Codeforces Round #334 (Div. 2)

A. Uncowed Forces

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Kevin Sun has just finished competing in Codeforces Round #334! The round was 120 minutes long and featured five problems with maximum point values of 500, 1000, 1500, 2000, and 2500, respectively. Despite the challenging tasks, Kevin was uncowed and bulldozed through all of them, distinguishing himself from the herd as the best cowmputer scientist in all of Bovinia. Kevin knows his submission time for each problem, the number of wrong submissions that he made on each problem, and his total numbers of successful and unsuccessful hacks. Because Codeforces scoring is complicated, Kevin wants you to write a program to compute his final score.

Codeforces scores are computed as follows: If the maximum point value of a problem is x, and Kevin submitted correctly at minute m but made w wrong submissions, then his score on that problem is $\max\left(0.3x, \left(1-\frac{m}{250}\right)x - 50w\right)$. His total score is equal to the sum of his scores for each problem. In addition, Kevin's total score gets increased by 100 points for each successful hack, but gets decreased by 50 points for each unsuccessful hack.

All arithmetic operations are performed with absolute precision and no rounding. It is guaranteed that Kevin's final score is an integer.

Input

The first line of the input contains five space-separated integers m_1, m_2, m_3, m_4, m_5, where m_i $(0 \le m_i \le 119)$ is the time of Kevin's last submission for problem i. His last submission is always correct and gets accepted.

The second line contains five space-separated integers w_1, w_2, w_3, w_4, w_5, where w_i $(0 \le w_i \le 10)$ is Kevin's number of wrong submissions on problem i.

The last line contains two space-separated integers h_s and h_u $(0 \le h_s, h_u \le 20)$, denoting the Kevin's numbers of successful and unsuccessful hacks, respectively.

Output

Print a single integer, the value of Kevin's final score.

Sample test(s)

input
20 40 60 80 100 0 1 2 3 4 1 0
output
4900

input
119 119 119 119 119 0 0 0 0 0 10 0
output
4930

Note

In the second sample, Kevin takes 119 minutes on all of the problems. Therefore, he gets $\left(1-\frac{119}{250}\right) = \frac{131}{250}$ of the points on each problem. So his score from solving problems is $\frac{131}{250}(500+1000+1500+2000+2500) = 3930$. Adding in $10 \cdot 100 = 1000$ points from hacks, his total score becomes $3930 + 1000 = 4930$.

B. More Cowbell

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

Kevin Sun wants to move his precious collection of n cowbells from Naperthrill to Exeter, where there is actually grass instead of corn. Before moving, he must pack his cowbells into k boxes of a fixed size. In order to keep his collection safe during transportation, he won't place more than **two** cowbells into a single box. Since Kevin wishes to minimize expenses, he is curious about the smallest size box he can use to pack his entire collection.

Kevin is a meticulous cowbell collector and knows that the size of his i-th ($1 \le i \le n$) cowbell is an integer s_i. In fact, he keeps his cowbells sorted by size, so $s_{i-1} \le s_i$ for any $i > 1$. Also an expert packer, Kevin can fit one or two cowbells into a box of size s if and only if the sum of their sizes does not exceed s. Given this information, help Kevin determine the smallest s for which it is possible to put all of his cowbells into k boxes of size s.

Input

The first line of the input contains two space-separated integers n and k ($1 \le n \le 2 \cdot k \le 100\,000$), denoting the number of cowbells and the number of boxes, respectively.

The next line contains n space-separated integers $s_1, s_2, ..., s_n$ ($1 \le s_1 \le s_2 \le ... \le s_n \le 1\,000\,000$), the sizes of Kevin's cowbells. It is guaranteed that the sizes s_i are given in non-decreasing order.

Output

Print a single integer, the smallest s for which it is possible for Kevin to put all of his cowbells into k boxes of size s.

Sample test(s)

input
2 1 2 5
output
7

input
4 3 2 3 5 9
output
9

input
3 2 3 5 7
output
8

Note

In the first sample, Kevin must pack his two cowbells into the same box.

In the second sample, Kevin can pack together the following sets of cowbells: $\{2, 3\}$, $\{5\}$ and $\{9\}$.

In the third sample, the optimal solution is $\{3, 5\}$ and $\{7\}$.

C. Alternative Thinking

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

Kevin has just recevied his disappointing results on the USA Identification of Cows Olympiad (USAICO) in the form of a binary string of length n. Each character of Kevin's string represents Kevin's score on one of the n questions of the olympiad—`'1'` for a correctly identified cow and `'0'` otherwise.

However, all is not lost. Kevin is a big proponent of alternative thinking and believes that his score, instead of being the sum of his points, should be the length of the longest alternating subsequence of his string. Here, we define an *alternating subsequence* of a string as a **not-necessarily contiguous** subsequence where no two consecutive elements are equal. For example, $\{0, 1, 0, 1\}$, $\{1, 0, 1\}$, and $\{1, 0, 1, 0\}$ are alternating sequences, while $\{1, 0, 0\}$ and $\{0, 1, 0, 1, 1\}$ are not.

Kevin, being the sneaky little puffball that he is, is willing to hack into the USAICO databases to improve his score. In order to be subtle, he decides that he will flip exactly one substring—that is, take a **contiguous** non-empty substring of his score and change all `'0'`s in that substring to `'1'`s and vice versa. After such an operation, Kevin wants to know the length of the longest possible alternating subsequence that his string could have.

Input

The first line contains the number of questions on the olympiad n ($1 \leq n \leq 100\,000$).

The following line contains a binary string of length n representing Kevin's results on the USAICO.

Output

Output a single integer, the length of the longest possible alternating subsequence that Kevin can create in his string after flipping a single substring.

Sample test(s)

input
8 10000011
output
5

input
2 01
output
2

Note

In the first sample, Kevin can flip the bolded substring '`100`**`00`**`011`' and turn his string into '`10011011`', which has an alternating subsequence of length 5: '**`1`**`0`**`01`**`1`**`01`**`1`'.

In the second sample, Kevin can flip the entire string and still have the same score.

D. Moodular Arithmetic

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

As behooves any intelligent schoolboy, Kevin Sun is studying psycowlogy, cowculus, and cryptcowgraphy at the Bovinia State University (BGU) under Farmer Ivan. During his Mathematics of Olympiads (MoO) class, Kevin was confronted with a weird functional equation and needs your help. For two fixed integers k and p, where p is an odd prime number, the functional equation states that

$$f(kx \mod p) \equiv k \cdot f(x) \mod p$$

for some function $f : \{0, 1, 2, \cdots, p-1\} \to \{0, 1, 2, \cdots, p-1\}$. (This equation should hold for any integer x in the range 0 to p - 1, inclusive.)

It turns out that f can actually be many different functions. Instead of finding a solution, Kevin wants you to count the number of distinct functions f that satisfy this equation. Since the answer may be very large, you should print your result modulo $10^9 + 7$.

Input

The input consists of two space-separated integers p and k ($3 \le p \le 1\,000\,000$, $0 \le k \le p$ - 1) on a single line. It is guaranteed that p is an odd prime number.

Output

Print a single integer, the number of distinct functions f modulo $10^9 + 7$.

Sample test(s)

input
3 2
output
3

input
5 4
output
25

Note

In the first sample, $p = 3$ and $k = 2$. The following functions work:

1. $f(0) = 0, f(1) = 1, f(2) = 2$.
2. $f(0) = 0, f(1) = 2, f(2) = 1$.
3. $f(0) = f(1) = f(2) = 0$.

E. Lieges of Legendre

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

Kevin and Nicky Sun have invented a new game called Lieges of Legendre. In this game, two players take turns modifying the game state with Kevin moving first. Initially, the game is set up so that there are n piles of cows, with the i-th pile containing a_i cows. During each player's turn, that player calls upon the power of Sunlight, and uses it to either:

1. Remove a single cow from a chosen non-empty pile.
2. Choose a pile of cows with even size $2 \cdot x$ ($x > 0$), and replace it with k piles of x cows each.

The player who removes the last cow wins. Given n, k, and a sequence $a_1, a_2, ..., a_n$, help Kevin and Nicky find the winner, given that both sides play in optimal way.

Input

The first line of the input contains two space-separated integers n and k ($1 \leq n \leq 100\,000$, $1 \leq k \leq 10^9$).

The second line contains n integers, $a_1, a_2, \ldots a_n$ ($1 \leq a_i \leq 10^9$) describing the initial state of the game.

Output

Output the name of the winning player, either "`Kevin`" or "`Nicky`" (without quotes).

Sample test(s)

input
2 1 3 4
output
Kevin

input
1 2 3
output
Nicky

Note

In the second sample, Nicky can win in the following way: Kevin moves first and is forced to remove a cow, so the pile contains two cows after his move. Next, Nicky replaces this pile of size 2 with two piles of size 1. So the game state is now two piles of size 1. Kevin then removes one of the remaining cows and Nicky wins by removing the other.