ICPC International Collegiate Programming Contest // 2024-2025
The 2024 ICPC Pacific Northwest Regional Contest

JETBRAINS
icpc global sponsor programming tools
upsilon pi epsilon honor society

TECHNICAL NOTES

Subject to Change
Updated: November 1, 2024

This document contains important information related to the environment for the Pacific Northwest Regional Contest. It is important that your team read and understand all the information below.

All Programs:

- The languages allowed in the contest are C, C++, Java, Kotlin, and Python 3.

- There is a limit of 65,535 bytes on the length of files submitted for judging.

- Your program must read its input from "standard input".

- Your program should send its output to "standard output". Your program may also send output to "standard error", but only output sent to "standard output" will be considered during judging.

- All program source code files and/or test data files which you create must be located in or beneath your "home directory". Your home directory will normally be named "**/home/team**" – but this may vary between sites so check with your site coordinator. You may create subdirectories beneath your home directory.

- If your program exits with a non-zero exit code, it will be judged as a Run Time Error.

- Each problem will specify a maximum CPU time limit your submission can use for a single test case. If the CPU time limit is exceeded, it will be judged as Time Limit Exceeded.

- Each problem will specify a maximum RAM (memory) limit your submission can use for a single test case. If the RAM limit is exceeded, it will be judged as Memory Limit Exceeded.

C/C++ Programs:

- Use the filename extension "`.cpp`" for C++ program files (extensions `.cc`, `.cxx,` and `.c++` can also be used). Use the filename extension "`.c`" for C program files.

Java Programs:

- **Do not** use *package* statements (that is, your solution should reside in the "default package".) Use the filename extension "**.java**" for all Java source files.

Kotlin Programs:

- **Do not** use *package* statements (that is, your solution should reside in the "default package"). Use the filename extension "**.kt**" for all Kotlin source files. Be sure to capitalize (at least) the first letter of the filename of your Kotlin source file.

Python Programs:

- Only Python 3 (but not Python 2) is supported (see information below about which version of Python 3 is supported). Use the filename extension "**.py**" for all Python source files.

- Python programs will be "syntax checked" when submitted; programs which fail the syntax check will receive a "Compilation Error" judgment response (for which no penalty currently applies, just as with C/C++/Java/Kotlin programs which fail to compile). See the sections below for information on how to perform a syntax check yourself in the same way as will be done by the Judges.

Note: in the following sections, the notation "**${files}**" means "the list of file names passed to the corresponding script as arguments". We *strongly* encourage you to compile your programs using the commands below, since these commands will use the same compiler options as the judging systems. Note that on the judging systems, your submission will be "*pinned*" to a single CPU (using the `taskset` command) - keep this in mind if you receive Time Limit Exceeded errors.

Command-Line Usage for C/C++ :

- To compile a C or C++ program from a command line, type the command

  compilegcc   progname.c   (for C programs) or

  compileg++   progname.cpp   (for C++ programs)

  where **progname.c** or **progname.cpp** is the name of your source code file.

  The **compilegcc** command is a script which invokes the GNU GCC compiler with the same options as those used by the Judges:
  ```
  -x c -g -O2 -std=gnu11 –static ${files} -lm
  ```

  The **compileg++** command is a script which invokes the GNU G++ compiler with the same options as those used by the Judges:
  ```
  -x c++ -g -O2 -std=gnu++20 –static ${files}
  ```

- To execute a C/C++ program after compiling it as above, type the command
  ```
  ./a.out
  ```

ICPC International Collegiate Programming Contest // 2024-2025
The 2024 ICPC Pacific Northwest Regional Contest

JETBRAINS
icpc global sponsor
programming tools

upsilon pi epsilon
honor society

Command-Line Usage for Java :

- To compile a Java program from a command line, type the command

      compilejava   Progname.java

  where **Progname.java** is the name of your source code file. This will compile the source code in the file **Progname.java**, and will produce a class file named **Progname.class**.   The **compilejava** command is a script which invokes the **javac** compiler with the same options as those used by the Judges:

      -encoding UTF-8 -sourcepath . -d . ${files}

- To execute a Java program after compiling it, type the command

      runjava   Progname

  where **Progname** is the name of the class containing your **main** method (your source code file name without the filename extension). The **runjava** command is a script which invokes the **java** command with the same options as those used by the Judges:

  -Dfile.encoding=UTF-8 -XX:+UseSerialGC -Xss64m -Xms1920m -Xmx1920m ${mainclass}

Command-Line Usage for Python 3 :

- To "compile" (syntax-check) a Python 3 program from a command line, type the command

      compilepython3   progname.py

  where **progname.py** is the name of your Python 3 source code file. The **compilepython3** command is a script which invokes the **PyPy3** Python 3 interpreter as follows:

      pypy3 -m py_compile ${files}

  which compiles (but does not execute) the specified Python program and displays the result (i.e., whether the compile/syntax-check was successful or not).

- To execute a Python 3 program from a command line, type the command

      runpython3   progname.py

  where **progname.py** is the name of your Python 3 source code file.   The **runpython3** command is a script which invokes the **pypy3** Python 3 interpreter passing to it the specified Python program file.

- Note that the above commands are precisely what the Judges will use to compile and execute Python 3 submissions.

Command-Line Usage for Kotlin :

- To compile a Kotlin program from a command line, type the command

    ```
    compilekotlin  Progname.kt
    ```

    where `Progname.kt` is the name of your Kotlin source code file. The `compilekotlin` command is a script which invokes the `kotlinc` compiler with the same arguments as those used by the Judges:

    ```
    -d . ${files}
    ```

- To execute a Kotlin program from a command line, type the command

    ```
    runkotlin  PrognameKt
    ```

    where `progname.kt` is the name of your Kotlin source code file (note the capitalization and the lack of a period in the `runkotlin` argument.) The `runkotlin` command is a script which invokes the Kotlin JVM with the following options (which are identical to what the Judges will use):

```
-Dfile.encoding=UTF-8 -J-XX:+UseSerialGC -J-Xss64m -J-Xms1920m -J-Xmx1920m ${mainclass}
```

IDEs and Editors

- The following IDEs (Integrated Development Environments) are available on the contest system from the *Applications*->*Development* menu: **CLion**, **Code::Blocks**, **Eclipse**, **IntelliJ IDEA**, **PyCharm, VS Code**.

- The following editors are available on the contest system from the *Applications*->*Development* menu: **Vim**, **Gvim**, **Emacs(GUI)**, **Emacs(Terminal)**, **GEdit**, **Geany**, **Kate**.

Documentation

- Documentation for each available programming language can be found on your machine using the "**Documentation**" bookmark provided on the address bar in the Firebox web browser..

Submissions

- Programs are submitted to the Judges using the *PC²* contest control system *Web Team Interface (WTI)*. The *WTI* can be accessed using the URL provided in the credentials email you received.

Scoreboards

- The current contest scoreboard (standings) can be viewed using the **Scoreboard** menu item on the PC² WTI. A more detailed scoreboard (showing site-specific standings and containing sample problem data for example) can be accessed via the Firefox browser (the Scoreboard URL will be provided prior to the start of the contest). Note that *all* scoreboards will be *frozen* with **one hour** remaining in the contest, or earlier at the discretion of the judges.

Printing

- Teams at all sites will have access to printing. There will be runners who will deliver printed output to your team workstation (teams will not have direct access to the printers).

- Every file you send to the printer MUST contain your team's NAME and TEAM NUMBER in a comment at or near the top of the file. Runners will not deliver printouts that do not identify the team to which the printout belongs.

- Specific additional instructions for printing may vary between sites; ask your site coordinator for additional information regarding printing at your site.

- Print jobs are limited to a few pages long; printing excessively long output will be deemed an activity detrimental to the contest and subject to disqualification.

Sample data and Problem Statements

- Sample data and problem statements for each problem will be accessible in machine-readable form via a link provided in your credential Email. You will be able to read or download individual items or a single .zip file containing the sample data and problem statements for all problems in the contest. You can unzip that file under your home directory where an individual folder for each problem will be created.

Files and Data Storage

- Any files that you create must be stored underneath the home directory of the **team** account (this does not apply to files automatically created by system tools such as editors).

Language and Tool Versions

- The contest image, as well as the judging system uses the following compiler and tool versions. If you are using a version of a compiler that is different from these, we cannot help you diagnose issues. If you are using a system other than the designated contest image, we cannot help diagnose issues. Compile or run-time errors can often be traced back to compiler version differences.

Compiler/Tool	Version
g++ (C++)	13.2.0
gcc (C)	13.2.0
java	21.0.4
kotlin	1.9.24
pypy3	7.3.15 with Python 3.9.18
Eclipse	2024-09 (4.33.0)
Jetbrains CLion	2024.2.2
Jetbrains IntelliJ	2024.2.2
Jetbrains PyCharm	2024.2.3