**Problem 5 – Bit Lock**

Your task is to write a program that tests a new kind of security lock which uses bitwise operations. The lock itself can be represented as a table with 8 rows and 12 columns (see example below), where each cell contains a single bit (**0** or **1**).

You will be given 8 integers (representing the rows of the table) on a single line, separated by a single space.

Afterwards, you will be given a **series of commands, between 1 and 30**, which will be in one of the following three formats:

* **"check [col]"**, where **[col] is a number**. Upon receiving this command you'll need to check how many 1 bits there are in column [col] and print their amount on the console.
* **"end"** denotes the end of input. Upon receiving this command you need to print all rows of the table (as numbers) on a single line, separated by a single space; print a space after the last number as well.
* **"[row] [direction] [rotations]"**, where [row] is a number; [direction] is a string, either "left" or "right"; and [rotations] is also a number. Upon receiving this command, you need to roll the bits at the specified row. Rolling once to the left means that all bits are moved once to the left, the bit at column 11 goes to column 0. Rolling once to the right means all bits are moved once to the right, the bit at column 0 goes to column 11. The **number of rotations** shows how many times you have to roll the bits on the specified row; it will be **between 0 and 360 inclusive.**

**Input**

The input data is read from the console.

* On the first line you will be given **8 integers, separated by a single space**.
* On the next lines you will be given **commands in the described formats**; the input ends when you receive the command "**end**".

**Output**

The output data must be printed on the console.

* Upon receiving the command "**check [col]**" you need to print a single line containing a number.
* Upon receiving "**end**" print the rows of the table as integers - on a single line, separated by a single space.

**Constraints**

* The numbers representing the rows of the table will be integers in the range [0…4095].
* **[row]** will be between [0…7] and [col] will be between [0…11], i.e. valid coordinates of the table.
* Time limit: 0.25 seconds. Allowed memory: 16 MB.

**Examples**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 0 3423 55 23 11 454 555 1234  check 4  5 right 2  end | 4  0 3423 55 23 11 2161 555 1234 |  | 3 7 4095 900 9 121 256 3222  check 6  check 11  0 right 13  2 left 3  check 2  end | 2  2  4  2049 7 4095 900 9 121 256 3222 |

Note: Tables represent the first example.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |  |  |  | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  |  |  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 3423 |  |  |  | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 3423 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 55 |  |  |  | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 55 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 23 |  |  |  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 23 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 11 |  |  |  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 11 |
| 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 454 |  |  |  | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 2161 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 555 |  |  |  | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 555 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1234 |  |  |  | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1234 |