Task 3. Rabbit

The Mad Hatter just lost his favorite rabbit (The White Rabbit, of course) somewhere in a sequence of N cells and is trying to find it. The cells are numbered with the integers 1 to N. In the beginning the rabbit is in a single unknown cell in the sequence and every second of The Hatter's search proceeds as follows:

1. First, he chooses a single cell from the sequence and checks it. We'll call this cell **the checked cell**. If the rabbit is there, the search ends.
2. Afterwards, the rabbit chooses to either stay in the same cell, or to hop to a neighbouring cell (i.e. one cell leftwards or one cell rightwards). **Note that it is possible for the rabbit to hop in the checked cell, if it's a neighbouring one; that does not end the search!**

The decisions of the rabbit are deterministic given its mood. In particular, the rabbit has the following two moods:

1. ***Scared*** mood – when the rabbit is in this mood, it moves **further away from the checked cell**. If it cannot move further away (i.e. it is in cell 1 or N), it stays in the same cell.
2. ***Curious*** mood – when the rabbit is in this mood, it moves **closer to the checked cell.** Note that it is always possible for the rabbit to move closer.

Note that the rabbit only acts according to the last checked cell and does not care about previous checked cells.

Since this is The Hatter's favorite rabbit, he knows its mood very well. In particular, he knows that the rabbit alternates between exactly S seconds of *scared mood* and C seconds of *curious mood*. For example, if $S = 2$ and $C = 1$, the mood of the rabbit would be the sequence $[scared, scared, curious, scared, scared, curious \dots]$.

The Hatter is very worried about his rabbit and asks you to write a program `rabbit.cpp` that computes a sequence of cells to check, such that the rabbit, regardless of its initial position, is guaranteed to be found.

Input

From the first line of the standard input, your program should read three integers: N, S and C, describing the number of cells and the behavior of the rabbit.

Output

On the first line of the standard output your program should print K, the number of seconds your search sequence takes. On the second line your program should print K integers in the range $[1, N]$, listing the cells checked in each second. Note that this sequence is allowed to have repeats.

Scoring

Let K be the number of seconds in your search sequence. If you attempt to check an invalid cell (i.e. outside of the range $[1, N]$), or if your check sequence doesn't **always** find the rabbit, you will receive 0 points for the test case and a *Wrong Answer* verdict. Otherwise, if a test gives R points, you will receive pR points where:

- $p = 0$, if $K > 2N$
- $p = 1$, if $K \leq T$
- $p = 0.3\left(\frac{T}{K}\right)^2$, otherwise

Here: $T = \frac{N(S+C)}{S+2\max(S,C)} + 3\max(S, C)$

Constraints

$2 \leq N \leq 10^4$
$0 \leq S, C \leq 50$

Test Information

- For 8% of the tests $S = 0, C = 1$
- For 12% of the tests $S = 1, C = 0$
- For 8% of the tests $S = 1, C = 1$

Sample test

Input	Output

`12 2 1`	`14`
	`2 5 3 2 6 1 2 11 12 12 8 10 12 6`

Sample test explanation

One can test that regardless of the starting position, this sequence always finds the rabbit. For example, consider the case where the rabbit starts in cell 8. The search would proceed as follows:

Second	*Checked Cell*	*Rabbit Mood (Before Moving)*	*Rabbit Move*
1	2	Scared	8 -> 9
2	5	Scared	9 -> 10
3	3	Curious	10 -> 9
4	2	Scared	10 -> 11
5	6	Scared	11 -> 12
6	1	Curious	12 -> 11
7	2	Scared	11 -> 12
8	11	Scared	12 -> 12
9	12	Curious	*Found*

For this solution we have $K > T$, since $K = 14$ and $T = 12$.
Thus, the part of the points received for this test is $p \approx 0.22$.