the 44th annual World Finals Moscow MIPT

icpc International Collegiate Programming Contest

Addendum to Judging Notes: Interactive Problems

What is an interactive problem?

In many respects, interactive problems are like any other problem – your program will read from standard input and print results to standard output. However, in an interactive problem, standard input and output are connected to another (judge) program, with which you have to communicate back and forth.

Output buffering

In most programming environments, program output is buffered to speed up I/O operations. With interactive problems, it is crucial to make sure the output is actually sent from your program and not simply stored in internal buffers. This typically means flushing the output buffers after each write.

- In C (or C++ using cstdio), you can use `fflush(stdout)`.
- A C++ output stream is flushed automatically each time you write the `endl` manipulator. When using other means or if you want to be sure, call `cout.flush()`.
- In Java and Kotlin, the `System.out` stream has so-called "auto-flush" functionality and its buffer is therefore flushed automatically with each newline character. When using other streams or if you want to be sure, invoke the `flush()` method of the stream.
- In Python, you can use `sys.stdout.flush()`.

Judging of interactive problems

Interactive problems are judged in a way similar to other problems, but there are some differences:

- When it attempts to read data, your program will wait until more data are available or until the judge program terminates the input (unless you read in a non-blocking way, which is beyond the scope of these notes). So if your program attempts to read more input than can currently be provided (e.g., because you forgot to flush your previous output, or because of some other reason), then the program will stall indefinitely and your submission will get Time Limit Exceeded.
- As usual, the judgement given to an incorrect submission is the first error discovered, but this does not always mean exactly the same thing as for traditional problems. For instance, if your submission to a traditional problem is too slow on a particular test case, you would get Time Limit Exceeded on that test case. With an interactive problem, the judgement may be Wrong Answer, if the solution exhibits an incorrect answer before it runs out of time.

The time limit for an interactive problem is how much time *your submission* may spend; the time spent by the judge program is not counted towards this.

Note that the judge program may behave in an adversarial way and adapt the input provided to your program based on your previous output, with the intent to discover errors in your algorithm.

Test support

For some problems, the judges may provide a simple testing tool simulating the evaluation. If this is the case, you will find such a tool where you normally see the sample data. Executing the tool with a "`-h`" option should describe how to use the tool. Of course, the testing tool will only implement some test scenarios and only some functionality of the real judge program.