How to Use the System

When Getting in Trouble with the System

Raise your hand to call a staff member and tell them about your trouble. Here are a few phrases that may be useful in typical situations:

- “We printed ____ (source code, debug output, …) about ____ minutes ago but have not received the printout yet.”
- “____ (browser, editor, …) does not start.”
- “Our ____ (computer, keyboard, mouse, …) stopped working.”

Contest staff will help you resolve the issue.

Do not use the Clarification in DOMjudge for system troubles. Use the Clarification only for questions about contest problems.

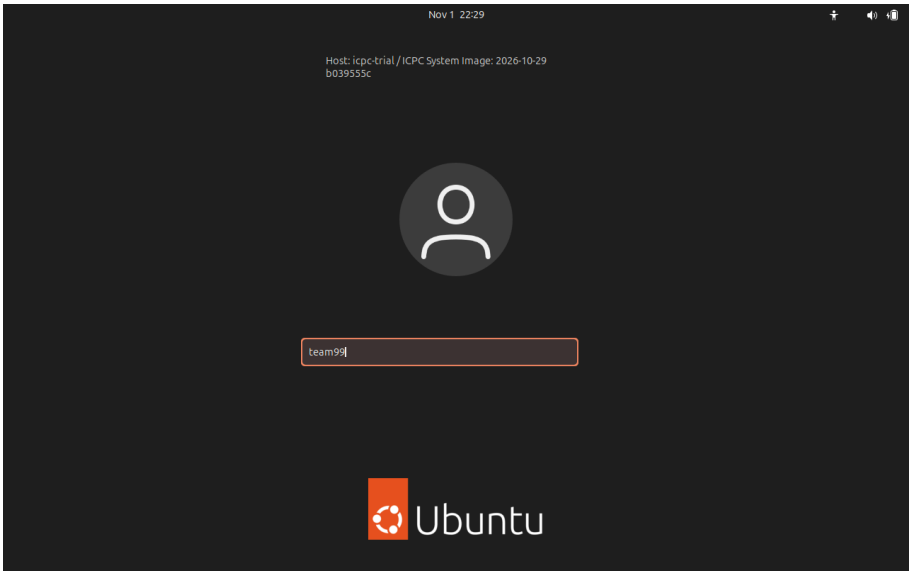
In the following explanations, the screenshots and texts may slightly differ from the actual interface.

1. Logging In to Your Computer

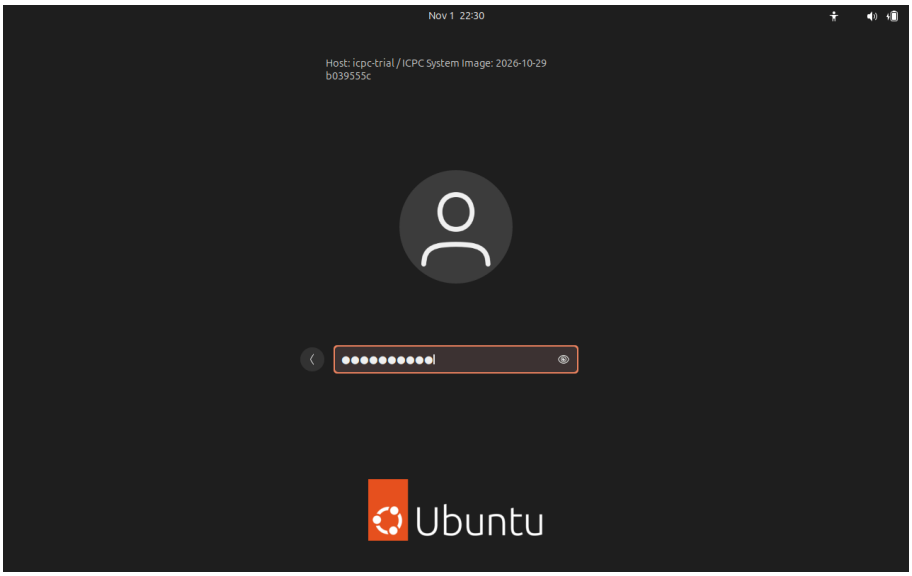
Your team ID and password are provided in a document in an envelope.

Note: You may use *different usernames and passwords* in the dress rehearsal session (on Saturday) and the main contest (on Sunday). You will receive a document of login information each day.

1. Type your username and press an `Enter` key.



2. Type your password and press an `Enter` key.



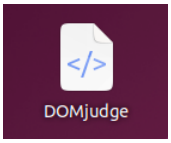
3. You should see a screen like this.



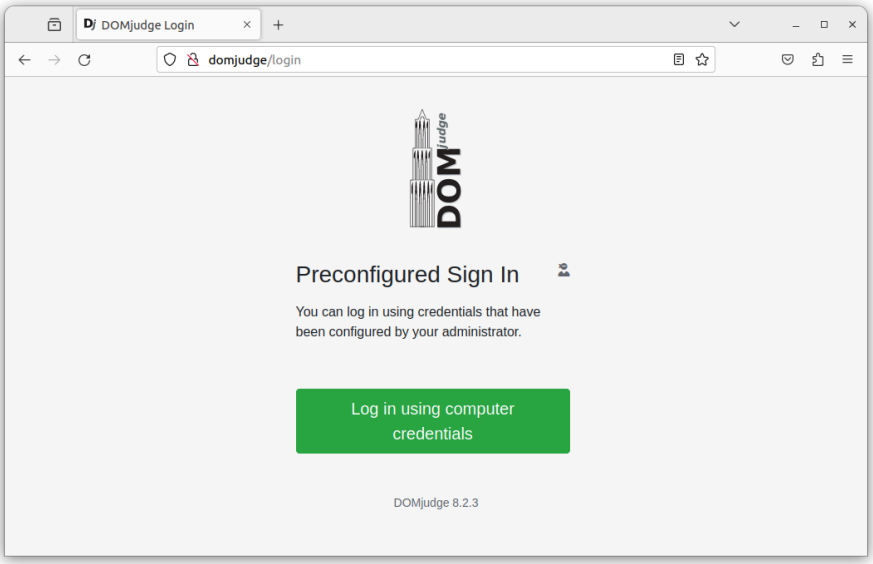
2. Logging In to the Contest System

We use DOMjudge 9.0.1 for the contest system this year.

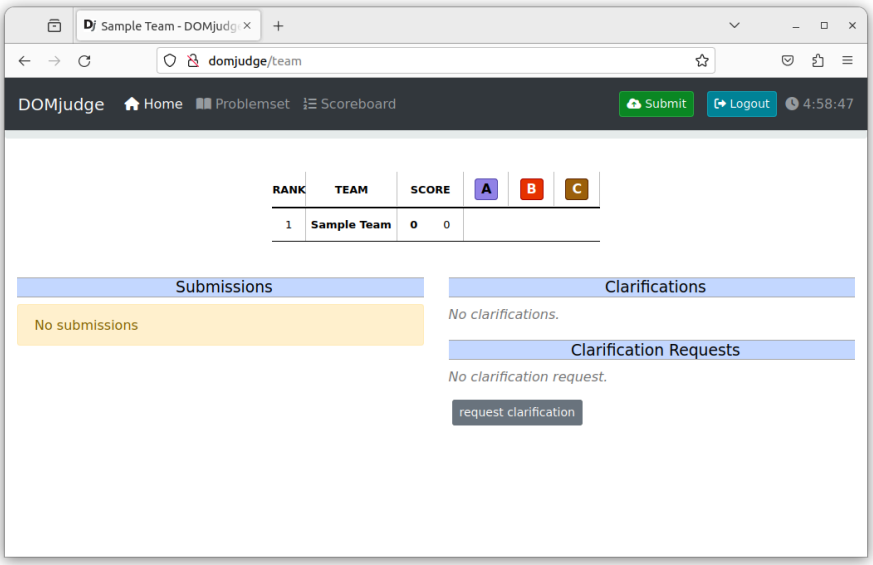
1. Double-click the `DOMjudge` shortcut on the desktop. It will open a new Firefox tab.



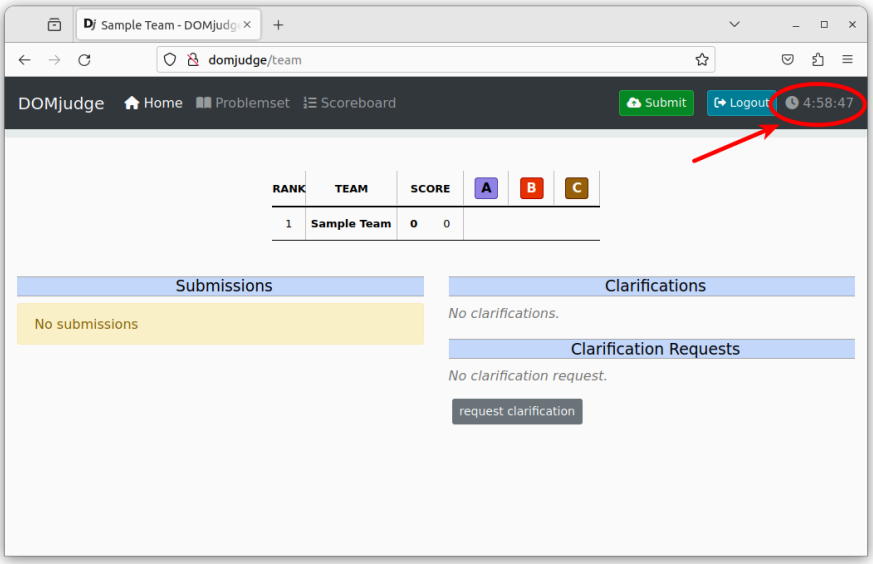
2. Click the `[Log in using computer credentials]` button. It will automatically log in as your team.



3. Once you are logged in, you will see a page like this.



4. You can check the remaining contest time at the upper-right corner.



After the successful log in, you can download problem statements and samples, submit your solution, see the scoreboard, request clarifications and more. Please check the "How to use DOMjudge contest system" section.

3. Printing Your Code and Debug Output

To print a text file, such as your code and debug output, open a terminal and run:

```
printfile $FILENAME
```

Here, `$FILENAME` is the name of a file you want to print. This command outputs "`Print request was sent.`" on success.

Your printouts will be delivered to your seat by staff. **Do not** attempt to pick them up by yourself; the printers are located in the staff area, which you cannot enter.

Do not use other methods, especially printing features in applications.

Only plain text is supported. **Do not** print anything else (e.g. PDF, HTML).

We do not have many printers. **Do not** attempt to print many pages. Printing the whole content of an STL document or a Java API document is a bad idea, for example.

4. Writing and Compiling Your Code

The system is set up with a number of IDEs and editors. Use your favorite. There is a shortcut on the desktop for each of them.

You can test your programs with the following commands. Just pass your source file(s) or Java class file(s) as the arguments. See the Appendix A for more detailed instructions.

- C: `compilegcc` / `runc`
- C++: `compileg++` / `runcpp`
- Java: `compilejava` / `runjava`
- Python 3: `compilepython3` / `runpython3`
- Kotlin: `compilekotlin` / `runkotlin`

5. Customizing the System

We strongly discourage you from changing system settings of the computer. If you choose to do so, you are responsible for all troubles it could cause, i.e. you will be doing it at your own risk.

Also be advised that the system will be reset to the original state between the rehearsal and the main contest. All files and customizations made during the rehearsal will be lost.

6. Taking Out Your Files after the Contest

We will pick up all the files under the home directory[*1] , except some special files such as files under the `.config` directory, and send them to you by email.

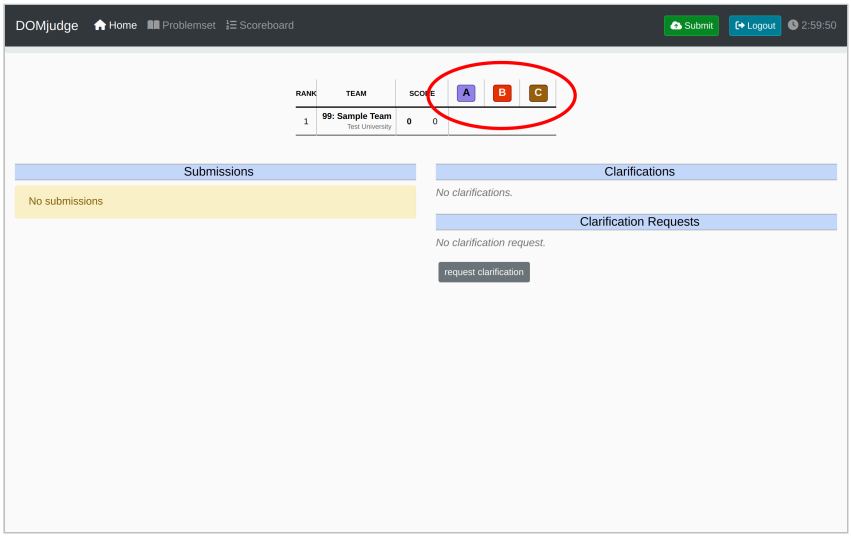
1. The home directory is usually the first directory when you open a shell in the terminal. For example, files on the desktop are also included.

How to use DOMjudge contest system

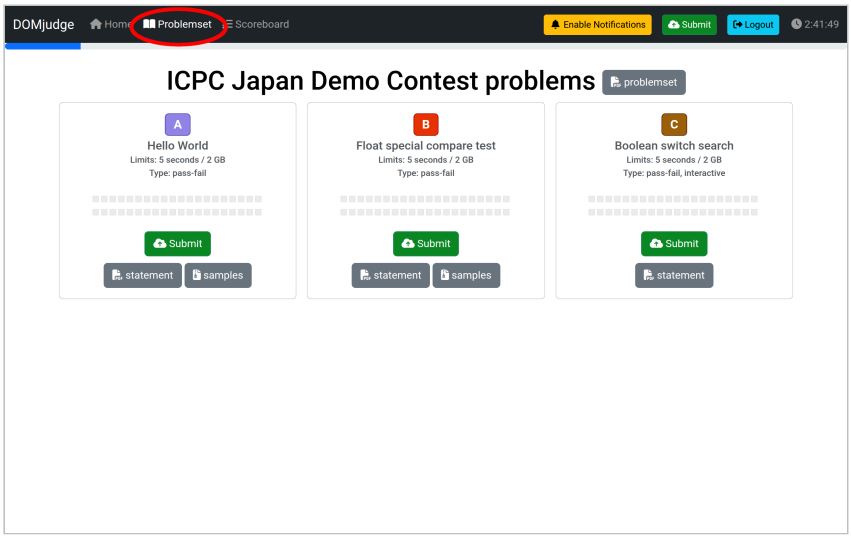
In the contest, we will use DOMjudge[*1] 9.0 as the contest system.

1. Download Problem Statement and Sample Input/Output

1. You can download a problem statement for each problem by clicking the problem ID at the center of the team page.

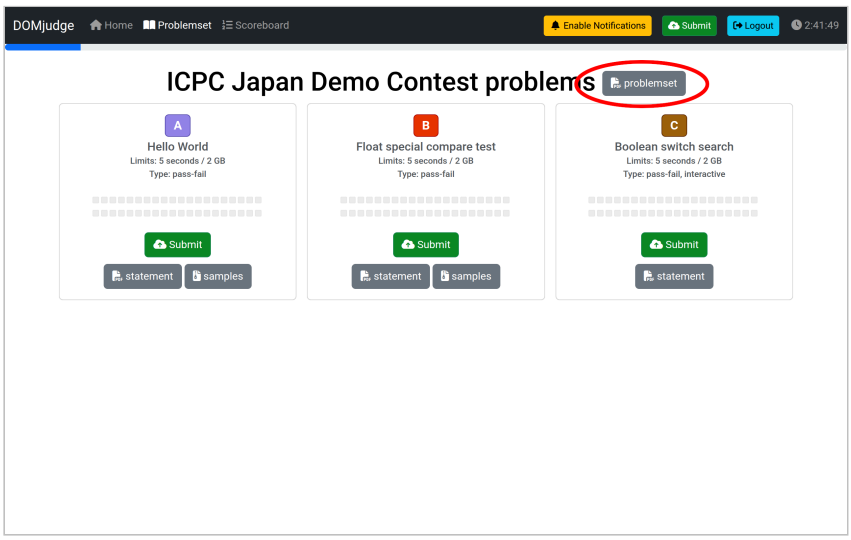


2. Also, by clicking `Problemset` at the top of the screen, you can see a list of problems. On this page, you can access a problem statement, a zip file containing sample data, and a submission dialog for each problem.



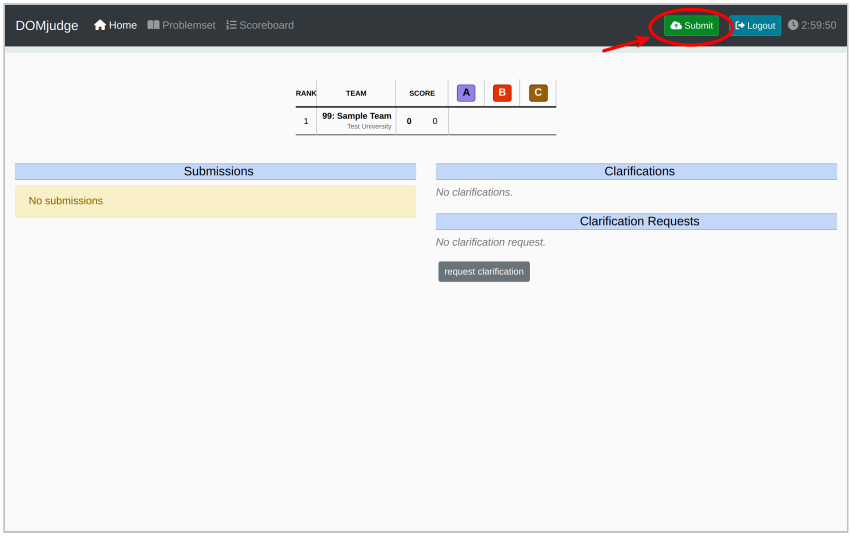
1. https://www.domjudge.org/

3. On the `Problemset` page, by clicking the `problemset` button on the right of the contest title, you can download a PDF file containing all the problem statements.

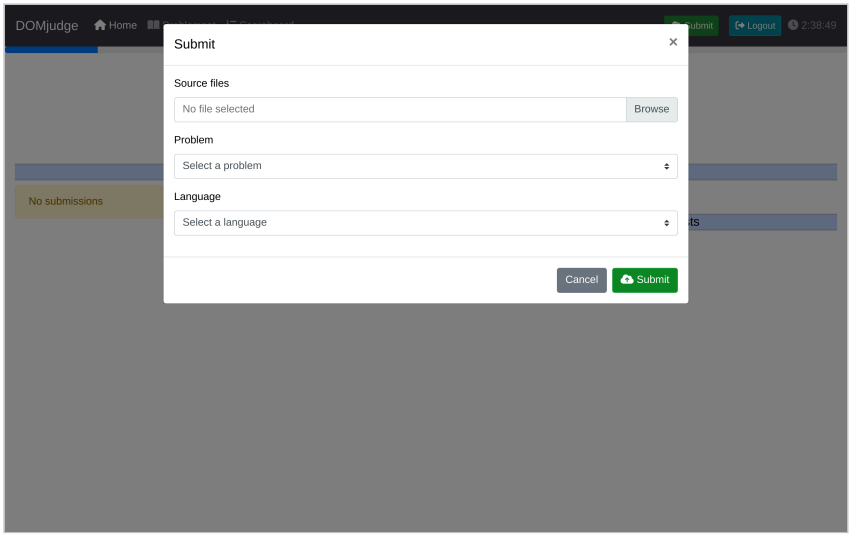


2. Submit

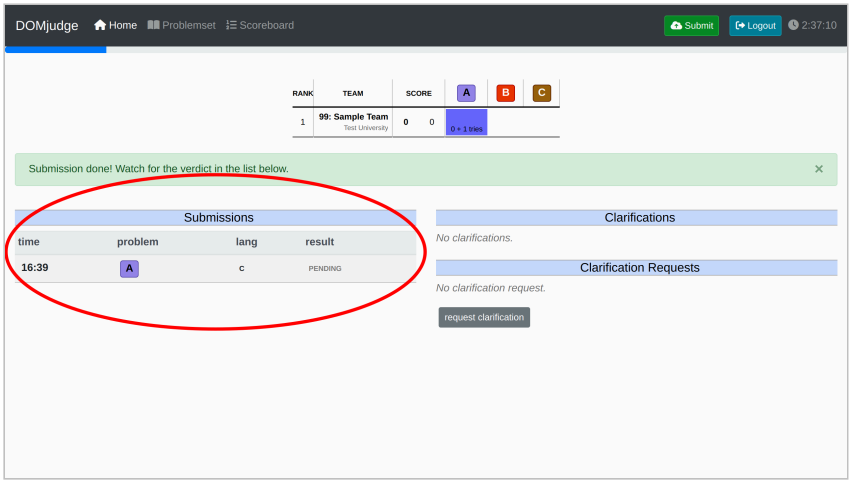
1. To submit, click the `Submit` button at the top right corner of the screen. A submission dialog will appear.



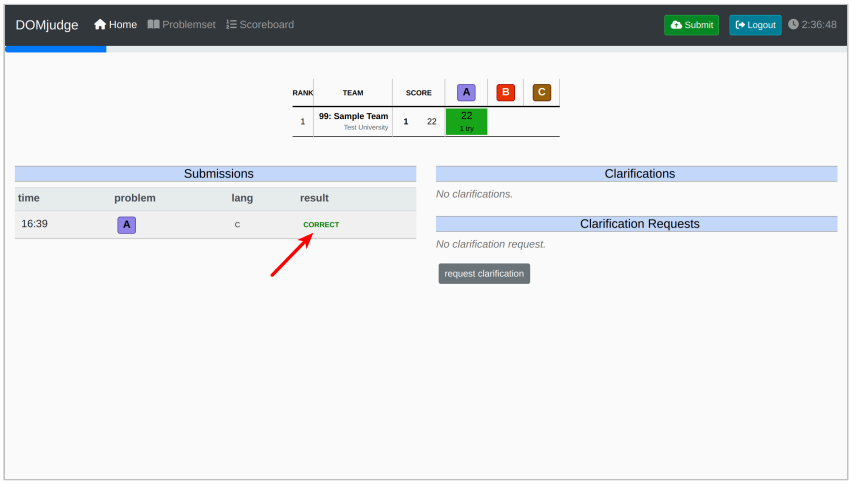
2. Select the source files and the problem you want to submit. The language is automatically detected from the file extension, but please make sure it is correct before submitting. Then, click the `Submit` button at the bottom right of the dialog. You can include multiple source files in a submission by holding the **`Ctrl`** key while selecting the files.



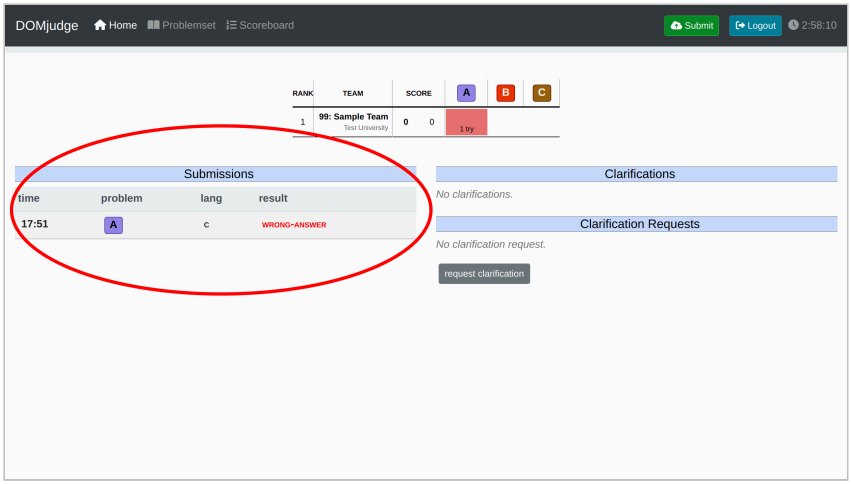
3. You can see submission results in the `Submissions` section of the team page. When you submit, it will be shown as `PENDING`, which means that the system is judging your submission.



4. When the judging completes, the result is updated on the team page. If the submission is judged to be correct, `CORRECT` will be shown.

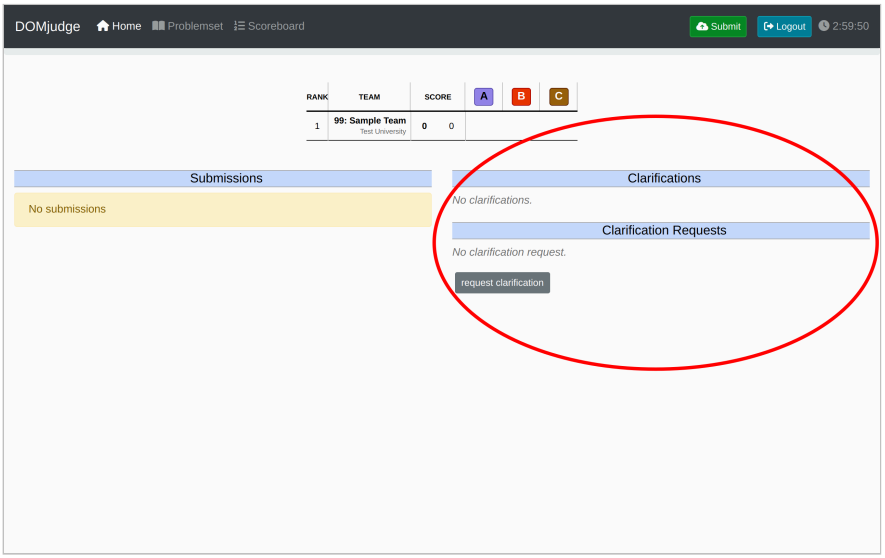


5. If the output of the submitted program is incorrect, `WRONG-ANSWER` will be shown. Other judging results, such as `TIMELIMIT` and `COMPILER-ERROR`, may be shown. Depending on the result, you may see the details by clicking the corresponding row. See the "Judging Notes" document for more information on judging results.



3. Request Clarification to the Judges

The `Clarifications` section of the team page shows messages and announcements from the judges.



If you have questions about ambiguities or potential issues in a problem statement during the contest, you can request a "clarification" to communicate with the judges.

- Before submitting a question, please carefully **re-read** the relevant problem statement.
- Clarification requests should focus on understanding the problem statement (e.g., "Is input X always positive?", "What does 'adjacent' mean in this context?"), not asking for hints ("How do I solve this?") or debugging help ("Why is my code getting a Wrong Answer?").

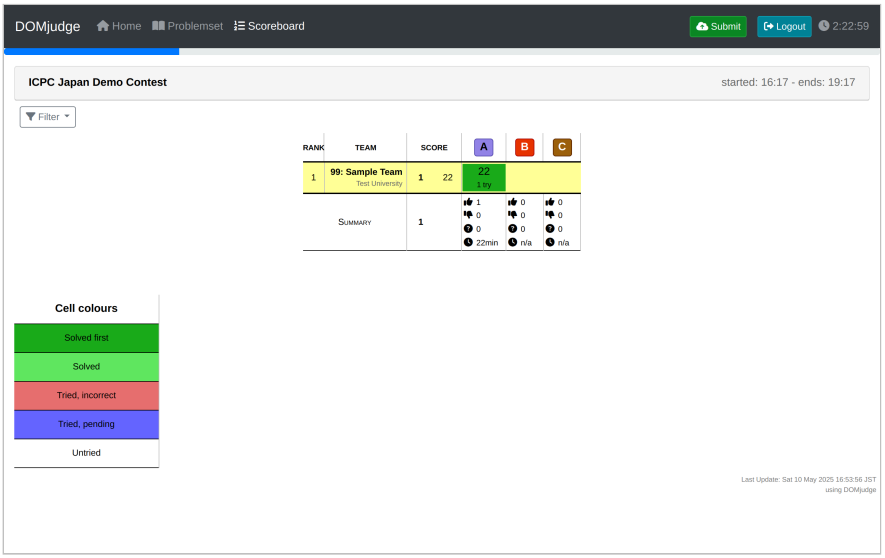
You can ask a question with the following steps.

1. Click the `request clarification` button. A dialog to send a question will appear.
2. Select an appropriate `Subject` from the options and write your question in the `Message` field. When finished, click the `Send` button.

Questions you send will be shown in `Clarification Requests` until an answer is received. After you receive the answer, it will be shown in the `Clarifications` section. Note that answers from the judges may be sent only to your team, or they may be made public to all participating teams.

4. Scoreboard

You can view the current scoreboard by clicking `Scoreboard` at the top of the screen.



The scoreboard shows the current submission status of each team. Each row represents the status of each team. The score column has two numbers:

- The number of problems solved correctly.
- The sum of the time to the first correct answer for each solved problem plus penalties. See the "Judging Notes" document for information on penalties.

For each problem, the following numbers are listed:

- The number of submissions.
- The time of the first correct submission since the contest starts (in minutes) if the problem has been solved.

The scoreboard will be frozen for the final hour of the contest to keep the final results secret until the award ceremony. During this freeze, judging results will be shown differently:

- On the public scoreboard (for spectators), the judging results of all submissions from all teams will be hidden.
- On your team's scoreboard, you will continue to see the full judging results for your own submissions, but all judging results from other teams will be hidden.

When a judging result is hidden, the display for any problem not solved before the freeze will change. The scoreboard will show the submission count as `x + y tries`, where `x` is the number of submissions made *before* the freeze, and `y` is the number of submissions made *during* the freeze.

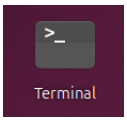
Appendix

1. Appendix A: How to Compile and Run Your Code

1.1. C/C++

We assume that you save your source file as `A.c` or `A.cpp` and an input file as `sample-A.1.in` in your home folder. We expect them to be saved there, not on the desktop, in this instruction.

1. Double-click the Terminal shortcut on the desktop.



2. Type as follows in the Terminal window (`↵` denotes an `Enter` key), depending on whether you are using C or C++. It compiles your code and generates a binary named `a.out`.
   - C:

     ```
     compilegcc A.c↵
     ```

   - C++:

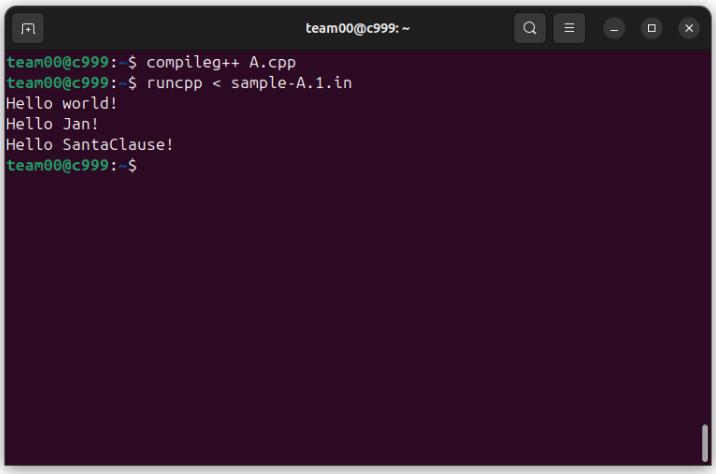
     ```
     compileg++ A.cpp↵
     ```

3. Type as follows to run your program (Use `runc` for C program instead of `runcpp`):

   ```
   runcpp < sample-A.1.in↵
   ```

   If you would like to save the output into a file named `my-A.1.out`:

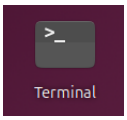
   ```
   runcpp < sample-A.1.in > my-A.1.out↵
   ```



1.2. Java

We assume that you save your source file as `A.java` and an input file as `sample-A.1.in` in your home folder. We expect them to be saved there, not on the desktop, in this instruction.

1. Double-click the Terminal shortcut on the desktop.



2. Type as follows in the Terminal window (`↵` denotes an `Enter` key). It compiles your code into Java class files (e.g. `A.class`).

```
compilejava A.java↵
```

3. Type as follows to run your program:

```
runjava A < sample-A.1.in↵
```

If you would like to save the output into a file named `my-A.1.out`:

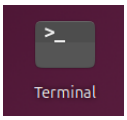
```
runjava A < sample-A.1.in > my-A.1.out↵
```



1.3. Python

We assume that you save your source file as `A.py` and an input file as `sample-A.1.in` in your home folder. We expect them to be saved there, not on the desktop, in this instruction.

1. Double-click the Terminal shortcut on the desktop.

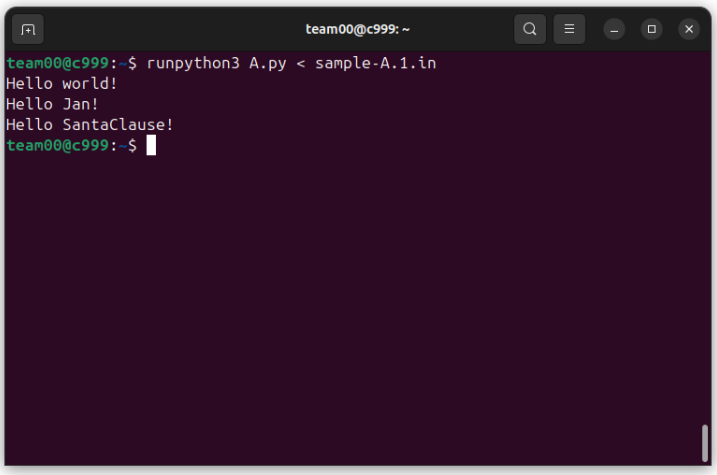


2. Type as follows in the Terminal window to run your program (↵ denotes an **Enter** key):

```
runpython3 A.py < sample-A.1.in↵
```

If you would like to save the output into a file named `my-A.1.out`:

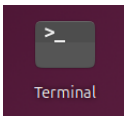
```
runpython3 A.py < sample-A.1.in > my-A.1.out↵
```



1.4. Kotlin

We assume that you save your source file as `A.kt` and an input file as `sample-A.1.in` in your home folder. We do not expect them to be saved there, not on the desktop, in this instruction.

1. Double-click the Terminal shortcut on the desktop.



2. Type as follows in the Terminal window (`↵` denotes an `Enter` key). It compiles your code into Java class files (e.g. `AKt.class`).

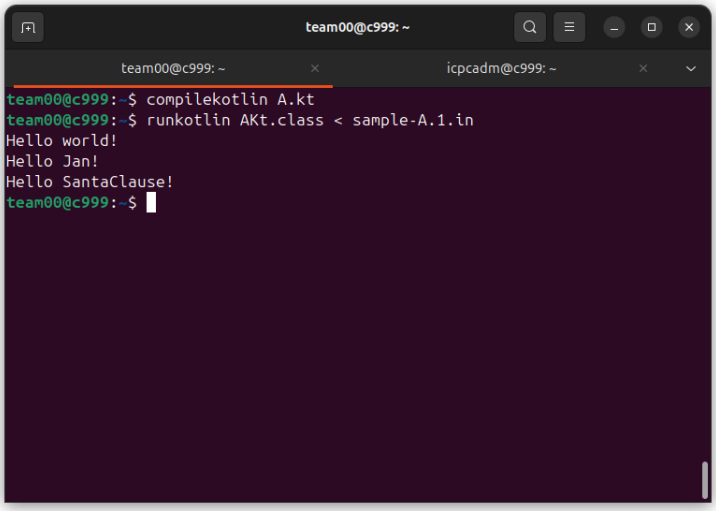
```
compilekotlin A.kt↵
```

3. Type as follows to run your program:

```
runkotlin AKt.class < sample-A.1.in↵
```

If you would like to save the output into a file named `my-A.1.out`:

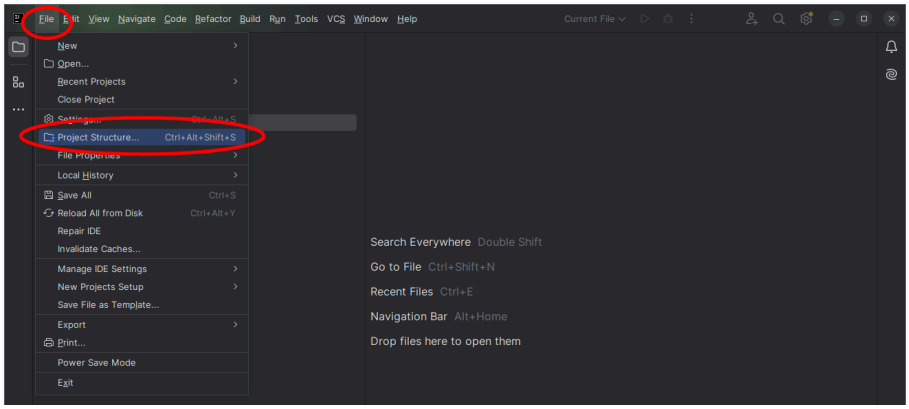
```
runkotlin AKt.class < sample-A.1.in > my-A.1.out↵
```



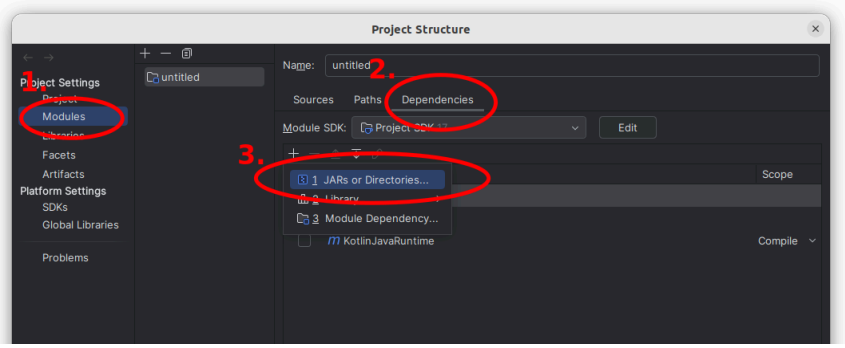
2. Appendix B: How to use Kotlin in IntelliJ IDEA

To use Kotlin's language features in IntelliJ IDEA (such as compiling, autocomplete, debugging, etc), you need to follow the steps below.

1. Open IntelliJ IDEA from the desktop icon, and create a new Kotlin project.
2. Click the top left "Main menu" (hamburger menu), then open the `[File]` menu and click the `[Project Structure]` option. You can alternatively open it by pressing **`Ctrl`**, **`Shift`**, **`Alt`**, and **`S`**.



3. Open `Modules` pane. From `Dependencies` tab, click `[+]` icon and click `[1. JARs or Directories...]`.



4. Add `/usr/local/lib` to the dependencies, and click `[OK]`.

