Problem G. Find a Better String

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	1024 mebibytes

This is an interactive problem. You have to use `flush` operation right after printing each line. For example, in C++ you should use the function `fflush(stdout)` or `cout.flush()`, in Java or Kotlin — `System.out.flush()`, and in Python — `sys.stdout.flush()`.

For a binary string s (a string consisting of the characters `0` and/or `1`) of length n, we define its *cost* as the number of integers x meeting the following constraint:

- there exist two integers i, j such that $1 \le i, j \le n$, $i + j = x$ and $s_i \ne s_j$.

For example, the cost of the string `11001` is 5: the suitable values of x are 4, 5, 6, 8, and 9.

The jury has a binary string b consisting of n characters. You are given the value of n, but you don't know b. Your goal is to find another string b' with **strictly greater cost than** b and the same length, or report that no such strings exist.

Before giving the answer, you can ask no more than 3 queries of the following form:

- `?` l r — ask the jury program to tell you the number of distinct characters between $b_l, b_{l+1}, \ldots, b_r$.

Giving the answer does not count as a query.

In every test case, the string b is fixed beforehand. In other words, the interactor is not adaptive.

Interaction Protocol

Initially, the jury program prints one line containing a single integer t ($1 \le t \le 500$) — the number of test cases.

Processing a test case starts with the jury program printing one line containing a single integer n ($3 \le n \le 100$) — the size of b.

To ask a query, you should print a line in the following format:

- `?` l r ($1 \le l \le r \le n$)

In response, the jury program will print one line containing a single integer — the number of distinct characters between $b_l, b_{l+1}, \ldots, b_r$. **If your query is incorrect or you have exceeded the number of queries you may ask, this integer will be** 0. **After receiving** 0 **as the answer, your program must terminate immediately, otherwise the verdict of your submission might be undefined.**

To give the answer, you should print a line in one of the two following formats:

- `NO` (if there is no string b' with the same length as b and strictly greater cost);
- `YES` b' (if at least one such string exists. You may choose any suitable string b').

After you give the answer, the jury program prints one line containing a single integer, which is:

- 0 if your answer is incorrect (in this case, **your program must terminate immediately, otherwise the verdict of your submission might be undefined**);
- 1 if your answer is correct (in this case, you may proceed to the next test case, or terminate the program if it was the last test case).

After printing anything, do not forget to flush the output buffer. Otherwise, you may get the `Idleness Limit Exceeded` verdict.

Example

standard input	standard output
3 // the number of test cases 3 // n = 3 1 // 1 distinct character in b[1..3] 1 // answer is correct 3 // n = 3 2 // 2 distinct characters in b[1..3] 1 // 1 distinct character in b[1..2] 1 // answer is correct 5 // n = 5 2 // 2 distinct characters in b[1..5] 1 // 1 distinct character in b[1..4] 1 // answer is correct	 ? 1 3 // asking about b[1..3] YES 010 // string 010 is better ? 1 3 // asking about b[1..3] ? 1 2 // asking about b[1..2] NO // there is no better string ? 1 5 // asking about b[1..5] ? 1 4 // asking about b[1..4] YES 11001 // string 11001 is better

Note

The example contains one possible way of interaction in a test where $t = 3$, and the strings in the test cases are `000`, `110` and `11110`, respectively. This test is the actual first test in the problem. Note that everything after the `//` sign is a comment that explains which line means what in the interaction. **The jury program won't print these comments in the actual problem, and you shouldn't print them**. The empty lines are also added for your convenience, **the jury program won't print them, and your solution should not print any empty lines**.