Codeforces Round #140 (Div. 1)

A. Flying Saucer Segments

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

An expedition group flew from planet ACM-1 to Earth in order to study the bipedal species (its representatives don't even have antennas on their heads!).

The flying saucer, on which the brave pioneers set off, consists of three sections. These sections are connected by a chain: the 1-st section is adjacent only to the 2-nd one, the 2-nd one — to the 1-st and the 3-rd ones, the 3-rd one — only to the 2-nd one. The transitions are possible only between the adjacent sections.

The spacecraft team consists of n aliens. Each of them is given a rank — an integer from 1 to n. The ranks of all astronauts are distinct. The rules established on the Saucer, state that an alien may move from section a to section b only if it is senior in rank to all aliens who are in the segments a and b (besides, the segments a and b are of course required to be adjacent). Any alien requires exactly 1 minute to make a move. Besides, safety regulations require that no more than one alien moved at the same minute along the ship.

Alien A is senior in rank to alien B, if the number indicating rank A, is more than the corresponding number for B.

At the moment the whole saucer team is in the 3-rd segment. They all need to move to the 1-st segment. One member of the crew, the alien with the identification number CFR-140, decided to calculate the minimum time (in minutes) they will need to perform this task.

Help CFR-140, figure out the minimum time (in minutes) that all the astronauts will need to move from the 3-rd segment to the 1-st one. Since this number can be rather large, count it modulo m.

Input

The first line contains two space-separated integers: n and m $(1 \le n, m \le 10^9)$ — the number of aliens on the saucer and the number, modulo which you should print the answer, correspondingly.

Output

Print a single number — the answer to the problem modulo m.

Sample test(s)

input
1 10
output
2

input
3 8
output
2

Note

In the first sample the only crew member moves from segment 3 to segment 2, and then from segment 2 to segment 1 without any problems. Thus, the whole moving will take two minutes.

To briefly describe the movements in the second sample we will use value $R_i \to j$, which would correspond to an alien with rank i moving from the segment in which it is at the moment, to the segment number j. Using these values, we will describe the movements between the segments in the second sample: $R_3 \to 2, R_3 \to 1, R_2 \to 2, R_3 \to 2, R_3 \to 3, R_2 \to 1, R_3 \to 2, R_3 \to 1, R_1 \to 2, R_3 \to 2, R_3 \to 3, R_2 \to 2, R_3 \to 2, R_3 \to 1, R_2 \to 3, R_3 \to 2, R_3 \to 3, R_1 \to 1, R_3 \to 2, R_3 \to 1, R_2 \to 2, R_3 \to 2, R_3 \to 3, R_2 \to 1, R_3 \to 2, R_3 \to 1$; In total: the aliens need 26 moves. The remainder after dividing 26 by 8 equals 2, so the answer to this test is 2.

B. Naughty Stone Piles

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

There are n piles of stones of sizes $a_1, a_2, ..., a_n$ lying on the table in front of you.

During one move you can take one pile and add it to the other. As you add pile i to pile j, the size of pile j increases by the current size of pile i, and pile i stops existing. The cost of the adding operation equals the size of the added pile.

Your task is to determine the minimum cost at which you can gather all stones in one pile.

To add some challenge, the stone piles built up conspiracy and decided that each pile will let you add to it not more than k times (after that it can only be added to another pile).

Moreover, the piles decided to puzzle you completely and told you q variants (not necessarily distinct) of what k might equal.

Your task is to find the minimum cost for each of q variants.

Input

The first line contains integer n $(1 \le n \le 10^5)$ — the number of stone piles. The second line contains n space-separated integers: $a_1, a_2, ..., a_n$ $(1 \le a_i \le 10^9)$ — the initial sizes of the stone piles.

The third line contains integer q $(1 \le q \le 10^5)$ — the number of queries. The last line contains q space-separated integers $k_1, k_2, ..., k_q$ $(1 \le k_i \le 10^5)$ — the values of number k for distinct queries. Note that numbers k_i can repeat.

Output

Print q whitespace-separated integers — the answers to the queries in the order, in which the queries are given in the input.

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.

Sample test(s)

input
5 2 3 4 1 1 2 2 3
output
9 8

Note

In the first sample one way to get the optimal answer goes like this: we add in turns the 4-th and the 5-th piles to the 2-nd one; then we add the 1-st pile to the 3-rd one; we add the 2-nd pile to the 3-rd one. The first two operations cost 1 each; the third one costs 2, the fourth one costs 5 (the size of the 2-nd pile after the first two operations is not 3, it already is 5).

In the second sample you can add the 2-nd pile to the 3-rd one (the operations costs 3); then the 1-st one to the 3-th one (the cost is 2); then the 5-th one to the 4-th one (the costs is 1); and at last, the 4-th one to the 3-rd one (the cost is 2).

C. Anniversary

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

There are less than 60 years left till the 900-th birthday anniversary of a famous Italian mathematician Leonardo Fibonacci. Of course, such important anniversary needs much preparations.

Dima is sure that it'll be great to learn to solve the following problem by the Big Day: You're given a set A, consisting of numbers $l, l+1, l+2, ..., r$; let's consider all its k-element subsets; for each such subset let's find the largest common divisor of Fibonacci numbers with indexes, determined by the subset elements. Among all found common divisors, Dima is interested in the largest one.

Dima asked to remind you that Fibonacci numbers are elements of a numeric sequence, where $F_1 = 1, F_2 = 1, F_n = F_{n-1} + F_{n-2}$ for $n \geq 3$.

Dima has more than half a century ahead to solve the given task, but you only have two hours. Count the residue from dividing the sought largest common divisor by m.

Input

The first line contains four space-separated integers m, l, r and k $(1 \leq m \leq 10^9; 1 \leq l < r \leq 10^{12}; 2 \leq k \leq r - l + 1)$.

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

Output

Print a single integer — the residue from dividing the sought greatest common divisor by m.

Sample test(s)

input
10 1 8 2
output
3

input
10 1 8 3
output
1

D. The table

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

Harry Potter has a difficult homework. Given a rectangular table, consisting of $n \times m$ cells. Each cell of the table contains the integer. Harry knows how to use two spells: the first spell change the sign of the integers in the selected row, the second — in the selected column. Harry's task is to make non-negative the sum of the numbers in each row and each column using these spells.

Alone, the boy can not cope. Help the young magician!

Input

The first line contains two integers n and m $(1 \leq n, \ m \leq 100)$ — the number of rows and the number of columns.

Next n lines follow, each contains m integers: j-th integer in the i-th line is $a_{i,j}$ $(|a_{i,j}| \leq 100)$, the number in the i-th row and j-th column of the table.

The rows of the table numbered from 1 to n. The columns of the table numbered from 1 to m.

Output

In the first line print the number a — the number of required applications of the first spell. Next print a space-separated integers — the row numbers, you want to apply a spell. These row numbers must be distinct!

In the second line print the number b — the number of required applications of the second spell. Next print b space-separated integers — the column numbers, you want to apply a spell. These column numbers must be distinct!

If there are several solutions are allowed to print any of them.

Sample test(s)

input
4 1 -1 -1 -1 -1
output
4 1 2 3 4 0

input
2 4 -1 -1 -1 2 1 1 1 1
output
1 1 1 4

E. Noble Knight's Path

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

In Berland each feudal owns exactly one castle and each castle belongs to exactly one feudal.

Each feudal, except one (the King) is subordinate to another feudal. A feudal can have any number of vassals (subordinates).

Some castles are connected by roads, it is allowed to move along the roads in both ways. Two castles have a road between them if and only if the owner of one of these castles is a direct subordinate to the other owner.

Each year exactly one of these two events may happen in Berland.

1. The barbarians attacked castle c. The interesting fact is, the barbarians never attacked the same castle twice throughout the whole Berlandian history.
2. A noble knight sets off on a journey from castle a to castle b (provided that on his path he encounters each castle not more than once).

Let's consider the second event in detail. As the journey from a to b is not short, then the knight might want to stop at a castle he encounters on his way to have some rest. However, he can't stop at just any castle: his nobility doesn't let him stay in the castle that has been desecrated by the enemy's stench. A castle is desecrated if and only if it has been attacked after the year of y. So, the knight chooses the k-th castle he encounters, starting from a (castles a and b aren't taken into consideration), that hasn't been attacked in years from $y + 1$ till current year.

The knights don't remember which castles were attacked on what years, so he asked the court scholar, aka you to help them. You've got a sequence of events in the Berland history. Tell each knight, in what city he should stop or else deliver the sad news — that the path from city a to city b has less than k cities that meet his requirements, so the knight won't be able to rest.

Input

The first input line contains integer n $(2 \le n \le 10^5)$ — the number of feudals.

The next line contains n space-separated integers: the i-th integer shows either the number of the i-th feudal's master, or a 0, if the i-th feudal is the King.

The third line contains integer m $(1 \le m \le 10^5)$ — the number of queries.

Then follow m lines that describe the events. The i-th line (the lines are indexed starting from 1) contains the description of the event that occurred in year i. Each event is characterised by type t_i $(1 \le t_i \le 2)$. The description of the first type event looks as two space-separated integers t_i c_i $(t_i = 1;\ 1 \le c_i \le n)$, where c_i is the number of the castle that was attacked by the barbarians in the i-th year. The description of the second type contains five space-separated integers: t_i a_i b_i k_i y_i $(t_i = 2;\ 1 \le a_i, b_i, k_i \le n;\ a_i \ne b_i;\ 0 \le y_i < i)$, where a_i is the number of the castle from which the knight is setting off, b_i is the number of the castle to which the knight is going, k_i and y_i are the k and y from the second event's description.

You can consider the feudals indexed from 1 to n. It is guaranteed that there is only one king among the feudals. It is guaranteed that for the first type events all values c_i are different.

Output

For each second type event print an integer — the number of the castle where the knight must stay to rest, or -1, if he will have to cover the distance from a_i to b_i without a rest. Separate the answers by whitespaces.

Print the answers in the order, in which the second type events are given in the input.

Sample test(s)

input
3 0 1 2 5 2 1 3 1 0 1 2 2 1 3 1 0 2 1 3 1 1 2 1 3 1 2
output
2 -1 -1 2

input
6 2 5 2 2 0 5 3 2 1 6 2 0 1 2 2 4 5 1 0
output

```
5
-1
```

Note

In the first sample there is only castle 2 on the knight's way from castle 1 to castle 3. When the knight covers the path 1 - 3 for the first time, castle 2 won't be desecrated by an enemy and the knight will stay there. In the second year the castle 2 will become desecrated, so the knight won't have anywhere to stay for the next two years (as finding a castle that hasn't been desecrated from years 1 and 2, correspondingly, is important for him). In the fifth year the knight won't consider the castle 2 desecrated, so he will stay there again.
