1101 To Add or to Multiply

The Industrial Computer Processor Company offers very fast, special purpose processing units tailored to customer needs. Processors of the a-C-m family (such as the 1-C-2 and the 5-C-3) have an instruction set with only two different operations:

A add a
M multiply by m

The processor receives an integer, executes a sequence of A and M operations (the program) that modifies the input, and outputs the result. For example, the 1-C-2 processor executing the program AAAM with the input 2 yields the output 10 (the computation is $2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 10$), while the 5-C-3 processor yields 51 with the same program and input ($2 \rightarrow 7 \rightarrow 12 \rightarrow 17 \rightarrow 51$).

You are an a-C-m programmer assigned to a top secret project. This means that you have not been told the precise computation your program should perform. But you are given particular values p, q, r, and s and the following conditions:

1. The input is guaranteed to be a number between p and q.

2. The output must be some number between r and s.

Given an a-C-m processor and the numbers p, q, r, and s, your job is to construct the shortest a-C-m program which, for every input x such that $p \leq x \leq q$, yields some output y such that $r \leq y \leq s$. If there is more than one program of minimum length, choose the one that come first lexicographically, treating each program as a string of A's and M's.

Input

The input contains several test cases. Each test case is given by a line with the six integers a, m, p, q, r, and s as described above ($1 \leq a, m, p, q, r, s \leq 10^9$, $p \leq q$ and $r \leq s$).
The last test case is followed by a line with six zeros.

Output

For each test case, display its case number followed by the best program as described above. Display the word "empty" if the best program uses no operations. Display the word 'impossible' if there is no program meeting the specifications.
Display the program as a sequence of space-separated strings, alternating between strings of the form 'nA' and strings of the form 'nM', where $n > 0$. Strings of the former type indicate n consecutive 'A' operations, and strings of the latter type indicate n consecutive 'M' operations.
Follow the format of the sample output.

Sample Input

```
1 2 2 3 10 20
1 3 2 3 22 33
3 2 2 3 4 5
5 3 2 3 2 3
0 0 0 0 0 0
```

Sample Output

```
Case 1: 1A 2M
Case 2: 1M 2A 1M
Case 3: impossible
Case 4: empty
```