Reverse

Consider a Two-Operation Machine (TOM for short) with nine registers, numbered $1 \ldots 9$. Each register stores a non-negative integer in the range $0 \cdots 1000$. The machine has two operations:

$S\ i\ j$	Store one plus the value of register i into register j. Note that i may equal j
$P\ i$	Print the value of register i.

A TOM program includes a set of initial values for the registers and a sequence of operations. Given an integer $N (0 \leq N \leq 255)$, generate a TOM program that prints the decreasing sequence of integers $N, N-1, N-2, \ldots 0$. The maximum number of consecutive S-operations should be as small as possible.

Example of a TOM program and its execution for $N = 2$:

Operation	New Regiter Values	Printed
	1 2 3 4 5 6 7 8 9	Value
Initial values	0 2 0 0 0 0 0 0 0	
P 2	0 2 0 0 0 0 0 0 0	2
S 1 3	0 2 1 0 0 0 0 0 0	
P 3	0 2 1 0 0 0 0 0 0	1
P 1	0 2 1 0 0 0 0 0 0	0

Input

There are several test cases. The first line of a test case contains K, an integer specifying the input case number. The second line of input contains N.

Ouput

The first line of output, for each test case, should be the string "FILE reverse K", where K is the case number. The second line of output should contain nine space-separated values representing the desired initial values of the registers, in order (register 1, then register 2, etc.). The rest of the output file should contain the ordered list of operations to be performed, one per line. Thus, the third line contains the first operation to perform, and so on. The last line of the file should be the one that prints 0. Each line should be a valid operation. The instructions should be formatted as in the example output.

Example

```
Input:
1
2
```

```
Output:
FILE reverse 1
0 2 0 0 0 0 0 0 0
P 2
S 1 3
P 3
P 1
FILE reverse 1
0 2 1 0 0 0 0 0 0
P 2
P 3
P 1
```