Technical Notes

Atlantic Canadian Programming Competition 2026

Please read these notes before the contest. They describe the contest system, the languages and the commands available on your machine.

Contest system

The contest runs on Kattis, at `https://atlantic-canada26.kattis.com`. Firefox opens on the contest page. The bookmarks bar and the Contest menu of the Applications menu lead to the contest, the problems and the standings. Log in with your team's Kattis username or email address and password, then submit your solutions from the page of each problem. External sign-in providers and webmail are unavailable on the contest machines. Ask staff if you cannot use the team's password. The sample input and output of each problem are on its page.

The contest starts on Friday, October 23, 2026 at 13:00 Atlantic time and lasts 5 hours. The clock of your machine is synchronized, so it shows the same time as Kattis.

All programs

- Your program reads its input from standard input and writes its output to standard output. Output sent to standard error is ignored, but a large amount of it can slow your program down.
- A program that ends with a non-zero exit code is judged as a run time error.
- The memory limit and the time limit are given on the page of each problem.

Languages

The languages of the contest are C, C++, Java and Python 3. The compilers on your machine are the same versions as on Kattis.

Language	Version	File extensions
C	gcc-15 15.2.0	`.c`
C++	g++-15 15.2.0	`.cpp`, `.cc`, `.C`, `.cxx`, `.c++`
Java	javac and java 25.0.4.1	`.java`
Python 3	PyPy 7.3.23 (Python 3.11.15)	`.py`, `.py3`

Compiling and running on your machine

You can test your programs any way you like. The commands below use the same compiler options as Kattis, so they are the closest match to the way your submissions are judged. They do not enforce the time and memory limits.

C

Compile `compilec prog.c`
Run `runc < input.txt`

`compilec` runs `gcc-15 -g -O2 -o a.out -std=gnu23 -static prog.c -lm`, which is the command of Kattis. `runc` runs `./a.out`. `compilegcc` is another name for `compilec`.

C++

Compile `compilecpp prog.cpp`
Run `runcpp < input.txt`

`compilecpp` runs the same command as Kattis.
`g++-15 -g -O2 -o a.out -std=gnu++23 -static -lrt -Wl,--whole-archive -lpthread -Wl,--no-whole-archive prog.cpp`
`runcpp` runs `./a.out`. `compileg++` is another name for `compilecpp`.

Java

Compile `compilejava Prog.java`
Run `runjava Prog < input.txt`

`compilejava` runs the same command as Kattis.
`javac -encoding UTF-8 -sourcepath . -cp "./*" -d . Prog.java`
`runjava` takes the name of the class that contains `main` and runs
`java -Dfile.encoding=UTF-8 -XX:+UseSerialGC -Xss64m -XX:TieredStopAtLevel=1 -XX:ActiveProcessorCount=2 -Xms2048m -Xmx2048m -cp ".:./*" Prog`

Kattis sets `-Xms` and `-Xmx` to the memory limit of the problem. `runjava` uses 2048 MB. To test with another limit, for instance 1024 MB, run
`MEMLIM=1024 runjava Prog`

We suggest not using a `package` statement.

Python 3

Check the syntax `compilepython3 prog.py`
Run `runpython3 prog.py < input.txt`

Kattis runs Python 3 programs with PyPy (`pypy3 prog.py`), not with the usual Python interpreter. `runpython3` runs `pypy3 prog.py` in the same way. The `python3` command of the machine is a different interpreter (CPython 3.14) and does not match Kattis. `compilepython3` only checks the syntax (`pypy3 -m py_compile prog.py`).

Running programs from an IDE

The contest compilers and interpreters are already configured in the IDEs below. The actions described here use the language versions and options on the previous pages. They do not enforce the problem's time or memory limits.

Save your source file before building, then type input in the terminal or Console/Run panel that opens. For Java, give the public class the same name as the file and leave out the `package` statement. Only submit source files to Kattis.

Geany

You can use Geany for C, C++, Java and Python 3. Press F9 to build your program or check its syntax, then F5 to run it in a terminal. For C and C++, F8 compiles without linking, so use F9 before F5.

Code::Blocks

Open your C or C++ source file and press F9 to build and run it. The default ACPC GCC 15 profile uses the appropriate compiler for each language. If the compiler-detection window appears on the first launch, confirm with **OK**.

VS Code

You can use VS Code for C, C++, Java and Python 3. Open VS Code from the Applications menu to work in `~/contest`, or open your own source folder. If VS Code asks whether you trust your own folder, choose **Yes**. Select your source file in the editor and press F6 for **ACPC: Build and run**. Type input in the Terminal panel. Ctrl+Shift+B only builds the program.

For Java, use F6 to compile with `javac` and the Kattis options. The ordinary Java Run action uses Eclipse's Java compiler instead.

CLion

You can use CLion for C, C++ and Python 3. Open your source folder, then your file, and wait for indexing to finish. For C and C++, use the green arrow beside `main`. For Python, right-click the file and choose **Run**. The default interpreter is PyPy. If the Evaluation Feedback window appears, choose **No Thanks**.

IntelliJ IDEA

Open the folder containing your Java file. Use the green arrow beside `main`, then choose **Run**. New projects use Java 25 and the runtime options given in the Java section.

PyCharm

Open the folder containing your Python file. Right-click the file and choose **Run**, or use the green arrow. The default interpreter is PyPy.

Eclipse

You can use Eclipse for C, C++, Java and Python 3. The Applications menu opens the contest workspace. Put your files in `~/contest`, expand the **contest** project and open your source file. From the external-tools dropdown, shown as a green arrow with a toolbox, choose **ACPC Build and run**. Type input in the Console panel.

For Java, use this action to compile with `javac` and the Kattis options. The ordinary Java Run action uses Eclipse's Java compiler instead. Select the source file again before starting another run. If you copy files into the folder, press F5 in the project tree to show them.

Project settings

No compiler path needs to be entered for the actions above. You can skip the theme tour if it appears when you first open CLion, IntelliJ IDEA or PyCharm.

An existing project, a different Eclipse workspace, a new Python virtual environment or a compiler you select yourself can change these settings. Use the supplied workspace and default interpreter, or check your program with the `compile*` and `run*` commands before submitting.

Your machine

- You are logged in automatically with the account `team`. Keep your files in your home directory (`/home/team`).
- You can use CLion, Code::Blocks, Eclipse, IntelliJ IDEA, PyCharm or VS Code. Vim, Gvim, Emacs, Geany, Kate, gedit and nano are also available. They are in the Applications menu, except nano, which runs in a terminal.
- The documentation of the languages is at `https://devdocs/` (bookmark "Language Documentation"). It has the documentation of C, C++, Java 25 and Python 3.11, and works without Internet access. In a terminal, `man 3 printf` shows the manual of a C library function and `pypy3 -m pydoc heapq` the documentation of a Python module.
- "Contest Help (offline)" in the bookmarks bar and the Contest menu contains searchable summaries about input and output, judging, local testing and common C, C++, Java and Python questions. It works without Internet access. The source links in the articles lead to the online Kattis help portal, which is unavailable during the contest.
- The network only gives access to the contest site, the Kattis status page and the documentation. Every other website is blocked.