Advice, Hints, and General Information from the NWERC Jury

- Your submissions will be run multiple times, on different input files. The order in which the input files are being used is fixed and does not change from one submission to another.

  If your submission is incorrect, the error message you get will be the error that occurred on the first input file on which you failed. E.g. if your submission is prone to crash but also incorrect, your submission may be judged as either "Wrong Answer" or "Run Time Error", depending on which is discovered first.
- Your submission will have exactly one processor core fully at its disposal while running. You should not attempt to use multithreading in your submissions. Attempts to start extra processes or threads may lead to a judgement of "Run Time Error".
- After receiving the judgement of your submission, you are able to view the output of your submission and the corresponding verdict of each of the samples in the DOMjudge interface.
- Each problem will have a stated time limit. This time limit is per input file and refers to CPU time, which includes time spent on slow parsing (e.g. using `Scanner` in Java).
- All judging machines are physically identical to the desktop PCs provided to you during the contest.
- The memory limit is the same for all problems and is 2048 MB.
- For problems with large I/O, it is in your best interest to use buffered input and output in Java. We have tried to make sure that the I/O-intensive problems are solvable using even slow I/O. But for some problems, it can be more difficult to get accepted using e.g. the `Scanner` class, since it consumes a significant amount of valuable CPU time that could otherwise be used by your algorithm. In most cases, a `BufferedReader` will be more efficient. Also consider explicit output buffering (e.g. by using the `BufferedWriter` class) to avoid flushing every line of output separately.
- We guarantee that every problem is solvable in C++, Python 3, Java, and Kotlin.

Problem Statements

We generally try to make the input and output formats as easy as possible to follow for you as contestants – input and output are meant to be the easiest parts of the problems!

- For input, multiple numbers or words on one input line will be separated by single spaces. There are no additional spaces, tabs or newlines.

- For output, unless explicitly stated in the problem statement, we do not require you to output exactly the right number of spaces and newlines. Errors in whitespace within reason will be accepted, but if all whitespace between two tokens is omitted, that is (of course) an error.
- For problems with real-valued output, we generally only require that your output is correct up to a certain absolute or relative error. In the case that both are specified, the largest of the two applies. For example, if the problem statement requests an "absolute or relative error of at most 10^{-6}", this means that:
  - If the correct answer is 0.005, any answer between 0.004999 and 0.005001 will be accepted. The absolute error of $\pm 10^{-6}$ is larger than the relative error of $\pm 5 \cdot 10^{-9}$ $(= 0.005 \cdot 10^{-6})$.
  - If the correct answer is 5000, any answer between 4999.995 and 5000.005 will be accepted. The relative error of ± 0.005 $(= 5000 \cdot 10^{-6})$ is larger than the absolute error of $\pm 10^{-6}$.
- For problems with real-valued output, any reasonable format for floating point numbers is acceptable. For instance, `17.000000`, `0.17e2`, and `17` are all acceptable ways of formatting the number 17. For the definition of reasonable, please use your common sense.
- Some problems will state: "If there are multiple valid solutions, you may output any one of them." It may be the case that, on the public sample inputs, your submission provides a different correct output than the corresponding sample output. In that case, you should manually check whether your output conforms to the given output constraints.

Interactive Problems

- The contest typically includes interactive problems. Each interactive problem is accompanied by a Python script to help with testing your solutions locally. The tool attempts to detect and report common errors, but it is not guaranteed that a program that passes the testing tool will be accepted. The tool can be found in the Kattis or DOMjudge interface, in the panel below the statement of the corresponding problem, named `testing_tool.py`. The source code of the tool contains instructions on its use. You will be able to test this during the practice session.
- In solutions to interactive problems, make sure to flush after every write to standard output. Failing to do so may result in the interactor waiting for input indefinitely, and your solution being judged as "Time Limit Exceeded".
- Verdicts for incorrect submissions to interactive problems may not be deterministic. We can guarantee the following: a verdict of "Wrong Answer" means that your submission printed something wrong, and a verdict of "Run Time Error" means that your submission returned a non-zero exit code. If your submission does both, you may get either verdict.