2024 ICPC World Finals Astana Programming Environment

This page describes current plans for the Programming Environment which will be available to each World Finals team. Please note that ***these plans are subject to change***. See the "Update History" list at the bottom of this page for information on the currency of what is shown here. Additional updates will be posted soon.

Each *team* at the World Finals will be provided with *one workstation*, shared between team members. Note that this is the traditional ICPC setup (not the unique setup of some previous World Finals where each team *member* was provided with a separate workstation). The configuration of each workstation will be as follows:

Hardware

- **System Unit**
  - Lenovo ThinkBook 14 G6, with 16GB RAM, 256GB SSD, and 13th Generation Intel® Core™ i5-13500H processor.
- **External screen**:
  - 1920×1080 Full HD monitor; model to be determined.
- **External keyboard**:
  - Lenovo Essential wired keyboard (black), US English (a keyboard image is posted here)
- **External Mouse**
  - Lenovo Essential wired 3-button mouse with scroll wheel (a mouse image is posted here)
- **Webcam:**

- To Be Determined (the webcam will be used to capture team member images during the entire contest, and may not be redirected or blocked).

No hardware substitutions will be allowed (that is, teams may not bring their own equipment onto the contest floor). This includes that teams may not substitute keyboards or other peripherals; all teams will use identical equipment, as described above, during the contest.

Software

The software configuration for the World Finals will consist of the following:

- **OS**:
  - Ubuntu 22.04.1 LTS Linux (64-bit). A list of packages included in the OS image can be found here.
- **Desktop**:
  - GNOME
- **Editors**
  - vi/vim
  - gvim
  - emacs
  - gedit
  - geany
  - kate
- **Languages**:
  - **Java** (Openjdk version 17.0.5 2022-10-18)
    - compiler flags: -encoding UTF-8 -sourcepath . -d . {files}
    - runtime flags: -Dfile.encoding=UTF-8 -XX:+UseSerialGC -Xss64m -Xms{memlim}m -Xmx{memlim}m
  - **C** (gcc 11.3.0 (Ubuntu 11.3.0-1ubuntu1~22.04))
    - compiler flags: -x c -g -O2 -std=gnu11 -static {files} -lm
  - **C++** (g++ 11.3.0 (Ubuntu 11.3.0-1ubuntu1~22.04))

        - compiler flags: -x c++ -g -O2 -std=gnu++20 -static {files}

    - **Python 3** (PyPy 7.3.10 with GCC 11.3.0 providing python 3.9.15) A list of the installed Python modules is available here.
        - Note that Python 2 is no longer supported at the World Finals.

    - **Kotlin** (Version 1.7.21)
        - compiler flags: -d . {files}
        - runtime flags: -Dfile.encoding=UTF-8 -J-XX:+UseSerialGC -J-Xss64m -J-Xms{memlim}m -J-Xmx{memlim}m

- **IDEs**:
    - **Eclipse** 4.13 (version 2022-12), configured with:
        - Java Development Tooling (JDT) version 3.18.1400.v20221123-1800 using Java as listed above.
        - C++ Development Tooling (CDT) version 11.0.0 using C++ as listed above.
        - PyDev Python Development Tooling version 10.0.2 using Python3 as listed above.

    - **IntelliJ** (IDEA Community Edition, version 2022.3), configured with:
        - Java as listed above
        - Kotlin as listed above

    - **CLion** (version 2022.3), configured with:
        - C/C++ as listed above

    - **Pycharm** Community Edition Python IDE (version 2022.3), configured with:
        - Python 3 as listed above

    - **Code::Blocks** (version 20.03-3.1), configured with:
        - C/C++ as listed above

    - **VS Code**(version 1.74.2 configured with
        - Microsoft C/C++ extension V1.15.4)

- NOTE: the Judges will compile and execute C/C++ programs using C/C++ as listed under **Languages**, above, NOT using Microsoft C/C++.

Compilation of Submissions

During the contest, teams will submit proposed solutions to the contest problems to the Judges using the DOMjudge contest control system (CCS). Source files submitted to the Judges will be compiled using the following command line arguments for the respective language:

- C:

```
gcc -x c -g -O2 -std=gnu11 -static ${files} -lm
```

- C++:

```
g++ -x c++ -g -O2 -std=gnu++20 -static ${files}
```

- Java:

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

- Python 3

```
pypy3 -m py_compile ${files}
```

- Kotlin

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

The "${files}" in the above commands represents the list of source files from the submission which will actually be compiled. Files with the following suffixes (and only files with these suffixes) will be submitted to the compiler:

- For C submissions: files ending with .c
- For C++ submissions: files ending with .cc, .cpp, .cxx, or .c++
- For Java submissions: files ending with .java
- For Python submissions: files ending with .py
- For Kotlin submissions: files ending with .kt

Execution of Submissions

For each language, if the above compilation step is successful then the submission will be executed as follows:

- For C/C++: the executable file generated by the compiler will be executed to generate the output of the submission.
- For Python 3: the main source file will be executed by the PyPy3 Python3 interpreter to generate the output of the submission.
- For Java: the compiled main class will be executed using the following command:

```
java -Dfile.encoding=UTF-8 -XX:+UseSerialGC -Xss64
```

- For Kotlin: the compiled main class will be executed using the following command:

```
kotlin -Dfile.encoding=UTF-8 -J-XX:+UseSerialGC -J
```

Compilation and execution as described above will take place in a "sandbox" on a dedicated judging machine. The judging machine will be as identical as possible to, and at least as powerful as, the machines used by teams. The sandbox will allocate 2GB of memory; the entire program, including its runtime environment, must execute within this memory limit. For interpreted languages (Java, Python, and Kotlin) the runtime environment includes the interpreter (that is, the JVM for Java/Kotlin and the Python interpreter for Python).

The sandbox memory allocation size will be the same for all languages and all contest problems. For Java and Kotlin, the above commands show the stack size and heap size settings which will be used when the program is run in the sandbox.

Building Your Own World Finals Machine

Instructions for building a system identical to what is planned for team use at the World Finals are posted ***HERE***. Please note that *the image created by those instructions is a* ***draft***, subject to changes as we approach the World Finals. See the ***Revised*** list at the bottom of that page to determine the most recent change date.

Reference Materials

The following packages will be available on team machines at the World Finals, and will be installed automatically as part of the steps listed under *Building Your Own World Finals Machine*, above.

- JDK JavaDocs
- C++ STL docs
- DOMjudge Team Guide
- Additional Information:
  - Judging Notes (To Be Updated)
  - Judging Notes Addendum (To Be Updated)
  - Technical Notes (To Be Updated)

We welcome all suggestions and comments. This configuration is subject to change until the final update. All questions about the system configuration should be directed to John Clevenger, ICPC Technical Director (clevenger@icpc.global).

Update History

May 5, 2024 Initial version, cloned from 47th WF Luxor description and updated with known 48th WF details.