Draupnir

Problem

Odin has some magical rings which produce copies of themselves. Each "X-day ring" produces one more X-day ring every X days after the day it came into existence. These rings come in six possible varieties: 1-day, 2-day, ..., all the way up to 6-day.

For example, a 3-day ring that came into existence on day 0 would do nothing until day 3, when it would produce another 3-day ring. Then, on day 6, each of those two rings would produce another 3-day ring, and so on.

You know that Odin had no rings before day 0. On day 0, some rings came into existence. At the end of day 0, Odin had R_i i-day rings, for each $1 \le i \le 6$. You know that $0 \le R_i \le 100$, for all i, and at least one of the R_i values is positive.

Fortunately, you also have access to the secret well of knowledge. Each time you use it, you can find out the *total* number of rings that Odin had at the end of a particular day between day 1 and day 500, inclusive. The well will give you the answer modulo 2^{63}, because even it can only hold so much information! Moreover, you can only use the well up to **W** times.

Your goal is to determine how many rings of each type Odin had at the end of day 0 — that is, you must find each of the R_i values.

Input and output

This is an interactive problem. You should make sure you have read the information in the [Interactive Problems section](#) of our FAQ.

Initially, your program should read a single line containing two integers **T**, the number of test cases, and **W**, the number of times you are allowed to use the well of knowledge per test case. Then, you need to process **T** test cases.

In each test case, your program processes up to **W** + 1 exchanges with our judge. You may make up to **W** exchanges of the following form:

- Your program outputs one line with a single integer D between 1 and 500, inclusive.
- The judge responds with one line with a single integer: the total number of rings that Odin had at the end of day D, modulo 2^{63}. If you send invalid data (e.g., a number out of range, or a malformed line), the judge instead responds with -1.

After between 0 and **W** exchanges as explained above, you must perform one more exchange of the following form:

- Your program outputs one line with six integers R_1, R_2, R_3, R_4, R_5, R_6, where R_i represents the number of i-day rings that Odin had at the end of day 0.
- The judge responds with one line with a single integer: 1 if your answer is correct, and -1 if it is not (or you have provided a malformed line).

After the judge sends -1 to your input stream (because of either invalid data or an incorrect answer), it will not send any other output. If your program continues to wait for the judge after receiving -1, your program will time out, resulting in a Time Limit Exceeded error. Notice that it is your responsibility to have your program exit in time to receive a Wrong Answer judgment instead of a Time Limit Exceeded error. As usual, if the memory limit is exceeded, or your program gets a runtime error, you will receive the appropriate judgment.

Limits

1 ≤ **T** ≤ 50.
Time limit: 20 seconds per test set.
Memory limit: 1GB.

Test set 1 (Visible)

W = 6.

Test set 2 (Hidden)

W = 2.

Testing Tool

You can use this testing tool to test locally or on our platform. To test locally, you will need to run the tool in parallel with your code; you can use our interactive runner for that. For more information, read the instructions in comments in that file, and also check out the Interactive Problems section of the FAQ.

Instructions for the testing tool are included in comments within the tool. We encourage you to add your own test cases. Please be advised that although the testing tool is intended to simulate the judging system, it is **NOT** the real judging system and might behave differently. If your code passes the testing tool but fails the real judge, please check the Coding section of the FAQ to make sure that you are using the same compiler as us.

Download testing tool

Sample Interaction

This interaction corresponds to Test set 1. Suppose that, unbeknownst to us, the judge has decided that Odin had one ring of each of the six types at the end of day 0.

```
t, w = readline_int_list()   // Reads 50 into t and 6 into w
printline 3 to stdout         // Asks about day 3.
flush stdout
n = readline_int()            // Reads 15 into n.
printline 1 to stdout         // Asks about day 1.
flush stdout
n = readline_int()            // Reads 7 into n.
printline 1 1 1 3 0 0 to stdout
flush stdout                  // We make a guess even though we could have
                              // queried the well up to four more times.
verdict = readline_int()      // Reads -1 into verdict (judge has decided our
                              //   solution is incorrect)
exit                          // Exits to avoid an ambiguous TLE error
```

Notice that even though the guess was consistent with the information we received from the judge, we were still wrong because we did not find the correct values.