Judging Details

Judge System

Your programs will be judged on the system once they're submitted.

- Your program must read input data from the standard input, and write its output to the standard output.
- Other outputs, e.g. writing to the standard error, will not be used for judging.
- You will never have to write to (open) a file, and are not allowed to do so.

Your programs will be run inside a *sandboxed environment*, i.e. with protections to prevent the system from being damaged. Specifically:

- Memory usage is limited to 2 GB in the environment. Note it is the total amount, not the amount you can use exclusively in your programs.
- The stack size is set unlimited (in C/C++), only capped by the total memory limit.
- Multi-processing or multi-threading is discouraged and unlikely beneficial, though not prohibited. Remember your programs will run on a single processor core. The total number of processes is limited to 64, including ones the system may create outside your programs.
- It is *never* recommended to run external commands. It is technically possible but probably does not work as you expect.

If you have no idea about what these mean — no worries. Just remember your programs should use the standard input and output, not files.

There are a couple more restrictions that apply:

- The total amount of source code must not exceed 256 KB in each submission.
- Your program must compile within 30 seconds.

See the DOMjudge team manual for more details about these restrictions.

Note about Platform

The judge system is running on Google Compute Engine, C2 machine type (`c2-standard-4`). For more information about Google Compute Engine, please visit the official website[*1].

1. https://cloud.google.com/compute/docs/cpu-platforms

Compilers & Options

The judge system uses the following compilers and execution environments (e.g., interpreters) with the following options. `"$@"` is substituted with your source file(s); `"$DEST"` is the name of the binary (which is `./a.out` by default) and is chosen arbitrarily by the system.

The **Run** commands indicated in the following table are for non-interactive problems. For interactive problems, standard input and output are connected to a judge program. See the "Note on Interactive Problems" section below for the details.

C	
Version	gcc (Ubuntu 11.3.0-1ubuntu1~22.04.1) 11.3.0
Compile	`gcc -x c -g -O2 -std=gnu11 -static -o "$DEST" "$@" -lm`
Run	`"$DEST" < infile > outfile`
C++	
Version	g++ (Ubuntu 11.3.0-1ubuntu1~22.04.1) 11.3.0
Compile	`g++ -x c++ -g -O2 -std=gnu++20 -static -o "$DEST" "$@"`
Run	`"$DEST" < infile > outfile`
Java	
Version	OpenJDK 17.0.5 2022-10-18 (build 17.0.5+8-Ubuntu-2ubuntu122.04)
Compile	`javac -encoding UTF-8 -sourcepath . -d . "$@"`
Run	`java -Dfile.encoding=UTF-8 -XX:+UseSerialGC -Xss64m -Xms1920m -Xmx1920m MainClass < infile > outfile`
Python 3 (PyPy)	
Version	Python 3.9.15 (7.3.10+dfsg-1~ppa2~ubuntu20.04, Dec 07 2022, 00:17:39) [PyPy 7.3.10 with GCC 9.4.0]
Compile	`pypy3 -m py_compile "$@"`
Run	`pypy3 "$@" < infile > outfile`
Kotlin	
Version	1.7.21 (JRE 17.0.5+8-Ubuntu-2ubuntu122.04)
Compile	`kotlinc -d . "$@"`
Run	`kotlin -Dfile.encoding=UTF-8 -J-XX:+UseSerialGC -J-Xss64m -J-Xms1920m -J-Xmx1920m MainClass < infile > outfile`

In Java and Kotlin, DOMjudge will detect the main class automatically; you do not have to name it `Main`. See the DOMjudge team manual for details.

In Python, **Compile** commands only verify the syntax. `*.pyc` files will not be used in the real run.

The compilers and the execution environments are also available on your workstation as the following commands:

- **C** — `compilegcc` (no `run` command)
- **C++** — `compileg++` (no `run` command)
- **Java** — `compilejava` / `runjava`
- **Python 3** — `compilepython3` / `runpython3`
- **Kotlin** — `compilekotlin` / `runkotlin`

Submission Results

The judges may have prepared multiple test cases for each problem. On each submission, DOMjudge decides one result for each test case. DOMjudge does *not* report results for each test case, but it reports one result for a submission, based on the following rules.

Results for test cases

For each test case, DOMjudge decides one of the following results:

- **CORRECT** - Your program ran successfully and passed the test case.
- **TIMELIMIT** — Your program did not finish within the time limit.
- **RUN-ERROR** — Your program crashed or exited with a non-zero exit status (e.g. because of missing `return 0;` in C/C++).
- **OUTPUT-LIMIT** — Your program produced excessive output (> 8 MB).
- **WRONG-ANSWER** — Your program neither crashed nor exceeded the time limit, but produced incorrect output.
- **NO-OUTPUT** — Your program did not produce any output.

See the DOMjudge team manual for more details about these results.

Results for submissions

For each submission, DOMjudge reports one of the following results:

Accepted

- **CORRECT** — Your program resulted in **CORRECT** for all test cases.

Rejected with 20-minutes penalty

- **TIMELIMIT**, **RUN-ERROR**, **OUTPUT-LIMIT** — If your program resulted in **TIMELIMIT**, **RUN-ERROR**, or **OUTPUT-LIMIT** for any test case, then that result is returned immediately.
- **WRONG-ANSWER** — Your program didn't result in any of the above three, but resulted in **WRONG-ANSWER** for some case(s).
- **NO-OUTPUT** — Your program didn't result in any of the above four, but resulted in **NO-OUTPUT** for some case(s).

Rejected with no penalty

The following results imply your program did not even start. You do not receive any penalty for these results.

- **COMPILE-ERROR** — Your program did not compile in the judging environment. You can consult the error message(s) on the submission details page.
- **TOO-LATE** — Your program was submitted after the contest was over. [*2]

Note on Interactive Problems

You may meet “interactive problems” in the contest. They are the same as other problems in a way that your program will read from standard input and print results to standard output. The difference is, the standard input and output are connected to a special program (judge program), with which you have to communicate back and forth. Unlike other problems where the input text is fixed for each test case, the input varies based on your previous outputs.

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/C++ with `stdio.h` (or `cstdio`), you can use `fflush(stdout)`. Writing `\n` does not mean it will get flushed.
- In C++ with `iostream`, an output stream is flushed automatically each time you write the `std::endl` manipulator. When using other means or if you want to be sure, call `std::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()`.

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 if your program attempts to read more input than can be provided currently (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 **TIMELIMIT**.

Note on Languages

The judges have solved all problems in languages from at least two of the three distinct language groups (Java/Kotlin, C/C++, and Python).

Note to Python Users

Only syntax errors will be reported as **COMPILE-ERROR**. Other types of errors, such as `NameError` or `ModuleNotFoundError`, will result in **RUN-ERROR** and incur a 20-minute penalty.

It is fine, though not needed, to start your scripts with an interpreter directive (line starting with `#!`, also known as shebang). [*3]

The full list of modules available in the judge system can be found in the following section.

---

2. Note that this does not mean your programs need to be judged before the end of the contest. Your programs will be judged as long as submitted (“queued”) within the contest time.
3. Some past versions of DOMjudge refused scripts that contain a shebang.

Available Python Modules

```
LanguageSelector    _sitebuiltins       formatter           pymacaroons
PIL                 _socket             fractions           pyparsing
__decimal           _sqlite3            ftplib              pypy_tools
__exceptions__      _sqlite3_build      functools           pypyjit
__future__          _sqlite3_cffi       future_builtins     pyrepl
__pypy__            _sre                gc                  pyrfc3339
_abc                _ssl                genericpath         pytz
_aix_support        _ssl_build          getopt              queue
_ast                _string             getpass             quopri
_audioop_build      _strptime           gettext             random
_audioop_cffi       _struct             gi                  re
_blake2             _structseq          glob                readline
_bootlocale         _sysconfigdata      graphlib            reportlab
_bootsubprocess     _syslog_build       greenlet            reprlib
_bz2                _syslog_cffi        grp                 requests
_cffi_backend       _testcapi           gzip                resource
_cffi_ssl           _testing            hashlib             rlcompleter
_codecs             _testmultiphase     heapq               runpy
_codecs_cn          _thread             hmac                sched
_codecs_hk          _threading_local    html                secrets
_codecs_iso2022     _vmprof             http                secretstorage
_codecs_jp          _warnings           httplib2            select
_codecs_kr          _weakref            identity_dict       selectors
_codecs_tw          _weakrefset         idlelib             setuptools
_collections        _winapi             idna                shelve
_collections_abc    _yaml               imaplib             shlex
_compat_pickle      abc                 imghdr              shutil
_compression        aifc                imp                 signal
_contextvars        antigravity         importlib           site
_continuation       apport              importlib_metadata  six
_cppyy              apport_python_hook  inspect             smtpd
_crypt              apt                 io                  smtplib
_csv                aptdaemon           ipaddress           sndhdr
_ctypes             aptsources          itertools           socket
_ctypes_test        argparse            jeepney             socketserver
_curses             array               json                sqlite3
_curses_build       ast                 jwt                 sre_compile
_curses_cffi        asynchat            keyring             sre_constants
_curses_panel       asyncio             keyword             sre_parse
_dbm                asyncore            language_support_pkgs ssl
_decimal_build      atexit              launchpadlib        stackless
_distutils_hack     audioop             lib2to3             stat
_distutils_system_mod base64              linecache           statistics
_ffi                bdb                 locale              string
_gdbm               binascii            logging             stringprep
_gdbm_build         binhex              lsb_release         struct
_gdbm_cffi          bisect              lzma                subprocess
_hashlib            blinker             macaroonbakery      sunau
_hpy_universal      builtins            macpath             symbol
_immutables_map     bz2                 macurl2path         symtable
_imp                cProfile            magic               sys
_io                 cairo               mailbox             sysconfig
_jitlog             calendar            mailcap             syslog
_ldb_text           certifi             marshal             tabnanny
_locale             cffi                math                tarfile
_lsprof             cgi                 mimetypes           telnetlib
_lzma               cgitb               mmap                tempfile
_lzma_build         chardet             modulefinder        termios
_lzma_cffi          chunk               more_itertools      test
_markupbase         cmath               msilib              textwrap
```

```
_marshal            cmd                 msvcrt              this
_md5                code                multiprocessing     threading
_minimal_curses     codecs              nacl                time
_multibytecodec     codeop              netrc               timeit
_multiprocessing    collections         nntplib             tkinter
_opcode             colorsys            ntpath              token
_operator           compileall          nturl2path          tokenize
_osx_support        concurrent          numbers             tputil
_overlapped         configparser        oauthlib            trace
_pickle_support     contextlib          olefile             traceback
_posixshmem         contextvars         opcode              tracemalloc
_posixshmem_build   copy                operator            tty
_posixshmem_cffi    copyreg             optparse            turtle
_posixsubprocess    cpyext              os                  turtledemo
_pwdgrp_build       crypt               parser              types
_pwdgrp_cffi        cryptography        pathlib             typing
_py_abc             csv                 pdb                 uaclient
_pydecimal          ctypes              pexpect             ufw
_pyio               ctypes_support      pickle              unicodedata
_pypy_generic_alias cupshelpers         pickletools         unittest
_pypy_interact      curses              pipes               urllib
_pypy_irc_topic     dataclasses         pkg_resources       urllib3
_pypy_openssl       datetime            pkgutil             uu
_pypy_testcapi      dbm                 platform            uuid
_pypy_util_build    dbus                plistlib            venv
_pypy_util_cffi     decimal             poplib              wadllib
_pypy_util_cffi_inner defer               posix               warnings
_pypy_wait          difflib             posixpath           wave
_pypy_winbase_build dis                 pprint              weakref
_pypy_winbase_cffi  distro              problem_report      webbrowser
_pypy_winbase_cffi64 distutils           profile             wsgiref
_pypyjson           doctest             pstats              xdrlib
_random             email               psutil              xml
_rawffi             encodings           pty                 xmlrpc
_resource_build     ensurepip           ptyprocess          yaml
_resource_cffi      enum                pwd                 zipapp
_scproxy            errno               py_compile          zipfile
_sha1               faulthandler        pyclbr              zipimport
_sha256             fcntl               pydoc               zipp
_sha3               filecmp             pydoc_data          zlib
_sha512             fileinput           pyexpat             zoneinfo
_signal             fnmatch             pygtkcompat
```