Knockout

The solitaire game *Knockout* is played as follows. The digits from 1 to 9 are written down in ascending order. In each turn, you throw a pair of six-sided dice; you sum the dice, and cross out some set of digits, of your choice, that sum to the same total. If you cannot, the game ends and your score is the remaining digits, taken as a single number. Otherwise, you throw the dice again and continue.

This game can be played to either minimize or maximize your score. Given a position of the game (what digits remain) and a roll of the dice, determine which digits you should remove and what your expected score would be for both versions of the game, assuming you make the best moves possible for whichever version you're playing for the remainder of the game. The expected score is the sum of all possible scores weighted by their probabilities (presuming optimum play).

Input

Each input will consist of a single test case. Note that your program may be run multiple times on different inputs.

Each test case will consist of a single line containing a string of digits $\boldsymbol{d}$ ($1 \le |\boldsymbol{d}| \le 9$) and two integers $\boldsymbol{a}$ and $\boldsymbol{b}$ ($1 \le \boldsymbol{a},\boldsymbol{b} \le 6$), all separated by spaces. The string of digits $\boldsymbol{d}$ will contain a subset of the digits 1..9 in ascending order, with no digit appearing more than once. This is the current state of the game. The integers $\boldsymbol{a}$ and $\boldsymbol{b}$ represent your current throw of the dice.

Output

Output two lines, each with two parts. First, output the digits that you eliminate with your throw of the dice, as a string of digits in ascending order. If you cannot eliminate any digits, output -1. Then, output the expected score as a real number rounded to five decimal places. Output a space between the parts.

The first line represents the best result when minimizing your score and the second line represents the best result when maximizing your score. Note that it is impossible for two different combinations of digits to yield the same expected score.

Sample Input	Sample Output
`1345 1 1`	`-1 1345.00000` `-1 1345.00000`

`12349 3 1`	`13 151.70370`
	`4 401.24546`