Codeforces Round #298 (Div. 2)

A. Exam

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

An exam for n students will take place in a long and narrow room, so the students will sit in a line in some order. The teacher suspects that students with adjacent numbers (i and $i + 1$) always studied side by side and became friends and if they take an exam sitting next to each other, they will help each other for sure.

Your task is to choose the maximum number of students and make such an arrangement of students in the room that no two students with adjacent numbers sit side by side.

Input

A single line contains integer n ($1 \le n \le 5000$) — the number of students at an exam.

Output

In the first line print integer k — the maximum number of students who can be seated so that no two students with adjacent numbers sit next to each other.

In the second line print k distinct integers $a_1, a_2, \ldots, a_k$ ($1 \le a_i \le n$), where a_i is the number of the student on the i-th position. The students on adjacent positions mustn't have adjacent numbers. Formally, the following should be true: $|a_i - a_{i+1}| \neq 1$ for all i from 1 to $k - 1$.

If there are several possible answers, output any of them.

Examples

input
6
output
6 1 5 3 6 2 4

input
3
output
2 1 3

B. Covered Path

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

The on-board computer on Polycarp's car measured that the car speed at the beginning of some section of the path equals v_1 meters per second, and in the end it is v_2 meters per second. We know that this section of the route took exactly t seconds to pass.

Assuming that at each of the seconds the speed is constant, and between seconds the speed can change at most by d meters per second in absolute value (i.e., the difference in the speed of any two adjacent seconds does not exceed d in absolute value), find the maximum possible length of the path section in meters.

Input

The first line contains two integers v_1 and v_2 ($1 \le v_1, v_2 \le 100$) — the speeds in meters per second at the beginning of the segment and at the end of the segment, respectively.

The second line contains two integers t ($2 \le t \le 100$) — the time when the car moves along the segment in seconds, d **($0 \le d \le 10$)** — the maximum value of the speed change between adjacent seconds.

It is guaranteed that there is a way to complete the segment so that:

- the speed in the first second equals v_1,
- the speed in the last second equals v_2,
- the absolute value of difference of speeds between any two adjacent seconds doesn't exceed d.

Output

Print the maximum possible length of the path segment in meters.

Examples

input
5 6 4 2
output
26

input
10 10 10 0
output
100

Note

In the first sample the sequence of speeds of Polycarpus' car can look as follows: 5, 7, 8, 6. Thus, the total path is $5 + 7 + 8 + 6 = 26$ meters.

In the second sample, as $d = 0$, the car covers the whole segment at constant speed $v = 10$. In $t = 10$ seconds it covers the distance of 100 meters.

C. Polycarpus' Dice

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

Polycarp has n dice $d_1, d_2, \ldots, d_n$. The i-th dice shows numbers from 1 to d_i. Polycarp rolled all the dice and the sum of numbers they showed is A. Agrippina didn't see which dice showed what number, she knows only the sum A and the values $d_1, d_2, \ldots, d_n$. However, she finds it enough to make a series of statements of the following type: dice i couldn't show number r. For example, if Polycarp had two six-faced dice and the total sum is $A = 11$, then Agrippina can state that each of the two dice couldn't show a value less than five (otherwise, the remaining dice must have a value of at least seven, which is impossible).

For each dice find the number of values for which it can be guaranteed that the dice couldn't show these values if the sum of the shown values is A.

Input

The first line contains two integers n, A ($1 \le n \le 2 \cdot 10^5, n \le A \le s$) — the number of dice and the sum of shown values where $s = d_1 + d_2 + \ldots + d_n$.

The second line contains n integers $d_1, d_2, \ldots, d_n$ ($1 \le d_i \le 10^6$), where d_i is the maximum value that the i-th dice can show.

Output

Print n integers $b_1, b_2, \ldots, b_n$, where b_i is the number of values for which it is guaranteed that the i-th dice couldn't show them.

Examples

input
2 8 4 4
output
3 3

input
1 3 5
output
4

input
2 3 2 3
output
0 1

Note

In the first sample from the statement A equal to 8 could be obtained in the only case when both the first and the second dice show 4. Correspondingly, both dice couldn't show values 1, 2 or 3.

In the second sample from the statement A equal to 3 could be obtained when the single dice shows 3. Correspondingly, it couldn't show 1, 2, 4 or 5.

In the third sample from the statement A equal to 3 could be obtained when one dice shows 1 and the other dice shows 2. That's why the first dice doesn't have any values it couldn't show and the second dice couldn't show 3.

D. Handshakes

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

On February, 30th n students came in the Center for Training Olympiad Programmers (CTOP) of the Berland State University. They came one by one, one after another. Each of them went in, and before sitting down at his desk, greeted with those who were present in the room by shaking hands. Each of the students who came in stayed in CTOP until the end of the day and never left.

At any time any three students could join together and start participating in a team contest, which lasted until the end of the day. The team did not distract from the contest for a minute, so when another student came in and greeted those who were present, he did not shake hands with the members of the contest writing team. Each team consisted of exactly three students, and each student could not become a member of more than one team. Different teams could start writing contest at different times.

Given how many present people shook the hands of each student, get a possible order in which the students could have come to CTOP. If such an order does not exist, then print that this is impossible.

Please note that some students could work independently until the end of the day, without participating in a team contest.

Input

The first line contains integer n ($1 \le n \le 2 \cdot 10^5$) — the number of students who came to CTOP. The next line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i < n$), where a_i is the number of students with who the i-th student shook hands.

Output

If the sought order of students exists, print in the first line "`Possible`" and in the second line print the permutation of the students' numbers defining the order in which the students entered the center. Number i that stands to the left of number j in this permutation means that the i-th student came earlier than the j-th student. If there are multiple answers, print any of them.

If the sought order of students doesn't exist, in a single line print "`Impossible`".

Examples

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

input
9 0 2 3 4 1 1 0 2 2
output
Possible 7 5 2 1 6 8 3 4 9

input
4 0 2 1 1
output
Impossible

Note

In the first sample from the statement the order of events could be as follows:

- student 4 comes in ($a_4 = 0$), he has no one to greet;
- student 5 comes in ($a_5 = 1$), he shakes hands with student 4;
- student 1 comes in ($a_1 = 2$), he shakes hands with two students (students 4, 5);
- student 3 comes in ($a_3 = 3$), he shakes hands with three students (students 4, 5, 1);
- students 4, 5, 3 form a team and start writing a contest;
- student 2 comes in ($a_2 = 1$), he shakes hands with one student (number 1).

In the second sample from the statement the order of events could be as follows:

- student 7 comes in ($a_7 = 0$), he has nobody to greet;
- student 5 comes in ($a_5 = 1$), he shakes hands with student 7;
- student 2 comes in ($a_2 = 2$), he shakes hands with two students (students 7, 5);

- students 7, 5, 2 form a team and start writing a contest;
- student 1 comes in($a_1 = 0$), he has no one to greet (everyone is busy with the contest);
- student 6 comes in ($a_6 = 1$), he shakes hands with student 1;
- student 8 comes in ($a_8 = 2$), he shakes hands with two students (students 1, 6);
- student 3 comes in ($a_3 = 3$), he shakes hands with three students (students 1, 6, 8);
- student 4 comes in ($a_4 = 4$), he shakes hands with four students (students 1, 6, 8, 3);
- students 8, 3, 4 form a team and start writing a contest;
- student 9 comes in ($a_9 = 2$), he shakes hands with two students (students 1, 6).

In the third sample from the statement the order of events is restored unambiguously:

- student 1 comes in ($a_1 = 0$), he has no one to greet;
- student 3 comes in (or student 4) ($a_3 = a_4 = 1$), he shakes hands with student 1;
- student 2 comes in ($a_2 = 2$), he shakes hands with two students (students 1, 3 (or 4));
- the remaining student 4 (or student 3), must shake one student's hand ($a_3 = a_4 = 1$) but it is impossible as there are only two scenarios: either a team formed and he doesn't greet anyone, or he greets all the three present people who work individually.

E. Berland Local Positioning System

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

In Berland a bus travels along the main street of the capital. The street begins from the main square and looks like a very long segment. There are n bus stops located along the street, the i-th of them is located at the distance a_i from the central square, all distances are distinct, the stops are numbered in the order of increasing distance from the square, that is, $a_i < a_{i+1}$ for all i from 1 to $n - 1$. The bus starts its journey from the first stop, it passes stops 2, 3 and so on. It reaches the stop number n, turns around and goes in the opposite direction to stop 1, passing all the intermediate stops in the reverse order. After that, it again starts to move towards stop n. During the day, the bus runs non-stop on this route.

The bus is equipped with the Berland local positioning system. When the bus passes a stop, the system notes down its number.

One of the key features of the system is that it can respond to the queries about the distance covered by the bus for the parts of its path between some pair of stops. A special module of the system takes the input with the information about a set of stops on a segment of the path, a stop number occurs in the set as many times as the bus drove past it. This module returns the length of the traveled segment of the path (or -1 if it is impossible to determine the length uniquely). The operation of the module is complicated by the fact that *stop numbers occur in the request not in the order they were visited but in the non-decreasing order*.

For example, if the number of stops is 6, and the part of the bus path starts at the bus stop number 5, ends at the stop number 3 and passes the stops as follows: $5 \to 6 \to 5 \to 4 \to 3$, then the request about this segment of the path will have form: $3, 4, 5, 5, 6$. If the bus on the segment of the path from stop 5 to stop 3 has time to drive past the 1-th stop (i.e., if we consider a segment that ends with the second visit to stop 3 on the way from 5), then the request will have form: $1, 2, 2, 3, 3, 4, 5, 5, 6$.

You will have to repeat the Berland programmers achievement and implement this function.

Input

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

The second line contains n integers ($1 \le a_i \le 10^9$) — the distance from the i-th stop to the central square. The numbers in the second line go in the increasing order.

The third line contains integer m ($1 \le m \le 4 \cdot 10^5$) — the number of stops the bus visited on some segment of the path.

The fourth line contains m integers ($1 \le b_i \le n$) — the sorted list of numbers of the stops visited by the bus on the segment of the path. The number of a stop occurs as many times as it was visited by a bus.

It is guaranteed that the query corresponds to some segment of the path.

Output

In the single line please print the distance covered by a bus. If it is impossible to determine it unambiguously, print -1.

Examples

input
6 2 3 5 7 11 13 5 3 4 5 5 6
output
10

input
6 2 3 5 7 11 13 9 1 2 2 3 3 4 5 5 6
output
16

input
3 10 200 300 4 1 2 2 3
output
-1

input
3 1 2 3 4 1 2 2 3

output
3

Note

The first test from the statement demonstrates the first example shown in the statement of the problem.

The second test from the statement demonstrates the second example shown in the statement of the problem.

In the third sample there are two possible paths that have distinct lengths, consequently, the sought length of the segment isn't defined uniquely.

In the fourth sample, even though two distinct paths correspond to the query, they have the same lengths, so the sought length of the segment is defined uniquely.

F. Simplified Nonogram

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

In this task you have to write a program dealing with nonograms on fields no larger than 5×20.

Simplified nonogram is a task where you have to build such field (each cell is either white or black) that satisfies the given information about rows and columns. For each row and each column the number of contiguous black segments is specified.

For example if size of the field is $n = 3, m = 5$, and numbers of contiguous black segments in rows are: $[2, 3, 2]$ and in columns are: $[1, 0, 1, 2, 1]$ then the solution may look like:

It is guaranteed that on each test in the testset there exists at least one solution.

Input

In the first line there follow two integers n, m ($1 \le n \le 5, 1 \le m \le 20$) — number of rows and number of columns respectively.

Second line contains n integers $a_1, a_2, \ldots, a_n$ where a_i is the number of contiguous black segments in i-th row of the field.

Similarly, third line contains m integers $b_1, b_2, \ldots, b_m$ where b_i is the number of contiguous black segments in the i-th column of the field.

It is guaranteed that there exists at least one solution.

Output

Output any possible solution. Output should consist of n lines each containing m characters. Denote white cell as "." and black cell as "*".

Examples

input
3 5 2 3 2 1 0 1 2 1
output
*.**. *.*.* *..**

input
3 3 2 1 2 2 1 2
output
. .*. *.*

input
3 3 1 0 1 2 2 2
output
*** ... ***