer k — the size of the subset.

In the second line, print k integers — the elements of the subset in any order.

If there are multiple optimal subsets, print any.

Examples

input

```
4
1 2 3 12
```

output

```
3
1 2 12
```

input

```
4
1 1 2 2
```

output

```
3
1 1 2
```

input

```
2
1 2
```

output

```
2
1 2
```

Note

In the first case, the optimal subset is $\{1, 2, 12\}$, which has mean 5, median 2, and simple skewness of $5 - 2 = 3$.

In the second case, the optimal subset is $\{1, 1, 2\}$. Note that repetition is allowed.

In the last case, any subset has the same median and mean, so all have simple skewness of 0.

F. Group Projects

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

There are n students in a class working on group projects. The students will divide into groups (some students may be in groups alone), work on their independent pieces, and then discuss the results together. It takes the i-th student a_i minutes to finish his/her independent piece.

If students work at different paces, it can be frustrating for the faster students and stressful for the slower ones. In particular, the *imbalance* of a group is defined as the maximum a_i in the group minus the minimum a_i in the group. Note that a group containing a single student has an imbalance of 0. How many ways are there for the students to divide into groups so that the total imbalance of all groups is at most k?

Two divisions are considered distinct if there exists a pair of students who work in the same group in one division but different groups in the other.

Input

The first line contains two space-separated integers n and k ($1 \le n \le 200$, $0 \le k \le 1000$) — the number of students and the maximum total imbalance allowed, respectively.

The second line contains n space-separated integers a_i ($1 \le a_i \le 500$) — the time it takes the i-th student to complete his/her independent piece of work.

Output

Print a single integer, the number of ways the students can form groups. As the answer may be large, print its value modulo $10^9 + 7$.

Examples

input

```
3 2
2 4 5
```

output

```
3
```

input

```
4 3
7 8 9 10
```

output

```
13
```

input

```
4 0
5 10 20 21
```

output

```
1
```

Note

In the first sample, we have three options:

- The first and second students form a group, and the third student forms a group. Total imbalance is $2 + 0 = 2$.
- The first student forms a group, and the second and third students form a group. Total imbalance is $0 + 1 = 1$.
- All three students form their own groups. Total imbalance is 0.

In the third sample, the total imbalance must be 0, so each student must work individually.

G. Raffles

time limit per test: 5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Johnny is at a carnival which has n raffles. Raffle i has a prize with value p_i. Each participant can put tickets in whichever raffles they choose (they may have more than one ticket in a single raffle). At the end of the carnival, one ticket is selected at random from each raffle, and the owner of the ticket wins the associated prize. A single person can win multiple prizes from different raffles.

However, county rules prevent any one participant from owning more than half the tickets in a single raffle, i.e. putting more tickets in the raffle than all the other participants combined. To help combat this (and possibly win some prizes), the organizers started by placing a single ticket in each raffle, which they will never remove.

Johnny bought t tickets and is wondering where to place them. Currently, there are a total of l_i tickets in the i-th raffle. He watches as other participants place tickets and modify their decisions and, at every moment in time, wants to know how much he can possibly earn. Find the maximum possible expected value of Johnny's winnings at each moment if he distributes his tickets optimally. Johnny may redistribute all of his tickets arbitrarily between each update, but he may not place more than t tickets total or have more tickets in a single raffle than all other participants combined.

Input

The first line contains two integers n, t, and q ($1 \le n, t, q \le 200\ 000$) — the number of raffles, the number of tickets Johnny has, and the total number of updates, respectively.

The second line contains n space-separated integers p_i ($1 \le p_i \le 1000$) — the value of the i-th prize.

The third line contains n space-separated integers l_i ($1 \le l_i \le 1000$) — the number of tickets initially in the i-th raffle.

The last q lines contain the descriptions of the updates. Each description contains two integers t_k, r_k ($1 \le t_k \le 2$, $1 \le r_k \le n$) — the type of the update and the raffle number. An update of type 1 represents another participant adding a ticket to raffle r_k. An update of type 2 represents another participant removing a ticket from raffle r_k.

It is guaranteed that, after each update, each raffle has at least 1 ticket (not including Johnny's) in it.

Output

Print q lines, each containing a single real number — the maximum expected value of Johnny's winnings after the k-th update. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-6}.

Namely: let's assume that your answer is a, and the answer of the jury is b. The checker program will consider your answer correct, if $\frac{|a-b|}{\max(1,b)} \le 10^{-6}$.

Examples

input

```
2 1 3
4 5
1 2
1 1
1 2
2 1
```

output

```
1.666666667
1.333333333
2.000000000
```

input

```
3 20 5
6 8 10
6 6 6
1 1
1 2
1 3
2 3
2 3
```

output

```
12.000000000
12.000000000
11.769230769
12.000000000
12.000000000
```

Note

In the first case, Johnny only has one ticket to distribute. The prizes are worth 4 and 5, and the raffles initially have 1 and 2 tickets, respectively. After the first update, each raffle has 2 tickets, so Johnny has expected value $\frac{5}{3}$ of winning by placing his ticket into the second raffle. The second update adds a ticket to the second raffle, so Johnny can win $\frac{4}{3}$ in the first raffle. After the final update, Johnny keeps his ticket in the first raffle and wins $\frac{4}{2}$.

In the second case, Johnny has more tickets than he is allowed to spend. In particular, after the first update, there are 7, 6, and 6 tickets in each raffle, respectively, so Johnny can only put in 19 tickets, winning each prize with probability $\frac{1}{2}$. Also, note that after the last two updates, Johnny must remove a ticket from the last raffle in order to stay under $\frac{1}{2}$ the tickets in the third raffle.