

Power Arrangers

Problem

Go, go, Power Arrangers! Everyone loves this team of five superhero high school students who wear the letters A, B, C, D, and E. When they stand side by side to confront evil monsters, they arrange their team in one of 120 possible different left-to-right orders, giving them various different tactical superpowers. They are even more popular than the Teenage Permutant Ninja Turtles!

Some critics of the show claim that the team only has its arrangement gimmick so that the owners of the show can sell 120 separate sets of 5 action figures, each of which features the team in a different left-to-right order, glued to a base so that the set cannot be rearranged. As an avid Power Arrangers fan, you have collected 119 of these sets, but you do not remember which set you are missing. Your 119 sets are lined up horizontally along a shelf, such that there are a total of $119 \times 5 = 595$ action figures in left-to-right order. You do not remember how the sets are arranged, but you know that the permutation of the sets is selected uniformly at random from all possible permutations, and independently for each case.

You do not want to waste any time figuring out which set you are missing, so you plan to look at the letters on at most **F** figures on the shelf. For instance, you might choose to look at the letter on the eighth figure from the left, which would be the third figure from the left in the second set from the left. When looking at a figure, you only get the letter from that one figure; the letters are hard to see, and the different team members look very similar otherwise!

After checking at most **F** figures, you must figure out which of the sets is missing, so you can complete your collection and be ready to face any possible evil threat!

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 **F**, the number of figures you are allowed to inspect per test case. Then, you need to process **T** test cases.

Within each test case, the missing set of figures is chosen uniformly at random from all possible sets, and the order of the remaining sets is chosen uniformly at random from all possible orders as well. Every choice is made independently of all other choices and of your inputs.

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

- Your program outputs one line containing a single integer between 1 and 595, inclusive, indicating which figure (in left-to-right order along the shelf) you wish to look at. As a further example, 589 would represent the fourth figure from the left in the second set from the right.
- The judge responds with one line containing a single uppercase letter A, B, C, D, or E, indicating the letter on that figure. If you sent invalid data (e.g., a number out of range, or a malformed line), the judge will instead respond with one line containing the single uppercase letter N.

Then, after you have made as many of the **F** exchanges above as you want, you must make one more exchange of the following form:

- Your program outputs one line containing a single string of five uppercase letters: the permutation corresponding to the missing set (e.g., CADBE).
- The judge responds with one line containing a single uppercase letter: Y if your answer was correct, and N if it was not (or you provided a malformed line). If you receive Y, you should begin the next test case, or stop sending input if there are no more test cases.

After the judge sends N 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 N , 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

$1 \leq T \leq 50$.

Time limit: 40 seconds per test set.

Memory limit: 1GB.

The missing set, and the order of the remaining sets, are chosen uniformly and independently at random.

Test set 1 (Visible)

$F = 475$.

Test set 2 (Hidden)

$F = 150$.

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.

```
t, f = readline_int_list()    // Reads 50 into t and 475 into f
println 10 to stdout          // Looks at the last figure in the second set
                               // from the left

flush stdout
n = readline_string()         // Reads B into n. Ooh, team member B! They may
                               // not have the leadership ability of A, or the
                               // technical skill of C, but they entertain the
                               // team with clever quips!

println 11 to stdout          // Looks at the first figure in the third set
                               // from the left

flush stdout
n = readline_string()         // Reads B into n. Notice that B is at the start
                               // of the third set, whereas they were at the
                               // end of the second set.

println 14 to stdout          // Looks at the fourth figure in the third set
                               // from the left

flush stdout
n = readline_string()         // Reads D into n. Silent and brooding, team
                               // member D nonetheless fights fiercely to
                               // protect their friends... and the world!

println ABCDE to stdout       // We foolishly make a wild guess even though we
```

```
flush stdout // could have looked at up to 472 more figures.
verdict = readline_string() // Reads N into verdict (judge has decided our
                             // solution is incorrect)
exit // Exits to avoid an ambiguous TLE error
```